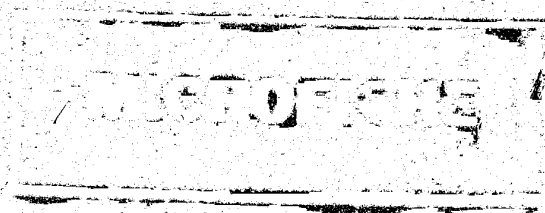


OPTIMAL MULTISHIFT PROPORTIONAL  
ROTATING SCHEDULES

William W. Stenzel, D.Sc.

The Institute for  
Public Program Analysis  
St. Louis, Missouri

43609  
curt



NCJRS

OCT 20 1977

ACQUISITIONS

WASHINGTON UNIVERSITY  
SEVER INSTITUTE OF TECHNOLOGY

---

OPTIMAL MULTISHIFT PROPORTIONAL

ROTATING SCHEDULES

BY

William W. Stenzel

Prepared under the direction of Nelson B. Heller, Ph.D.

---

A dissertation presented to the Sever Institute of  
Washington University in partial fulfillment  
of the requirements for the degree of

DOCTOR OF SCIENCE

November, 1976

Saint Louis, Missouri

The material in this project was prepared under Grant No. 72-NI-99-1064 from the Manpower Development Assistance Division, Office of Criminal Justice Assistance, Law Enforcement Assistance Administration, U. S. Department of Justice. Researchers undertaking such projects under Government sponsorship are encouraged to express freely their professional judgment. Therefore, points of view or opinions stated in this document do not necessarily represent the official position or policy of the U. S. Department of Justice.

WASHINGTON UNIVERSITY  
SEVER INSTITUTE OF TECHNOLOGY

---

ABSTRACT

---

OPTIMAL MULTISHIFT PROPORTIONAL  
ROTATING SCHEDULES

by William W. Stenzel

---

ADVISOR: Nelson B. Heller, Ph.D.

---

December, 1976

Saint Louis, Missouri

---

This work describes: (1) the development of a systematic procedure for the design of optimal multishift proportional rotating (PR) schedules which provide a given distribution of manpower by shift and day of the week; (2) the identification of preferred schedule attributes and quantitative measures for each that can be used in a multi-criteria decision model to discriminate between alternative schedules; (3) the incorporation of the design procedures into a set of computer programs capable of constructing optimal and near-optimal multishift PR schedules based on a given set of required and preferred schedule properties; and (4) the use of the computerized procedures to design manpower schedules for several units of the St. Louis Metropolitan Police Department.

The sequential process for the design of one-shift PR schedules, first proposed by Heller, consists of: (1) the aggregation of recreation days into recreation periods of acceptable lengths; (2) the distribution of each recreation period set over the days of the week to match the required manpower allocation; (3) the identification of the work period length defined by each ordered pair of recreation periods;

and (4) the selection of sequences of recreation periods which yield schedules with the required number of work days.

The clustering of recreation days into periods is a variant of the combinatoric problem of representing a positive integer  $n$  as the sum of two or more positive integers. A simple enumeration scheme is described for generating all distinct partitions. The distribution of recreation periods over the days of the week is schematically represented as a cyclic graph and a branching procedure is developed for enumerating all graphs for each partition. Using the information contained in a separation matrix associated with each graph, a branch-and-bound algorithm is used to enumerate all sequences of recreation periods which correspond to feasible one-shift schedules.

Multishift PR schedules are formed by designing a small set of dominating one-shift schedules for each shift tour, selecting one schedule from each set, and placing the schedules in proper rotation sequence. An "artificial" recreation period is used to control the placement of recreation periods at the beginning and end of each shift schedule. A simple branching process is used to enumerate optimal and near-optimal multishift schedules from the set of dominating schedules for each shift tour.

A survey of 21 police agencies and a review of scheduling literature is used to identify schedule attributes which can be used to characterize individual schedules. Quantitative measures for each attribute are used to design optimal schedules in two ways: acceptability measures are used to screen out schedules which fail to satisfy minimum requirements, and preference measures are used to determine preferable schedules from among acceptable candidates.

Acceptable schedules are those which: (1) preserve a given manpower allocation by shift and day of the week, (2) satisfy upper and lower limits on the lengths of all work and recreation periods, and (3) have a recreation period of acceptable length at each shift changeover point. Preference measures utilized include: (1) the number and frequency of weekend recreation periods, (2) the size and frequency of maximum length work periods, (3) the range between maximum and minimum length work periods, and (4) the number of preferable recreation periods (where preferability is determined by period length and days of the week covered).

Applications of the computerized design procedures are presented for two units of the St. Louis Metropolitan Police Department: the Evidence Technician Unit, and the Radar-Vascar and Motorcycle subunits of the Traffic Safety Unit. Comparisons of the computer designed schedules with work patterns designed by police personnel indicate the superiority of the computer designed schedules in terms of several important schedule properties.

TABLE OF CONTENTS

No.		Page
1.	Introduction . . . . .	1
1.1	Problem Background. . . . .	1
1.2	Police Manpower Scheduling. . . . .	5
1.2.1	The Demand for Police Service by Day and Shift. . . . .	5
1.2.2	Commonly Used Police Schedules . . . . .	6
1.2.3	Desirable Police Schedule Features . . . . .	11
1.3	Types of Manpower Schedules . . . . .	13
1.3.1	Shift Scheduling . . . . .	13
1.3.2	Days-Off Scheduling. . . . .	14
1.3.2.1	Fixed Schedules . . . . .	16
1.3.2.2	Cyclic Schedules. . . . .	17
1.3.2.3	Multishift Schedules. . . . .	19
1.4	Objectives and Outline of the Thesis. . . . .	23
1.4.1	Proportional Rotating Schedules. . . . .	23
1.4.2	Thesis Objectives. . . . .	27
1.4.3	Basic Assumptions and Conditions . . . . .	31
1.4.4	Thesis Outline . . . . .	32
1.5	Previous Research on Manpower Scheduling. . . . .	33
1.5.1	Introduction . . . . .	33
1.5.2	Event Scheduling . . . . .	34
1.5.3	Vehicle Scheduling . . . . .	36
1.5.4	Manpower Scheduling. . . . .	38
1.5.4.1	Measuring the Demand for Service and Setting Minimum Manning Requirements. . . . .	39
1.5.4.2	Shift Scheduling. . . . .	41
1.5.4.3	Days-Off Scheduling . . . . .	51

TABLE OF CONTENTS  
(continued)

No.		Page
2.	The Measurement of PR Schedules . . . . .	57
2.1	Introduction . . . . .	57
2.2	PR Schedule Attributes . . . . .	60
2.2.1	Introduction . . . . .	60
2.2.2	Weekend Recreation Periods. . . . .	60
2.2.3	Recreation Periods. . . . .	62
2.2.4	Work Periods. . . . .	68
2.2.5	Other Schedule Features . . . . .	71
2.2.6	Summary of PR Schedule Measures . . . . .	75
2.3	Sequential Lexicographic Comparison of PR Schedules . . . . .	82
2.3.1	Introduction. . . . .	82
2.3.2	Sequential Lexicographic Elimination. . . . .	82
2.4	Multiple Criteria Decision Making Procedures . . . . .	87
2.4.1	Introduction. . . . .	87
2.4.2	Collective Procedures . . . . .	88
2.4.3	Component Procedures. . . . .	90
	2.4.3.1 Conjunctive and Disjunctive Constraints. . . . .	91
	2.4.3.2 Dominance. . . . .	91
	2.4.3.3 Lexicography . . . . .	92
3.	The Design of Optimal PR Schedules. . . . .	95
3.1	Introduction . . . . .	95
3.2	Schedule Design Data Requirements. . . . .	96
3.2.1	Manpower Allocation . . . . .	96
3.2.2	Schedule Design Features. . . . .	101
3.3	The Design of Optimal One-Shift PR Schedules . . . . .	102

TABLE OF CONTENTS  
(continued)

No.		Page
3.3.1	Specification of Sets of Recreation Periods of Acceptable Length . . . . .	103
3.3.2	The Distribution of Recreation Periods Over Days of the Week. . . . .	104
3.3.3	Specification of Work Periods of Acceptable Length. . . . .	107
3.3.4	Enumeration of Acceptable Sequences of Work and Recreation Periods . . . . .	110
3.4	Overview of the Computer Program Logic for Designing Optimal One-Shift PR Schedules. . . . .	114
3.5	The Design of Optimal Multishift PR Schedules	116
3.5.1	Optimal Non-Cyclic, One-Shift PR Schedules. . . . .	116
3.5.2	Optimal Multishift Schedules . . . . .	132
3.6	Overview of the Computer Program Logic for Enumerating Optimal Multishift PR Schedules . . . . .	134
4.	The Enumeration of Acceptable Recreation Periods . . . . .	137
4.1	Introduction. . . . .	137
4.2	Basic Concepts of the Partitioning Algorithm. . . . .	140
4.2.1	Ranking Definition . . . . .	140
4.2.2	Step-Wise Search Procedure . . . . .	142
4.2.2.1	Fill Procedure. . . . .	142
4.2.2.2	Backtrack Procedure . . . . .	147
4.3	Partitioning Algorithm Logic. . . . .	153
4.4	Efficiency of the Partitioning Algorithm. . . . .	156

TABLE OF CONTENTS  
(continued)

No.	Page
4.4.1 Measure of Relative Efficiency . . . .	156
4.4.2 Upper Bounds on the Total Number of Partitions. . . . .	158
4.4.3 Upper Bounds on the Total Number of Permutations . . . . .	166
4.5 Use of the Partitioning Algorithm to Control Some PR Schedule Attributes . . . . .	170
5. Allocation of Individual Recreation Periods to Days of the Week . . . . .	174
5.1 Introduction. . . . .	174
5.2 Overview of the Cyclic Graph Enumeration Algorithm . . . . .	178
5.2.1 Tabular Representation of Cyclic Graphs . . . . .	178
5.2.2 Cyclic Graph Enumeration Logic . . . .	180
5.3 Basic Concepts of the Cyclic Graph Enumeration Algorithm . . . . .	188
5.3.1 The Number of Recreation Period Starts for Each Day of the Week. . . .	188
5.3.2 Enumeration of All Start Arrangements for Each Day of the Week . . . . .	189
5.3.3 Reduced Star Diagram . . . . .	195
5.4 Cyclic Graph Enumeration Algorithm Logic. . .	197
5.4.1 Iterative Process. . . . .	197
5.4.2 Two Example Problems . . . . .	201
5.4.3 Enumeration Algorithm Performance Characteristics. . . . .	218
5.4.4 Enumeration Algorithm Logic Summary. .	222
5.5 Weekend Recreation Periods. . . . .	225
5.5.1 Weekend Classification of Recreation Periods. . . . .	226
5.5.2 Acceleration of the Design Process Based on the Number of Full Weekend Recreation Periods . . . . .	232

TABLE OF CONTENTS  
(continued)

No.		Page
6.	The Enumeration of One-Shift Cyclic PR Schedules . . . . .	238
6.1	Introduction . . . . .	238
6.2	Number of Feasible Schedules from Each Cyclic Graph . . . . .	239
6.3	Work Periods Defined by the Sequence of Recreation Periods " . . . . .	245
6.4	Separation Matrix. . . . .	249
6.5	Use of the Separation Matrix to Construct One-Shift, Cyclic Schedules. . . . .	254
6.5.1	Selection of the Appropriate Separation Matrix . . . . .	257
6.5.2	Characteristics of the Designated Matrix Entries Corresponding to Feasible Schedules. . . . .	259
6.5.3	Analogy of the One-Shift Scheduling Problem to the Travelling Salesman Problem . . . . .	261
6.6	An Algorithm for the Enumeration of One-Shift Cyclic Schedules . . . . .	267
6.6.1	Enumeration of Feasible Sequences of Recreation Periods - The Branching Process . . . . .	268
6.6.1.1	Distinct Cyclic Schedules. . . . .	281
6.6.1.2	Cyclic Schedule Length . . . . .	285
6.6.2	Enumeration of Feasible Sequences of Recreation Periods - The Bounding Process . . . . .	290
6.6.3	The Branch-and-Bound Algorithm - A Sample Problem. . . . .	302
6.6.4	Summary of the Enumeration Algorithm. . . . .	304

TABLE OF CONTENTS  
(continued)

No.	Page
6.7 Acceleration Techniques for the Enumeration of One-Shift Cyclic Schedules . . . . .	308
6.7.1 Identical Recreation Periods . . . . .	309
6.7.2 Work Period Lengths... . . . .	320
6.7.3 Matrix Conditioning. . . . .	329
7. The Design of Multishift PR Schedules. . . . .	338
7.1 Introduction. . . . .	338
7.2 Multishift Schedule Properties. . . . .	339
7.3 Classification of Non-Cyclic, One-Shift Schedules . . . . .	348
7.4 Design and Construction of Dominating Non-Cyclic Schedules. . . . .	353
7.4.1 Introduction . . . . .	353
7.4.2 Artificial Recreation Periods. . . . .	354
7.4.3 Use of the Expanded Separation Matrix. . . . .	360
7.4.4 Modified Matrix Examples . . . . .	366
7.4.5 Verification of the Properties of the Corresponding Set of Cyclic Schedules. . . . .	372
7.4.6 The Design of Optimal Non-Cyclic Schedules. . . . .	374
7.5 The Enumeration of Optimal Multishift Schedules . . . . .	382
7.5.1 Introduction . . . . .	382
7.5.2 The Enumeration Process. . . . .	383
7.5.3 A Sample Problem . . . . .	386
7.5.4 Acceleration Techniques. . . . .	389
7.5.4.1 Growth Characteristics of the Enumeration Process . . . . .	389
7.5.4.2 Selection of the Level 1 Shift . . . . .	395
7.5.4.3 Ordering the Dominating Schedules for Each Shift. . . . .	398
7.5.4.4 Enumeration of Optimal and Near-Optimal Multishift Schedules . . . . .	404
7.5.5 Computer Code. . . . .	406

TABLE OF CONTENTS  
(continued)

No.		Page
8.	Conclusions . . . . .	407
8.1	Introduction . . . . .	407
8.2	Major Results. . . . .	407
8.2.1	Measures of Manpower Schedules. . . . .	407
8.2.2	Construction of Optimal One-Shift Schedules . . . . .	409
8.2.2.1	One-Shift Cyclic Schedules . . . . .	410
8.2.2.2	One-Shift Non-Cyclic Schedules. . . . .	415
8.2.3	Enumeration of Optimal and Near-Optimal Multishift PR Schedules. . . . .	423
8.3	Applications of Computerized Manpower Scheduling . . . . .	435
8.3.1	Evidence Technician Unit, St. Louis Police Department . . . . .	435
8.3.2	Traffic Safety Unit, St. Louis Police Department . . . . .	443
8.4	Future Work. . . . .	459
9.	Acknowledgements. . . . .	461
10.	Appendices. . . . .	463
Appendix 10.1	Survey of Police Manpower Scheduling Practices. . . . .	464
10.1.1	Introduction. . . . .	464
10.1.2	Survey Instrument and Implementation. . . . .	464
10.1.3	Survey Results. . . . .	468
10.1.3.1	General Schedule Properties . . . . .	468
10.1.3.2	Workload Distribution and Manpower Allocation . . . . .	468
10.1.3.3	Shift Rotation Properties . . . . .	471
10.1.3.4	Schedule Preference Measures. . . . .	474

TABLE OF CONTENTS  
(continued)

No.		Page
	Appendix 10.2 Derivation of the Equation for the Exact Number of Distinct, Feasible One-Shift PR Schedules for a Given Cyclic Graph . . . . .	477
10.2.1	Notation. . . . .	477
10.2.2	Recreation Period Interactions. . . . .	477
10.2.2.1	Recreation Periods that Begin on the Same Day of the Week . . . . .	478
10.2.2.2	Recreation Periods that Begin on the Same Day of the Week and Have the Same Length. . . . .	479
10.2.2.3	Non-Start Recreation Days . . . . .	480
10.2.3	Exact Number of Distinct Schedules for One Class of Cyclic Graphs. . . . .	483
10.2.4	Summary . . . . .	487
11.	Bibliography. . . . .	488
12.	Vita. . . . .	496

LIST OF TABLES

No.		Page
1.1	Percentage Distribution of Calls-for-Service by Shift and Day of the Week, St. Louis Metropolitan Police Department, 1971 . . . . .	7
1.2	Percentage Distribution of Police Workload by Shift for Ten United States Cities . . . . .	8
1.3	Required On-Duty Manpower Levels by Shift and Day of the Week. . . . .	25
2.1	Twenty-One Recreation Periods Classified by Length and Starting Day of the Week with $U_R=4$ and $L_R=2$ . . . . .	65
2.2	Daily Manpower Allocation for a Seven-Week Schedule . . . . .	66
2.3	Number of Recreation Period Types in the PR Schedules in Figures 2.1 and 2.2 . . . . .	68
2.4	Time Off Between Shift Assignments for "Backward" Rotating Schedules . . . . .	72
2.5	Time Off Between Shift Assignments for "Forward" Rotating Schedules . . . . .	72
2.6	Standard Deviations of the Work to Recreation Period Length Ratios for the Schedules in Figures 2.1 and 2.2. . . . .	75
2.7	Preference Measure Values for Multishift Schedules I, II, III, and IV in Figures 2.4, 2.5, 2.6, and 2.7. . . . .	81
2.8	Preference Rankings of the Nine Schedule Measures Used to Design PR Schedules. . . . .	84
2.9	Preference Rankings for Individual Recreation Periods Used to Design PR Schedules. . . . .	85
3.1	Allocation of 95 Manshifts Over Three Shifts and Seven Days of the Week . . . . .	99
3.2	Distribution of the Recreation Days Corresponding to the Manpower Allocation in Table 3.1 by Shift and Day of the Week. . . . .	99
3.3	Daily Manpower Allocation for a Five Week Schedule	124

LIST OF TABLES  
(continued)

No.	Page
4.1 Permutations Enumerated to Partition Eight Recreation Days ( $R=8$ ) into Periods of Four, Three, and Two Days in Length ( $L=(4,3,2)$ ) . . . .	155
4.2 Performance Statistics for Several Examples of the Partitioning Algorithm. . . . .	157
4.3 Values of $P_{\ell}(R)$ . . . . .	160
4.4 Upper Bounds on the Total Number of Partitions $B(R,L)$ Enumerated by the Partitioning Algorithm .	169
5.1 Tabular Representation of the Cyclic Graph in Figure 5.2. . . . .	179
5.2 Tabular Representation of the Cyclic Graph in Figure 5.3. . . . .	180
5.3 Maximum and Minimum Number of Starts for Each Day of the Week for the Star Diagram in Figure 5.1. . . . .	190
5.4 Distinct Start Arrangements for Wednesday, Selected as the First Day from the Star Diagram in Figure 5.1 . . . . .	191
5.5 Start Arrangements for Two Recreation Periods, Enumerated by the Partitioning Algorithm, for Wednesday of the Star Diagram in Figure 5.1 . . .	193
5.6 Start Arrangements for One Recreation Period, Enumerated by the Partitioning Algorithm for Wednesday of the Star Diagram in Figure 5.1 . . .	193
5.7 Upper and Lower Bounds on the Number of Period Starts for Each Day Based on the Star Diagram in Figure 5.7 . . . . .	203
5.8 Upper and Lower Bounds on the Number of Period Starts for Each Day Based on the Reduced Star Diagram in Figure 5.9 . . . . .	208
5.9 Tabular Representation of the First Solution for the Cyclic Graph Enumeration Example (Step 5 in Figure 5.8). . . . .	210

LIST OF TABLES  
(continued)

No.	Page
5.10 Upper and Lower Bounds on the Number of Period Starts for Each Day Based on the Reduced Star Diagram in Figure 5.11 . . . . .	212
5.11 Tabular Representation of the Second Solution for the Cyclic Graph Enumeration Example (Step 14 in Figure 5.8) . . . . .	214
5.12 Tabular Representation of the Third Solution for the Cyclic Graph Enumeration Example (Step 25 in Figure 5.8) . . . . .	216
5.13 Performance Characteristics of the Cyclic Graph Enumeration Algorithm for Ten Sample Problems with Uniform Recreation Day Distributions. . . . .	220
5.14 Performance Characteristics of the Cyclic Graph Enumeration Algorithm for Ten Sample Problems with Non-Uniform Recreation Day Distributions. . . . .	221
5.15 Weekend Classification for Recreation Periods from One to Six Days in Length and Beginning on Each Day of the Week . . . . .	227
5.16 Manpower Allocation for a Ten-Week PR Schedule with a Maximum of One Full Weekend Recreation Period per Rotation Period . . . . .	229
5.17 Manpower Allocation for a Ten-Week PR Schedule with a Maximum of Two Full Weekend Recreation Periods per Rotation Period. . . . .	229
6.1 Work Period Lengths for the Schedule in Figure 6.2 . . . . .	247
6.2 Work Period Lengths for the Schedule in Figure 6.3 . . . . .	248
6.3 Work Period Lengths for the Schedule in Figure 6.8 . . . . .	255
6.4 Number of Distinct Values for Each Separation Matrix Entry . . . . .	258

LIST OF TABLES  
(continued)

No.		Page
7.1	Preference Measures Used to Rank Non-Cyclic Schedules . . . . .	375
7.2	Dominating Schedule Sets for the Three-Shift Sample Problem. . . . .	387
7.3	Number of Schedules and Nodes Enumerated for Different Changeover Period Limits and Number of Shifts. . . . .	396
7.4	Estimated Number of Active Nodes for Cyclic Permutations of the Shift Sequence for Three Sample Problems . . . . .	399
8.1	Daily Manpower Allocation for a Nine-Week Schedule. . . . .	410
8.2	Daily Manpower Allocation for the Day Watch . . .	416
8.3	Number of Partitions, Cyclic Graphs, and Dominating Schedules Enumerated for the Day Watch Data in Figure 8.4. . . . .	422
8.4	Number of Partitions, Cyclic Graphs, and Dominating Schedules Enumerated for the Watch Data in Figures 8.4, 8.8, and 8.9 . . . . .	426
8.5	Number of ETU Radio Assignments by Shift and Day of the Week, January 1, 1972 - October 10, 1972 .	437
8.6	Manpower Allocation by Shift and Day of the Week for the 1973 Evidence Technician Unit Schedule. .	438
8.7	Manpower Allocation by Shift and Day of the Week for the Seventeen Men Assigned to the 1973 ETU PR Schedule . . . . .	440
8.8	Number of ETU Radio Assignments by Shift and Day of the Week, September 1, 1972 - June 30, 1973. .	444
8.9	Manpower Allocation by Shift and Day of the Week for the 1974 Evidence Technician Unit Schedule. .	445

LIST OF TABLES  
(continued)

No.	Page
8.10 Number of Reported Accidents by Shift and Day of the Week Covered by the Radar-Vascar and Motorcycle Units, Traffic Safety Unit, City of St. Louis, 1971 . . . . .	449
8.11 Manpower Allocation by Shift and Day of the Week for the 1973 Radar-Vascar Unit Schedule. . . . .	450
8.12 Comparison of the Computer-Designed Radar-Vascar Unit Schedule for 1973 with the Unit's Manually-Designed Schedules for 1972 and 1973 . . . . .	454
8.13 Manpower Allocation by Shift and Day of the Week for the 1973 Motorcycle Unit Schedule. . . . .	455
8.14 Comparison of the Computer-Designed Motorcycle Unit Schedule for 1973 with the Unit's Manually-Designed Schedule for 1972 and 1973. . . . .	458

Appendices:

10.1.1 Police Departments Surveyed, Traffic Institute Northwestern University, June 1972 . . . . .	466
10.1.2 General Schedule Properties Based on Survey Questions 1, 2, and 3. . . . .	469
10.1.3 Workload Distribution and Manpower Allocation (Survey Questions 5 and 6) . . . . .	470
10.1.4 Shift Rotation Properties Based on Survey Questions 4 and 7. . . . .	472
10.1.5 Schedule Preference Measures Based on Survey Questions 8, 9, and 10 . . . . .	475

LIST OF FIGURES

No.		Page
1.1	Sample Seven-Bracket Schedule . . . . .	15
1.2	Two-Shift Multishift Schedule . . . . .	20
1.3	Nineteen-Week Multishift PR Schedule. . . . .	26
1.4	Nineteen-Week Multishift PR Schedule. . . . .	28
1.5	Nineteen-Week Multitour, Multishift PR Schedule .	29
2.1	Sample Schedule Based on the Manpower Allocation in Table 2.2. . . . .	67
2.2	Sample Schedule Based on the Manpower Allocation in Table 2.2. . . . .	67
2.3	Defining Criteria for Feasible, Acceptable, and Preferable Sets of PR Schedules . . . . .	77
2.4	Sample Multishift Schedule I. . . . .	79
2.5	Sample Multishift Schedule II . . . . .	79
2.6	Sample Multishift Schedule III. . . . .	80
2.7	Sample Multishift Schedule IV . . . . .	80
3.1	Star Diagram with Eleven Nodes. . . . .	105
3.2	Cyclic Graph Based on the Star Diagram in Figure 3.1. . . . .	106
3.3	Elementary Separation Matrix Based on the Cyclic Graph in Figure 3.2 . . . . .	107
3.4	Modified Separation Matrix Based on the Cyclic Graph in Figure 3.2 . . . . .	109
3.5	Five-Week, One-Shift PR Schedule Based on the Recreation Period Sequence {1,2,3,4}. . . . .	111
3.6	Five-Week, One-Shift PR Schedule Based on the Recreation Period Sequence {1,3,4,2}. . . . .	113
3.7	Flow Diagram for the Computerized Design of Optimal One-Shift PR Schedules (EXEC Computer Code) . . . . .	115

LIST OF FIGURES  
(continued)

No.		Page
3.8	Four-Week PR Schedule . . . . .	118
3.9	Four-Week PR Schedule . . . . .	118
3.10	Seven-Week Multishift Schedule Commencing with the Four-Week Schedule in Figure 3.8. . . . .	119
3.11	Seven-Week Multishift Schedule Commencing with the Four-Week Schedule in Figure 3.9. . . . .	119
3.12	Reduced Star Diagram, Nine Nodes. . . . .	127
3.13	Cyclic Graph Based on the {4,3,2,1,1} Partition .	127
3.14	Expanded Elementary Separation Matrix Based on the Cyclic Graph in Figure 3.13 . . . . .	128
3.15	Modified Separation Matrix with Two Dedicated Entries Based on the Cyclic Graph in Figure 3.13. .	129
3.16	Non-Cyclic PR Schedule Based on the Recreation Period Sequence {a,1,2,3,4,5} . . . . .	131
3.17	Non-Cyclic PR Schedule Based on the Recreation Period Sequence {a,1,3,2,4,5} . . . . .	131
3.18	Flow Diagram for the Computerized Design of Multishift PR Schedules (MERGE Computer Code) . .	136
5.1	Star Diagram with Eleven Nodes. . . . .	174
5.2	Cyclic Graph with Four Recreation Periods Based on the Star Diagram in Figure 5.1 . . . . .	176
5.3	Alternate Cyclic Graph Based on the Same Recreation Period Lengths and Star Diagram as the Cyclic Graph in Figure 5.2. . . . .	177
5.4	Schematic Representation of the Branching Process for the Cyclic Graph Enumeration Algorithm. . . .	186
5.5	Reduced Star Diagram, One Wednesday Node and One Thursday Node Removed from the Star Diagram in Figure 5.1. . . . .	196

LIST OF FIGURES  
(continued)

No.		Page
5.6	Basic Logic Flow for the Cyclic Graph Enumeration Algorithm . . . . .	198
5.7	Initial Star Diagram for the Cyclic Graph Enumeration Example . . . . .	203
5.8	Tree Structure for the Cyclic Graph Enumeration Example . . . . .	205
5.9	Reduced Star Diagram Corresponding to the Assignment of Two Period Starts (One Three-Day Period and One Two-Day Period) to Day C . . . . .	207
5.10	First Cyclic Graph Solution (Step 5 in Figure 5.8)	211
5.11	Reduced Star Diagram Corresponding to the Assignment of Two 2-Day Recreation Periods to Day C . . . . .	212
5.12	Second Cyclic Graph Solution (Step 14 in Figure 5.8) . . . . .	214
5.13	Third Cyclic Graph Solution (Step 25 in Figure 5.8) . . . . .	216
5.14	Star Diagram with Twelve Nodes. . . . .	217
5.15	Thirteen Cyclic Graphs Enumerated for the Star Diagram in Figure 5.14 ( $L=(3,2)$ and $P=(2,3)$ ). . . . .	219
6.1	Cyclic Graph with Five Recreation Periods . . . . .	239
6.2	PR Schedule Based on the Five Recreation Periods in Figure 6.1 . . . . .	240
6.3	PR Schedule Based on the Five Recreation Periods in Figure 6.1 . . . . .	242
6.4	Cyclic Graph with Eight Recreation Periods. . . . .	245
6.5	Elementary Separation Matrix for the Cyclic Graph in Figure 6.1 . . . . .	250

LIST OF FIGURES  
(continued)

No.	Page
6.6 Placement of Recreation Periods 1 and 2 for Minimum Separation . . . . .	251
6.7 Placement of Recreation Periods 1 and 2 for Minimum Separation Plus One Week . . . . .	252
6.8 PR Schedule Based on the Cyclic Graph in Figure 6.1 . . . . .	254
6.9 Modified Separation Matrix Obtained from the Elementary Matrix in Figure 6.5. . . . .	255
6.10 Distance Matrix for a Five-City Travelling Salesman Problem . . . . .	262
6.11 Tree of Solutions for the Enumeration of Feasible Sequences of Recreation Periods for a 3x3 Separation Matrix. . . . .	271
6.12 Sample Cyclic Graph with Three Recreation Periods.	274
6.13 Separation Matrix for the Cyclic Graph in Figure 6.12. . . . .	275
6.14 PR Schedule Constructed with the {1,2,3,1} Sequence of Recreation Periods and the Separation Matrix in Figure 6.13. . . . .	276
6.15 PR Schedule Constructed with the {2,3,1,2} Sequence of Recreation Periods and the Separation Matrix in Figure 6.13. . . . .	278
6.16 PR Schedule Constructed with the {3,1,2,3} Sequence of Recreation Periods and the Separation Matrix in Figure 6.13. . . . .	278
6.17 PR Schedule Constructed with the {1,3,2,1} Sequence of Recreation Periods and the Separation Matrix in Figure 6.13. . . . .	279
6.18 PR Schedule Constructed with the {3,2,1,3} Sequence of Recreation Periods and the Separation Matrix in Figure 6.13. . . . .	279

LIST OF FIGURES  
(continued)

No.	Page
6.19 PR Schedule Constructed with the {2,1,3,2} Sequence of Recreation Periods and the Separation Matrix in Figure 6.13 . . . . .	280
6.20 Cyclic Graph with Four Recreation Periods. . . . .	296
6.21 Separation Matrix Based on the Cyclic Graph in Figure 6.20 . . . . .	296
6.22 Reduced Separation Matrices for Partial Sequences {1,3,...} and {1,3,2,...} Based on the Initial Separation Matrix in Figure 6.21 . . . . .	299
6.23 Partial Tree Structure for the Enumeration of One-Shift Schedules Based on the Separation Matrix in Figure 6.21. . . . .	303
6.24 Five-Week PR Schedule Enumerated as Solution 1 in Figure 6.23 . . . . .	305
6.25 Five-Week PR Schedule Enumerated as Solution 2 in Figure 6.23 . . . . .	305
6.26 Reduced Separation Matrix Associated with Each Node in the Partial Tree Structure in Figure 6.23. . . . .	306
6.27 Cyclic Graph with Two Pairs of Identical Recreation Periods . . . . .	310
6.28 Separation Matrix Based on the Cyclic Graph in Figure 6.27. . . . .	311
6.29 PR Schedule Based on the {1,2,3,4,1} Sequence of Recreation Periods and the Separation Matrix in Figure 6.28 . . . . .	311
6.30 PR Schedule Based on the {1,2,4,3,1} Sequence of Recreation Periods and the Separation Matrix in Figure 6.28 . . . . .	312
6.31 Partial Tree Diagram for the Enumeration of the Schedules in Figures 6.29 and 6.30 . . . . .	315

LIST OF FIGURES  
(continued)

No.	Page
6.32 Tree Diagram for the Enumeration of Recreation Period Sequences Based on the Cyclic Graph in Figure 6.27 . . . . .	318
6.33 PR Schedule Based on the {1,3,2,4,1} Sequence of Recreation Periods and the Separation Matrix in Figure 6.28 . . . . .	319
6.34 PR Schedule Based on the {1,3,4,2,1} Sequence of Recreation Periods and the Separation Matrix in Figure 6.28 . . . . .	319
6.35 Partitioning Sets for a Non-Integer Valued Average Work Period Length. . . . .	323
6.36 Partitioning Sets for an Integer Valued Average Work Period Length. . . . .	325
6.37 Original 4×4 Separation Matrix and Reduced (6,7) Matrix. . . . .	327
6.38 Conditioned Reduced Matrix from Figure 6.37 . . .	333
6.39 Original 4×4 Separation Matrix and Reduced (6,8) Matrix. . . . .	335
7.1 Sample Three-Shift PR Schedule. . . . .	340
7.2 Three One-Shift Schedules Used in the Multishift Schedule in Figure 7.1. . . . .	343
7.3 Sample Three-Shift PR Schedule. . . . .	345
7.4 Actual Nine-Week Two-Shift Schedule . . . . .	355
7.5 Perception of the Nine-Week Schedule in Figure 7.4 by a Supervisor Permanently Assigned to Shift A. . . . .	355
7.6 Sample Cyclic Graph with Four Recreation Periods.	357
7.7 Sample Cyclic Graph with an Artificial Recreation Period. . . . .	357

LIST OF FIGURES  
(continued)

No.		Page
7.8	Elementary Separation Matrix for the Cyclic Graph in Figure 7.6 . . . . .	358
7.9	Expanded Elementary Separation Matrix for the Cyclic Graph in Figure 7.7. ....	358
7.10	Four Schedule Types for the Placement of Beginning and Ending Recreation Periods in Non-Cyclic Schedules. . . . .	362
7.11	Modified Separation Matrix for a Type I Schedule (0,0) Based on the Elementary Separation Matrix in Figure 7.9. . . . .	367
7.12	Modified Separation Matrix for a Type II Schedule (b,c) Based on the Elementary Separation Matrix in Figure 7.9 . . . . .	367
7.13	Modified Separation Matrix for a Type III Schedule (0,e) Based on the Elementary Separation Matrix in Figure 7.9. . . . .	368
7.14	Modified Separation Matrix for a Type IV Schedule (b,e) Based on the Elementary Separation Matrix in Figure 7.9 . . . . .	368
7.15	Two Type II Non-Cyclic Schedules Based on the Modified Separation Matrix in Figure 7.12 . . . .	371
7.16	A Type III Non-Cyclic Schedule Based on the Modified Separation Matrix in Figure 7.13 . . . .	371
7.17	Eleven Step Procedure for the Algorithmic Construction of Dominating Non-Cyclic Schedules .	376
7.18	Reduced Star Diagram. ....	380
7.19	Reduced Cyclic Graph. . . . .	381
7.20	Schematic Diagram of the Tree Search Procedure for Constructing Multishift Schedules . . . . .	384
7.21	Tree Diagram for the Three-Shift Search Example .	388

LIST OF FIGURES  
(continued)

No.		Page
7.22	F Values and Screening Factors (f) for the Multishift Schedule Enumeration Procedure as a Function of the Upper and Lower Limits on the Length of Changeover Recreation Periods . . .	394
8.1	Initial Data for a Nine-Week Cyclic Schedule, EXEC Code Printout. . . . .	411
8.2	Nine-Week Cyclic Schedule, EXEC Code Printout . .	412
8.3	Preference Measures for the Nine-Week Cyclic Schedule in Figure 8.2, EXEC Code Printout. . . .	414
8.4	Initial Data, Day Watch, EXEC Code Printout . . .	417
8.5	Number of Partitions and Cyclic Graphs Enumerated for the (0,2) Set of Schedules for the Day Watch, EXEC Code Printout . . . . .	418
8.6	Dominant Schedule and Preference Measures for the (0,2) Set of Schedules for the Day Watch, EXEC Code Printout . . . . .	420
8.7	Enumeration Statistics and Result for the (4,2) Set of Schedules for the Day Watch, EXEC Code Printout. . . . .	421
8.8	Initial Data for the Afternoon Watch, EXEC Code Printout. . . . .	424
8.9	Initial Data for the Night Watch, EXEC Code Printout. . . . .	425
8.10	Initial Data for the Three Watch Example, MERGE Code Printout . . . . .	428
8.11	Preference Measures for Top Ten Ranked Multishift Schedules with Six Weekend Recreation Periods and Three Consecutive Working Weekends, MERGE Code Printout. . . . .	429
8.12	Top Ranked Multishift Schedule, MERGE Code Printout. . . . .	431

LIST OF FIGURES  
(continued)

No.		Page
8.13	Preference Measures for the Multishift Schedule in Figure 8.12, MERGE Code Printout . . . . .	432
8.14	Tenth Ranked Multishift Schedule, MERGE Code Printout. . . . .	433
8.15	Preference Measures for the Multishift Schedule in Figure 8.14, MERGE Code Printout . . . . .	434
8.16	1973 Multishift PR Schedule for the Evidence Technician Unit, St. Louis Metropolitan Police Department, MERGE Code Printout . . . . .	441
8.17	Preference Measures for 1973 ETU Schedule in Figure 8.16, MERGE Code Printout. . . . .	442
8.18	1974 ETU Multishift PR Schedule, MERGE Code Printout. . . . .	446
8.19	Preference Measures for the 1974 ETU Schedule in Figure 8.18, MERGE Code Printout. . . . .	447
8.20	Multishift PR Schedule for the Radar-Vascar Unit, St. Louis Metropolitan Police Department, MERGE Code Printout . . . . .	451
8.21	Preference Measures for the Radar-Vascar Unit Schedule in Figure 8.20, MERGE Code Printout. . .	452
8.22	Multishift PR Schedule for the Motorcycle Unit, St. Louis Metropolitan Police Department, MERGE Code Printout . . . . .	456
8.23	Preferences Measures for the Motorcycle Unit Schedule in Figure 8.22, MERGE Code Printout. . .	457
 Appendices:		
10.1.1	Police Manpower Scheduling Survey Instrument. . .	465
10.2.1	Cyclic Graph with Five Recreation Periods . . . .	479
10.2.2	PR Schedule Based on the Five Recreation Periods in Figure 10.2.1. . . . .	481

LIST OF FIGURES  
(continued)

No.	Page
10.2.3 Second PR Schedule Based on the Five Recreation Periods in Figure 10.2.1 . . . . .	481
10.2.4 Cyclic Graph with Seven Recreation Periods. . . .	484

# OPTIMAL MULTISHIFT PROPORTIONAL ROTATING SCHEDULES

## 1. INTRODUCTION

### 1.1 PROBLEM BACKGROUND

In recent years, increasing attention has been devoted to the development and use of new varieties of manpower schedules. In place of the traditional work pattern of eight hours per day, Monday through Friday, an increasing number of private companies and public agencies are experimenting with four-day work weeks (1,2,3,4,5)\*, three-day work weeks (5,6,7,8), split and part-time shifts (9,10,11,12), variable starting and stopping work hours (13), rotating and cyclic schedules (14,15,16,17,18,19,20,21,22, 23,24,25,26,27), and proportional schedules (9,10,12,14,15, 16,17,18,19,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38, 39,40,41).

Many factors have contributed to increased interest in new kinds of manpower schedules. One of the most important has been the significant increase in the number of persons employed in service-oriented industries where the demand for service, even if predictable, cannot be modified to fit employee work patterns. In the public sector, the most

---

\*The numbers in parentheses in the text indicate references in the Bibliography.

notable examples of services which cannot be backlogged or inventoried are the delivery of emergency police (1,16,17,18, 28,29), fire, and medical services. In the private sector, services which are tied to customer demand include those provided by telephone operators (9,30,31,39,40), taxi drivers, toll booth attendants (10,26,34,42), baggage handlers (27), airline reservationists, bus drivers (11), refuse collectors (19), air line crews (43,44,45), nurses (20,21,22,23,24,35,36, 37,38), and in some cases, retail clerks.

Pressure for new and better manpower schedules has originated from both management and worker groups. With spiraling labor costs, administrators have sought to improve productivity by rearranging work patterns to provide the same level of performance with a reduced number of personnel, and by improving individual service levels. At the same time, workers both in public and private agencies are demanding benefits that are not exclusively tied to economic issues, but rather relate to employee satisfaction and morale. In many instances these demands are for new kinds of manpower schedules which (1) require less time on monotonous jobs, (2) require less scheduled overtime, (3) reduce the amount of commuting, and (4) provide more leisure time for family activities.

The impetus for the development and use of new kinds of manpower schedules is not limited to service industries.

Manufacturing industries which require continuous utilization of valuable machines and equipment often operate six or seven days a week and the kinds of manpower schedules used are often an important labor-management issue; examples include the steel, chemical, and petroleum industries plus many computer installations.

The development of new schedule design methodologies to replace rule-of-thumb procedures has progressed slowly however. This has been due, in part, to the increasing awareness that the development of new methods for designing schedules which can adequately satisfy both labor and management constraints is a complicated task. Barriers to the successful implementation of new kinds of schedules have included (1) a myriad of Federal laws, state regulations, city ordinances, and union agreements which regulate overtime, undertime, starting hours, stopping hours, lunch and break times, split shifts, and the use of full- and part-time workers\*; (2) the resistance of management and

---

\* As an example of increasing Federal involvement in local scheduling practices, the scheduling of municipal employees such as police officers and firemen would have become even more difficult if the United States Supreme Court had upheld the new provisions of the Fair Labor Standards Act adopted by Congress in 1974. A suit challenging the constitutionality of the law was brought before the Court by the International City Managers Association, the National League of Cities, the National Conference of Governors, and 18 states during the 1975-1976 term of the Court. The new provisions were ruled unconstitutional in June 1976. If upheld, the provisions would have required overtime payment to municipal employees who worked more than a specified number of hours per pay period.

employee groups to changes in work patterns which are perceived as detrimental; and (3) the inertia of tradition. In addition, changing work patterns for large numbers of persons may require adjustments to work schedules for support personnel such as cafeteria workers, maintenance staff, and transportation and traffic related workers (such as bus drivers and police personnel). Understandably, both management and employee organizations, aware that new schedules may have economic and social impacts that are not immediately obvious, have tended to take a cautious, wait-and-see attitude when new work patterns are proposed.

Within the last decade, an increasing number of efforts have been made to develop systematic approaches to the design of acceptable manpower schedules. A variety of heuristic and algorithmic procedures have been proposed; a review of these efforts is presented in section 1.5 of this chapter.

This thesis discusses the construction and design of one kind of manpower schedule: proportional rotating schedules (hereafter referred to as PR schedules) which can satisfy many of the scheduling problems cited above. The procedures described in this work have been successfully used to design schedules for the St. Louis Metropolitan Police Department, among other applications in the police area, and much of the discussion in this thesis will be within the context of scheduling police manpower. The basic concepts of schedule design and most of the schedule attributes discussed, however,

are applicable to other service agencies in both the public and private sector.

## 1.2 POLICE MANPOWER SCHEDULING

Scheduling manpower for police services is a multifaceted planning activity which includes measuring the demand for services, determining the distribution of available manpower which best meets the demand, and finally constructing personnel work rosters or schedule sheets (hereafter called schedules) which achieve the desired manning through acceptable patterns of work days and days off for each police officer.

In some areas of police work, the services demanded are not urgent and can be conveniently postponed. For these cases, simple schedules such as the common 8 A.M. to 5 P.M. five-day work week can be used. For emergency service demands, however, manpower scheduling is more difficult. Most such services require an immediate response and requests for such services arrive around the clock, seven days a week. A wide variety of work schedules has been devised by police agencies to properly staff the watches\* of the week.

### 1.2.1 The Demand for Police Service by Day and Shift

Although the demand for police services is a random phenomenon, the pattern of call arrivals exhibits regular

---

\* "Watch" can be used interchangeably with "shift". Watch is the more commonly used expression to describe police manpower schedules, and refers either to a particular set of consecutive hours for each day of the week (e.g., the day, afternoon, and night watches) or to the set of work shifts for the entire week (e.g., "The week consists of 21 watches").

cycles on a daily, weekly, and even seasonal basis. These cycles permit police agencies to compute reasonably accurate forecasts of the number of calls for each day of the week and each shift. Table 1.1 shows the distribution of calls for police service by day and shift of the week in St. Louis during 1971. Almost half of the calls (45.8 percent) occurred during the afternoon shift, and approximately one-third of the calls (33.6 percent) occurred on Friday and Saturday. The most dramatic change in daily workload occurs on the night shift where the daily average of the watch workload is only 11.4 percent for Sunday through Thursday, but increases to an average of 21.5 percent for Friday and Saturday.

Police statistics from other cities show a similar distribution of call activity by day and shift. Table 1.2, taken from a 1972 survey conducted by the author indicates the distribution of workload by shift for 10 United States cities.

#### 1.2.2 Commonly Used Police Schedules

Most police departments use some form of a three-shift schedule. In rural areas or for specialized services such as traffic enforcement, two-shift schedules may be used. Schedules with more than three shifts are used by some agencies to achieve a distribution of manpower more closely resembling the actual demand for service.

The type of service required as well as the volume of requests vary with the time of day. Late afternoon and night

Table 1.1

Percentage Distribution of Calls-For-Service by Shift and  
Day of the Week, St. Louis Metropolitan Police  
Department, 1971\*

Day	Shift			Daily
	Day (7 A.M.- 3 P.M.)	Afternoon (3 P.M.- 11 P.M.)	Night (11 P.M.- 7 A.M.)	
Monday	14.2%	14.3%	11.7%	13.6%
Tuesday	14.0	14.2	11.5	13.5
Wednesday	14.0	14.1	11.5	13.4
Thursday	14.8	14.8	13.1	14.4
Friday	14.5	15.4	21.8	16.7
Saturday	15.6	15.4	21.3	16.9
Sunday	12.9	11.8	9.1	11.5
Total	100.0%	100.0%	100.0%	100.0%
Percentage by shift	29.6	45.8	24.6	100.0

Source: Computer Division, St. Louis Metropolitan  
Police Department, St. Louis, Missouri.

\*Based on 439,716 radio calls (both directed incident  
and directed assist calls) from January 1, 1971 through  
December 31, 1971.

Table 1.2

Percentage Distribution of Police Workload by Shift for  
Ten United States Cities

City	Shift*		
	Day	Afternoon	Night
Norman, Oklahoma	25%	50%	25%
Rockville, Maryland	38	46	16
Fort Lauderdale, Florida	30	50	20
Bethlehem, Pennsylvania	39	37	24
Kalamazoo, Michigan	25	60	15
Ocean City, Maryland	21	46	33
Waterbury, Connecticut	25	35	40
Kansas City, Missouri	33	39	28
Millersville, Maryland	35	45	20
Hartford, Connecticut	35	50	15
Unweighted Average	30.6%	45.8%	23.6%

Note: Survey conducted by the author of police officials attending a planning seminar at Northwestern University in June 1972 (see appendix 10.2).

\*Starting hours for the three shifts differ slightly among the departments. Generally, the day shift begins between 6 A.M. and 8 A.M., the afternoon shift begins between 2 P.M. and 4 P.M., and the night shift begins between 10 P.M. and midnight.

assignments are considered by many police officers to be less desirable for work assignments since they interfere with normal social and family life. Police administrators differ as to whether these circumstances imply a need for officers to rotate periodically through the shifts or to be permanently assigned to a particular shift.

Shift rotating or multishift schedules are the most common type of police schedule in use today\*. Officers generally spend an equal amount of time, called a "tour", on each shift, although the exact length of the time varies considerably from department to department -- from one week to several months. Since changing shifts requires adjustment of personal habits for each officer and his family, frequent changes are usually considered undesirable. Extended periods on the same shift, however, are also considered undesirable, particularly during the summer months when crime and calls for service increase. Hence the length of time spent on each shift usually represents a compromise between these two unattractive features.

A schedule which is manned solely by tours of equal length on each shift will produce equal manning levels on each shift, a distinct disadvantage for most departments. Despite this, such schedules are widely used because of their simplicity and the lack of readily available and acceptable alternatives.

---

\* Sixteen of the 21 departments surveyed by the author indicated that officers rotate through the shifts on a regular basis (see appendix 10.1).

The workload imbalance between different shifts and days of the week has motivated some departments to use permanent shift assignments for all officers, with the number of officers assigned to each shift proportional to the workload. Shift assignments are made by seniority in some cases and by officer choice in others; officers change shifts only by promotion, assignment to a special detail on another shift, or by special request (e.g., to accommodate part-time college programs). Some departments employ both fixed and rotating assignments: for example, San Francisco and Boston both reported in 1968 that they used fixed assignments for the day watch and rotating assignments for the afternoon and night watches (46)\*.

The match between manpower and workload may be improved by scheduling overlapping shifts so that officers assigned to different shifts are on-duty at the same time during the busier hours of the day. Two popular versions of this type of schedule are the four-shift or overlay schedule, and the "4-10" plan. The four-shift schedule employs the usual three shifts plus a fourth shift which usually begins midway through the afternoon shift and ends halfway through the night shift. This plan provides extra manpower during the peak demand period of each day (usually from 7 P.M. to 3 A.M.). In the

---

\* Seven of the 16 departments surveyed by the author that reported use of rotating shift assignments also indicated that some officers are permanently assigned to specific shifts.

4-10 plan, officers work four 10-hour days each week instead of five 8-hour days. This plan produces a total overlap time of six hours per day which can be placed at the peak demand period at the expense of manpower reductions during other hours (i.e., since the total number of manhours to be allocated per week is fixed, longer shift hours result in fewer men on duty per shift) (1,2). Both four-shift and 4-10 schedules can be based on either fixed or rotating shift assignments or on a combination of both.

Many systems are used to schedule officers' days off.\* In a few departments, officers are permanently assigned certain days of the week as recreation days, but in most departments a revolving or cyclic pattern of recreation and work days is used. Some departments schedule fewer men off duty on busier days, but since this complicates the scheduling process and also usually reduces the number of weekends off, many departments settle for equal manning levels for each day of the week.

#### 1.2.3 Desirable Police Schedule Features

In addition to the sometimes felt need for providing police manning levels that are proportional to the demand for service by shift and day of the week, police manpower schedules must also often satisfy a variety of statutory

---

\* The term "days-off" will be used interchangeably with the expression "recreation days," the latter being more commonly used to describe police schedules. Consecutive recreation days are defined as recreation periods.

and administrative constraints. Also, there are a number of schedule attributes that, although not required, are usually considered desirable by police administrators.

These attributes include:

- (1) identical (or equivalent) schedules for all officers,
- (2) shift tours of acceptable length,
- (3) the scheduling of a recreation period whenever an officer rotates to a new shift assignment,
- (4) acceptable work and recreation period lengths (i.e., elimination of very short or long periods),
- (5) the presence of an adequate number of weekend recreation periods distributed evenly over the schedule to minimize the number of consecutive working weekends\*,
- (6) the ability to schedule holiday recreation days at the beginning or end of regularly scheduled recreation periods (police officers routinely receive time off for holidays on days other than the actual holiday date),
- (7) the design of vacation schedules which maximize officer choice and minimize manning level disruptions, and
- (8) the design of supervisory schedules which maximize the "unity of command" between each officer and his supervisor.

In most police departments today, police schedule designers have no systematic way of incorporating these

---

\* A weekend recreation period is a recreation period that includes both Saturday and Sunday. A working weekend is a weekend in which an officer must work either Saturday or Sunday or both days.

features into the schedules that are being used. Further, research efforts to date into the design and construction of new varieties of manpower schedules have frequently ignored or have been unable to include them. The schedule design procedures described in this thesis represent, in some cases, the first systematic inclusion of the attributes listed above into an algorithmic approach to schedule design.

### 1.3 TYPES OF MANPOWER SCHEDULES

Two broad categories of schedule design problems exist: shift scheduling and days-off scheduling.

#### 1.3.1 Shift Scheduling

In shift scheduling problems, service demands are determined for small units of time (usually one hour periods) and then used to find the best shift times for each day of the week given the requirement that each shift must consist of a specific number of consecutive time units (e.g., eight hours). This type of scheduling is often posed as an allocation problem in which the objective is to determine, by specifying the shift hours, the minimum number of officers required to provide adequate manning levels for each hour of the day or week. Among the variables that may be specified in a shift scheduling solution are shift starting times, shift length, lunch times, break periods, and the number of men for each shift. Many variations of this problem exist and a variety of analytical and heuristic techniques have been

used to find adequate shift designs. (See section 1.5 for a review of previous research efforts.) The important fact to be noted about shift scheduling problems is that they deal only with the allocation of the entire workforce rather than with the scheduling of individual officers; the solutions to these problems specify when and how many officers should be on-duty, but do not indicate the day-to-day work patterns of individual officers. The resolution of this latter problem is called days-off scheduling.

### 1.3.2 Days-Off Scheduling

Days-off scheduling is the primary topic of this thesis. To describe this type of scheduling problem, it is useful to define some basic elements of manpower scheduling. A schedule bracket is a sequence or pattern of work and recreation days; the length of a bracket is equal to the total number of days in the sequence. A group of men is a subset of the workforce defined by the fact that all of the men within the group work the same bracket (or pattern of work and recreation days). A group of men may contain only one man. All of the schedules derived in this thesis are based on schedule brackets that are one week long.\* The common Monday through Friday work schedule, if repeated every week, is a seven-day one-bracket schedule that can be represented as:

---

\* This thesis also uses the convention that each seven-day schedule bracket begins on Monday.

MON.	TUE.	WED.	THUR.	FRI.	SAT.	SUN.
					R	R

where each "R" represents an off-duty or recreation day and each blank represents a work day.

Schedules containing two or more brackets can be used to provide manning on all seven days of the week. For example, a uniform manning level for each day of the week can be achieved with the seven-bracket schedule shown in figure 1.1. Each bracket represents a different pattern of work and recreation days. This schedule is used by assigning a group of men to each schedule bracket. Five groups of men are on duty each day of the week (indicated by the five blanks or on-duty days in each column of the schedule

	M	T	W	T	F	S	S
1	R	R					
2		R	R				
3			R	R			
4				R	R		
5					R	R	
6						R	R
7	R						R

Figure 1.1

Sample Seven-Bracket Schedule

chart). As a result, if an equal number of men is assigned to each group, an equal number of men will be on duty each day of the week.

Multibracket schedules can be used in two ways: either each group of men can be permanently assigned to one schedule bracket (i.e., each group works the same seven-day pattern every week), or the groups of men can be rotated through all of the brackets. Each of these procedures is discussed below.

#### 1.3.2.1 Fixed Schedules

A manpower schedule is said to be used as a fixed schedule if each group of men work the same schedule bracket every week: for example, if the schedule in figure 1.1 is used as a fixed schedule, the men in bracket 3 receive Wednesday and Thursday off every week, while the men in bracket 6 receive Saturday and Sunday off each week. By placing different numbers of men in each bracket, fixed schedules can approximate variations in daily demand: for example, if two men are assigned to each bracket in figure 1.1, then 10 of the 14 men assigned to the schedule will be on-duty each day of the week (2 men for each of the five brackets), but if these 14 men are reassigned so that five men work bracket 7, two men work brackets 1, 2, and 3 each, and only one man works brackets 4, 5, and 6 each, the number of men on-duty each day of the week becomes:

Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.
7	10	10	11	12	12	8

This daily manpower distribution may be useful if the service demands are greater on Friday and Saturday, and lighter on Sunday and Monday. Baker (12,25,33,41), Tibrewala et al. (32), Guha (34), and Guha and Browne (26) have all developed algorithms for determining the number of men to assign to each bracket in order to meet daily manning requirements.

The advantages of fixed schedules are their relative simplicity, and their ability to approximate variable daily service demands. The most serious drawback, however, is the inequity of the days off schedules for individual officers; in figure 1.1, for example, men assigned to brackets 1, 2, 3, or 4 never receive a recreation day on either Saturday or Sunday while men assigned to bracket 6 receive both of these days off every week.

#### 1.3.2.2 Cyclic Schedules

A manpower schedule is said to be used as a cyclic schedule if each group of men works each schedule bracket in sequence. This can be illustrated using the schedule in figure 1.1 as a cyclic schedule: if an officer works bracket 1 during the first week of the schedule, instead of returning to Monday of the same bracket, he rotates to the Monday of schedule bracket 2 for the second week, rotates to the Monday of schedule bracket 3 for the third

week, and so on; after working bracket 7 in the seventh week, the officer rotates back to bracket 1 for the eighth week. More precisely, a schedule is defined to be cyclic if in the  $j^{\text{th}}$  week the  $i^{\text{th}}$  group of men (identified as the group of men who worked bracket  $i$  during week 1) work the same schedule as did the  $i+j-1$ st group of men (modulo the number of brackets) during week 1 (14).

In a cyclic schedule with  $m$  brackets, each group of officers spends one week on each bracket during each  $m$ -week period. The number of weeks required for one complete cycle through all of the brackets is defined as the schedule period (e.g., the schedule in figure 1.1 has a seven-week period). Cyclic schedules have the advantage that during each period of the schedule, every officer works the same pattern of work and recreation days. As an example, if the schedule in figure 1.1 is used as a cyclic schedule, each officer receives one weekend off during each seven week period. Cyclic schedules are usually more complicated than fixed schedules, and as a result, are more difficult to design.

When a cyclic schedule is used, equal (or nearly equal) numbers of men are usually assigned to each group. As a result, each schedule bracket has the same number of men assigned to it each week; this causes the number of men off-duty each day of the week to remain constant during

each week of the schedule. If unequal numbers of men are assigned to each group, the number of men off duty each day changes from week to week. For cyclic schedules with equally manned groups, the proportion of total manpower off-duty any day of the week is indicated by the fraction of brackets that contain recreation days on that day of the week. For each day of the week in figure 1.1, two of the seven brackets contain recreation days, indicating that two-sevenths of the workforce will be off duty each day\*. Unless otherwise indicated, it is assumed throughout the remainder of this thesis that equal numbers of men are assigned to each group.

#### 1.3.2.3 Multishift Schedules

Shift rotating or multishift schedules can be constructed by combining cyclic schedules designed for each shift. As an example, consider the multishift schedule in figure 1.2. The schedule consists of 13 brackets, seven on shift A and six on shift B. Since the multishift schedule is used a cyclic schedule, as each group of men completes one week on each bracket, it rotates down to the next bracket and the group completing the last bracket rotates back to bracket 1. Two groups of men change shifts each week: the men completing bracket 7 (on shift A)

---

\* If the number of men assigned to each group varies, the fraction of men on recreation each day of the week will average, over the schedule period, the fraction obtained with equally manned groups, but will vary from week to week -- usually an undesirable schedule feature.

	M	T	W	T	F	S	S
Shift A	1	R	R				
	2		R	R			
	3			R	R		
	4				R	R	
	5					R	R
	6						R
	7	R					R
Shift B	8	R	R				
	9		R	R	R		
	10				R	R	
	11					R	R
	12						R
	13						R

Figure 1.2

Two-Shift Multishift Schedule

rotate to bracket 8 (on shift B), and the men completing bracket 13 (on shift B) rotate to bracket 1 (on shift A). The period of the multishift schedule is 13 weeks; and each officer spends seven weeks on shift A and six weeks on shift B during each 13-week period.

Since one group of men rotates to and one group of men rotates from each shift during each week of the schedule, the number of groups assigned to schedule brackets within each shift remains constant (e.g., in figure 1.2, during each week seven groups are assigned to shift A and six groups are assigned to shift B). In general, the fraction of the workforce (assuming equally manned groups) assigned to schedule brackets within a shift equals the fraction of schedule brackets for the shift within the entire schedule; hence, in figure 1.2,  $7/13$  of the total workforce is assigned to shift A and  $6/13$  of the workforce is assigned to shift B.

The fraction of manpower on duty for each shift each day of the week equals the fraction of all groups used in the schedule that are on duty within a shift each day (e.g.,  $5/13$  of the workforce is on duty on shift A and  $4/13$  of the workforce is on duty on shift B each Tuesday). In general, as the fraction of the workforce assigned to (i.e., the fraction of schedule brackets within) a shift increases, the fraction of manpower on duty on that shift each day of the week also increases.

The multishift schedule in figure 1.2 can be viewed as the combination of two cyclic schedules: schedule A consisting of the seven brackets for shift A, and schedule B consisting of the six brackets for shift B. Each can be used as a cyclic schedule by itself: at the end of each week in schedule A the men in bracket 7 cycle back to bracket 1, and at the end of each week in schedule B the men in bracket 13 cycle back to bracket 8. Schedule A has a seven-week period and schedule B has a six-week period. When used together in a multishift schedule, however, schedules A and B are not used as cyclic schedules. Rather, they are said to be used as non-cyclic schedules to denote the fact that although groups of men rotate through each schedule bracket, they do not cycle within each shift (i.e., at the end of each week in the multishift schedule, the men in bracket 7 rotate to bracket 8 instead of bracket 1, and the men in bracket 13 rotate to bracket 1 instead of bracket 8).

A basic constraint imposed on all of the schedules derived in this thesis is the requirement that all on-duty days in a schedule bracket represent the same shift. This requirement, combined with the condition that all schedule brackets are seven days long and begin on Monday, has two effects:

- (1) every schedule derived in this thesis, whether fixed, cyclic, non-cyclic, or multishift consists of an integral number of weeks, and

- (2) every shift changeover point in multishift schedules used in this thesis occurs after the last Sunday of each non-cyclic schedule.

Some research on the design of multishift schedules in which shift tours are not constrained to an integral number of weeks has been reported by Bodin (15), Ignall et al. (19), Smith (23), Baker (25), and Guha and Browne (26).

#### 1.4 OBJECTIVES AND OUTLINE OF THE THESIS

##### 1.4.1 Proportional Rotating Schedules

This thesis concerns the design of manpower schedules which may possess any or all of the following basic properties:

- (1) manning levels that are proportional to service demands by day of the week;
- (2) manning levels that are proportional to service demands by shift; and
- (3) equitable schedules for all officers.

These properties can be achieved with rotating multishift schedules (described in section 1.3). Equality of schedules for individual officers is achieved by using cyclic schedules; proportional manning by day of the week is achieved by arrangement of the recreation days; and proportional manning by shift is accomplished by assigning a greater number of brackets to busier shifts. Such schedules are called proportional rotating schedules by Heller (16), and this terminology will be followed in this thesis\*. When

---

\*As noted earlier, the abbreviation "PR schedules" is also frequently used.

discussing one-shift schedules, a PR schedule is a cyclic schedule which distributes manpower according to daily service demand, and has a schedule period equal to the number of brackets in the schedule. Multishift PR schedules provide manning levels proportional to the service demands by shift. The period of a multishift PR schedule is equal to the sum of the lengths (i.e., number of brackets) of all shift schedules included.

To illustrate these concepts, consider the task of designing a PR schedule for 19 police officers which matches the manpower allocation shown in table 1.3. This allocation provides for 92 on-duty manshifts per week. A PR schedule to match these manning levels is shown in figure 1.3. One man is assigned to each group, and shift rotation is used, resulting in a schedule period of 19 weeks. To achieve proportionality by shift, six brackets have been used for the night and day shifts each, and seven brackets have been used for the afternoon shift. As a result, during the 19 week period each officer spends six weeks on the day shift, six weeks on the night shift, and seven weeks on the afternoon shift. Three men change shifts at the end of each week. Note that the total number of officers assigned to each shift remains constant each week.

More difficult to detect in this schedule is the fact that the numbers of officers on-duty each day of the week on each shift match the manning level requirements shown in table 1.3. As an example, in the day shift schedule,

Table 1.3

Required On-Duty Manpower Levels By Shift and Day of the Week

Shift	Start Hour	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Total (%)
Day	7 A.M.	5	4	4	4	5	4	4	30 (32.4)
Afternoon	3 P.M.	5	5	5	5	5	5	4	34 (41.1)
Night	11 P.M.	4	4	4	4	4	4	4	28 (26.5)
Total	—	14	13	13	13	14	13	12	92 (100.0)

Note: This manning level distribution was used to schedule 19 officers in the Evidence Technician Unit of the St. Louis Metropolitan Police Department in 1973 (see section 8.3.1).

Shift		M	T	W	T	F	S	S
Night	1	R	R	R	R			
	2					R	R	R
	3	R						
	4		R	R				
	5				R	R		
	6						R	R
Afternoon	7							R
	8	R	R					
	9			R	R			
	10					R	R	R
	11	R						
	12		R	R	R			
Day	13					R	R	R
	14	R						
	15		R	R	R			
	16						R	R
	17							
	18		R	R	R			
	19					R	R	R

Figure 1.3

Nineteen-Week Multishift PR Schedule

five on-duty brackets are used on Monday and Friday and four on-duty brackets are used for each of the remaining days of the week. The work periods in this schedule vary in length from six to eight days, and the recreation periods vary in length from two to four days, except for one seven-day period that begins in week 19 and ends in week 1\*.

The PR schedule of figure 1.3 is not the only multishift schedule that satisfies the manning requirements. A second PR schedule that also matches the manpower allocation is shown in figure 1.4; each shift schedule in this alternate schedule is different from the corresponding shift schedule shown in figure 1.3. (The seven-day mini-vacation is now in weeks 6 and 7.) Still another PR schedule that satisfies the manning requirements is shown in figure 1.5. This PR schedule differs from the first two shown in that each shift tour has been divided into two subtours. As a result, the maximum number of consecutive weeks worked on any shift is only four weeks.

#### 1.4.2 Thesis Objectives

Despite the ability of PR schedules to provide equitable manpower schedules that also preserve proportional manning levels, their adoption and use by police departments or other public and private agencies to date has been minimal.

---

\* This long recreation period was designed into the schedule to provide one "mini-vacation" every 19 weeks.

Shift		M	T	W	T	F	S	S
Night	1	R						
	2		R	R	R			
	3					R	R	R
	4	R						
	5		R	R				
	6				R	R	R	R
Afternoon	7	R	R	R				
	8				R	R		
	9						R	R
	10							R
	11	R	R	R				
	12				R	R	R	R
	13							
Day	14	R	R					
	15			R	R			
	16						R	R
	17							
	18		R	R	R			
	19					R	R	R

Figure 1.4

Nineteen-Week Multishift PR Schedule

Shift		M	T	W	T	F	S	S
Night A	1	R	R	R				
	2				R	R		
	3						R	R
Afternoon A	4							R
	5	R						
	6		R	R				
	7				R	R	R	R
Day A	8							
	9		R	R	R			
	10						R	R
Night B	11	R	R					
	12			R	R			
	13					R	R	R
Afternoon B	14	R						
	15		R	R	R			
	16					R	R	R
Day B	17	R						
	18		R	R				
	19				R	R	R	R

Figure 1.5

Nineteen-Week, Multitour, Multishift PR Schedule

The technical deficiencies that have contributed to a lack of experimentation with PR schedules by police administrators have included the following:

- (1) No algorithmic procedures have existed for constructing optimal PR schedules. (Heller's earlier work (16) defines the elements of the construction process, but does not describe a procedure for designing optimal schedules.)
- (2) The number of solutions (schedules) that may exist for a given manning distribution and set of constraints is not easily determined -- there may be none, or hundreds of alternatives to select from.
- (3) Very little work has been done to identify and systematize the attributes and features that make one schedule preferable to another. As a result, no procedures have existed to focus schedule design efforts on the most preferable schedules when large numbers of alternative solutions are available.

These deficiencies were the problems addressed in this work. Specifically, the objectives of this thesis were:

- (1) the development of a systematic procedure for the design of multishift PR schedules which match a specified distribution of manpower by shift and day of the week;
- (2) the identification of preferred schedule attributes and specific quantitative measures for each that can be used to discriminate between alternate PR schedules satisfying all given constraints; and
- (3) the incorporation of the schedule design procedures into a set of computer programs capable of finding "optimal" (most preferred) PR schedules based on the required manning levels, and a given set of required and preferred schedule attributes.

#### 1.4.3 Basic Assumptions and Conditions

This section identifies the major assumptions and conditions that are used throughout this thesis. (Some of these conditions have been noted above, but are identified again for completeness.)

With regard to the design of PR schedules, it is assumed in this thesis that:

- (1) all officers work under the same set of work rules (i.e., the rules governing hours worked per week, number of recreation days, etc.);
- (2) each officer can perform all required duties (i.e., the men are interchangeable); and
- (3) the total number of men included in the schedule remains constant.

In addition, it is assumed that the following information is available to the schedule designer:

- (1) the total number of men to be assigned to each shift; and
- (2) the required manning levels for each shift by day of the week.

Finally, the following conditions apply to every PR schedule designed with the procedures described in this thesis:

- (1) each schedule consists of a sequence of seven-day brackets with an equal number of men assigned to each;
- (2) each schedule consists of an integral number of brackets (weeks);
- (3) in multishift schedules, a shift changeover occurs after the last Sunday in each non-cyclic schedule;

- (4) the "optimal" schedule is the PR schedule which is judged to be "most preferable" based on its attributes;
- (5) the schedules can be designed to automatically include paid holidays as additional recreation days; and
- (6) any accounting for random losses of manpower due to illness, special assignments, compensatory time-off, etc., is accomplished through adjustment of the manning requirements (e.g., keeping them high enough so that operations can continue adequately when losses occur).

#### 1.4.4 Thesis Outline

The thesis is divided into eight chapters. The remainder of this one reviews previous research in manpower scheduling. Chapter 2 identifies schedule attributes that can be used to distinguish alternative PR schedules, specifies operational preference measures which assess these attributes, and discusses the use of a multiple criteria decision model for determining the optimal schedule.

Chapter 3 outlines the methodology for constructing PR schedules. The measures identified in chapter 2 are used in constraints which screen out unacceptable schedules, and in preference criteria to identify the most desirable schedule.

Chapters 4 through 7 describe enumeration algorithms used in the sequential design process to construct PR schedules. An algorithm used to cluster recreation days into periods of acceptable length is described in chapter 4. A procedure for enumerating all unique distributions of the

recreation periods over the days of the week is described in chapter 5; and a branch-and-bound procedure for enumerating optimal one-shift cyclic schedules is presented in chapter 6. The design and construction of optimal and near-optimal multishift schedules is described in chapter 7.

Several PR schedules, designed using computer programs based on the algorithms described in this thesis, are illustrated in chapter 8. The illustrative schedules shown were developed for the St. Louis Metropolitan Police Department. Appendix 10.1 contains detailed information relating to a survey of 21 police agencies conducted in the summer of 1972. Included in the appendix are the survey instrument, a list of participating departments, and a summary of the survey results. A comprehensive list of previous research in manpower scheduling is presented in the Bibliography.

## 1.5 PREVIOUS RESEARCH ON MANPOWER SCHEDULING

### 1.5.1 Introduction

Manpower scheduling problems represent one segment of a large body of literature that deals with problems of scheduling events, vehicles, and men. The methodologies used to attack these problems relate to their integrality constraints: combinatorics, integer programming, dynamic programming, network theory, and implicit enumeration. Grouped under the general heading of scheduling one can find such diverse problems as: the proper sequencing of events, the identification of specific routes for vehicles,

the allocation of equipment and manpower, and the design of work schedules for individual employees. Although this thesis deals only with manpower scheduling (specifically "days-off" scheduling), it is useful to review briefly the kinds of problems included in and the characteristics of event and vehicle scheduling. This review, in turn, will serve as an introduction to the features and difficulties that characterize manpower scheduling.

#### 1.5.2 Event Scheduling

Event scheduling problems are characterized by a specific task or set of tasks which either must be accomplished once, or repeated continuously. For problems with multiple tasks such as project scheduling and job shop problems, the solutions (schedules) indicate the proper sequence in which each event should occur or each task should be performed; the timing or scheduling of the events is determined by the sequential relationship of one event to another. The objective of such problems usually is to minimize either the total time (or cost) required to complete all tasks, or the total delay time from a given due date to completion of the entire project or of each task (47).

For event scheduling problems with a repeating or continuing task, the critical variable to be determined is usually the proper time for performing the task relative to a predictable demand; examples of this type of scheduling include inventory control, and equipment maintenance problems.

For inventory control problems, the task to be performed is the reorder decision (the unknown is when to reorder), and the demand is the rate of inventory depletion. (See Hadley and Whitin (48), and Hillier and Lieberman (49).) In equipment maintenance problems, the task to be scheduled is the replacement or maintenance of a specific piece of equipment, and the demand is the deterioration rate of the item (see Sasiemi, Yaspan, Friedman (50)). For both problems, the objective is to determine an optimal balance between the cost of performing the task and the cost associated with unmet demand if the inventory is depleted or equipment fails. In this class of problems, the schedule timing is determined by the relationship between each occurrence of the task and the state of the inventory (or equipment).

In summary, event scheduling problems are characterized by relatively simple objective functions that usually depend on well-defined quantifiable variables (e.g., project time, delay time, cost, etc.), and by relatively few constraints on the solution variables. As a result, feasible solutions are usually easily found, often by observation or with simple manual techniques. The methodological difficulties associated with event scheduling problems arise when the optimal solution is sought from among the feasible solutions.

These general concepts can be illustrated by a classic constrained optimization problem not usually considered as a scheduling problem: the travelling salesman problem.

The salesman's tour through each city can be interpreted as a schedule of events or tasks in which the object is to minimize the total travel time required for the trip; the schedule is defined by the sequence of cities visited. Finding a feasible solution to a travelling salesman problem is trivial, determining the optimal solution obviously is not.

### 1.5.3 Vehicle Scheduling

Vehicle scheduling problems are an extension of repeating event problems with additional constraints imposed primarily by the characteristics of a service vehicle. Typical problems in this category of scheduling research include the determination of "schedules" for city buses, refuse collection trucks (51), street sweepers (52,53), airlines (54), and school buses (55). Common to all of these applications is the need to determine a set of routes for the vehicles which satisfy both the demand for service and the constraints imposed by the quantity and performance characteristics of the vehicles. The objective functions and constraint sets associated with vehicle scheduling problems are frequently more difficult to formulate than those for event scheduling problems because of the existence of multiple goals which are often conflicting and difficult to quantify.

As an example, the design of a satisfactory school bus schedule depends on the number of children to be transported, the location of home and school for each child, the school schedule for each child, the street patterns within the

area of interest (e.g., the existence of one-way streets, dead-end streets, etc.), the capacity of each bus, and the average bus travel speed. This information must be used to determine bus schedules (routes and timetables) in response to a variety of objectives which include: limiting the maximum travel time for each child, minimizing the number of busses required, and maximizing the utilization of each bus (55).

Vehicle scheduling problems are usually solved in response to daily or weekly service demand cycles. The length of the demand cycle determines the number of distinct schedules that must be found. If the basic unit of time for each schedule is one day and the demand cycle repeats daily, then only one schedule is needed since it can be used repetitively. If the demand cycle is weekly, however, a different schedule may be needed for each day of the week. A critical point, in relation to manpower scheduling, is that the vehicles are usually not constrained by "work rules" which limit the number of hours they may be used each day or the number of days they may be used each week. Sometimes such constraints are imposed by maintenance requirements. The usual absence of such rules simplifies the vehicle scheduling problem (as compared to manpower scheduling) in two ways:

- (1) the demand for service can be separated into small time blocks (e.g., a day) and a schedule can be obtained for each block without regard to demand for service or schedules for other time blocks; and
- (2) once schedules are obtained for each time block, they can be joined together to form a vehicle schedule for the entire demand cycle; this is possible because vehicle shift assignments may be changed as needed and few limits are placed on the number of consecutive days the vehicle is in service.

Employee work rules cause manpower scheduling to be significantly more complex than vehicle scheduling. These rules relate mainly to the kinds of work and recreation patterns which may be assigned to individual workers (i.e., statutory and physical restrictions on the number of hours worked per day, the number of consecutive days worked, the frequency of shift changes, etc.). Previous research on this type of scheduling is discussed in the following section.

#### 1.5.4 Manpower Scheduling

Procedures for designing proportional manpower schedules which indicate the work and recreation patterns for individual officers may be analyzed as consisting of four steps:

- (1) analysis of the demand for service for time units that are equal to or shorter than the length of the work shifts to be used;
- (2) analysis of the constraints on the level of manpower that must be available for each time unit;
- (3) aggregation of individual time units into shifts or work patterns, and allocation of available manpower to the shifts based on manning requirements for each time unit; and

- (4) design of days-off schedules consistent with the manpower allocation by shift and day of the week.

#### 1.5.4.1 Measuring the Demand for Service and Setting Minimum Manning Requirements

This section is concerned with the first two steps identified above. The discussion is limited to service-on-demand environments in which neither the demand for or the availability of service can be inventoried without incurring serious costs such as loss of revenue, property, or life (e.g., the situations experienced by toll collectors, firemen, and police officers).

An analysis of the demand for service for scheduling purposes requires (1) a measure of the demand; (2) a historical record of demand sufficiently detailed to permit reliable forecasts of future demand cycles, and (3) a procedure for determining the manning level required for each time unit in order to satisfy the demand.

Measurement of the demand for service and its translation into manning requirements has been investigated for a variety of applications. Edie (42) analyzed the flow of vehicular traffic through toll stations operated by the New York Port Authority. Using data on the numbers of autos serviced in 15 minute intervals and on the numbers of autos waiting for service every 15 minutes, he constructed a queuing model of delay time (the time between arrival and service) based on the traffic rate and the number of toll

collectors available. He then used the delay model to compute the number of toll collectors required for a given traffic volume, to minimize the fraction of arrivals that would be delayed more than 11 seconds.

Ignall, Kolesar and Walker (19) studied the demand for refuse collection service in New York City. The amount of refuse to be collected follows a weekly cycle with more refuse appearing on Monday and Tuesday than on the other days of the week. The demand measure was then used to schedule refuse trucks and collectors.

In designing schedules for police manpower, the appropriate measure of service demand depends upon the type of police unit. Butterworth and Howard (28) used the number of reported accidents in the Monterey area of California to allocate units of the California State Highway Patrol. Heller (17) used the same measure to design schedules for the Traffic Safety Unit of the St. Louis Metropolitan Police Department. In the same report, Heller used calls for evidence service to schedule officers in the Evidence Technician Unit. Heller (56) also suggested that the seriousness of called-for-service incidents might be used to measure the demand for patrol manpower. Kolesar et al. (29) used hourly calls for police service as input to a queueing model of New York City police operations which treats each patrol unit as a server. The model was used to estimate the number of patrol units needed each hour to

insure that no more than a specified fraction of the calls would be queued.

Discussions of the design of cyclic schedules for nurses are given by Maier-Rothe and Wolfe (24), and Warner (37,38). Warner presents a mixed-integer quadratic programming model for allocating nursing care by ward, shift, and skill category. The quality of the allocation is measured by how well it matches the demand for service, which is estimated using a Markovian model based on the number of patients (classified according to their nursing needs) and the amount of time each patient class requires from each type of nurse.

#### 1.5.4.2 Shift Scheduling

Once the desired manpower levels for each time period have been determined it is necessary to identify (1) specific work patterns or shifts to be used and (2) the number of men to be assigned to each (step 3 in the four step procedure for designing proportional manpower schedules). The term work pattern or shift is used to describe a collection of consecutive time periods worked by the same person (e.g., 8 A.M. to 4 P.M. is a work pattern consisting of eight one-hour time periods; if no constraints are placed on start hours, there are 24 different eight-hour work patterns in a day). Work rules may govern shift starting times, shift lengths, the use of lunch and other relief periods within each shift, and the use of split shifts.

Additional rules may apply to part-time workers. The objective of the shift scheduling problem is to find the set of shifts which satisfies the manning and work rule requirements with the minimum number of men. An alternate form of the problem deals with a fixed number of men and seeks the set of shifts which best matches the allocated manpower and the desired manning levels.

Mathematically, shift scheduling problems can be expressed as one of two forms of the set covering problem. The first was suggested by Dantzig (57) in connection with Edie's analysis of toll station traffic patterns (42):

$$\begin{aligned} \text{Pl:} \quad & \min_x Z = c'x \\ & \text{subject to } Ax \geq r \\ & \quad x \geq 0 \\ & \quad \text{all } x_j, j = 1, 2, \dots, s \text{ integer} \end{aligned}$$

where:

$A = t \times s$  matrix, where rows correspond to time units, and columns correspond to distinct work patterns; if  $a_{ij} = 1$ , shift  $j$  covers time unit  $i$ ; if  $a_{ij} = 0$ , shift  $j$  does not cover time unit  $i$ .

$r = t \times 1$  vector, where component  $r_i$  equals the minimal manning requirement (i.e., number of men) for time unit  $i$ .

$x = s \times 1$  vector, where component  $x_j$  equals the number of men assigned to shift  $j$ .

$c = s \times 1$  vector, where  $c_j$  equals the cost of assigning one man to shift  $j$ .

The total number of men on duty during time unit  $i$ ,  $w_i$ , is given by

$$w_i = A_i x$$

where  $A_i$  is the  $i^{\text{th}}$  row of the  $A$  matrix. The value of  $r$  is obtained from the analysis of the service demand for each time unit. If  $c$  is set equal to 1 (the unity vector) the objective reduces to a minimization of the total number of men required to satisfy the constraints.

The second form of the shift scheduling problem is used when the total number of men ( $M$ ) to be allocated is known,

$$\text{P2:} \quad \min_x z = f(r, w)$$

$$\text{subject to: } Ax = w$$

$$1x = M$$

$$x \geq 0$$

$$\text{all } x_j, j = 1, 2, \dots, s \text{ integer}$$

where  $f(r, w)$  = objective function based on the required manning ( $r$ ) and allocated manpower ( $w$ ) for each time unit.

$w = t \times 1$  vector, where component  $w_i$  equals the total number of men on duty during the  $i^{\text{th}}$  time unit.

$A, r, x$  = defined above for P1.

The difficulties encountered in finding optimal solutions to either P1 or P2 arise from the integrality of the  $x_j$  values and the size of the A matrix. The number of columns in A (and hence the number of components in  $x$ ) can become enormous as more work rules are considered and more distinct shift patterns are created. To illustrate, Guha and Browne (26) describe the following case for toll collectors for the New York Port Authority:

If employees can start work at any ten minute period from 6 A.M. to Midnight ... this would mean 6 start times for each of 18 hours, 7 days per week or 756 possible starting times.

For each starting time, there are a variety of possible work patterns. A personal break 1 to 3 hours after the start would yield 12 possibilities and a meal break starting in a two hour band, e.g., 4th or 5th hour, another 12 possibilities for over 100,000 variables ( $756 \times 12 \times 12$ ) ignoring a possible second short break that might be required in some cases.

The number of variables (or columns in the A matrix) can become even larger if split shifts or part-time workers are used. Consequently, direct analytical approaches based on current integer programming codes are not feasible. To overcome these problems, a variety of heuristic and analytic procedures have been attempted; several of these are described below.

Kolesar et al. (29) analyzed the shift assignments of police patrol units by solving problem P1 with a reduced A matrix: shifts were constrained to begin on the hour and starting times for lunch were limited to four points during each shift. By utilizing the special structure of the resulting constraint matrix, they were able to solve P1 as a mixed-integer problem in which only a small number of the variables were forced to be integers. Using data on hourly calls for service for one day from a New York City police precinct, they first determined the minimum number of units required to meet hourly manning levels  $r$  by permitting the 8-hour shifts to begin at any hour. (The solution, used only as a benchmark, involved 24 units starting at 16 different starting times -- an administrator's nightmare.) This optimal but impractical solution was then used to evaluate alternative solutions which were further restricted to having no more than five shift starting hours. Interestingly, they were still able to find a solution that required only 24 patrol units. Working with the three standard shifts used by the New York Police Department, the minimum number of units increased to 29. They were able to demonstrate that with the addition of two extra shifts, the number of patrol units could be reduced from 29 to 24, a 17.2 percent reduction, without a significant loss in performance.

Byrne and Potts (10) described an eight-step algorithm for finding daily shift schedules for toll collectors. The

A matrix used is extremely large, since it is necessary to include lunch and relief periods, and part-time shifts. The initial problem (P1) is solved as a linear programming problem ignoring the integrality requirements. Beginning with this solution, the numbers of full and part-time workers are examined and integer values are selected for each which satisfy the requirement that the number of part-time workers must not exceed a specified proportion of the full-time workers. The problem is solved again with these additional constraints. Additional non-integer variables are examined and modified. This process continues until integer values are obtained for all of the variables. While optimality is not claimed, the authors document the use of the procedure on 70 test problems and indicate that shift schedules derived with this algorithm have been implemented by the Adelaide Bus Company in Sidney, Australia for both bus drivers and toll collectors.

Also working with the problem of scheduling toll collectors, Guha and Browne (26) developed an algorithm for solving problem P1 when each column of the A matrix defines a shift covering exactly  $m$  consecutive time units (i.e., each work pattern has the same length and ignores time-off for lunch or breaks). Optimality is proven for this method.

Segal (31) obtains solutions to P1 by interpreting it in terms of a network flow problem. He describes a three-phase algorithm for obtaining acceptable shift schedules

for telephone operators. In the initial phase, a out-of-kilter algorithm is used to find solutions to a simplified version of P1 which ignores lunch and break times in each shift (thereby significantly reducing the size of the A matrix). In the next phase, a new problem is defined based on the "demand" for lunch and break periods within each tour. As much of this demand as possible is first satisfied by manpower exceeding the minimum requirements, using the manning levels indicated by the "ideal" solution of phase 1. In the final phase, unfilled demand is smoothed over adjacent time units to reduce demand "peaks", and the reduced problem is formulated as a network flow. An "ideal" solution is obtained, and the new shifts are added to those already generated in phases 1 and 2. Phases 2 and 3 are repeated until a lunch and break period is defined for each shift. Segal does not claim that the algorithm will always find the optimal solution, but reports that comparison of results obtained from a computerized version of this procedure with manual methods used in the past showed that the algorithm (1) reduced the total cost of producing schedules, (2) reduced the number of operators required, and (3) improved the quality of the schedules.

Henderson and Berry (39,40) also examined a variety of heuristic methods for determining shift schedules for telephone operators. The heuristics include several methods for selecting a small subset of column vectors  $A_j$  from the

A matrix which are used to create a reduced A matrix in problem P1. Two vector selection procedures were found to be most useful: random selection and maximal difference (defined below). If more than 40 columns were selected, Henderson and Berry found that reduced A matrices constructed with randomly selected columns yielded the best solutions to P1. If fewer than 40 columns were selected, the authors found the following method to be most effective: after the first column is selected randomly, each subsequent column is chosen on the basis that it is maximally different from the set of columns already selected (i.e., if  $A_1, A_2, \dots, A_{j-1}$  have previously been selected,  $A_j$  is selected from among all unselected columns  $A_k$  if it maximizes  $\max_k \sum_{i=1}^{j-1} |A_i - A_k| \cdot 1$ ).

The solution to P1 (with the reduced A matrix) is obtained using a two-step procedure: (1) the linear programming solution to P1 is obtained and all non-integer  $x_j$  are rounded up; and (2) several heuristic procedures are used to identify  $x_j$  that can be reduced (i.e., to identify operators on shift  $j$  that can be eliminated). The authors also investigated the use of a random solution in place of the LP solution in step 1. The LP solution to P1 with the complete A matrix is a lower bound on the optimal value of Z. Henderson and Berry studied the two-step procedure on 99 different problems involving A matrices constructed with different numbers of columns and based on various column selection strategies. They report that near-optimal

solutions can be obtained when the A matrix contains 100 or more columns, and that the procedure used to select the initial solution should depend primarily on the nature of the requirements vector r: for low manning levels, randomly selected solutions produced superior results, and for high manning levels, the LP solution was preferred. In (39), Henderson and Berry replaced the heuristic two-step procedure with a branch-and-bound algorithm to find the optimal solution. They report solving the P1 problem with A matrices containing 100 shift patterns and 72 20-minute time periods covering the 24 hour day.

Luce (9) examined P1 with an objective function of the form:

$$\min_x Z = \sum_{i=1}^t |r_i - w_i|$$

where  $r_i$  and  $w_i$  equal the preferred and actual manning levels for time unit  $i$  ( $w_i = A_i x$ ). This objective function yields the value of  $x$  that provides the best match of allocated manpower to preferred levels. Luce describes a heuristic sequential procedure for finding this value of  $x$  by adding one manshift at a time to a partial solution (i.e., identifying shift  $j$  and incrementing  $x_j$  by one). The next manshift to be included at each step is the work pattern that produces the greatest decrease in  $z$ ; the process stops when  $\sum_{i=1}^t w_i$  equals  $\sum_{i=1}^t r_i$ . Luce also examined an objective function based on the percentage

difference between each  $r_i$  and  $w_i$  (i.e.,  $\min_x Z = \left( \sum_{i=1}^t 1 - \frac{w_i}{r_i} \right)^2$ ).

Although the procedure does not yield optimal results, Luce argues for its use on the basis of computational simplicity and efficiency: an integer solution is always obtained in only one iteration.

Luce's algorithm is easily adopted for use on a fixed manpower problem (P2 above) by replacing the termination rule,  $\sum w_i = \sum r_i$ , with the constraint  $\sum x_i = M$ . In this form, Luce's heuristic becomes methodologically identical to an earlier formulation of this same problem by Heller (16). Heller uses an objective function of the form

$$\min_x Z = \sum_{i=1}^t (f_i^r - f_i^w)^2$$

where  $f_i^r$  equal the fraction of the manpower required for time units  $i$ , and  $f_i^w$  equals the fraction of the manpower allocated to time unit  $i$ . (The optimal value of  $z$  rarely equals zero because of the integrality constraints on  $x$ .) Heller devises a simple constructive procedure (similar to that later proposed by Luce) to find an optimal solution: one officer is added at a time on the basis of the maximum decrease in the complete derivative of  $z$  at each step. In (17), Heller and Stenzel extend this algorithm by introducing upper and lower limits on the number of officers assigned to each shift (i.e.,  $L_i \leq x_i \leq U_i$ ).

More recently, Butterfield and Howard (28) have solved the P2 problem using an objective function of the form:

$$\min_x z = \sum_{i=1}^t f(r_i, w_i)$$

where

$$f(r_i, w_i) = \begin{cases} (f_i^r - f_i^w)^2 & \text{if } f_i^r \geq f_i^w \\ 0 & \text{otherwise} \end{cases}$$

Also, rather than fix the total number of men to be used, they replace the constraint  $1'x = M$  with  $1'x = 100$  and redefine  $x_i$  to be the percentage of manpower on duty during time unit  $i$ . As a result, the integrality requirements on  $x$  can be dropped and non-integer values of  $w_i = A_i x$  exist. Optimal solutions are obtained using Rosen's gradient projection method.

#### 1.5.4.3 Days-Off Scheduling

The final step in the design of manpower schedules is the specification of work and recreation patterns for individual workers which match the designated manning levels by shift and day of the week. As noted earlier, the principal constraints on the characteristics of individual schedules (and hence on the solutions to the schedule design problem) are personnel work rules.

The problem of designing days-off schedules for fixed schedules can be stated analytically in a form identical to

problem P1 above.

$$\text{P3:} \quad \min_x z = c'x$$

$$\text{subject to} \quad Ax \geq r$$
$$x \geq 0$$

all  $x_j$ ,  $j = 1, 2, \dots, b$  integer

where:

$A = d \times b$  matrix, where each row  $A_i$  represents one day of the week (i.e.,  $d = 1, 2, \dots, 7$ ) and each column  $A_j$  represents a distinct seven day pattern of work and recreation days ( $a_{ij} = 1$  indicates that day  $i$  is a work day for pattern  $j$ ,  $a_{ij} = 0$  indicates a recreation day); each column can be used as a bracket in a fixed schedule.

$x = b \times 1$  vector, where component  $x_j$  equals the number of men assigned to bracket  $j$ .

$r = d \times 1$  vector, where component  $r_i$  equals the minimum number of men required for day  $i$ .

$c = b \times 1$  vector, where component  $c_j$  equals the cost of assigning one man to bracket  $j$ .

As in P1, if  $c = 1$ , the problem is reduced to minimizing the total number of men assigned to the  $b$  brackets of the schedule. Formulation P3 incorporates work and recreation period constraints into the columns of the  $A$  matrix (period lengths are indicated by consecutive ones and zeroes in each column). To illustrate using a simple example, assume that each officer must receive two consecutive days off

each week. The A matrix for this case, assuming no restrictions on which two days are used, is:

$$A = \begin{array}{c} \text{work patterns} \\ \left[ \begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right] \begin{array}{l} \text{Monday} \\ \text{Tuesday} \\ \text{Wednesday} \\ \text{Thursday} \\ \text{Friday} \\ \text{Saturday} \\ \text{Sunday} \end{array} \end{array}$$

The first column corresponds to a work pattern with Monday and Tuesday off, the second column uses Tuesday and Wednesday, and so on; a total of seven distinct work patterns (brackets) exist.

Tibrewala, Philippe, and Browne (32) presented a manual procedure for solving P3 with the A matrix illustrated above. Further work on this problem has been reported by Baker (33); Baker, Crabill, and Magazine (41); Rothstein (58), using an LP approach; and Guha and Browne (26), who adapted the initial work done by Tibrewala et al. to find shift schedules for problem P1.

As discussed earlier, fixed schedules having different work patterns in each bracket may provide good schedules for some officers and poor schedules for others. Some authors have attempted to produce more equitable schedules by having officers cycle through the brackets. Although this eliminates some inequities, it introduces another problem, since recreation and work periods are no longer

defined completely within each bracket and may be of unacceptable length.

Two survey papers on manpower scheduling, Bodin (14,15), and Baker (25) discussed the state of the art in days-off scheduling. Both papers indicate that no adequate mathematical statement or formulation of the problem yet exists. In (15), Bodin used the results of four previous research efforts in days-off scheduling to develop a generalized model of manpower scheduling. Bodin's model consists of four submodels which he labels (1) the allocation model, (2) the grouping model (to determine the specific recreation periods to be used), (3) the pattern model (to determine the sequence of work and recreation periods), and (4) the shift model. Almost all of the literature to date on the design of cyclic days-off schedules is based on heuristic and analytical techniques which are dependent on the special characteristics or features associated with the service (occupation) of interest to the authors.

Maier-Rothe and Wolfe (24) present an excellent review of the difficulties encountered in designing acceptable cyclic days-off schedules for nurses. Although the authors explicitly identify (in mathematical form) many of the basic constraints that define the nursing scheduling problem, they could not find a direct solution; rather they produced cyclic schedules with a five-week period by trading off the constraints heuristically to identify preferred solutions.

More recently, Smith (23) reported finding suitable cyclical days-off schedules for nurses using a five-phase algorithm coded for interactive use on a time-share computer. The five phases are (1) determination of staff requirements by day of week and shift; (2) determination of a feasible set of two-day recreation periods by solving a set of simultaneous linear equations (an approach first used by Monroe (27)) with adjustments to staff requirements if needed to solve the equations; (3) use of an heuristic procedure to designate the sequence of recreation periods; (4) designation of shift assignments for each work day; and (5) "fine tuning" the schedule using a series of heuristics to eliminate work rule violations.

Other heuristic procedures for designing cyclical schedules are reported by Altman (59), who studied scheduling refuse collectors in New York City; Guha and Browne (26), who studied scheduling toll collectors for the New York City Port Authority; and Bennet and Potts (60), who studied scheduling conductors for the Adelaide Bus Company in Australia.

The most extensive development of a systematic design procedure capable of incorporating a variety of work rules has been done by Heller (16). He describes a sequential process for the design of police schedules. The basic steps of the process are:

- (1) determination of specific manning levels for each shift tour for a fixed number of men assigned to the operation;
- (2) use of "cyclic graphs" to determine acceptable distributions of recreation periods for each shift (i.e., determining sets of recreation periods which preserve the designated manpower allocation); each cyclic graph is found by trial and error; and
- (3) use of a "separation matrix" for determining the proper sequence of recreation and work periods.

This thesis builds on Heller's findings by contributing more sophisticated and comprehensive algorithmic procedures for enumerating the sets of acceptable recreation periods and cyclic graphs, and for deriving optimal schedules using the separation matrix. Optimality is extended beyond Heller's original formulation to include additional schedule attributes of interest to police administrators, and seemingly applicable in other scheduling contexts.

## 2. THE MEASUREMENT OF PR SCHEDULES

### 2.1 INTRODUCTION

As indicated in Chapter 1 the basic design process for the construction of PR schedules was first identified by Heller (16). Use of Heller's procedure, however, does not insure (1) that any schedules exist for a given allocation of manpower, or (2) that a solution will be found, even if one or more schedules exist. Despite these limitations, when the presence of one or more PR schedules can be reasonably assumed, Heller's procedure has proven to be quite effective for enumerating small numbers of schedules\*. In addition, the procedure can be readily understood and used by persons without technical backgrounds after a brief training period.

Limitations of the method, however, include the following:

- (1) the procedure has no natural termination point (i.e., the user has no way of knowing when he has exhausted all possibilities); and
- (2) the procedure provides the user with only limited capability to find preferred schedules when many alternate solutions exist.

---

\* Computational experience acquired during this research suggests that the good results obtained using Heller's procedure are due, in part, to the fact that for manpower allocations associated with many practical scheduling problems, large numbers of feasible schedules usually exist.

With the algorithms developed in this thesis (and described in chapters 4 through 7), it is possible to examine all feasible PR schedules for any given manpower allocation\*. This capability insures that a feasible schedule, if any exists, will always be found. It also introduces the need for additional mechanisms to screen out large numbers of feasible schedules that may exist so that the enumeration procedure can be used as efficiently as possible to find PR schedules which are both feasible and, in some sense, the most preferred.

In this chapter, a procedure is described for screening out "unacceptable" (defined below) schedules and for discriminating between alternate acceptable schedules in terms of relevant schedule properties. To state this more precisely, let  $a_1$  and  $a_2$  each represent an acceptable PR schedule from the set A of all acceptable schedules for a given manpower allocation. (Acceptable schedules are, by definition, feasible.) Let  $m(a_1)$  and  $m(a_2)$  represent measure vectors based on the properties of schedules  $a_1$  and  $a_2$  respectively. The principal questions addressed in this chapter are:

- (1) What schedule attributes and measures should be used to construct  $m(a_1)$  and  $m(a_2)$ ? and
- (2) What decision process can be used to determine whether schedule  $a_1$  is "better" (i.e., superior, preferred, more desirable, etc.) than schedule  $a_2$ ?

---

\* "Feasible" schedules are those which satisfy the required manning levels by shift and day of the week.

It is a basic assumption of this thesis that no one criterion exists which adequately measures the desirability of all manpower schedules. This assumption follows directly from the observation that each alternative schedule is characterized by a variety of attributes and that the relative importance of each depends upon the environment in which the schedule is implemented and the preferences of individual officers. The objective of this chapter is to identify relevant schedule measures which, when used collectively, will indicate, in a valid and consistent manner, the preferability of one PR schedule over another.

In the following sections, three schedule attributes are introduced and quantitative measures are identified for each. Each measure is categorized into one of two groups. The first group consists of measures which are used only to determine whether a schedule is acceptable. As an example, manpower schedules are almost always designed with specific limits, both upper and lower, on the length of individual work periods. The upper and lower limits are quantitative measures which can be used to classify feasible schedules as either acceptable or unacceptable; feasible PR schedules which contain one or more work periods that exceed either of these limits are defined to be unacceptable.

The measures in the first group are used collectively to define a subset of acceptable schedules. The measures in the second group are used to quantify the "preferability"

of each acceptable schedule in the subset; these measures serve as the components in the measure (or preference) vector  $m(a_i)$  associated with each acceptable schedule.

The simultaneous use of several measures to determine preferable schedules requires a multicriteria decision model. Such a model, a sequential lexicographic decision process based on the preference vector associated with each schedule, is described in the final section of this chapter.

## 2.2 PR SCHEDULE ATTRIBUTES

### 2.2.1 Introduction

As indicated in the review of manpower scheduling research, many authors have identified schedule attributes that reflect the preferences, policies, and requirements for a variety of implementation environments. Although varying slightly in form depending upon the implementing agency, most preference criteria are related to one of the following schedule attributes:

- (1) the nature of weekend recreation periods;
- (2) the nature of all recreation periods, and
- (3) the nature of the work periods.

These attributes and quantitative measures for each are discussed below.

### 2.2.2 Weekend Recreation Periods

Consecutive days off on the weekend (Saturday and Sunday) are almost always viewed as an important schedule property. Weekend days are often the only time when

leisure activities involving several family members can be arranged. For workers in occupations which routinely require some weekend work, the regular scheduling of weekend recreation periods (i.e., recreation periods which include both Saturday and Sunday) is also considered an important schedule attribute. In police departments in which only a limited number of weekend periods can be scheduled, officers prefer to have these periods distributed as uniformly as possible. This may be particularly important for younger officers who are unable to obtain summertime vacations because of seniority rules governing vacation selection.

In this thesis, two measures are used to distinguish PR schedules in terms of their weekend recreation periods. These are:

- (1) the number of weekend periods, and
- (2) the maximum number of consecutive working weekends.

The first measure is the number of weekend recreation periods contained in one rotation period of a schedule. This measure is independent of the nature or "quality" of the weekend periods themselves (i.e., the length of the other days of the week included in each weekend period are not considered); the only requirement for a recreation period to be counted is that it must include both Saturday and Sunday\*.

---

\* To avoid the complications introduced by long recreation periods that cover consecutive weekends, it is assumed in the remainder of this thesis that recreation periods are always less than seven days in length.

If  $W_f$  denotes the number of weekend recreation periods in a schedule; the preference rule used in this thesis is that the greater the number of weekend periods (i.e., the higher the  $W_f$  value), the "better" the schedule. (In chapter 5, a method is described for determining the maximum value of  $W_f$  for any schedule based on a given allocation of manpower by day of the week.)

The second measure is used to quantify the distribution of weekend recreation periods over each rotation period of the schedule. The preference for uniformly distributed weekend recreation periods is equivalent to a preference for minimizing the maximum number of consecutive working weekends (denoted as CWW). (Any weekend not covered by a weekend recreation period is defined as a working weekend.) The preference rule used in this thesis is that the lower the CWW value, the better the schedule.

### 2.2.3 Recreation Periods

Alternative schedules that are "equal" in terms of the number and distribution of their weekend recreation periods (i.e., they have equal values for both  $W_f$  and CWW) may be considerably "unequal" in terms of the lengths of their respective weekend recreation periods, and in terms of the lengths and days of the week covered by their non-weekend periods.

As an example, a schedule with four-day weekend recreation periods would be considered more preferable than a

schedule containing an equal number of two-day weekend periods. As another example, suppose two schedules are identical in all respects except for the placement of one three-day recreation period; if one schedule has the period covering Thursday-Friday-Saturday, and the second schedule has the period covering Tuesday-Wednesday-Thursday, it is likely that the first schedule would be preferred because of the particular three days covered\*.

It is important to note that for both of these examples, the schedule selected would depend upon the preferences of the individual schedule designer, and that the purpose in this discussion is not to determine what these preferences should be, but rather to make explicit the schedule attributes and measures on which such decisions can be based.

Two measures of the recreation periods contained within a schedule (whether weekend periods or not) are used in this thesis. These are:

- (1) upper and lower limits on the lengths of recreation periods; and
- (2) a set of preference ratings for periods with acceptable lengths based on the length and position of each period within the week.

The first measure is a bounding criterion used to define the subset of acceptable schedules. The second measure is

---

\* It should be noted that these two schedules would not produce the same manpower allocation by day of the week.

based on a preference rating of the recreation periods that can appear in acceptable schedules. The preference rating for each recreation period is based on its length (i.e., the number of recreation days in the period) and its position within the week, (i.e., on which day of the week the period begins).

To illustrate, let  $U_R = 4$  and  $L_R = 2$  where  $U_R$  and  $L_R$  are the upper and lower limits respectively on recreation period lengths; hence, for the example limits, only schedules with recreation periods that are two, three, or four days long would be acceptable. The number of distinguishable recreation periods that can appear in an acceptable schedule equals  $7k$  where  $k$  is the number of distinct period lengths (i.e.,  $k = U_R - L_R + 1$ ). In this example,  $k = 3$  and a total of 21 distinct recreation periods (see table 2.1) can be used to measure the preference of each acceptable schedule.

Two seven-week schedules, both satisfying the daily manpower allocation in table 2.2 are shown in figures 2.1 and 2.2. Both schedules are acceptable in terms of their recreation periods (i.e., all of the periods in both schedules satisfy the  $U_R = 4$  and  $L_R = 2$  limits), and both schedules have the same number and distribution of weekend periods (i.e.,  $W_f = 1$  and  $CWW = 3$ ). These schedules do differ, however, in the nature of their recreation periods, both in length and days of the week covered. A comparative

Table 2.1

Twenty-One Recreation Periods Classified by  
Length and Starting Day of the Week with  
 $U_R=4$  and  $L_R=2$

Period Number	Period Length (Days)	Starting Day of the Week
1.	4	Monday
2.	4	Tuesday
3.	4	Wednesday
4.	4	Thursday
5.	4	Friday
6.	4	Saturday
7.	4	Sunday
8.	3	Monday
9.	3	Tuesday
10.	3	Wednesday
11.	3	Thursday
12.	3	Friday
13.	3	Saturday
14.	3	Sunday
15.	2	Monday
16.	2	Tuesday
17.	2	Wednesday
18.	2	Thursday
19.	2	Friday
20.	2	Saturday
21.	2	Sunday

Table 2.2

Daily Manpower Allocation for a Seven-Week Schedule

Number of Men	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.
On Duty	5	5	5	5	5	5	4
On Recreation	2	2	2	2	2	2	3
Total	7	7	7	7	7	7	7

assessment of these schedules can be based, in part, on the number of each type of distinct recreation period that exists in each schedule. To illustrate, both schedules contain five recreation periods (see table 2.3); two periods are common to both: a four-day period beginning on Thursday and a two-day period beginning on Tuesday. The remaining three periods in each schedule are different: the four-day period that starts on Friday in figure 2.1 begins on Sunday in figure 2.2; the three-day period that starts on Tuesday in the first schedule, begins on Saturday in the second schedule; and finally, the two-day period that starts on Sunday in figure 2.1 begins on Thursday in figure 2.2. A

	M	T	W	T	F	S	S
1	R						
2		R	R	R			
3					R	R	R
4	R						
5		R	R				
6				R	R	R	R
7							R

Figure 2.1

Sample Schedule Based on the Manpower  
Allocation in Table 2.2

	M	T	W	T	F	S	S
1	R	R	R				
2				R	R		
3						R	R
4	R						
5		R	R				
6				R	R	R	R
7							R

Figure 2.2

Sample Schedule Based on the Manpower  
Allocation in Table 2.2



**CONTINUED**

**1 OF 6**

Table 2.3

Number of Recreation Period Types in the  
PR Schedules in Figures 2.1 and 2.2

Period Number (Table 2.1)	Period Length (Days)	Start Day	Number of Periods	
			Figure 2.1	Figure 2.2
4.	4	Thursday	1	1
5.	4	Friday	1	0
7.	4	Sunday	0	1
9.	3	Thursday	1	0
13.	3	Saturday	0	1
16.	2	Tuesday	1	1
18.	2	Thursday	0	1
21.	2	Sunday	1	0

decision model to select the preferable schedule based on the number of each type of recreation period is presented in section 2.3.

#### 2.2.4 Work Periods

Another attribute that is important in selecting one schedule over another is the nature of the work periods within each schedule. In environments where manpower allocations are driven by demand levels that vary by shift and day of the week, manpower work schedules often contain work periods that are longer and shorter than the "normal" five-day period. As indicated in the survey of police officers conducted for this thesis (see appendix 10.1), some departments use schedules that require officers to

work as many as ten consecutive days (the average maximum work period length reported was 6.5 days). As recently as 1973, the St. Louis Metropolitan Police Department used a manpower schedule for field operations that required officers to work 13 consecutive days at least once during the year (61). Not surprisingly, police officers who work such schedules complain that their performance deteriorates during the last day of such periods.

In contrast, very short work periods (e.g., only one or two days) are also considered unattractive because officers have little opportunity to adjust to the work routine; such periods may be perceived as interruptions to the preceding and following recreation periods.

This discussion suggests another bounding measure to determine the subset of acceptable schedules, upper and lower limits on work period lengths.

A number of additional preference measures based on work periods are also used in this thesis to characterize individual schedules. They include:

- (1) the maximum length work period;
- (2) the number of times the maximum length work period appears in one rotation period of the schedule;
- (3) the uniformity of the work period lengths as measured by the range of the lengths (i.e., the difference between the maximum and minimum length periods);

- (4) the distribution of long and short work periods as measured by the maximum number of work days in consecutive work periods; and
- (5) the number of times the consecutive period maximum appears in one rotation period of the schedule.

The first two measures are used to identify PR schedules which minimize both the length of the longest work period and the number of time it appears in the schedule. The third measure reflects the preference for uniformity in work period lengths. (If all of the periods have the same length, the range equals zero.) The fourth and fifth measures reflect the preference for alternating long and short work periods when the range of the work periods does not equal zero. The measures are used to identify schedules which minimize both the maximum number of days that appear in consecutive work periods and the number of times the maximum value occurs.

To illustrate these measures, consider the schedule in figure 2.1. The sequence of work period lengths is {7,7,7,7,6}. The maximum length period is seven days (measure 1) and the maximum value occurs four times (measure 2). The range of the schedule is one day (measure 3), and the maximum two-period total is 14 days (measure 4) which occurs three times (measure 5). The PR schedule in figure 2.2 has exactly the same sequence of work period lengths and hence with regard to work periods, is identical to the schedule in figure 2.1.

### 2.2.5 Other Schedule Features

Two additional schedule features exist which are not conveniently classified under any of the attributes discussed above. These features, however, are important schedule characteristics, and are incorporated into the PR schedule design methodology described in this thesis. The first characteristic is applicable only to multishift PR schedules; the second is based on a schedule property defined by the sequence of lengths of the work and recreation periods in a schedule.

Rotating multishift schedules require each officer to rotate periodically from one shift to another; such rotations force each officer to readjust his personal off-duty schedule to accommodate his new shift assignment. Some departments which use shift rotating schedules are not able to schedule changeover recreation periods for each shift changeover point in the schedule (i.e., a recreation period that separates the last work day on the old shift from the first work day on the new shift). To compensate for the absence of such scheduled time-off, most departments use a "backward" rotation through the shifts (i.e., night to afternoon to day) to insure that each officer receives a minimum of eight hours off between work assignments (see table 2.4). (See table 2.5 for the consequences of a "forward" shift rotation with no intervening recreation periods.)

Table 2.4

Time Off Between Shift Assignments for  
"Backward" Rotating Schedules\*

Shift Rotation	Time Off Between Assignments (Hours)
Night to afternoon	8
Afternoon to day	8
Day to night	32

\*Based on a three-shift schedule for  
a "police day" beginning with the day shift.

Table 2.5

Time Off Between Shift Assignments for  
"Forward" Rotating Schedules\*

Shift Rotation	Time Off Between Assignments (Hours)
Day to afternoon	24
Afternoon to night	24
Night to day	0

\*Based on a three-shift schedule for  
a "police day" beginning with the day shift.

The presence of a recreation period between shift assignments can minimize or eliminate many of the difficulties associated with shift changeover: personal adjustments can be more easily made, short change-over times are eliminated, and each officer begins his new assignment by coming off a recreation period instead of experiencing the fatiguing effects of working "straight through."

Expressed as a preference measure, this attribute can be quantified by noting the fraction of shift changeovers that are covered by changeover recreation periods. (A fraction of 1.0 would indicate that every changeover point in the schedule was covered.) As a bounding measure, acceptable multishift schedules can be defined as those schedules which have a changeover recreation period for at least a specified minimum fraction of all changeover points.

The desirability of this attribute lead to the decision early in this study, to investigate the feasibility of using the most stringent measure: that is, a bounding condition that required the presence of a changeover recreation period at every shift changeover point. Computational experience has verified the feasibility of this decision. It is possible, for most applications, to find multishift schedules which satisfy this requirement; and when schedules cannot be found, it is almost always due to another attribute.

The second schedule feature concerns the relative positions of the work and recreation periods within each schedule. The importance of this feature lies in the fact that officers prefer schedules which place long recreation periods after long work periods (i.e., schedules which are "orderly"). The measure adopted in this study to quantify this characteristic is based on the set of ratios formed by dividing the length of each work period by the length of the recreation period immediately following it. The standard deviation of these ratios is used as the "orderliness" measure for each schedule. Although the specific numeric value obtained for a schedule is not in itself significant, the measure enables comparisons to be made with other schedules.

The ratios and standard deviations for the schedules in figures 2.1 and 2.2 are shown in table 2.6. The lower standard deviation for the schedule in figure 2.1 reflects the slight superiority of this schedule in terms of the sequence of work and recreation periods. The schedule in figure 2.1 has both of its four-day recreation periods following seven-day work periods; the schedule in figure 2.2 has its two four-day periods following seven and six-day work periods.

Table 2.6

Standard Deviations of the Work to Recreation  
Period Length Ratios for the Schedules in  
Figures 2.1 and 2.2

Work Period Length (Days) (1)	Following Recreation Period Length (Days) (2)	Ratio (1)/(2)	Standard Deviation of Ratios
--	---	------------------	------------------------------------

Figure 2.1

7	3	2.33	0.69
7	4	1.75	
7	2	3.50	
7	4	1.75	
6	2	3.00	

Figure 2.2

7	2	3.50	0.84
7	3	2.33	
7	2	3.50	
7	4	1.75	
6	4	1.50	

#### 2.2.6 Summary of PR Schedule Measures

This section summarizes the schedule attributes and measures that are used in this thesis to design optimal PR schedules. To more clearly indicate how these measures are used in the design process that begins with a manpower allocation by shift and day of the week, and concludes with a PR schedule that matches that allocation, it is convenient to consider the design process in terms of three

sets of schedules. (The discussion that follows is schematically represented in figure 2.3.)\*

Given any manpower allocation, the set of feasible schedules contains all cyclic schedules which match the required allocation. Although the number of feasible schedules that exists for any allocation is indeterminate (there may be none), for most applications, multiple feasible schedules exist. (With no constraints on the lengths of either the work or recreation periods, it is usually possible to produce a feasible schedule by hand in only a short time using trial and error methods.)

The second set of schedules is a subset of the collection of all feasible schedules for a given manpower allocation. The subset of acceptable schedules includes only feasible schedules which possess each of the following characteristics:

- (1) a recreation period of acceptable length at every shift changeover point (only applicable to multishift schedules);
- (2) a set of work periods each of which satisfies the upper and lower limits on work period lengths; and
- (3) a set of recreation periods each of which satisfies the upper and lower limits on recreation period lengths.

A feasible schedule that does not possess each of these features is defined to be unacceptable.

---

\* Figure 2.3 is a conceptual model of the design process which is useful in explaining the application of individual schedule measures. It is not, however, an accurate representation of the actual computational process by which preferable PR schedules are obtained.

Set

Defining Criteria

Feasible Schedules:

Cyclic schedules which match a given manpower allocation by shift and day of the week.

Acceptable Schedules:

Feasible schedules which possess each of the following:

- (1) a recreation period at every shift changeover point (if applicable);
- (2) a set of work periods, each of which satisfies the upper and lower limits on work period lengths; and
- (3) a set of recreation periods each of which satisfies the upper and lower limits on recreation period lengths.

Preferable Schedules:

Acceptable schedules which are ranked according to the following preference criteria:

- (1) number of weekend recreation periods,
- (2) maximum number of consecutive working weekends,
- (3) maximum work period length,
- (4) number of maximum length work periods,
- (5) work period range,
- (6) number of each type of recreation period,
- (7) maximum number of days in consecutive work periods,
- (8) number of maximum length work period pairs, and
- (9) standard deviation of the work to recreation period length ratios.

Figure 2.3

Defining Criteria for Feasible, Acceptable, and  
Preferable Sets of PR Schedules

After the set of acceptable schedules has been identified, preference measures for weekend recreation periods, work periods, and all other recreation periods are used to establish the preferability of each schedule. The ordering of the acceptable schedules according to a preference function produces a set of preferable PR schedules. (This third set of schedules is formed, not by eliminating acceptable schedules, but rather by quantifying the desirability of each in terms of its attributes.)

To illustrate the differences that can exist between acceptable schedules, consider the four multishift PR schedules in figures 2.4 through 2.7. Each schedule satisfies the same manpower allocation (i.e., each is feasible); and each schedule also possesses (1) a recreation period at every changeover point, (2) acceptable work periods (the limits set in this case were that every period be between three and nine days long), and (3) acceptable recreation periods (every period was required to be either two, three, or four days long). Despite these similarities, these schedules differ in a number of ways. The values for nine preference measures for each schedule are shown in table 2.7.

To determine which of the four PR multishift schedules is the most desirable requires a multicriteria decision model that incorporates the preference measures shown in table 2.7. The decision model used in this thesis is discussed in the following section.



Shift		M	T	W	T	F	S	S
Night	1	R	R	R				
	2				R	R		
	3						R	R
	4							
	5		R	R	R			
Aft.	6	R	R					
	7			R	R			
	8					R	R	R
	9							
	10		R	R	R	R		
Day	11						R	R
	12	R	R					
	13			R	R			
	14					R	R	R
	15						R	R
	16				R	R		
	17		R	R				R

Figure 2.4

Sample Multishift Schedule I

Shift		M	T	W	T	F	S	S
Night	1	R	R	R				
	2				R	R		
	3						R	R
	4							
	5		R	R	R			
Aft.	6	R	R					
	7			R	R			
	8				R	R	R	R
	9							
	10		R	R				
Day	11					R	R	R
	12	R						
	13		R	R				
	14				R	R	R	R
	15						R	R
	16				R	R		
	17		R	R				R

Figure 2.5

Sample Multishift Schedule II

Shift		M	T	W	T	F	S	S
Night	1	R	R	R				
	2				R	R		
	3						R	R
	4							
	5		R	R	R			
Aft.	6	R	R	R				
	7				R	R		
	8					R	R	R
	9							
	10		R	R	R			
Day	11						R	R
	12	R	R					
	13			R	R			
	14					R	R	R
	15						R	R
	16				R	R		
	17		R	R				R

Figure 2.6

Sample Multishift Schedule III

Shift		M	T	W	T	F	S	S
Night	1	R	R	R				
	2				R	R		
	3						R	R
	4							
	5		R	R	R			
Aft.	6	R	R	R	R			
	7				R	R		
	8					R	R	R
	9							
	10		R	R				
Day	11						R	R
	12	R						
	13		R	R				
	14				R	R	R	R
	15						R	R
	16				R	R		
	17		R	R				R

Figure 2.7

Sample Multishift Schedule IV

Table 2.7

Preference Measure Values for Multishift  
Schedules I, II, III and IV in  
Figures 2.4, 2.5, 2.6, and 2.7

Preference Measure	Multishift Schedule (Figure Number)			
	I (2.4)	II (2.5)	III (2.6)	IV (2.7)
1. Number of weekend recreation periods	5	5	5	5
2. Maximum number of consecutive working weekends	4	4	4	4
3. Maximum length work period (days)	8	8	8	9
4. Number of maximum length work periods	2	3	3	1
5. Number of recreation periods of each type: Length (days)/Start Day				
4/Fri.	0	1	0	0
4/Thu.	0	2	0	1
4/Sat.	1	0	1	0
3/Fri.	2	0	2	1
3/Sat.	0	0	0	1
2/Sat.	2	2	2	2
4/Wed.	0	0	0	0
3/Thu.	0	0	0	0
4/Sun.	1	1	1	1
3/Sun.	0	0	0	0
2/Fri.	0	0	0	0
2/Sun.	0	0	0	0
6. Work period range (days)	5	5	5	6
7. Maximum number of days in consecutive work periods	15	16	16	17
8. Number of maximum length work period pairs	3	1	1	1
9. Standard deviation of the work to recreation period length ratios	0.87	1.00	0.91	1.02

## 2.3 SEQUENTIAL LEXICOGRAPHIC COMPARISON OF PR SCHEDULES

### 2.3.1 Introduction

In the preceding section, several preferential schedule measures were introduced. This section presents a decision process for determining whether schedule  $a_1$  is "more preferred", "equal to", or "less preferred" than schedule  $a_2$  in terms of these preference measures. The selection of a decision process is difficult because of the diverse nature of the schedule attributes to be compared. As an example, suppose that in comparing schedules  $a_1$  and  $a_2$ , it is found that  $a_1$  has more weekend recreation periods than  $a_2$ , but also contains work periods that are longer than any work periods in  $a_2$ . The tradeoff between the number of weekend recreation periods and the maximum length work period cannot be resolved by purely analytic procedures but should, in some manner, reflect the relative importance of these attributes to the schedule designer.

### 2.3.2 Sequential Lexicographic Elimination

To utilize the individual preferential schedule measures discussed above and to incorporate the relative importance assigned to each measure by schedule designers, a sequential elimination procedure using a lexicographic ordering of the schedule measures is used in this thesis. The lexicographic method requires that the schedule measures (i.e., the components of the measure vector) be ranked in terms of importance, and that the values of each measure be placed, at a minimum,

on an ordinal scale. Once the most important measure is selected, the schedule(s) having the best value for this measure is(are) determined. If a single such schedule exists, it is selected as the best; if multiple schedules have the best value for the specified measure, these schedules are then compared with respect to the second most important measure. This process is continued on a measure-by-measure basis until either a single schedule emerges or two or more schedules, equal among all measures, are found. Repeating this process on the set of schedules not already ranked, eventually yields a ranking order for all of the alternative schedules. As indicated, the lexicographic procedure only requires that the components of the measure vector be ranked in order of their importance. Hence the inter-component measurement scale is ordinal and the only input required from the schedule designer is a set of preference rankings for the schedule measures to be considered.

To implement this procedure, nine preference measures were identified and used as components in the measure vector; the nine measures and their relative importance as used in this thesis are indicated in table 2.8. Each of these measures has been discussed above, and the use of each is self-explanatory except for the measure of the recreation period set (preference ranking number 5 in table 2.8). This measure is formulated as a subvector whose components represents the number of times each type of recreation

Table 2.8

Preference Rankings of the Nine Schedule Measures  
Used to Design PR Schedules

Preference Rank	Schedule Measure
1.	Number of weekend recreation periods
2.	Maximum number of consecutive working weekends
3.	Maximum length work period (days)
4.	Number of maximum length work periods
5.	Recreation period measure based on the number of each type of recreation period*
6.	Work period range (days)
7.	Maximum number of days in consecutive work periods
8.	Number of maximum length work period pairs
9.	Standard deviation of the work to recreation period length ratios

\*See table 2.9 for the preference rankings used to measure each set of recreation periods.

period (defined by length and start day) appears in the schedule. The ordering of the components in the subvector is determined by the relative importance given to each period type. The importance rankings used in this thesis are shown in table 2.9.

Table 2.9

Preference Rankings for Individual Recreation Periods  
Used to Design PR Schedules

Preference Rank	Period Length (Days)	Start Day
1.	4	Friday
2.	4	Thursday
3.	4	Saturday
4.	3	Friday
5.	3	Saturday
6.	2	Saturday
7.	4	Wednesday
8.	3	Thursday
9.	4	Sunday
10.	3	Sunday
11.	2	Friday
12.	2	Sunday

These rankings are used for all of the schedules discussed in this thesis and are applicable for schedules that limit the recreation period lengths to two, three, or four days. (The method is easily extended to periods with other lengths.) The use of these rankings parallels the lexicographic process described above. If two schedules have equal values for the first four components of their measure vectors, their respective recreation period subvectors are compared. The comparison is done component-by-component (in the order indicated in table 2.9) until a difference between like-ordered components is found; the schedule with the higher value for this component is defined as the preferred schedule. If the subvectors are equal (i.e., the like-ordered components are equal for all 12 components), the comparison continues with the sixth ranked measure (work period range) in the measure vector.

To illustrate the use of the sequential process, consider the comparison of the four PR schedules in figures 2.4 through 2.7. The complete measure vector for each schedule is shown in table 2.7. (The components for each schedule are read from top to bottom in order of decreasing importance.) All four schedules have identical values for the first two measures: each schedule contains exactly five weekend recreation periods and a maximum of four consecutive working weekends. For the next most important measure (i.e., maximum work period length), schedules I,

II, and III are still equal, but schedule IV is less preferable by virtue of its nine-day work period. Examination of the next measure (i.e., the number of maximum length work periods) for schedules I, II, and III indicates that schedules II and III both contain three eight-day work periods while schedule I contains only two. Hence both schedules II and III are judged to be less preferable than schedule I. As the only remaining schedule of the original four, schedule I is defined as the most preferred.

To determine the second preferred schedule, the measure vector components for schedules II and III are compared. Based upon the measure vectors in table 2.7, schedule II is preferred to schedule III because it contains one four-day recreation period beginning on Friday while schedule III contains none. (Schedule II, in fact, possesses the most preferable set of recreation periods of the four schedules, but is ranked below schedule I because of less attractive work period properties.)

## 2.4 MULTIPLE CRITERIA DECISION MAKING PROCEDURES

### 2.4.1 Introduction

The design and selection of an "optimal" schedule, like many social and political decisions often cannot be realistically based on only one criterion. Rather, as seen in the discussion above for comparing schedules, a variety of factors must be considered which usually are not directly comparable (e.g., which is more important: one

additional recreation weekend period or one less day in the maximum length work period?). In response to the growing demand for more sophisticated decision tools, a variety of multiple criteria decision making procedures have been developed. This section presents a brief review of these procedures in order to identify the specific advantages of the sequential elimination procedure used in this thesis.

Multiple criteria decision methods can be divided into two categories: collective procedures which assume that the individual components of the measure vector can be unified either explicitly or implicitly into a one-dimensional measure for each alternative schedule, and component methods which compare schedules strictly on a component-by-component basis.\* Both of these methods are discussed below.

#### 2.4.2 Collective Procedures

Explicit collective procedures are characterized by the requirement that an explicit compensatory objective function must be defined which numerically relates all of the components of the measure vector. All weighting schemes, regardless of how diverse they appear, fall into this category. Despite their individual differences, all weighting procedures possess the following characteristics:

- (1) a set of available alternatives with specified measures and measure values;

---

\*The discussion that follows is based to a large extent on an excellent survey paper by Kenneth R. MacCrimmon entitled: "An Overview of Multiple Objective Decision Making". (62)

- (2) a process for obtaining numeric values for each measure and numeric weights across measures;
- (3) a well-specified objective function for aggregating the numerically scaled preferences into a single number for each alternative; and
- (4) a rule for selecting the alternatives on the basis of the highest (or best) value.

Weighting methods can be grouped into two subcategories according to the method used for determining the preferences of the decision-maker. Preferences can be obtained by:

- (1) inference based on statistical analysis of past choices; or
- (2) direct questioning of the decision-maker about each component.

Inferential methods include both linear regression (63,64), and analysis of variance (64,65,66). Direct questioning methods have the advantage of not requiring a history of past decisions made under similar circumstances, and the disadvantage of requiring the decision-maker to verbalize his preference. A variety of weighting models based upon direct questioning exist. Some of the more commonly utilized include trade-offs (67), simple additive weighting (56,68,69), and modified weighting schemes (67,70,71,72,73,74,75).

More advanced among explicit collective procedures are mathematical programming schemes which incorporate constraints on individual or groups of attributes and are able to handle measure vectors with large numbers of

components. These methods include linear programming, goal programming (76,77,78), and interactive programming (79,80).

Implicit collective procedures do not externalize a specific function to relate the components of each measure vector, but assume that such a function can be operationally defined by the preferences of the decision-maker. These methods are characterized by the manner in which preferences are obtained and utilized to find the best decision; Delphi methods and the Churchman-Ackoff pair-by-pair decision procedure(s) are both implicit collective methods.

#### 2.4.3 Component Procedures

Component procedures seek to avoid the difficulties frequently cited with collective procedures. With explicit collective methods, it is necessary to establish numerical relationships between components which often conceptually defy such comparisons; implicit collective methods assume that the decision-maker possesses a consistent set of preferences which can be, in some manner, extracted, and frequently require that the decision-maker be available for direct participation in the resolution process.

Component procedures are characterized by comparisons which are made exclusively between like components of the measure vectors of two alternatives. Included among these methods are conjunctive and disjunctive constraints, dominance, and the method used in this study, lexicographic sequential elimination.

#### 2.4.3.1 Conjunctive and Disjunctive Constraints

In this method, the decision-maker establishes standards for the values of certain components. In a conjunctive form, all of the standards must be satisfied for the alternative to be judged acceptable. (The acceptability or bounding measures described earlier in this chapter are used as conjunctive constraints.) In the disjunctive form, only one standard of a specified group must be met for an alternative to be acceptable.

Since each alternative must satisfy a number of standards in the conjunctive method, the procedure serves primarily as a filtering device to screen out all but the most desirable alternatives. Kleinmuntz (81) utilizes conjunctive and disjunctive decision models to describe the decision process expert clinicians use to categorize persons into "adjusted" and "maladjusted" categories on the basis of the MMPI profiles. Other applications of constraint usage are given by Clarkson (82) for portfolio decisions, by Smith and Greenlaw (83) for personnel selection decisions, and by Bettmen (84) for consumer choices. Combs (85) and Dawes (86) present theoretical consideration about the method.

#### 2.4.3.2 Dominance

In conjunctive and disjunctive constraint procedures, each alternative is compared to a minimal set of standards for some or all of the components. Dominance methods

provide for direct comparisons between pairs of alternative schedules. If one alternative has component values that are at least as good as those of another alternative for all of the components and if it has one or more values that are better, the first alternative is said to "dominate" the second and the second alternative can be eliminated. Although this procedure is one of the least controversial decision methods, it often fails to eliminate many alternatives. Most decision-makers use dominance whenever appropriate as an initial filtering device to reduce the set of viable alternatives.

#### 2.4.3.3 Lexicography

Although lexicography is also a form of sequential elimination, the method differs from both constraint and dominance methods; the comparison of alternatives is accomplished by comparing the values of like components in a specified order. As a result, the lexicographic procedure requires that the decision-maker rank the components of the measure vector in terms of importance, and that the values for each component be placed, at least, on an ordinal scale. A complete ranking for all alternatives is achieved using the process described in section 2.3.2.

Extensions of this method include allowing for bands of imperfect discrimination so that one alternative will not be judged better than another because of a slightly

better value for one attribute. This "semiorder" processing is described by Luce (87) and Tversky (88). The theoretical aspects of lexicography are discussed by Luce (87), Shepard (89), and Fishburn (90).

The most important condition associated with the use of the lexicographic method is the requirement that a strict and consistent hierarchical ordering of the components exists which can be externalized by the decision-maker. In addition, the method also imposes the constraint that if component a is judged to be more important than component b and if b is judged to be more important than component c, then it is assumed that component a is also more important than all possible combinations of components b and c. In fact, component a is assumed to be more important than the cumulative value of all components that are individually less important than a.

The lexicographic method does, however, possess several distinct advantages as a decision tool for determining preferable PR schedules. These include:

1. The method requires the direct comparison of only like components (e.g., the number of weekend recreation periods for a schedule is compared only with the corresponding number in alternative schedules). The method avoids entirely the necessity of determining, either implicitly or explicitly, compensatory relationships between unlike components.

2. The method permits the decision-maker to indicate which attributes he values and the relative importance he assigns to each.
3. The component-by-component decision process is particularly well-suited for the sequential construction methods used in this thesis to design PR schedules. Partially designed schedules available at each stage of the design process can be ranked according to the component values that are known; this ranking in turn can be used to discard sets of partially designed schedules which can only produce complete schedules that are less preferable.
4. The use of the bounding measures (conjunctive constraints) serves to minimize the negative aspects of the hierarchical ordering of the components imposed by the lexicographic procedure (i.e., the method is only used to rank schedules which already satisfy acceptability constraints).
5. The method is easily understood by schedule designers who must (1) identify the schedule attributes and measures they wish to use, (2) provide a preference rating for each measure, and (3) specify the constraint values for the bounding measures.

The use of the lexicographic decision method for the design of optimal PR schedules is described in chapter 3.

### 3. THE DESIGN OF OPTIMAL PR SCHEDULES

#### 3.1 INTRODUCTION

Although the feasibility of constructing acceptable PR schedules by hand for relatively small schedules (e.g., four or five weeks long) has been demonstrated by Heller (16), improved design methods are needed to provide schedule designers with more systematic and reliable procedures for constructing schedules which satisfy given manpower allocations and are preferred in terms of relevant schedule properties. Without such methods, the design of PR schedules is highly dependent upon the skills, resources, and experience of individual schedule designers. Improved design methods, such as those described in this thesis, can contribute to the process of shifting the effort required for the design of manpower schedules from one of "finding" feasible schedules (which, hopefully, also possess desirable features) to the more fruitful area of identifying important schedule properties and using them to "design" the best schedules possible.

This chapter describes each major step in the design and construction of PR schedules. This overview of the design process serves as an introduction to the material presented in the next four chapters which describe the algorithms used for each step of the process. The remainder of this chapter is divided into three sections:

the first describes the data requirements for the design process; the next describes a four-step procedure for the design of optimal one-shift PR schedules; and the final section describes the process for the design of optimal multishift PR schedules.

### 3.2 SCHEDULE DESIGN DATA REQUIREMENTS

#### 3.2.1 Manpower Allocation

The schedule design procedures begin with an allocation, by shift and day of the week, of the total number of work tours (or manshifts) per week for a specified number of officers. Cyclic schedules which preserve this weekly allocation are defined to be feasible. To indicate the elements of a manpower allocation more clearly (and also to introduce notation which will be used throughout this chapter), we define the following quantities:

$N$  = number of officers\*;

$s$  = number of shifts;

$N_i$  = number of men assigned to each shift ( $i=1, 2, \dots, s$ );

$f$  = the average number of work days expected from each officer per week;

---

\*In designing PR schedules, it is more precise to speak of the number of schedule brackets to be allocated to each shift rather than the number of officers. For convenience, however, the presentations in this and subsequent chapters are based on the assumption that one man is assigned to each bracket. This causes no loss of meaning or generality, and permits the use of the more commonly used terms "men" and "manshifts" in place of "brackets" and "bracketshifts".

$W$  = number of manshifts (i.e., on-duty tours),  
each week over  $s$  shifts for  $N$  officers;

$W_i$  = number of manshifts each week on shift  $i$   
for  $N_i$  officers;

$w_{ij}$  = number of manshifts (officers) assigned to  
work shift  $i$  on day  $j$ ;

$R$  = number of recreation days each week over  $s$   
shifts for  $N$  officers;

$R_i$  = number of recreation days each week on shift  
 $i$  for  $N_i$  officers; and

$r_{ij}$  = number of officers on recreation from shift  
 $i$  on day  $j$ .

The following relationships hold between these quantities:

(1) the number of officers,

$$N = \sum_{i=1}^s N_i = \frac{1}{7} \sum_{i=1}^s (W_i + R_i) = \frac{1}{7} \sum_{i=1}^s \sum_{j=1}^7 (w_{ij} + r_{ij});$$

(2) the number of on-duty manshifts per week,

$$W = \sum_{i=1}^s W_i = \sum_{i=1}^s \sum_{j=1}^7 w_{ij} \approx fN^*;$$

(3) the number of recreation days per week,

$$R = 7N - W = \sum_{i=1}^s R_i = \sum_{i=1}^s \sum_{j=1}^7 r_{ij};$$

(4) the number of on-duty manshifts for shift  $i$   
per week,

$$W_i = \sum_{j=1}^7 w_{ij} \approx fN_i; \text{ and}$$

---

\*Since the number of manshifts expected from each officer per week is usually not equal to an integral number of shifts (due to the "amortization" of time off for holidays, if included, over each week of the schedule), the product  $fN$  is usually not an integer.

- (5) the number of recreation days for shift  $i$  per week,

$$R_i = 7N_i - W_i = \sum_{j=1}^7 r_{ij} . \quad (3.1)$$

As an example, consider a work force of 20 men ( $N=20$ ) in which each officer works an average of 4.75 days per week ( $f=4.75$ ). These values indicate that the 20 officers work 95 on-duty manshifts each week (i.e.,  $W = fN = (4.75) \times (20) = 95$ ). A sample allocation of these 95 manshifts over three shifts ( $s = 3$ ) and seven days of the week is illustrated in table 3.1. The distribution of the corresponding 45 recreation days per week ( $R = 7N - W = (7) \times (20) - 95 = 140 - 95 = 45$ ) is shown in table 3.2. A PR schedule designed to preserve the allocations shown in tables 3.1 and 3.2 would have a 20-week rotation period in which each officer would be assigned to shift 1 for seven weeks, to shift 2 for eight weeks, and to shift 3 for five weeks. Over the 20 week period, each officer has 95 work days and 45 recreation days.

The allocation problem and the algorithmic procedures used to obtain satisfactory manpower distributions have been reviewed in chapter 1 (section 1.5.4). All of the methods discussed in chapter 1 focus on the derivation of the  $w_{ij}$  values (i.e., the number of officers on-duty for each shift and day of the week). Determination of the

Table 3.1

Allocation of 95 Manshifts Over Three Shifts and  
Seven Days of the Week

Shift (i=)	Number of Men Assigned to Shift (N <sub>i</sub> =)	Number of Men On Duty (w <sub>ij</sub> )							Total Number of Manshifts
		Mon. (j=1)	Tue. (j=2)	Wed. (j=3)	Thu. (j=4)	Fri. (j=5)	Sat. (j=6)	Sun. (j=7)	
1	7	4	4	5	5	6	6	3	33
2	8	5	5	5	5	7	7	4	38
3	5	3	3	3	4	4	4	3	24
	N=20								W=95

Table 3.2

Distribution of the Recreation Days Corresponding to the  
Manpower Allocation in Table 3.1 by Shift and  
Day of the Week

Shift (i=)	Number of Men Assigned to Shift (N <sub>i</sub> =)	Number of Men on Recreation (r <sub>ij</sub> )							Total
		Mon. (j=1)	Tue. (j=2)	Wed. (j=3)	Thu. (j=4)	Fri. (j=5)	Sat. (j=6)	Sun. (j=7)	
1	7	3	3	2	2	1	1	4	16
2	8	3	3	3	3	1	1	4	18
3	5	2	2	2	1	1	1	2	11
	N=20								R=45

$w_{ij}$ 's, however, does not define a unique set of  $N_i$  values (i.e., the number of officers assigned to each shift). To illustrate, consider the manpower allocation shown in table 3.1. The  $w_{ij}$  values for shift 1 indicate that a minimum of six on-duty officers must be assigned to that shift in order to satisfy the manpower requirements for Friday and Saturday. Similarly, the maximum values for shift 2 and 3 indicate that a minimum of seven officers must be assigned to the second shift (to work on Friday and Saturday) and a minimum of four officers must be assigned to the third shift (for Thursday, Friday and Saturday). The sum of these three minimum values equals 17, leaving three of the 20 officers unassigned.

In (16) Heller determines shift designations for unassigned officers by using their shift assignment to minimize the quantity  $\sum_{i=1}^S \left( \frac{W_i}{R_i} - \frac{W_i}{R_i} \right)^2$  where  $W/R$  equals the ratio of work to recreation days for the entire schedule. The motivation for this objective function is to equalize, as nearly as possible, the ratio of work to recreation days ( $W_i/R_i$ ) for each shift. The  $W_i$  values, determined from the  $w_{ij}$  values, are fixed, but the  $R_i$  value for each shift is a function of both  $W_i$  and  $N_i$ ; and the final  $N_i$  value for each shift is dependent on the placement of "unassigned" officers. Hence, using result (3.1), the objective function can be written as:

$$\min_{N_i} z = \left( \sum_{i=1}^S \frac{W_i}{R} - \frac{W_i}{R_i} \right)^2 = \sum_{i=1}^S \left( \frac{W_i}{R} - \frac{W_i}{7N_i - W_i} \right)^2 \quad (3.2)$$

In tables 3.1 and 3.2, the  $R$  and  $W_i$  values are

$$R = 95/45 \approx 2.11,$$

$$W_1 = 33, W_2 = 38, W_3 = 24, \text{ and}$$

the objective function is minimized when one unassigned officer is placed on each shift; i.e.,

$$N_1 = 7, W_1/R_1 = 2.0625;$$

$$N_2 = 9, W_2/R_2 = 2.1111;$$

$$N_3 = 5, W_3/R_3 = 2.1818; \text{ and}$$

$$z = .007.$$

Determination of the  $N_i$  values completes specification of the manpower allocation. The PR schedule design process described in the following sections uses the  $N_i$  and  $w_{ij}$  values computed in this way to identify the manpower allocation that must be preserved by feasible PR schedules.

### 3.2.2 Schedule Design Features

The schedule attributes used in the design procedures described in this thesis were discussed in detail in chapter 2 (see section 2.2.6); the specific preference structure utilized was shown in table 2.8. This structure has been incorporated into the computer code developed

for this thesis and used for all of the schedules presented herein.

In addition to the manpower allocation data discussed above, the data required for use of the computerized procedure are:

- (1) upper ( $U_W$ ) and lower ( $L_W$ ) limits on work period lengths; and
- (2) upper ( $U_R$ ) and lower ( $L_R$ ) limits on recreation period lengths.

For multishift PR schedules, the design data required are:

- (1) a set of candidate schedules for each shift tour to be included in the multishift schedule (this set is described in section 3.5.1), and
- (2) upper ( $U_{CR}$ ) and lower ( $L_{CR}$ ) limits on changeover recreation period lengths.

### 3.3 THE DESIGN OF OPTIMAL ONE-SHIFT PR SCHEDULES

The sequential design of one-shift PR schedules consists of the following steps:

- (1) the specification of distinct sets of recreation periods of acceptable length;
- (2) the distribution of each recreation period set over the days of the week;
- (3) the specification of work periods of acceptable length; and
- (4) the enumeration of feasible sequences of work and recreation periods.

Each of these steps is discussed below.

### 3.3.1 Specification of Sets of Recreation Periods of Acceptable Length

The total number of recreation days which each officer receives during his  $N_i$  weeks on shift  $i$  is given by

$R_i = \sum_{j=1}^7 r_{ij}$ . To produce days-off schedules for individual officers, these  $R_i$  days must be aggregated into recreation periods of lengths which satisfy the upper and lower limits set by the schedule designer. Determining a set of recreation periods, each of which satisfies these limits, is equivalent to finding an unordered partition  $P$  of  $R_i$  such that each member  $\{P_q^k\}$  of the  $k^{\text{th}}$  partition satisfies the requirement:  $L_R \leq P_q^k \leq U_R$ , and that all members of  $P^k$  sum to  $R_i$  (i.e.,  $\sum_q P_q^k = R_i$ ).

To illustrate, consider a manpower allocation for a five-week shift tour containing 11 recreation days (see shift 3 in table 3.2). If the recreation period length limits are  $L_R = 2$  and  $U_R = 4$  respectively, then only four distinct sets (or partitions) of the 11 recreation days exist which satisfy the requirement that  $\sum_q P_q^k = 11$ , and  $2 \leq P_q^k \leq 4$  for each member. The four sets are:

Partition (k=)	Member Period Lengths ( $P_q^k$ )
1	(4, 4, 3)
2	(4, 3, 2, 2)
3	(3, 3, 3, 2)
4	(3, 2, 2, 2, 2)

Although 56 distinct partitions can be created from the 11 recreation days (assuming only that each member is a positive integer less than or equal to 11), only four partitions satisfy the acceptability criteria imposed by the upper and lower limits on period lengths. As a result, each acceptable five-week PR schedule that matches the specified manpower allocations in table 3.2 must use one of these four sets of recreation periods.

The algorithmic enumeration of all partitions for a given  $R_i$  value and a specified set of limits,  $L_R$  and  $U_R$ , is presented in chapter 4.

### 3.3.2 The Distribution of Recreation Periods Over Days of the Week

After the recreation days have been aggregated into periods of acceptable length, each set of periods must be distributed over the days of the week in a way which matches the required daily allocation of recreation days. This distribution must specify the day of the week on which each recreation period will begin.

As an example, consider the second partition derived above which consists of four recreation periods with lengths: {4,3,2,2}; and assume that these four periods must be distributed over the days of the week according to the recreation day allocation shown for shift 3 in table 3.2 (i.e., over the five-week tour, each officer receives a total of 11 recreation days: two each on Monday, Tuesday, Wednesday,

and Sunday, and one each on Thursday, Friday and Saturday).

Following the procedure developed by Heller, the required daily allocation for the 11 recreation days can be schematically represented in a star diagram (see figure 3.1). Each day of the week is represented by a ray, and the number of nodes on each ray corresponds to the number of recreation days each officer receives on that day during the five-week schedule. Recreation periods can be represented by drawing lines to connect nodes on adjacent rays. A star diagram which has its nodes connected into a set of recreation periods is called a cyclic graph. The cyclic graph in figure 3.2 is based on the star diagram in figure 3.1, and consists of one four-day period (Thursday-Friday-Saturday-Sunday), one three-day period (Monday-

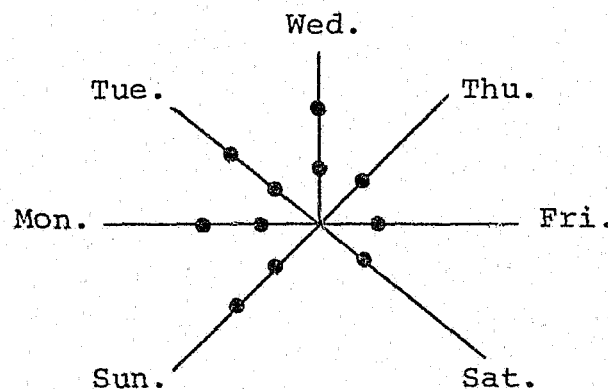


Figure 3.1

Star Diagram with Eleven Nodes

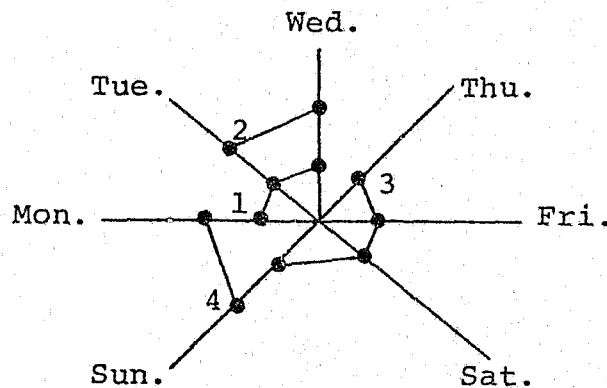


Figure 3.2

Cyclic Graph Based on the Star  
Diagram in Figure 3.1

Tuesday-Wednesday), and two two-day periods (Tuesday-Wednesday and Sunday-Monday); these four periods correspond to the second partition derived above. This cyclic graph defines one distribution of the recreation periods which matches the required daily allocation.

The cyclic graph in figure 3.2, however, may not be the only distribution of the  $\{4,3,2,2\}$  partition that can be formed on the 11-node star diagram in figure 3.1. The exact number of distinct cyclic graphs that can be enumerated from each partition is indeterminate: there may be several or none at all. An enumeration algorithm for generating all distinct cyclic graphs for a given daily allocation of recreation days and a specific partition of recreation days is described in chapter 5.

### 3.3.3 Specification of Work Periods of Acceptable Length

Again following Heller, associated with each cyclic graph is a square matrix  $S$  (called an elementary separation matrix) whose entries  $\{s_{ij}\}$  indicate the minimum number of work days that can occur between each ordered pair of recreation periods,  $i$  and  $j$ , defined in the graph. The number of rows and columns in the matrix equals the number of recreation periods in the graph, and each matrix entry is determined by the graph.

As an example, the elementary separation matrix, associated with the cyclic graph in figure 3.2 is shown in figure 3.3. Since there are four recreation periods in the graph, the matrix has four rows and columns. The (1,4) matrix entry indicates that a minimum of three work days

		Recreation Period			
		1	2	3	4
Recreation Period	1	-	5	0	3
	2	4	-	0	3
	3	0	1	-	6
	4	6	0	2	-

Figure 3.3

Elementary Separation Matrix Based on the  
Cyclic Graph in Figure 3.2

must separate the end of period 1 (Monday-Tuesday-Wednesday) and the beginning of period 4 (Sunday-Monday); the three on-duty days correspond to the three rays in the graph which separate the last day of period 1 (Wednesday) and the first day of period 4 (Sunday). All of the matrix entries can be obtained in a similar manner. The entries on the main diagonal of the matrix are not defined. For cyclic graphs with seven rays, all matrix entries in the elementary matrix lie between zero and six.

An infinite number of modified matrices can be generated from each elementary matrix by using the fact that if ordered periods  $(i,j)$  can be placed  $s_{ij}$  days apart, they can also be placed  $s_{ij} + 7k$ ,  $k = 1, 2, \dots$  days apart (e.g., periods 1 and 4 in figure 3.2 can be 3 days apart, or 3 days plus one week apart (10 days), or 3 days plus two weeks apart (17 days), etc.).

The selection of one value for each matrix entry (and hence the selection of one separation matrix) is accomplished using the upper and lower limits on work period lengths (i.e., each matrix entry is restricted to the interval,  $L_w \leq s_{ij} \leq U_w$ )\*. To illustrate, assume that the work period limits,  $L_w = 4$  and  $U_w = 8$ , are applied to the

---

\* Since consecutive values for each  $s_{ij}$  differ by seven days, the interval  $[L_w, U_w]$  will define only one  $s_{ij}$  value, at most, if  $U_w - L_w < 6$ . It may happen that no  $s_{ij}$  value lies within the interval.

elementary matrix in figure 3.3. The modified matrix obtained with these limits is shown in figure 3.4. Three possibilities exist for each entry in the transition from figure 3.3 to figure 3.4: (1) the entry is unchanged since it lies within the interval  $[4,8]$  (e.g., entries  $(1,2)$ ,  $(2,1)$ ,  $(3,4)$ , and  $(4,1)$ ); (2) the entry must be increased by seven to fall within the interval  $[4,8]$  (e.g., entries  $(1,3)$ ,  $(2,3)$ ,  $(3,1)$ ,  $(3,2)$ , and  $(4,2)$ ); or (3) the entry is voided because none of the values,  $s_{ij} + 7k$  lie in  $[4,8]$  (e.g., entries  $(1,4)$ ,  $(2,4)$ , and  $(4,3)$ ).

		Recreation Period			
		1	2	3	4
Recreation Period	1	-	5	7	-
	2	4	-	7	-
	3	7	8	-	6
	4	6	7	-	-

Figure 3.4

Modified Separation Matrix Based on the  
Cyclic Graph in Figure 3.2

The significance of the modified separation matrix is that it indicates the length of the acceptable work period, if one exists, separating each ordered pair of recreation periods in the cyclic graph. Hence, the modified separation matrix of figure 3.4 identifies the only nine work ~~periods~~ that can be used to produce PR schedules based on the four recreation periods of figure 3.2 and the upper and lower limits on work period lengths,  $U_w$  and  $L_w$ . A more thorough discussion of separation matrices and their use in the final step of designing optimal one-shift PR schedules is presented in chapter 6.

#### 3.3.4 Enumeration of Acceptable Sequences of Work and Recreation Periods

The final step suggested by Heller for the design of optimal one-shift schedules is the specification of acceptable sequences of work and recreation periods based on information contained in a modified separation matrix. Since each schedule produced must include each recreation period exactly once, each distinct sequence (or permutation) of the recreation periods defines a schedule candidate.

As an example, consider the four recreation periods used to construct the matrix of figure 3.4; these four periods define the following six sequences:\*

---

\*The first period selected for cyclic schedules is arbitrary; hence  $n$  periods produce, at most,  $(n-1)!$  sequences. The presence of identical recreation periods (i.e., periods with the same length and starting day) further reduces the number of distinct sequences (see chapter 6 and appendix 10.2).

<u>Sequence No.</u>	<u>Sequence</u>
1	{1, 2, 3, 4}
2	{1, 2, 4, 3}
3	{1, 3, 2, 4}
4	{1, 3, 4, 2}
5	{1, 4, 2, 3}
6	{1, 4, 3, 2}

Each sequence of recreation periods defines a sequence of ordered pairs which identify specific matrix entries; for example, the sequence {1,2,3,4} corresponds to the sequence of ordered pairs: {(1,2), (2,3), (3,4), (4,1)} which identifies the four work period lengths {5,7,6,6}. This particular combination of recreation and work periods produces the five-week PR schedule shown in figure 3.5. The first recreation period in the sequence, period 1, covers Monday, Tuesday, and Wednesday of week 1 followed by the five-day work period that separates periods 1 and 2

		M	T	W	T	F	S	S
Week	1	1 R	R	R				
	2		2 R	R				
	3				3 R	R	R	R
	4							4 R
	5	R						
Number of officers on duty		3	3	3	4	4	4	3

Figure 3.5

Five-Week, One-Shift PR Schedule Based on the Recreation Period Sequence {1,2,3,4}

(matrix entry (1,2))\* . Period 2 covers Tuesday and Wednesday, and is followed by the seven-day work period that separates recreation periods 2 and 3 (matrix entry (2,3)). This process is continued until all four recreation periods have been used as defined by the sequence {1,2,3,4}.

It is useful at this point to note the properties of the schedule in figure 3.5:

- (1) the schedule contains 24 work days aggregated into four work periods of acceptable lengths which match the required daily allocation (see table 3.1, shift 3); and
- (2) the schedule contain 11 recreation days aggregated into four recreation periods of acceptable lengths which match the required daily allocation (see table 3.2, shift 3).

It is also important to note that the sequence of recreation periods {1,2,3,4} produces the schedule in figure 3.5 for the following reasons:

- (1) each ordered pair of recreation periods defined by the sequence specifies an acceptable work period (i.e., each pair of recreation periods identifies a valid entry in the separation matrix); and
- (2) the sum of the lengths of the four work periods defined by the sequence equals the number of work days (i.e., 24) required in the manpower allocation.

If either of these conditions are not satisfied by the schedule produced from a sequence of recreation periods,

---

\*The first recreation period in the sequence is by definition always placed in week 1; any week can be used, however, since the specific labelling of the weeks is arbitrary in a cyclic schedule.

an acceptable PR schedule does not exist for that sequence. In fact, of the six sequences of recreation periods originally identified for the matrix in figure 3.4, only two produce acceptable schedules: sequence 1 (schedule shown in figure 3.5) and sequence 4 (schedule shown in figure 3.6); each of the other sequences includes one or more matrix entries which do not define an acceptable work period length (i.e., the sequence contains a voided matrix entry).

The use of the modified separation matrix to enumerate sequences of recreation periods which yield acceptable schedules is discussed in chapter 6. An analogy between the generation of acceptable PR schedules from a separation matrix and the enumeration of tours for the travelling salesman problem is used to develop an implicit enumeration scheme to systematically search for optimal PR schedules.

	M	T	W	T	F	S	S
1	1 R	R	R				
2				3 R	R	R	R
3							4 R
4	R						
5		2 R	R				
Number of officers on duty	3	3	3	4	4	4	3

Figure 3.6

Five-Week, One-Shift PR Schedule Based on the  
Recreation Period Sequence {1,3,4,2}

### 3.4 OVERVIEW OF THE COMPUTER PROGRAM LOGIC FOR DESIGNING OPTIMAL ONE-SHIFT PR SCHEDULES

A computer program, entitled EXEC and written by the author, incorporates and extends each of the algorithms discussed above for the sequential design of one-shift PR schedules. A flow diagram illustrating the logic of the program is shown in figure 3.7. The four steps identified in the flow diagram correspond to the four procedures discussed above: the partitioning of recreation days into periods (step 1), the generation of cyclic graphs for each partition (step 2), the construction of a modified separation matrix for each graph (step 3), and the use of each matrix to enumerate acceptable PR schedules (step 4). The logic of the EXEC program utilizes the preference structure indicated in table 2.8. The current version of the program does not permit the user to reorder the preference rankings or in any way to modify the preference measures themselves\*.

The computer program, written in FORTRAN IV, was originally tested on an IBM 7040 computer and later modified for use on an IBM 360/65 system. The code has been extensively tested and has been used to design one-shift PR schedules up to nine weeks long. Examples of the program printout are presented in chapter 8.

---

\* Such a generalized program would have necessitated a greater programming effort than was deemed necessary since the primary objective of this thesis was to demonstrate the feasibility of computerized design procedures using a specified preference structure.

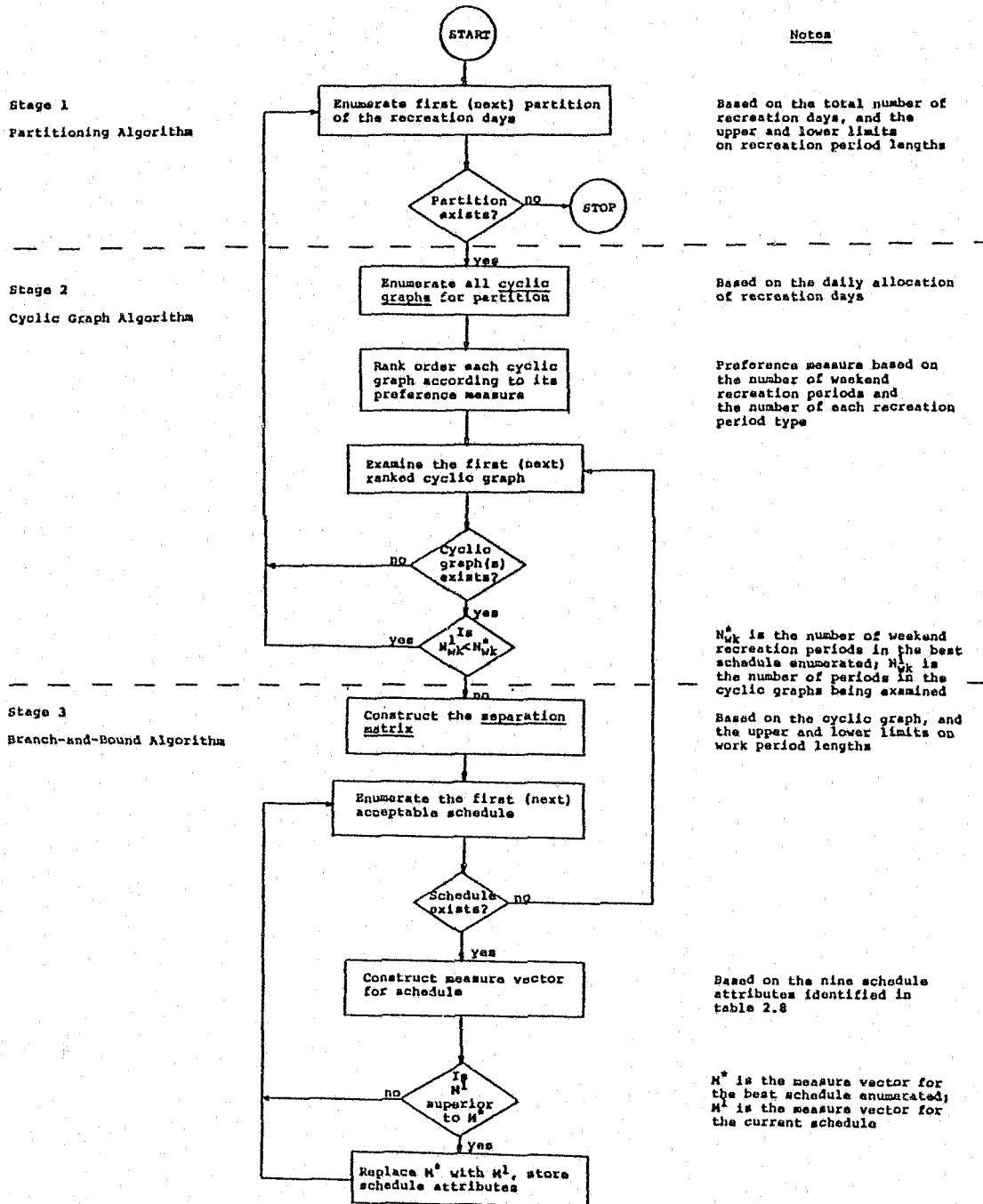


Figure 3.7

Flow Diagram for the Computerized Design of  
Optimal One-Shift PR Schedules  
(EXEC Computer Code)

### 3.5 THE DESIGN OF OPTIMAL MULTISHIFT PR SCHEDULES

A procedure for the design of multishift PR schedules has been developed by the author. It involves a two-step process. First, a small set of non-cyclic, one-shift schedules are designed for each shift tour to be included in the multishift schedule. The non-cyclic, one-shift schedules are constructed by solving a modified cyclic scheduling problem using the design procedures described in section 3.3. In the second step, combinations of the one-shift schedules are systematically examined to determine the most preferable multishift schedule. Each of these steps is discussed below.

#### 3.5.1 Optimal Non-Cyclic, One-Shift PR Schedules

Cyclic schedules, by definition, have no natural beginning and ending point; although the sequence or order of the brackets is important, which bracket is identified as week 1 is not. After each  $n$  weeks (the rotation period of the schedule), each officer has rotated through each bracket once, and none of the characteristics or measures of the schedule are dependent upon bracket labels.

Multishift PR schedules are cyclic over all shift tours (i.e., after  $m$  weeks, the multishift schedule period, each officer has been assigned to each bracket of the schedule for one week), and the labeling of the "first" week of the multishift schedule is arbitrary. The individual shift schedules, however, are not cyclic. Each shift schedule

has a specific beginning and ending bracket; the first or initial week of each shift schedule is defined as the first bracket worked in that schedule after rotating from the previous shift. Similarly, the last or final week of each shift schedule is defined as the last bracket worked before rotating to the next shift. The characteristics of the multishift schedule are dependent upon the specific brackets within each shift schedule which are used to begin and end the shift tour. Changing the first and last brackets of one shift schedule (e.g., by rotating the brackets within that shift tour) may alter the preferability of the multishift schedule.

To illustrate, consider the two one-shift schedules shown in figures 3.8 and 3.9. If used as cyclic schedules, these PR schedules are equivalent; (i.e., have identical properties); the only difference between them is that different labels have been used on the brackets. Each bracket in figure 3.8 can be rotated backwards one week to obtain the schedule in figure 3.9 (bracket 1 becomes bracket 4). These two schedules, however, may not produce equivalent multishift schedules. This may occur because the properties of a multishift schedule are dependent upon the characteristics of the first and last weeks of each shift schedule. As an example, in figures 3.10 and 3.11, the two one-shift schedules are each used for shift A in

	M	T	W	T	F	S	S
1	R	R					
2			R	R			
3					R	R	
4						R	R

Figure 3.8

Four-Week PR Schedule

	M	T	W	T	F	S	S
1			R	R			
2					R	R	
3						R	R
4	R	R					

Figure 3.9

Four-Week PR Schedule

		M	T	W	T	F	S	S
Shift A (figure 3.8)	1	R	R					
	2			R	R			
	3					R	R	
	4						R	R
Shift B (identical to shift B in figure 3.11)	5						R	R
	6				R	R		
	7		R	R				

Figure 3.10

Seven-Week Multishift Schedule Commencing with  
the Four-Week Schedule in Figure 3.8

		M	T	W	T	F	S	S
Shift A (figure 3.9)	1			R	R			
	2					R	R	
	3						R	R
	4	R	R					
Shift B (identical to shift B in figure 3.10)	5						R	R
	6				R	R		
	7		R	R				

Figure 3.11

Seven-Week Multishift Schedule Commencing with  
the Four-Week Schedule in Figure 3.9

a multishift schedule with the same three-week schedule for shift B. Examination of these multishift schedules indicates several differences; the schedule in figure 3.10 has a recreation period at each shift changeover point and contains no work periods that are longer than seven days while the schedule in figure 3.11 does not have a recreation period at either changeover point, and contains one ten-day work period.

The differences between the multishift schedules in figure 3.10 and 3.11 can be more clearly explained if three classes of recreation periods are defined for non-cyclic, one-shift schedules:

- (1) a beginning period that precedes the first work day of a shift;
- (2) an ending period that follows the last work day of a shift; and
- (3) interior periods which lie between the first work day and last work day of the shift.

Let  $b_i$  and  $e_i$  equal the lengths of the beginning and ending recreation periods for shift  $i$  (i.e., these lengths indicate the number of recreation days at the beginning and end of a non-cyclic schedule)\*.

The requirement that a recreation period of acceptable length separate the last work day on shift  $i$  and the first

---

\* If the first day of the shift schedule (Monday) is a work day, the beginning recreation period is defined to have length zero (i.e.,  $b_i=0$ ). Similarly, if the last day of the shift schedule (Sunday) is a work day, the ending recreation period is defined to have length zero (i.e.,  $e_i=0$ ). If both  $b_i$  and  $e_i$  equal zero, every recreation period in the schedule is an interior period.

work day on shift  $i+1$  can be stated in terms of the lengths of the ending and beginning recreation periods for shift  $i$  and  $i+1$ ; i.e.,

$$L_{CR} \leq e_i + b_{i+1} \leq U_{CR}, \quad L_{CR}, U_{CR} \geq 0 \text{ and integer} \quad (3.3)$$

where  $U_{CR}$  and  $L_{CR}$  represent the upper and lower limits on the length of the changeover recreation periods. (These limits are not necessarily the same as the limits,  $U_R$  and  $L_R$ , imposed on the lengths of interior periods.) The limits in (3.3) further imply that

$$0 \leq e_i, b_{i+1} \leq U_{CR} \quad e_i, b_{i+1} \text{ integer} \quad (3.4)$$

If, for convenience, the same upper and limit limits  $U_{CR}$  and  $L_{CR}$  are used for all changeover and interior recreation periods in a multishift schedule (i.e.,  $U_R = U_{CR}$  and  $L_R = L_{CR}$ ), then result (3.4) holds for all beginning and ending recreation periods in the schedule. Hence, a complete statement of the limits on the lengths of all recreation periods in each non-cycle shift schedule is:

(1) beginning period,

$$0 \leq b_i \leq U_{CR} \quad i = 1, 2, \dots, s \quad b_i \text{ integer} \quad (3.5)$$

(2) ending period,

$$0 \leq e_i \leq U_{CR} \quad i = 1, 2, \dots, s \quad e_i \text{ integer} \quad (3.6)$$

(3) interior period(s),

$$L_R \leq l_{ij} \leq U_R \quad i = 1, 2, \dots, s \quad (3.7)$$

$l_{ij}$  integer

where  $l_{ij}$  represents the  $j^{\text{th}}$  interior period in the  $i^{\text{th}}$  shift tour.

Since each non-cyclic shift schedule contains, by definition, only one beginning and one ending recreation period, the lengths of these periods can be used to classify each non-cyclic schedule: that is, let  $(b_i, e_i)$  represent the collection of all non-cyclic schedules for shift  $i$  that have a beginning period equal to length  $b_i$  and ending period equal to length  $e_i$ . The only non-cyclic schedules of interest, however, are those which satisfy conditions (3.5) and (3.6) above; these are the only schedules which can be used in a multishift schedule with limits  $U_{CR}$  and  $L_{CR}$ . The number of such  $(b_i, e_i)$  sets equals  $(U_{CR}+1)^2$ . The corresponding set of two-number pairs is  $\{(0,0), (0,1), \dots, (0, U_{CR}), (1,0), (1,1), \dots, (U_{CR}, U_{CR}-1), (U_{CR}, U_{CR})\}$ .

In chapter 7, a procedure for determining the optimal schedule for each set of non-cyclic schedules having the same  $b_i$  and  $e_i$  values is presented, based on the one-shift design algorithm described in chapter 6. The optimal schedule for each set is determined using the following procedure:

- (1) a corresponding set of cyclic schedules is defined in which each schedule has a one-to-one correspondence to a schedule in the set  $(b_i, e_i)$ ;
- (2) the optimal cyclic schedule in the corresponding set is found using the algorithm described in chapter 6 (and outlined above in section 3.3); and
- (3) the optimal, non-cyclic schedule is defined by the properties of the optimal cyclic schedule obtained from the corresponding set.

The one-to-one correspondence between cyclic schedules in the corresponding set and non-cyclic schedules in each  $(b_i, e_i)$  set is achieved with the addition of an "artificial" recreation period of length zero to each cyclic graph that contains a  $b_i$  length period beginning on Monday and an  $e_i$  length period ending on Sunday. Each augmented graph, in turn, produces an expanded separation matrix which possesses one additional row and column. The artificial recreation period represents the off-shift time between the last Sunday and the first Monday of a non-cyclic shift schedule. The placement of the beginning and ending recreation periods is accomplished by "dedicating" selected matrix entries, which forces these recreation periods to appear immediately after and before the artificial period in each schedule enumerated from the separation matrix.

To illustrate the procedures outlined above, consider the problem of finding the optimal non-cyclic, one-shift

schedule for the  $(b_i, e_i) = (1,1)$  set based on the manpower allocation in table 3.3. (This is the same allocation used in the examples in section 3.3 and corresponds to shift 3 in tables 3.1 and 3.2). The procedure involves the following four steps.

1. Specification of All Sets of Acceptable Schedules

Since the optimal schedule must contain two one-day recreation periods (one at the beginning of the shift and one at the end of the shift), two recreation days are removed from the recreation days to be partitioned. The remaining recreation days are then aggregated into sets or partitions of acceptable lengths for interior recreation

Table 3.3

Daily Manpower Allocation for a Five-Week Schedule

Number of Men	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Total
On Duty	3	3	3	4	4	4	3	24
On Recreation	2	2	2	1	1	1	2	11
Total	5	5	5	5	5	5	5	35

periods. Full partitions are formed by adding the two one-day periods to each set. To illustrate, the nine recreation days to be partitioned from the allocation shown in table 3.3 can be grouped into three sets:

<u>Partition</u>	<u>Period Lengths</u>
1	{4, 3, 2}
2	{3, 3, 3}
3	{3, 2, 3, 2}

Adding the two one-day periods to each set yields three full partitions of the 11 recreation days:

<u>Partition</u>	<u>Period Lengths</u>
1	{4, 3, 2, 1, 1}
2	{3, 3, 3, 1, 1}
3	{3, 2, 2, 2, 1, 1}

## 2. Distribution of the Recreation Periods Over the Days of the Week

As discussed in section 3.3.2, there may be several distinct cyclic graphs associated with each partition; to continue the current example, consider the first partition above: {4,3,2,1,1}. The enumeration of every cyclic graph for this partition (and for every other partition) is simplified by the fact that two of the periods (the two one-day periods) must appear on particular days of the week. As a result, the remaining three periods {4,3,2} must be arranged over the nine nodes that remain in the star diagram after one Sunday and one Monday node have been used. This reduced star diagram is illustrated in

figure 3.12. The two one-day periods are added to each cyclic graph formed on the reduced star diagram to produce each complete graph. One graph for the first partition is shown in figure 3.13.

The dashed line in figure 3.13 represents an artificial recreation period which indicates time spent on other shifts. The artificial period always begins on Monday (the first day of the first week off a shift schedule) and always ends on Sunday (the last day before the beginning of a shift schedule). Regardless of the actual number of off-shift weeks in the schedule (which equals the period of the multishift schedule minus the length of the shift schedule being designed), the artificial period has length zero since it represents time spent on other shifts and does not contribute recreation days to the shift of interest. (The zero length of an artificial recreation period is indicated by the fact that the period does not use or connect any nodes in the graph.)

### 3. Specification of Acceptable Work Periods

The elementary separation matrix associated with the augmented cyclic graph in figure 3.13 is constructed following the procedures described in section 3.3.3. The expanded matrix, containing an additional row and column corresponding to the artificial period, is shown in figure 3.14.

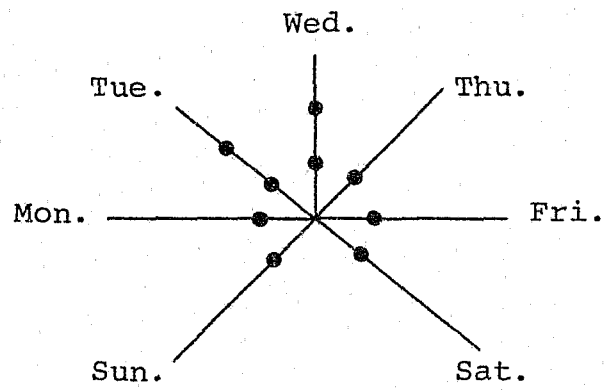


Figure 3.12

Reduced Star Diagram, Nine Nodes

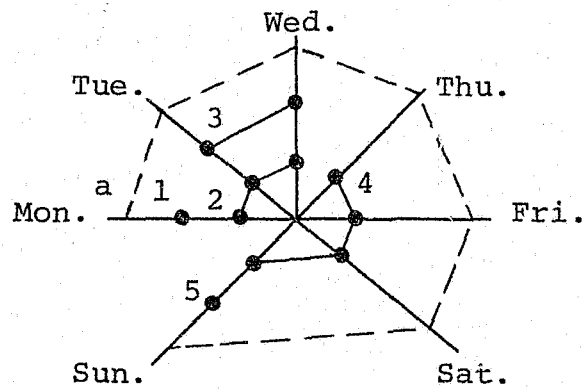


Figure 3.13

Cyclic Graph Based on the {4,3,2,1,1} Partition

		Recreation Period					
		1	2	3	4	5	a
Recreation Period	1	-	6	0	2	5	6
	2	4	-	5	0	3	4
	3	4	4	-	0	3	4
	4	0	0	1	-	6	0
	5	0	0	1	3	-	0
	a	0	0	1	3	6	0

Figure 3.14

Expanded Elementary Separation Matrix Based  
on the Cyclic Graph in Figure 3.13

Since the artificial period represents off-shift time, it is used as a reference period for positioning the beginning and ending recreation periods. The beginning recreation period (period 1) in figure 3.13 must start on the Monday of the first week of the shift schedule; this requirement can be met by placing the beginning period "next to" the end of the artificial period by placing a zero length work period between them (i.e., by setting entry  $(a,1) = 0$ ). In a similar manner, the ending period (period 5 in figure 3.13) can be placed on the last Sunday of the shift schedule by putting a zero length work period between the end of period 5 and the beginning of the artificial period (i.e., by setting entry  $(5,a) = 0$ ).

With these entry values, every sequence of recreation periods that defines an acceptable schedule and uses the entries  $(a,1)$  and  $(5,a)$  will produce a non-cyclic schedule with  $b_i = 1$  and  $e_i = 1$ . The inclusion of these two matrix entries in every sequence can be insured by dedicating the  $(a,1)$  and  $(5,a)$  entries\*. The modified separation matrix obtained by dedicating the  $(a,1)$  and  $(5,a)$  entries, and using the work period length limits  $U_w = 8$  and  $L_w = 4$  on all other entries is shown in figure 3.15. The dedicated entries are circled, and all voided entries are indicated with a dash.

		Recreation Period					
		1	2	3	4	5	a
Recreation Period	1	-	6	7	-	5	-
	2	-	-	5	7	-	-
	3	-	4	-	7	-	-
	4	-	7	8	-	6	-
	5	-	-	-	-	-	0
	a	0	-	-	-	-	-

Figure 3.15

Modified Separation Matrix with Two Dedicated Entries Based on the Cyclic Graph in Figure 3.13

---

\* Entry  $s_{ij}$  is said to be dedicated if every other entry in row  $i$  and column  $j$  is voided; this insures the use of  $s_{ij}$  in every schedule since each sequence of periods must contain one entry from each row and column of the separation matrix.

#### 4. Enumeration of Acceptable Sequences of Work and Recreation Periods

Each sequence of periods enumerated from the modified matrix in figure 3.15 must satisfy the properties described in section 3.3.4 in order to produce an acceptable schedule. These properties are:

- (1) each matrix entry defined in the sequence must represent a work period of acceptable length (excepting dedicated entries); and
- (2) the sum of all work period lengths must equal the total number of required work days for the shift.

Only two sequences of periods from the matrix in figure 3.15 satisfy both of these conditions: {a,1,2,3,4,5}, and {a,1,3,2,4,5}.\* The five-week schedules associated with these sequences are shown in figures 3.16 and 3.17; these schedules not only satisfy the daily manpower allocation specified in table 3.3, and the upper and lower limits imposed on work and recreation periods (interior periods only), but also possess one-day recreation periods at both the beginning and end of the schedule.

The complete enumeration of all non-cyclic schedules with  $b_i = 1$  and  $e_i = 1$  for a given manpower allocation can be obtained by using the procedures described above on each cyclic graph produced from each partition of the recreation

---

\*The first recreation period following the artificial period is, by convention, the first period in the schedule. As a result, every sequence of periods begins with period a.

	M	T	W	T	F	S	S
1	1 R						
2	2 R	R	R				
3		3 R	R				
4				4 R	R	R	R
5							5 R
Number of officers on duty	3	3	3	4	4	4	3

Figure 3.16

Non-Cyclic PR Schedule Based on the  
Recreation Period Sequence {a,1,2,3,4,5}

	M	T	W	T	F	S	S
1	1 R						
2		3 R	R				
3	2 R	R	R				
4				4 R	R	R	R
5							5 R
Number of officers on duty	3	3	3	4	4	4	3

Figure 3.17

Non-Cyclic PR Schedule Based on the  
Recreation Period Sequence {a,1,3,2,4,5}

days. Such an enumeration permits identification of the optimal or dominating schedule for the  $(1,1)$  set.\* Repeating this process for each of the  $(U_{CR}+1)^2$  sets for shift  $i$  produces, at most,  $(U_{CR}+1)^2$  schedules, each optimal over a particular set  $(b_i, e_i)$ . (Some sets may not contain any schedules.) Repeating this procedure for each shift yields a pool of optimal non-cyclic, one-shift schedules from which optimal multishift schedules can be constructed.

### 3.5.2 Optimal Multishift Schedules

Multishift schedules are constructed by selecting one non-cyclic schedule for each shift tour and arranging these schedules in the desired shift rotation sequence. The only acceptability requirement that must be satisfied between each pair of shift schedules,  $i$  and  $i+1$  is:

$$L_{CR} \leq e_i + b_{i+1} \leq U_{CR} \quad (3.8)$$

which insures that a recreation period of acceptable length exists at the shift changeover point between shift tours  $i$  and  $i+1$ . Each sequence of one-shift schedules that satisfies condition (3.8) at every shift changeover is defined to be an acceptable multishift PR schedule.

The total number of sequences of one-shift schedules that must be examined depends upon the number of schedules  $S$  generated for each shift and the number of shift tours  $T$

---

\* Optimal non-cyclic schedules are determined with the same basic measure vector components and preference structure used to determine optimal cyclic schedules.

in each rotation period. The maximum number of sequences to be examined is  $S^T$ , which assumes that a non-cyclic schedule is found for every set  $(b_i, e_i)$  for each shift. Using  $S = (U_{CR}+1)^2$ , the maximum number of sequences,  $S_m$  becomes:

$$S_m = S^T = [(U_{CR}+1)^2]^T = (U_{CR}+1)^{2T} \quad (3.9)$$

Result (3.9) indicates that, except for small values of  $U_{CR}$  and  $T$ , complete enumeration is not reasonable (e.g., a three-shift, one-tour multishift schedule ( $T = 3$ ) with  $U_{CR} = 4$  produces  $(4+1)^6 = 5^6 = 15,625$  sequences to be examined; while a three-shift, two-tour schedule ( $T = 3 \times 2 = 6$ ) produce as many as  $5^{12} \approx 2.4 \times 10^8$  sequences).

The enumeration of all sequences of non-cyclic schedules is avoided by using a strategy of constructing each sequence one shift schedule at a time. The properties of each partial sequence can be used to identify implicitly large sets of multishift schedules which are either unacceptable or undesirable. As a result, only sequences for the most preferable schedules are completely enumerated.

The implicit examination and elimination of both unacceptable and undesirable multishift schedules are achieved by the requirement that before each shift schedule can be added to a partial sequence, three conditions must be satisfied:

- (1) acceptability - the shift schedule to be added must be compatible with the shift schedule that precedes it (i.e., there must be a changeover recreation period of acceptable length); \*
- (2) weekend recreation periods - the new shift schedule must contribute a sufficient number of weekend recreation periods; and
- (3) consecutive working weekends - the new shift schedule must not produce an unacceptable number of consecutive working weekends.

If the shift schedule to be added fails any of these conditions, it is rejected, and an alternate schedule is sought for that tour. If none exists, the process returns to the previous shift tour and replaces that schedule. Each time a shift schedule satisfying all three conditions is found, the schedule is added to the current sequence, and an acceptable schedule is sought for the next shift tour.

Each multishift schedule enumerated is ranked with the preference measures and structure described in chapter 2. A complete description of the enumeration process is presented in chapter 7. Several illustrative multishift schedules are presented in chapter 8.

### 3.6 OVERVIEW OF THE COMPUTER PROGRAM LOGIC FOR ENUMERATING OPTIMAL MULTISHIFT PR SCHEDULES

The enumeration of optimal non-cyclic, one-shift schedules is accomplished with the EXEC program described in section 3.4. To enumerate multishift schedules, a

---

\* Since the completed multishift schedule is cyclic, the last shift schedule added to the sequence must be compatible, with the preceding shift schedule and following shift schedule (i.e., the first schedule in the sequence).

second program, entitled MERGE, is used. A flow diagram indicating the major logic steps of the MERGE program is shown in figure 3.18. Coded in FORTRAN IV and implemented on an IBM 360/65 system, the MERGE program has been successfully used to construct multishift schedules that are 20 weeks long and contain as many as six shift tours. Examples of the computer printout are presented in chapter 8.



**Notes**

W=number of weekend recreation periods  
C=maximum number of consecutive working weekends

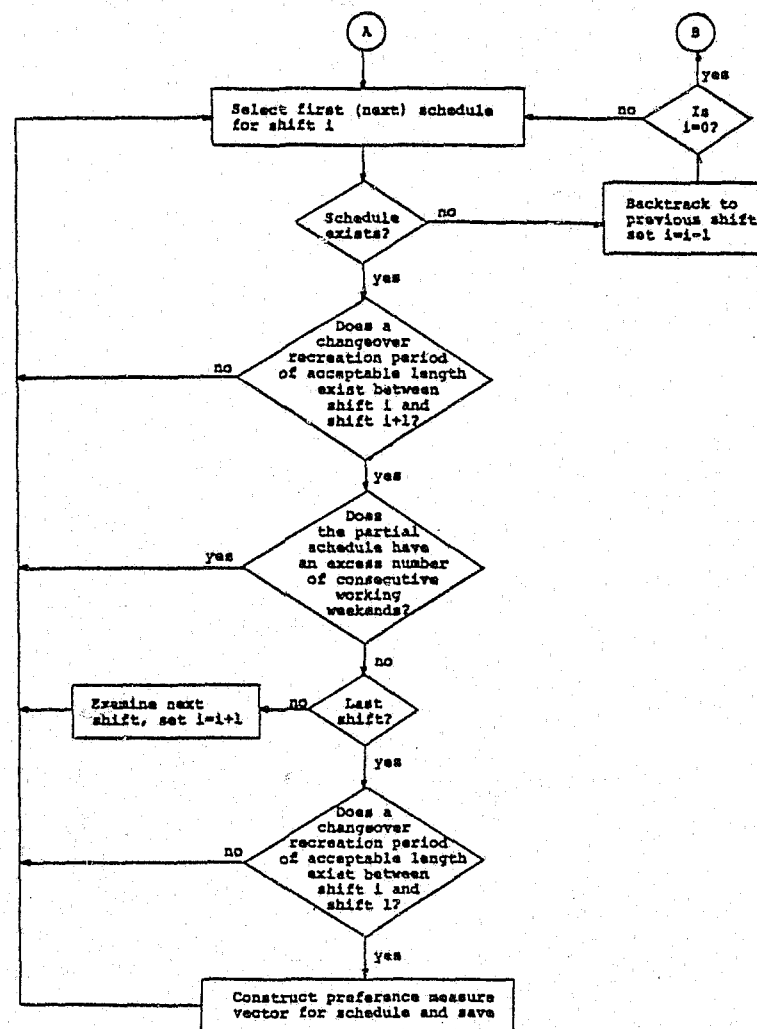
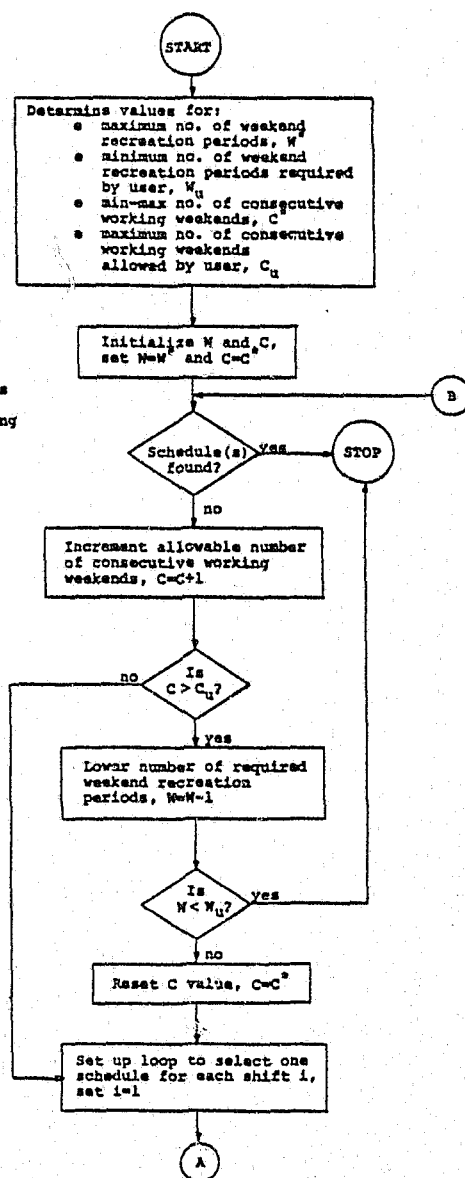


Figure 3.18

Flow Diagram for the Computerized Design of Multishift PR Schedules (MERGE Computer Code)

#### 4. THE ENUMERATION OF ACCEPTABLE RECREATION PERIODS

##### 4.1 INTRODUCTION

This chapter describes an enumeration algorithm to determine distinct partitions of the recreation days allocated to a shift. Each partition consists of clusters of recreation days which can be used as periods of consecutive recreation days in the design of work schedules. The enumeration scheme presented determines all partitions consisting of a given set of period lengths. As an example, eight recreation days can be divided into four unique partitions consisting of periods of only two, three or four days in length; i.e.,

$$\begin{aligned}8 &= 4 + 4 \\8 &= 4 + 2 + 2 \\8 &= 3 + 3 + 2 \\8 &= 2 + 2 + 2 + 2.\end{aligned}$$

Determining all distinct partitions for a total number of recreation days  $R$  is equivalent to finding all vectors  $(n_1, n_2, \dots, n_k)$  such that

$$R = n_1 \ell_1 + n_2 \ell_2 + \dots + n_k \ell_k \quad (4.1)$$

$$\text{integer } n_i \geq 0, i = 1, 2, \dots, k$$

where

$\ell_i$  = number of days in recreation period  
length  $i$ ;  $i = 1, 2, \dots, k$ ; there are  
 $k$  distinct lengths,  $\ell_i > 0$ .

$n_i$  = number of periods of length  $\ell_i$ .

Without loss of generality, the set of lengths can be arranged in decreasing order; i.e.,

$$\ell_1 > \ell_2 > \dots > \ell_k > 0.$$

In the example above,  $k = 3$  and  $\ell_1 = 4$ ,  $\ell_2 = 3$  and  $\ell_3 = 2$ .

The four partitions presented in the example can be presented as the vectors:  $(2,0,0)$ ,  $(1,0,2)$ ,  $(0,2,1)$  and  $(0,0,4)$ .

The set of partitions which satisfies equation (4.1) for  $k$  lengths can be considered as a subset of all permutations of the components of the  $k$ -dimensional row vector  $N = (n_1, n_2, \dots, n_k)$ . In addition, associated with each permutation of the components is a nonnegative number  $d$  defined as

$$d = |R-S| = |R-NL'| \geq 0 \quad (4.2)$$

where

$$L = (\ell_1, \ell_2, \dots, \ell_k).$$

$S$  = equals the number of recreation days given by the vector product of permutation  $N$  and length  $L$ .

Let  $P$  represent the set of all permutations of  $N$  and let  $P_j$  represent the subset of permutations with  $d = j$ . Hence the set of all permutations can be expressed as

$$P = \bigcup_{d=0}^{\infty} P_d.$$

Each permutation which satisfies equation (4.1) represents a partition, which by definition implies that  $R = NL'$  or equivalently that  $d = 0$ , and hence all partitions of  $R$  for a given set of period lengths are members of the subset  $P_0$ . The converse is also true; i.e., all members of  $P_0$  are partitions of  $R$ . As a result, determining the set of partitions which satisfy equation (4.1) is equivalent to finding all members of the subset  $P_0$ .

The enumeration algorithm described below restricts the search for partitions of  $R$  to a small number of  $P_i$  subsets which always includes the subset  $P_0$ . This characteristic of the algorithm ensures its relative efficiency in enumerating all partitions for a given number of recreation days and a specified set of period lengths.

The remainder of this chapter is divided into four sections. The first introduces the basic definitions and procedures on which the partitioning algorithm is based; the important properties of each concept are identified and non-trivial results are proven. The next section outlines each step of the algorithm, presents a detailed example, and describes computational experience with a variety of problems. In the next section, several results from number theory are used to discuss the relative efficiency of the algorithm. In the concluding section, the relationships between several schedule attributes and

various parameters and procedures of the algorithm are identified.

#### 4.2 BASIC CONCEPTS OF THE PARTITIONING ALGORITHM

The algorithm described in this chapter for the enumeration of all partitions for a given number of recreation days and a specified set of period lengths relies upon the following concepts:

- (1) a ranking definition which defines a unique ordering for all permutations of the components of  $N$ , and
- (2) a step-wise procedure for the systematic, implicit examination of each permutation.

Each of these concepts are discussed below.

##### 4.2.1 Ranking Definition

The ordering of all  $k$ -dimensional permutations is easily accomplished with the use of the following definition:

Ranking Definition. Given a specific length vector  $L$ , permutation  $N^+ = (n_1^+, n_2^+, \dots, n_k^+)$  is defined to be greater than  $N^* = (n_1^*, n_2^*, \dots, n_k^*)$  if there exists an  $i'$  such that  $n_i^+ = n_i^*$  for  $i = 1, 2, \dots, i'-1$  and  $n_{i'}^+ > n_{i'}^*$  for  $i = i'$ . For values of  $i > i'$ , the relationships between  $n_i^+$  and  $n_i^*$  are unconstrained. If  $n_i^+ = n_i^*$  for all  $i$ , then  $N^+$  is said to be equal to  $N^*$ .

As an example, using this ranking definition, the four partitions identified above for eight recreation days would have the following order:

$$\begin{array}{ll} N^1 = (2,0,0) & \text{greatest} \\ N^2 = (1,0,2) & \downarrow \\ N^3 = (0,2,1) & \\ N^4 = (0,0,4) & \text{least} \end{array}$$

It should be noted that the ranking definition does not require that the elements of the length vector  $L$  be arranged in any specific order; and in fact, a change in the order of the elements of  $L$  in the example above would produce a new ordering of the four partitions. Regardless, however, of which order of period lengths is used the ranking definition will always produce an unique ordering of all permutations.

The partitioning algorithm enumerates all partitions for a given value of  $R$  and specific  $L$  vector by initiating its search with a permutation which is greater than or equal to the highest ranking permutation which is also a partition. The algorithm then systematically examines a series of permutations, each lower in rank than the one before it, until a permutation is found which is lower in rank than the lowest ranking partition. This systematic or step-wise procedure and the properties of the procedure which insure that all partitions will be found (without the explicit enumeration of all permutations) are described below.

#### 4.2.2 Step-Wise Search Procedure

The step-wise procedure for the implicit enumeration of all permutations involves the alternate use of two distinct operations:

- (1) a "fill" procedure to determine a unique permutation from any given partial permutation (defined below), and
- (2) a backtrack procedure which uses a complete permutation as the basis from which to construct a new partial permutation.

##### 4.2.2.1 Fill Procedure

Given any k-dimensional partial permutation of N (i.e., a permutation designated as  $(n_1, n_2, \dots, n_p, \cdot, \cdot, \dots, \cdot)$ , in which only the first p elements (where p is any non-negative integer less than k) are specified), a unique complete permutation can be found by "filling" the remaining k-p elements,  $n_{p+1}, \dots, n_k$ , with

$$n_k = \max \left\{ \left\lfloor \frac{R-S_{i-1}}{l_i} \right\rfloor, 0 \right\} \quad i = p+1, p+2, \dots, k \quad (4.3)$$

where, as previously defined,

R = total number of recreation days  
to be partitioned

$l_i$  =  $i^{\text{th}}$  element of the length vector L

$S_{i-1} = \sum_{j=1}^{i-1} l_j n_j$  = "i-1 partial sum"

$$\left\lfloor \frac{R-S_{i-1}}{l_i} \right\rfloor = \text{greatest integer} \leq \frac{R-S_{i-1}}{l_i}$$

As an example, let  $R = 14$  and  $L = (4, 3, 2)$ . If a partial permutation is given with  $n_1 = 1$ , represented as  $(1, \cdot, \cdot)$ , the fill procedure produces

$$n_2 = \left\lfloor \frac{14-4}{3} \right\rfloor = 3$$

and

$$n_3 = \left\lfloor \frac{14-13}{2} \right\rfloor = 0.$$

The resulting permutation  $(1, 3, 0)$  is a member of the subset  $P_1$  (i.e.,  $d = |R-S| = |14-13| = 1$ ) and, as a result, is not a partition of  $R$ .

The fill procedure has two important properties:

- (1) it can be used to determine an initial permutation for the stepwise procedure which is greater than or equal to the highest ranking partition for a given  $R$  and  $L$ ; and
- (2) used with the proper class of partial permutations, the procedure will produce only permutations which belong to  $P_i$  subsets which satisfy  $0 \leq i \leq k-1$  (i.e., the procedure will explicitly enumerate only partitions and "near partitions").

Both of these properties are discussed below.

#### Fill Property 1

If the fill procedure is used to complete the  $k$ -dimensional partial permutation with  $p = 0$ , the resulting permutation is greater than or equal to the highest ranking partition for a given  $R$  and  $L$ .

Proof:

The proof is by contradiction. Let  $N'$  represent the completed permutation determined from the  $p = 0$  partial permutation, with equation (4.3), and let  $N^*$  represent a partition that is greater than  $N'$  (i.e., that has a higher rank). Beginning with  $i = 1$ , examine the corresponding  $n_i$  components for both permutations. Since  $N^* > N^1$ ,  $n_i^*$  must either be greater than or equal to  $n_i^1$ . The following argument will show that  $n_i^* = n_i^1$ . The  $n_i^1$  component, determined from equation (4.3) implies that

$$n_i^1 = \left\lceil \frac{R-S_0}{\ell_1} \right\rceil \leq \frac{R-S_0}{\ell_1} + 1 < n_i^1 + 1.$$

If  $n_i^*$  is greater than  $n_i^1$ , then  $n_i^* \geq n_i^1 + 1$

and

$$\frac{R-S_0}{\ell_1} < n_i^1 + 1 \leq n_i^*$$

hence

$$\frac{R-S_0}{\ell_1} < n_i^*$$

and

$$R < S_0^* + n_i^* \ell_1$$

or

$$R < S_1^* \tag{4.4}$$

Since  $N^*$  is a partition (i.e.,  $S_k^* = R$ ), and since  $S_0^* \leq S_1^* \leq \dots \leq S_k^*$ , result (4.4) is a contradiction. Hence  $n_i^*$  cannot be greater than  $n_i^1$ . Repeating this argument for  $i = 2, 3, \dots, k$  indicates that  $n_i^*$  must equal  $n_i^1$  for all  $i$ , and hence  $N^*$  cannot be greater than  $N^1$ . This result,

however, contradicts the initial assumption that  $N^* > N^1$  and the property is proven.

### Fill Property 2

If the fill procedure is used to complete a  $k$ -dimensional partial permutation ( $p$  components specified) with partial sum  $S_p \leq R$ , then the sum  $S_k$  of the components of the  $p$  complete partition satisfies the condition

$$0 \leq R - S_k \leq \min\{R - S_p, \ell_k - 1\}.$$

Proof:

The proof is divided into two parts. The first part establishes that  $0 \leq R - S_k$  and the second verifies that  $R - S_k \leq \min\{R - S_p, \ell_k - 1\}$ .

(1) First show that  $0 \leq R - S_{p+1}$ .

$$\text{Let } n'_{p+1} = \frac{R - S_p}{\ell_{p+1}}, \text{ where } n'_{p+1} \geq 0$$

and may be non-integer.

Hence

$$\begin{aligned} R - S_p &= n'_{p+1} \ell_{p+1} \\ R &= S_p + n'_{p+1} \ell_{p+1} \end{aligned}$$

Now by definition

$$\left\lfloor \frac{R - S_p}{\ell_{p+1}} \right\rfloor \leq \frac{R - S_p}{\ell_{p+1}}$$

$$\text{or } n_{p+1} \leq n'_{p+1}$$

$$n_{p+1} \ell_{p+1} \leq n'_{p+1} \ell_{p+1}, \ell_{p+1} \geq 0$$

$$S_p + n_{p+1} \ell_{p+1} \leq S_p + n'_{p+1} \ell_{p+1}$$

$$S_{p+n_{p+1}} \leq R$$

$$S_{p+1} \leq R$$

$$\text{and } 0 \leq R - S_{p+1}.$$

(2) Now establish that  $R - S_k \leq \min\{R - S_p, l_{k-1}\}$

(a) By definition

$$S_k = \sum_{j=1}^k n_j l_j = \sum_{j=1}^p n_j l_j + \sum_{j=p+1}^k n_j l_j$$

$$S_k = S_p + \sum_{j=p+1}^k n_j l_j$$

Hence

$S_k \geq S_p$  since  $n_j l_j \geq 0$  for  
 $j = p+1, p+2, \dots, k$  and

$$R - S_k \leq R - S_p.$$

(b) Again by definition

$$\left[ \frac{R - S_{k-1}}{l_k} \right] \leq \frac{R - S_{k-1}}{l_k}$$

$$\left[ \frac{R - S_{k-1}}{l_k} \right] + I_k = \frac{R - S_{k-1}}{l_k}, \quad 0 \leq I_k < 1$$

$$n_k + I_k = n_k' \quad (n_k' \text{ may be non-integer})$$

$$n_k l_k + I_k l_k = n_k' l_k$$

$$S_{k-1} + n_k l_k + I_k l_k = S_{k-1} + n_k' l_k$$

$$S_k + I_k l_k = R$$

$$R - S_k = I_k l_k$$

$$R - S_k < l_k.$$

Since both  $R$  and  $S_k$  are integers

$$R - S_k \leq l_k - 1.$$

Hence the results of (a) and (b) produce

$$R - S_k \leq \min\{R - S_p, l_{k-1}\}.$$

Property 2 is particularly important because  $d = R - S_k$ , and hence when any partial permutation with  $S_p \leq R$  is filled using the procedure described above, the final permutation belongs to a subset of permutation with  $0 \leq d \leq \ell_k - 1$ . For example, if  $\ell_k = 2$ , every complete permutation (obtained from a partial permutation with  $S_p \leq R$ ) belongs to either the  $P_0$  or  $P_1$  subsets. If  $\ell_k = 1$ , every filled permutation which starts from a partial permutation with  $R - S_p \geq 0$  belongs to  $P_0$  and is a partition.

#### 4.2.2.2 Backtrack Procedure

Given any full permutation  $N^1$ , this procedure uses  $N^1$  to find a new partial permutation which when completed, as described above, yields a new permutation  $N^2$ , with two important properties:

- (1)  $N^2$  is lower in rank than  $N^1$ , and
- (2) no permutation  $N^3$  belonging to the set  $P_0$  exists which satisfies the condition  $N^1 > N^3 > N^2$ ; i.e., no partition will have been passed over in stepping from  $N^1$  to  $N^2$ .

Using any permutation  $N^1 = (n_1, n_2, \dots, n_k)$ , a new permutation  $N^2$  satisfying the conditions above can be obtained in the following manner:

- (1) Examine the components of  $N^1$  in reverse order (i.e.,  $n_k, n_{k-1}, n_{k-2}, \dots$ ). If all  $n_i = 0$ , stop. ( $N^1$  is already the minimum ranking permutation.) If one or more  $n_i \geq 0$ , continue with step (2).

- (2) Find  $i = p$  for the first non-zero entry such that  $n_p > 0$  and  $n_i = 0$  for  $i = p+1, p+2, \dots, k$ .
- (3) Define a partial permutation with  $p$  components specified from  $N^1$  by using  $n_i$  for  $i = 1, 2, \dots, p-1$ , and  $n_p - 1$  for the  $p^{\text{th}}$  entry.
- (4) Find the  $N^2$  permutation by filling the partial permutation produced in step (3) using the fill procedure described above.

Both backtrack properties are discussed below.

#### Backtrack Property 1

Given any complete permutation  $N^f$ , use of the backtrack and fill procedures will produce a second permutation  $N^{f+1}$  which is lower in rank than  $N^f$ .

Proof:

By construction  $N^f$  and  $N^{f+1}$  have identical entries for  $n_i$ ,  $i = 1, 2, \dots, p-1$ . Also by construction  $n_p^{f+1} = n_p^f - 1$  which by the ranking definition implies that  $N^{f+1} < N_k^f$  regardless of what values are assigned to the remaining  $k-p$  components of  $N^{f+1} \{n_{p+1}^{f+1}, n_{p+2}^{f+1}, \dots, n_k^{f+1}\}$ .

#### Backtrack Property 2

Given any permutation  $N^1$  and the next lower ranking permutation  $N^2$  determined with the backtrack and fill procedures described above, no permutation  $N^3$  exists which is a member of the set  $P_0$  (i.e., a partition) and satisfies the condition  $N^1 > N^3 > N^2$ .

Proof:

To verify that no partition exists between  $N^1$  and  $N^2$ , assume that such a permutation  $N^3$  can exist and consider its construction on a component by component basis:

- (a) Since the first  $p-1$  terms of  $N^1$  and  $N^2$  are identical (i.e.,  $n_i^1 = n_i^2$  for  $i = 1, 2, \dots, p-1$ ), in order for the inequality  $N^1 > N^3 > N^2$  to hold, the first  $p-1$  terms of  $N^3$  must be  $n_i^3 = n_i^1 = n_i^2$ ,  $i = 1, 2, \dots, p-1$ .
- (b) For the  $n_p^3$  component, the entry must be  $n_p^3 = n_p^2 = n_{p-1}^1$  since
  - (i) if  $n_p^3 > n_p^1$ , then  $N^3$  would be greater than  $N^1$ , which would contradict the initial assumption  $N^1 > N^3$ ,
  - (ii) if  $n_p^3 = n_p^1$ ,  $N^3$  will never be less than  $N^1$  since  $n_i^1 = 0$  for  $i = p+1, p+2, \dots, k$ , and
  - (iii) if  $n_p^3 < n_{p-1}^1$ ,  $N^3$  would be less than  $N^2$  since by construction  $n_p^2 = n_{p-1}^1$  which would contradict the initial assumption  $N^3 > N^2$ .

- (c) The specification of the first  $p$  entries in  $N^3$  insures that  $N^3 < N^1$  since  $n_i^3 = n_i^1$  for  $i = 1, 2, \dots, p-1$  and  $n_p^3 < n_p^1$ . No relative ranking can be determined yet for  $N^2$  and  $N^3$  since  $n_i^3 = n_i^2$  for  $i = 1, 2, \dots, p$ . In determining the value for each remaining component of  $N^3$ , beginning with  $n_{p+1}^3$ , each term must be set equal to either  $n_i^3 = n_i^2$  or  $n_i^3 > n_i^2$ . Once the first  $n_i^3 > n_i^2$  assignment can be made, the ranking  $N^3 > N^2$  is insured.
- (d) A restriction on the construction of the  $N^3$  partition, however, is the requirement that  $S_i \leq R$  for all  $i$  in  $N^3$ . If  $S_i > R$  for any  $i$ , a valid partition (i.e.,  $S_k = R$ ) cannot be obtained; e.g., if  $S_i > R$ , then

$$S_k = S_i + \sum_{i=i'+1}^k n_i \ell_i,$$

$S_i \geq S_i$ , ( $\ell_i > 0$ ,  $n_i \geq 0$ , all  $i$ )  
and  $S_k > R$ , indicating that  $N_3$  is not a partition.

(e) First consider the assignment  $n_{p+1}^3 > n_{p+1}^2$ ;  
i.e.,  $n_{p+1}^3 = n_{p+1}^2 + h$  ( $h = 1, 2, \dots$ ). To  
show that this assignment leads to a  
contradiction for any  $h$  requires the  
following inequality based on  $N^2$ :

$$R - S_{p+1}^2 - l_{p+1} < 0.$$

(i) Since every term in  $N^2$  was derived using  
the fill procedure, each  $S_p$  sum must  
satisfy

$$\frac{R - S_p^2}{l_{p+1}} \geq \left\lceil \frac{R - S_p^2}{l_{p+1}} \right\rceil$$

$$\frac{R - S_p^2}{l_{p+1}} = \left\lceil \frac{R - S_p^2}{l_{p+1}} \right\rceil + I_{p+1}, \quad 0 \leq I_{p+1} < 1$$

$$n_{p+1}^{2'} = n_{p+1}^2 + I_{p+1} \quad (n_{p+1}^{2'} \text{ may be non-integer})$$

$$n_{p+1}^{2'} l_{p+1} = n_{p+1}^2 l_{p+1} + I_{p+1} l_{p+1}$$

$$S_p^2 + n_{p+1}^{2'} l_{p+1} = S_p^2 + n_{p+1}^2 l_{p+1} + I_{p+1} l_{p+1}$$

$$R = S_{p+1}^2 + I_{p+1} l_{p+1}$$

$$R - S_{p+1}^2 = I_{p+1} l_{p+1}$$

$$R - S_{p+1}^2 < l_{p+1}$$

and

$$R - S_{p+1}^2 - l_{p+1} < 0. \quad (4.5)$$

(ii) Now examine the result of setting

$$n_{p+1}^3 = n_{p+1}^2 + h.$$

$$n_{p+1}^3 l_{p+1} = n_{p+1}^2 l_{p+1} + h l_{p+1} \\ (h = 1, 2, \dots)$$

$$S_p^3 + n_{p+1}^3 l_{p+1} = S_p^2 + n_{p+1}^2 l_{p+1} \\ + h l_{p+1}, \quad (S_p^3 \text{ equals } S_p^2 \\ \text{by construction})$$

$$S_{p+1}^3 = S_{p+1}^2 + h l_{p+1}$$

$$R - S_{p+1}^3 = R - S_{p+1}^2 - h l_{p+1} \quad (4.6)$$

Using inequality (4.5) from above, on the right side of (4.6)

$$R - S_{p+1}^2 - h l_{p+1} \leq R - S_{p+1}^2 - l_{p+1} < 0 \\ (h > 0, l_i > 0)$$

which leads to

$$R - S_{p+1}^3 < 0$$

or

$$S_{p+1}^3 > R.$$

This result violates the requirement that

$S_i \leq R$  for all  $i$  (see (d) above). Hence

$n_{p+1}^3$  must be set equal to  $n_{p+1}^2$ . Repeating

this same argument for  $i = p+2, p+3, \dots, k$

indicates that  $n_i^3$  must equal  $n_i^2$  for

$i = p+1, p+2, \dots, k$  which leads to  $N^3 = N^2$ ,

a contradiction of the initial condition

that  $N^3 > N^2$ . Hence, no partition  $N^3$  exists

such that  $N^1 > N^2 > N^3$ .

#### 4.3 PARTITIONING ALGORITHM LOGIC

The basic concepts introduced above are used in an enumeration algorithm to generate all partitions for a given number of recreation days  $R$  and a fixed set of period lengths  $L$ . The basic steps of the algorithm are:

- Step 1. Arrange the components of the length vector  $L$  in decreasing order.
- Step 2. Find the first permutation using the fill procedure for the  $p = 0$  partial permutation. If the initial permutation is a partition, save the result.
- Step 3. Use the backtrack procedure to find a new partial permutation. Begin the search for the first non-zero entry at  $i = k-1$ . (The  $i = k$  term can be ignored since  $S_k \leq R$  for all filled permutations, and reducing  $n_k$  by 1 cannot produce a new  $S'_k = R$ .) If no non-zero entries exist for  $i = k-1, k-2, \dots, 1$  the process is terminated.
- Step 4. Fill the partial permutation. If the result is a partition, save the answer. Return to Step 3.

As an example, consider partitioning eight recreation days ( $R = 8$ ) into periods of length four, three, or two days ( $L = (4, 3, 2)$ ).

1. Step 1. Order the  $\ell_i$ 's,  $L = (4, 3, 2)$ .
2. Step 2. Find the first permutation:

$$n_1 = \frac{R-S_0}{\ell_1} = \frac{8-0}{4} = 2,$$

$$n_2 = \frac{R-S_1}{\ell_2} = \frac{8-8}{3} = 0,$$

$$\text{and } n_3 = \frac{R-S_2}{\ell_3} = \frac{8-8}{2} = 0.$$

Hence,  $N^1 = (2, 0, 0)$ . Since  $d = R - S_3 = 8 - 8 = 0$  for this example,  $N^1$  is a member of  $P_0$  and a partition of  $R$ . (Note that if  $S_j = R$ , for  $j < k$ , then  $n_i = 0$ , for  $i = j+1, j+2, \dots, k$ .)

3. Step 3. Use the backtrack procedure on  $N^1$  to find a partial permutation. Since the first non-zero entry is  $n_1$ , it is reduced by one to create the partial permutation  $(1, \cdot, \cdot)$ .
4. Step 4. Fill the partial permutation to obtain  $N^2 = (1, 1, 0)$ . Since  $S_3 = 7 \neq R$ ,  $N^2$  is not a partition.
5. Step 5. The next partial permutation is  $(1, 0, \cdot)$ .

6. Step 4. The fill procedure produces a third permutation  $N^3 = (1,0,2)$  which is a partition ( $S_3 = 1(4)+0(3)+2(2) = 8 = R$ ).

Steps 3 and 4 are continued until permutation  $N^2 = (0,0,4)$  is obtained. Since no non-zero entries exist for  $i = k-1, k-2, \dots, 1$ , the algorithm is terminated. A summary of this example is presented in table 4.1.

Table 4.1

Permutations Enumerated to Partition Eight Recreation Days ( $R=8$ ) into Periods of Four, Three, and Two Days in Length ( $L=(4,3,2)$ )

Permutation Number	Partial Permutation	Full Permutation	Subset $P_d$ ( $d=R-S_3$ )*
1.	(.,.,.)	(2,0,0)	$P_0$
2.	(1,.,.)	(1,1,0)	$P_1$
3.	(1,0,.)	(1,0,2)	$P_0$
4.	(0,.,.)	(0,2,1)	$P_0$
5.	(0,1,.)	(0,1,2)	$P_1$
6.	(0,0,.)	(0,0,4)	$P_0$

\* $S_3 = 4n_1 + 3n_2 + 2n_3$  where  $n_i$  equals the number of recreation periods of length  $\ell_i$ .

Table 4.2 summarizes the results of applying the algorithm to several problems with different values for R and L. The efficiency factor noted in the table is discussed in section 4.4.

#### 4.4 EFFICIENCY OF THE PARTITIONING ALGORITHM

The partitioning algorithm described in section 4.3 can be used to enumerate all partitions for a given number of recreation days and a fixed set of recreation period lengths. Whether it is desirable or feasible to enumerate all partitions depends upon the relative efficiency of the algorithm and the total number of partitions that exist for a given set of initial conditions. If the effort required to produce each partition is very high or if the total number of partitions is extremely large, it may be necessary to introduce a screening process which can isolate partitions which yield preferable schedules.

##### 4.4.1 Measure of Relative Efficiency

One measure of the relative efficiency of the partitioning algorithm is shown in table 4.2. The efficiency measure is defined to be the ratio of the total number of partitions found to the total number of permutations generated by the algorithm. The ratio serves as a direct measure of the relative effort required to find each partition. A high ratio indicates that most of the permutations generated are partitions. In table 4.2, the

Table 4.2

Performance Statistics for Several Examples of the Partitioning Algorithm

	Total Number of Recreation Days				
	8	16	26	52	104
Length Vector $L=(4,3,2)$					
(1) Total number of partitions (class $P_0$ )	4	10	21	70	252
(2) Total number of permutations (classes $P_0$ and $P_1$ )	6	17	37	131	486
(3) Efficiency=(1)/(2)	.667	.588	.568	.534	.519
Length Vector $L=(6,5,4,3)$					
(1) Total number of partitions (class $P_0$ )	2	7	19	104	666
(2) Total number of permutations (classes $P_0, P_1$ , and $P_2$ )	5	19	54	299	1,949
(3) Efficiency=(1)/(2)	.400	.368	.352	.348	.342

efficiency measure for  $L = (4,3,2)$  declines from .667 for  $R = 8$  to only .519 for  $R = 104$ . For  $L = (6,5,4,3)$ , the efficiency measure declines from .400 to .342 as  $R$  is increased from 8 to 104. These results indicate that the algorithm is more efficient for low values of  $l_k$ .\*

#### 4.4.2 Upper Bounds on the Total Number of Partitions

To determine the total number of partitions for a given set of initial conditions, several results from combinatorics are useful. In number theory a partition of a positive integer  $R$  is a representation of  $R$  as a sum of positive integers, i.e.,

$$R = x_1 + x_2 + \dots + x_l, \quad x_i > 0 \\ i = 1, 2, \dots, l.$$

To discuss the number of partitions, it is necessary to distinguish between ordered and unordered partitions. The number of unordered partitions refers to the number of distinct sets of positive integers which sum to a given positive integer. The arrangement or order of the integers within each set is not distinguishable. The number of ordered partitions refers to the sum of the number of distinct arrangements that can be made with each set of positive integers that sum to a given



**CONTINUED**

**2 OF 6**

positive integer. As an example, there are five unordered partitions (or sets of integers) which sum to four:  $\{4\}, \{3,1\}, \{2,2\}, \{2,1,1\}, \{1,1,1,1\}$ . These five sets, however, can be used in eight distinct arrangements (i.e., eight ordered partitions); the eight are:

$$\begin{aligned} 4 &= 4 \\ 4 &= 3+1 = 1+3 \\ 4 &= 2+2 \\ 4 &= 2+1+1 = 1+2+1 = 1+1+2 \\ 4 &= 1+1+1+1 \end{aligned}$$

The number of ordered partitions is easily found; i.e., to divide  $R$  into  $\ell$  ordered parts is equivalent to the number of ways of putting  $(\ell-1)$  markers into the  $(R-1)$  spaces between  $R$  dots (91). This number is  $\binom{R-1}{\ell-1}$ .<sup>\*</sup> For the total number of ordered partitions, (i.e., into any number of parts), a marker may or may not be placed in each of the  $R-1$  spaces. This yields a total of  $2^{R-1}$  ordered partitions.

Considerable work has been done on the problem of counting the number of unordered partitions. Let  $p_{\ell}(R)$  designate the number of unordered partitions of  $R$  into  $\ell$  parts. To determine  $p_{\ell}(R)$  for small values of  $\ell$  and any value of  $R$ , the following recursive formula can be used (92):

---

<sup>\*</sup>As an example, the integer 6 can be represented with 5 ordered partitions, each containing exactly two parts; i.e.,  $\binom{6-1}{2-1} = 5$ :  $5+1$ ,  $4+2$ ,  $3+3$ ,  $2+4$ , and  $1+5$ . There are only three unordered partitions of 6 with exactly two parts:  $\{5,1\}, \{4,2\}$ , and  $\{3,3\}$ .

$$p_\ell(R) = p_\ell(R-\ell) + p_{\ell-1}(R-\ell) + \dots + p_1(R-\ell) \quad (4.7)$$

The initial conditions are  $p_\ell(R) = 0$  for  $R < \ell$  and  $p_\ell(\ell) = 1$  since the only way to represent  $\ell$  as a sum of  $\ell$  positive integers is to write it as the sum of  $\ell$  1's. Formula (4.7) can be used to derive a table  $p_\ell(R)$ 's as shown in table 4.3.

Table 4.3

Values of  $P_\ell(R)$

Number of Parts ( $\ell$ )	Integer Value (R)							
	1	2	3	4	5	6	7	8
1	1	1	1	1	1	1	1	1
2	0	1	1	2	2	3	3	4
3	0	0	1	1	2	3	4	5
4	0	0	0	1	1	2	3	5
5	0	0	0	0	1	1	2	3
6	0	0	0	0	0	1	1	2
7	0	0	0	0	0	0	1	1
8	0	0	0	0	0	0	1	1

Each partition of  $R$  can be schematically illustrated with a diagram consisting of a row of dots for each part, putting the largest part at the top and the rest in decreasing size below it. Hence  $15 = 5+5+3+2$  can be diagrammed

$$\begin{array}{rcccccc}
 \cdot & \cdot & \cdot & \cdot & \cdot & 5 \\
 \cdot & \cdot & \cdot & \cdot & \cdot & + 5 \\
 \cdot & \cdot & \cdot & & & + 3 \\
 \cdot & \cdot & & & & + 2 \\
 \hline
 4 & + & 4 & + & 3 & + & 2 & + & 2 & = & 15 .
 \end{array}$$

Associated with each diagram is another partition consisting of parts represented by the columns rather than the rows. In the diagram above, the second partition consists of  $15 = 4+4+3+2+2$ . Two partitions related in this manner are called conjugates of each other. It is easily seen that a one-to-one correspondence exists between each pair of conjugate partitions. This property leads to the following result:

Theorem 4.1

The number of partitions of an integer  $R$  into  $\ell$  parts is equal to the number of partitions of  $R$  into parts the greatest of which is  $\ell$ .

Proof:

The conjugate of each partition with  $\ell$  parts is a partition whose greatest part is  $\ell$ . The one-to-one correspondence between each pair of conjugates establishes the theorem (92).

This theorem adds a dual meaning to each  $p_{\ell}(R)$ . In addition to representing the number of unordered partitions of  $R$  into  $\ell$  parts, it also equals the number of unordered partitions of  $R$  in which the greatest is  $\ell$ . As an example, in table 4.3,  $p_{\ell}(7)$  indicates that there are four partitions of 7 with exactly three parts; i.e.,

$$\begin{aligned} 7 &= 5 + 1 + 1 \\ 7 &= 4 + 2 + 1 \\ 7 &= 3 + 3 + 1 \\ 7 &= 3 + 2 + 2 . \end{aligned}$$

It also indicates that there are four partitions of 7 with 3 as the greatest part; i.e.,

$$\begin{aligned} 7 &= 3 + 3 + 1 \\ 7 &= 3 + 2 + 2 \\ 7 &= 3 + 2 + 1 + 1 \\ 7 &= 3 + 1 + 1 + 1 + 1 . \end{aligned}$$

This second meaning for each  $p_{\ell}(R)$  can be used to establish upper bounds on the number of ways that  $R$  recreation days can be partitioned into recreation periods of lengths  $L = (\ell_1, \ell_2, \dots, \ell_k)$ . Let each unordered partition of  $R$  be placed in set  $R_{\ell}^G$  if the greatest part of the partition equals  $\ell$ ; as a result, the set of all partitions of  $R$  can be expressed as the sum of the sets  $R_{\ell}^G$  for  $\ell = \ell_1, \ell_2, \dots, \ell_k$ . The maximum number of partitions in each set  $R_{\ell}^G$  is given by  $p_{\ell}(R)$ , and the maximum total number of partitions of  $R$  for all values of  $\ell$  equals  $\sum_{\ell_i} p_{\ell}(R)$ .

Hence an upper bound  $B(R, L)$  on the total number of ways that  $R$  recreation days can be divided into the lengths in  $L = (\ell_1, \ell_2, \dots, \ell_k)$  is given by

$$B(R, L) = \sum_{\text{all } \ell_i} p(\ell_i) . \quad (4.8)$$

As an example, if  $R = 8$  and  $L = (4, 3, 2)$

$$\begin{aligned} B &= p_4(8) + p_3(8) + p_2(8) \\ &= 5 + 5 + 4 \quad (\text{from table 4.3}) \\ B &= 14. \end{aligned}$$

The actual number of partitions for  $R = 8$  and  $L = (4, 3, 2)$  is four (see table 4.2). The  $p_{\ell}(R)$  value for the number of partitions for each set  $R^G_{\ell}$  assumes that all lengths less than or equal to  $\ell$  are available for use in the partitions. If the  $L$  vector does not contain every length less than  $\ell$ , each  $p_{\ell}(R)$  value used will overestimate the actual number of partitions and as a result, the  $B(R, L)$  value will be an overestimate. In the special case when the  $k$ -dimensional vector  $L$  does contain all lengths from 1 to  $k$ , each  $p_{\ell}(R)$  value indicates the exact number of partitions and the resulting  $B(R, L)$  value yields an exact count of the total number of partitions.

Since one-day recreation periods are frequently not used, it is also useful to derive an estimating procedure to determine the number of partitions of  $R$  with parts that

are less than or equal to  $\ell$  and greater than one. Let  $p_{\ell}^*(R)$  represent the number of such partitions for a given  $R$  and  $\ell$ . The determination of  $p_{\ell}^*(R)$  is quite simple with the help of the following result:

Theorem 4.2

$$p_{\ell}^*(R) = p_{\ell}(R) - p_{\ell}(R-1), \quad R = 2, 3, \dots$$

Proof:

Divide all the partitions of  $p_{\ell}(R)$  into two mutually exclusive groups: I and II. Let group I consist of all partitions which contain one or more parts that are equal to one, and let group II contain all partitions which do not belong to group I (i.e., which do not contain a part equal to one). Hence

$$p_{\ell}(R) = N_I + N_{II}$$

where  $N_I$  and  $N_{II}$  indicate the number of partitions in I and II. By definition  $p_{\ell}^*(R) = N_{II}$ , hence

$$p_{\ell}(R) = N_I + p_{\ell}^*(R)$$

$$p_{\ell}^*(R) = p_{\ell}(R) - N_I$$

It will be established that there is a one-to-one correspondence between each partition in group I and each partition in the set  $R-1 G_{\ell}$ .

(a) By definition each partition in group I is a member of the set  $R G_\ell$  and contains at least one part equal to unity. Removing one part equal to unity produces a member of  $R-1 G_\ell$  (i.e., the parts of the new partition sum to  $R-1$  and the greatest part remains  $\ell$ ).

(b) Adding one part equal to unity to each member of  $R-1 G_\ell$  produces a member of  $R G_\ell$  and a partition in group I (i.e., the new sum of the parts is  $R$  and the greatest part remains  $\ell$ ).

Hence the total number of partitions in group I equals the total number of partitions in  $R-1 G_\ell$ . Therefore  $N_I = p_\ell(R-1)$  and

$$p_\ell^*(R) = p_\ell(R) - p_\ell(R-1).$$

With this result, if the length vector  $L$  does not contain  $\ell_k = 1$ , each  $p_\ell(R)$  in (4.8) can be replaced with  $p_\ell^*(R)$  to obtain a more accurate estimate  $B^*(R, L)$  of the maximum number of partitions:

$$B^*(R, L) = \sum_{\text{all } \ell_i} p_\ell^*(R) = \sum_{\text{all } \ell_i} (p_\ell(R) - p_\ell(R-1)) \quad (4.9)$$

Using the  $B^*(R, L)$  bound for  $R = 8$  and  $L = (4, 3, 2)$  produces

$$\begin{aligned}
 B^*(R,L) &= p_4^*(8) + p_3^*(8) + p_2^*(8) \\
 &= p_4(8) - p_4(7) + p_3(8) - p_3(7) + p_2(8) - p_2(7) \\
 &= 5 - 3 + 5 - 4 + 4 - 3 \\
 &= 2 + 1 + 1
 \end{aligned}$$

$$B^*(R,L) = 4$$

The  $B^*(R,L)$  value for this example yields an exact count of the total number of partitions. In fact, the  $B^*(R,L)$  value will be exact whenever the  $k$ -dimensional vector  $L$  contains all integers from 2 through  $k$ .

#### 4.4.3 Upper Bounds on the Total Number of Permutations

It was shown in section 4.2 that the enumerating algorithm will produce only permutations which belong to one of the subsets  $\{P_0, P_1, P_2, \dots, P_{\ell_k-1}\}$  where  $\ell_k$  is the  $k^{\text{th}}$  entry in the length vector  $L$ . This result followed directly from the fact that all of the permutations enumerated satisfy the condition  $0 \leq d \leq \ell_k-1$ . This same condition can be used to determine the range on the sum,  $S_k$ , of the parts of each permutation; i.e.,

$$\begin{aligned}
 0 &\leq d \leq \ell_k-1 \\
 0 &\leq R-S_k \leq \ell_k-1 \\
 R-\ell_k+1 &\leq S_k \leq R \quad .
 \end{aligned} \tag{4.10}$$

The inequalities in (4.10) can be used to obtain upper bounds,  $B_T(R,L)$ , on the number of permutations generated by the algorithm based on the  $B(R,L)$  limits observed above; i.e.,

$$B_T(R, L) = \sum_{S_k=R-\ell_k+1}^R B(S_k, L)$$

or

$$B_T(R, L) = \sum_{S_k=R-\ell_k+1}^R \sum_{\text{all } \ell_i} p_{\ell_i}(S_k) \quad (4.11)$$

For  $R = 8$  and  $L = (4, 3, 2)$ ,  $B_T(R, L)$  becomes

$$\begin{aligned} B_T(R, L) &= \sum_{S_k=8-2+1}^8 B(S_k, L) \\ &= \sum_{S_k=7}^8 B(S_k, L) \\ &= B(7, L) + B(8, L) \\ &= p_4(7) + p_3(7) + p_2(7) + p_4(8) + p_3(8) + p_2(8) \\ &= 3 + 4 + 3 + 5 + 5 + 4 \\ B_T(R, L) &= 24. \end{aligned}$$

If the length vector  $L$  contains  $\ell_k = 1$ , then

$$B_T(R, L) = \sum_{S_k=R-1+1}^R B(S_k, L)$$

$$B_T(R, L) = \sum_{S_k=R}^R B(S_k, L)$$

$$B_T(R, L) = B(R, L)$$

and the upper bound on the number of permutations equals the upper bound on the number of partitions. This result is not unexpected since as noted above, if unity is included in  $L$ , every permutation enumerated is a partition.

If  $L$  does not contain  $\ell_k = 1$ , theorem 4.2 can be used to obtain a more accurate estimate for  $B_T(R, L)$  by replacing each  $B(S_k, L)$  with  $B^*(S_k, L)$ ; i.e.,

$$B_T(R, L)^* = \sum_{S_k=R-\ell_k+1}^R \prod_{\text{all } \ell_i} p_{\ell_i}^*(S_k)$$

or

$$B_T(R, L)^* = \sum_{S_k=R-\ell_k+1}^R \prod_{\text{all } \ell_i} (p_{\ell_i}(S_k) - p_{\ell_i}(S_k - 1)) \quad (4.12)$$

Paralleling the characteristics of the upper bound  $B^*(R, L)$  for the number of partitions,  $B_T^*(R, L)$  will yield an exact count of the number of permutations if the length vector consists of  $L = (k+1, k, k-1, \dots, 2)$ .

Table 4.4 presents upper bounds on the number of partitions and permutations enumerated for several combinations of length vectors and recreation days. The upper bounds presented in table 4.4 indicate the relative efficiency of the partitioning algorithm for a variety of period lengths and recreation days.

Table 4.4

Upper Bounds on the Total Number of Partitions,  $B(R,L)$ , and Permutations,  $B_T(R,L)$ , Enumerated by the Partitioning Algorithm

Length Vector (L)	Equation Number	Total Number of Recreation Days (R)				
		8	16	26	52	104
(3,2,1)**	(4.8)	10*	30	70	252	954
	(4.11)	10	30	70	252	954
(4,3,2,1)**	(4.8)	15	64	206	1,285	8,991
	(4.11)	15	64	206	1,285	8,991
(3,2)**	(4.9)	2	3	5	9	18
	(4.12)	3	6	9	18	35
(4,3,2)**	(4.9)	4	10	21	70	252
	(4.12)	6	17	37	131	486
(5,4,3,2)**	(4.9)	5	17	50	286	1,901
	(4.12)	8	31	94	654	3,944
(4,3)	(4.9)	3	9	20	69	251
	(4.12)	7	23	54	194	727
(5,4,3)	(4.9)	4	16	49	285	1,900
	(4.12)	9	42	134	810	5,582
(6,5,4,3)	(4.9)	5	25	96	853	9,773
	(4.12)	11	63	253	2,385	28,287

\*Top number equals upper bound on the number of partitions; bottom number equals the upper bound on the number of permutations.

\*\*Bounds equal the exact number of partitions and permutations produced for this length vector.

If an average of two recreation days per week is assumed, the results in table 4.4 are applicable for schedules varying in length from 4 to 52 weeks. The upper bounds on the numbers of permutations and partitions are well within the computational and printing capabilities of modern high-speed computer equipment.

#### 4.5 USE OF THE PARTITIONING ALGORITHM TO CONTROL SOME PR SCHEDULE ATTRIBUTES

The enumeration of acceptable sets of recreation periods is the initial step in the sequential procedure for the design of one-shift PR schedules. This section identifies PR schedule attributes which can be controlled by manipulating the parameters and procedures of the partitioning algorithm. These attributes include:

1. Upper and lower limits on recreation period lengths

This limiting condition is equivalent to specifying the first and last entries in the length vector  $L$  (where  $l_1 > l_2 > \dots l_k$ ).

2. Specific lengths for the recreation periods

This condition is merely an extension of setting upper and lower limits on period lengths. Specifying each valid length is equivalent to specifying each component in the length vector.

3. Upper and lower limits on the total number of periods.

This condition is equivalent to bounding the sum  $\sum_{i=1}^k n_i$  and can be easily incorporated into the basic partitioning algorithm. As each  $n_i$  in a partial permutation is determined, the aggregate number of periods is compared with the upper limit. If the total number of periods,  $n_1$  through  $n_i$ , exceeds the limit,  $n_i$  and  $n_{i+1}, \dots, n_k$  are set equal to zero and a new partial permutation is determined. If a complete enumeration has too few periods, it is excluded.

4. Upper and lower limits on the number of periods of each length.

This limiting condition is easily incorporated into the enumeration algorithm. As each  $n_i$  is computed, either with  $n_i = \left\lceil \frac{R-S_{i-1}}{\ell_i} \right\rceil$  in the fill procedure or with  $n_i = n_{i-1} - 1$  in the backtrack procedure, the new result is compared with the bounds for that length. If  $n_i$  exceeds its upper limit, it is reduced to that limit and the fill procedure is completed. If  $n_i$  falls below the lower limit,  $n_i$  and  $n_{i+1}, \dots, n_k$  are set equal to zero and a new partial permutation is determined.

5. Exact numbers of periods for specific lengths.

This condition, equivalent to setting the upper and lower limits on the number of periods for each specified length equal to each other, can be incorporated into the partitioning algorithm as described in 4 above. A more

direct procedure, however, can be used based on a modification of the original problem defined by  $R$  and  $L$ .

Each partition  $P$  for the original problem can be considered as the union of two subpartitions:  $P_1$  and  $P_2$ . The two subpartitions are distinguishable by the length vector used for each. The length vector  $L_1$  for the  $P_1$  subpartition consists only of period lengths for which the exact number of periods has been specified (i.e.,  $L_1 = \{\ell_i | \ell_i \in L \text{ and } n_i \text{ is known}\}$ ). The length vector  $L_2$  for each  $P_2$  subpartition consists of all periods lengths in  $L$  not included in  $L_1$  (i.e.,  $L_2 = \{\ell_i | \ell_i \in L \text{ and } \ell_i \notin L_1\}$ ). The  $n_i$  values for each  $\ell_i \in L_2$  are unspecified.

Since the  $n_i$  values for the  $L_1$  vector are specified, only one  $P_1$  subpartition exists. In addition, the number of recreation days,  $R_1$  and  $R_2$ , partitioned by  $P_1$  and  $P_2$  respectively is constant for every partition, i.e.,

$$R_1 = \sum_{\ell_i \in L_1} \ell_i n_i$$

and

$$R_2 = R - R_1 .$$

As a result, the task of determining each partition  $P_2$  is equivalent to finding each subpartition  $P_2$  and joining it with  $P_1$ . Determining all  $P_2$  subpartitions is equivalent to a reduced problem defined by  $R_2$  and  $L_2$ .

As an example, consider the problem defined by  $R = 15$  and  $L = (4, 3, 2)$ . If exactly one 4-day period is desired (i.e.,  $n_1 = 1$ ), then  $L_1 = (4)$ ,  $R_1 = 4$ ,  $L_2 = (3, 2)$ , and  $R_2 = 11$ . Application of the partitioning algorithm to the problem defined by  $R_2$  and  $L_2$  yields two  $P_2$  partitions:  $(3, 1)$  and  $(1, 4)$ . Adding the  $P_1$  partition to each yields two complete partitions of  $R$ , each with exactly one 4-day period:  $(1, 3, 1)$  and  $(1, 1, 4)$ .

5. ALLOCATION OF INDIVIDUAL RECREATION PERIODS  
TO DAYS OF THE WEEK

5.1 INTRODUCTION

After individual recreation periods have been determined by the partitioning algorithm, each period must be assigned to specific days of the week in a way that matches the allocation of recreation days by day of the week. To illustrate, consider the schematic representation in figure 5.1 of an allocated daily distribution of recreation days for one shift. Each ray in the star diagram represents one day of the week. The number of nodes on each ray corresponds to the number of recreation days allocated to

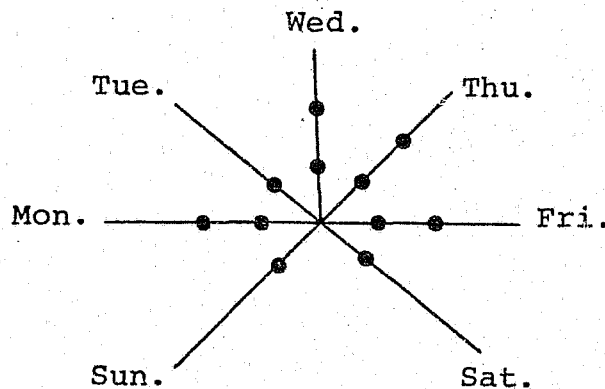


Figure 5.1

Star Diagram with Eleven Nodes

that day of the week. The star diagram indicates that each man will receive a total of 11 recreation days while on the shift: one day off on Tuesday, Saturday, and Sunday, and two days off on Monday, Wednesday, Thursday, and Friday. The star diagram also indicates the number of men assigned to this shift who will be off duty each day of the week: one man will be off duty each Tuesday, Saturday, and Sunday; and two men will be off duty each Monday, Wednesday, Thursday and Friday.

Assigning recreation periods to specific days of the week is equivalent to joining adjacent nodes in a star diagram into clusters. As an example, if the 11 nodes of the star diagram in figure 5.1 are partitioned into three 3-day periods and one 2-day period ( $11 = 3+3+3+2$ ), the star diagram in figure 5.2 indicates one distribution of these four periods over the days of the week. The 3-day periods start on Monday, Wednesday, and Saturday while the 2-day period begins on Thursday.

A star diagram which has its days clustered together into recreation periods is called a cyclic graph. In the degenerate case in which all recreation periods are one day long, the cyclic graph and its star diagram are identical. As outlined in chapter 3, cyclic graphs can be used to construct separation matrices which indicate the number of consecutive work days between each ordered pair of recreation periods. Each separation matrix, in turn,

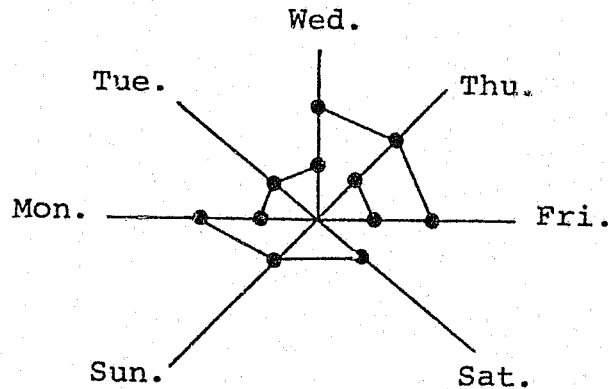


Figure 5.2

Cyclic Graph with Four Recreation Periods  
Based on the Star Diagram in Figure 5.1

is used to determine one-shift schedules for the recreation periods represented in the cyclic graph. A detailed discussion of the construction and use of separation matrices is presented in chapter 6.

The problem of determining all distributions of a given set of recreation periods over the allocated recreation days for each day of the week is equivalent to the problem of enumerating all cyclic graphs for a given star diagram and set of recreation periods. No formula for determining the number of cyclic graphs that exist for a given star diagram and set of recreation periods is known. Experience has shown that there may be no graphs in some cases, while

in others many exist. As an example, figure 5.3 illustrates a second cyclic graph for the recreation periods and star diagram used in the cyclic graph in figure 5.2. In this second graph, the 2-day period begins on Monday instead of Thursday and the 3-day period which begins on Monday in figure 5.2 now begins on Wednesday. It is important to note that the only differences between the cyclic graphs in figures 5.2 and 5.3 are the start days for two of the recreation periods; the total number and lengths of the periods used and the allocation of the individual recreation days over the days of the week are identical in both graphs.

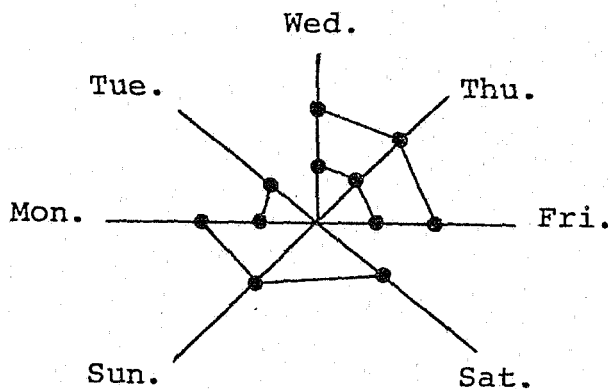


Figure 5.3

Alternate Cyclic Graph Based on the Same  
Recreation Period Lengths and Star Diagram  
As the Cyclic Graph in Figure 5.2

The following sections describe an algorithm which enumerates all cyclic graphs for a given star diagram and set of recreation periods by systematically examining all feasible start arrangements of the recreation periods over all days of the week. The algorithm described in this chapter was first proposed by Dr. Philip Zwart while a faculty member in the Department of Applied Mathematics and Computer Science at Washington University, and a consultant to the scheduling research project in which the author developed many of the results presented here. The algorithm was subsequently programmed and implemented by the author.

## 5.2 OVERVIEW OF THE CYCLIC GRAPH ENUMERATION ALGORITHM

This section describes the basic logic of the algorithm for the enumeration of all cyclic graphs for a given star diagram and set of recreation periods. The algorithm identifies each cyclic graph by specifying the starting day of the week for each recreation period.

### 5.2.1 Tabular Representation of Cyclic Graphs

All of the information contained in a cyclic graph can be represented in tabular form by specifying the length and start day of the week for each recreation period. For example, table 5.1 describes the cyclic graph shown in figure 5.2. The table indicates there are three 3-day recreation periods with one each beginning on Monday,

Table 5.1

Tabular Representation of the Cyclic Graph in Figure 5.2

Period Length (Days)	Number of Periods	Number of Starts						
		Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.
3	3	1	0	1	0	0	1	0
2	1	0	0	0	1	0	0	0

Wednesday, and Saturday, and one 2-day period beginning on Thursday. Construction of the cyclic graph from the information in table 5.1 would reveal the number of recreation days allocated to each day of the week i.e., the underlying star diagram.

As a second example, table 5.2 presents the tabular representation of the cyclic graph in figure 5.3. This second cyclic graph is based on the same star diagram and recreation periods used in the cyclic graph described in table 5.1. Comparison of tables 5.1 and 5.2 indicates that the only differences between the two cyclic graphs are the start assignments of the individual recreation periods; in table 5.2, the 2-day period begins on Monday instead of Thursday and two 3-day periods begin on Wednesday instead of one on Wednesday and one on Monday.

Table 5.2

Tabular Representation of the Cyclic Graph in Figure 5.3

Period Length (Days)	Number of Periods	Number of Starts						
		Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.
3	3	0	0	2	0	0	1	0
2	1	1	0	0	0	0	0	0

Since both cyclic graphs use the same star diagram and recreation periods, the only variables needed to identify distinct cyclic graphs are the start days for the individual recreation periods. The algorithm described in this chapter is a systematic procedure for implicitly examining all start arrangements of the recreation periods over all days of the week.

### 5.2.2 Cyclic Graph Enumeration Logic

Branching processes have been successfully used to enumerate solutions for a wide variety of problems and applications which require the implicit examination of a large, but finite number of possible values for each variable of the problem (93,94). Such processes begin with a single set which contains all feasible solutions to the

problem and an indeterminate number of infeasible solutions. This initial set is subdivided into two or more mutually exclusively subsets by specifying a different value for the same variable for each subset; the number of subsets created equals the number of distinct values that the variable can assume. Each subset represents a more restricted problem since all of the solutions within each subset have the additional constraint imposed by the specific value assigned to one variable. Each subset is divided again into two or more subsets with the introduction of a second constraint defined by the assignment of a specific value to another variable. All of the solutions within each second subset must now satisfy two constraints.

This process of creating new subsets with the specification of a value for one variable at each division continues until either:

- (1) one or more infeasibility conditions are encountered which indicate that regardless of what values are assigned to the remaining variables, no feasible solutions will be found; or
- (2) all of the variables are given specific values and a feasible solution is obtained.

When either an infeasibility condition is encountered or a solution is found, the value of the most recently specified variable is changed and a new solution is sought. If all possible values for the most recently specified variable

have been examined, the variable is left unassigned and the algorithm returns to the next most recently selected variable. This process continues until all subsets of solutions have been examined.

The efficiency of such branching algorithms relies on the recognition and use of infeasibility conditions which permit early identification of large subsets (i.e., with few specified variables) which contain no feasible solutions. If an objective function is used which attaches a value to each feasible solution, bounds can be computed for each subset which can also be used to identify and discard subsets of solutions which can not possess the optimal solution.

The algorithm used to enumerate all cyclic graphs for a given star diagram and set of recreation periods is a branching process which uses a set of variables indexed by day of the week and type of recreation period. The "value" assigned to each variable is the number of starts of its corresponding recreation period type on a specified day of the week; the "type" of recreation period refers to its length; e.g., all 2-day periods are considered the same type. Let  $S_{ij}$  represent the number of  $i$  type periods which begin on day  $j$  ( $j = 1$  represents Monday,  $j = 2$  represents Tuesday, ...,  $j = 7$  represents Sunday); and let  $S_j$  represent a  $k$ -dimensional row vector which indicates the number of

each type of recreation period assigned to day  $j$ ; i.e.,

$$S_j = (s_{1j}, s_{2j}, \dots, s_{kj})$$

where  $k$  represents the number of types of recreation periods. (The  $k$  period types in this chapter are equivalent to the  $k$  lengths defined for the partitioning algorithm in chapter 4.) In table 5.1,  $k = 2$  and the  $S_j$  vector or start arrangement for each day of the week is  $S_1 = (1,0)$ ,  $S_2 = (0,0)$ ,  $S_3 = (1,0)$ ,  $S_4 = (0,1)$ ,  $S_5 = (0,0)$ ,  $S_6 = (1,0)$ , and  $S_7 = (0,0)$ .

Let  $S$  be the matrix defined by  $S = (S_1', S_2', \dots, S_7')$ . Then for table 5.1,

$$S = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}. \quad (5.1)$$

The  $S$  matrix is merely an alternative statement of the tabular format introduced above to identify distinct cyclic graphs. The matrix contains seven columns, one for each day of the week and for table 5.1, two rows, one for each recreation period type (length). The total number of recreation periods which begin on day  $j$  is given by the sum of the entries in column  $j$  in the  $S$  matrix. The total number of recreation periods of length  $\ell_i$  is given by the sum of the entries in row  $i$ . Let  $N = (n_1, n_2, \dots, n_k)$  represent a partition where  $n_i$  indicates the number of periods of length  $\ell_i$ , then it can be shown that

$$N' = \begin{pmatrix} n_1 \\ n_2 \\ \vdots \\ n_k \end{pmatrix} = S \cdot 1 .$$

where 1 is the seven-dimensional unity vector. As an example, using the matrix in (5.1),

$$N' = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \end{pmatrix} ,$$

which indicates that the start arrangement in the S matrix for table 5.1 has three starts for the  $i = 1$  type period (3-day periods,  $\ell_1 = 3$ ) and one start for the  $i = 2$  type period (2-day periods,  $\ell_2 = 2$ ).

These results also confirm the observation that the tabular format introduced above for the representation of cyclic graphs is an alternative notation for presenting the information contained within each graph; i.e., both identify the recreation period lengths  $L$ ; the number of each period type  $N$ ; and the specific start arrangement for each day of the week  $S_j$ .

The branching process described in this chapter uses the  $L$  and  $N$  vectors as constraints, and identifies each cyclic graph by specifying each  $S_j$  vector within the  $S$  matrix associated with each graph. The branching algorithm determines each cyclic graph by assigning, in step-wise

fashion, specific values to the components of  $S_j$ . The number of distinct start arrangements that must be examined for each day  $S_j$  is the product of the number of distinct values that can be assigned to each component of  $S_j$ . Each possible start arrangement for a day represents a different subset of cyclic graphs.

The generalized branching process is schematically illustrated in figure 5.4. From the set of all solutions, day  $J$  (i.e.,  $j = J$ ) is selected as the first day for examination. Hence, the first constraint used to subdivide the set of all solutions is based on the assignment of specific values to the components of the  $S_j^!$  column of the  $S$  matrix. The number of distinct values or start arrangements that can be assigned to  $S_j$  determines the number of subdivisions that can be made of the original set. For the generalized diagram in figure 5.4, four start arrangements ( $J1, J2, J3$ , and  $J4$ ) are shown for day  $J$ . The procedure for systematically enumerating all start arrangements for each day is described below.

The circled number above each node indicates the actual sequence in which the nodes are generated by the enumeration algorithm; each start arrangement or node for day  $J$  is fathomed by the algorithm before the next start arrangement for day  $J$  is examined. Node 1 represents the subset of all solutions with start arrangement  $J1$  assigned to day

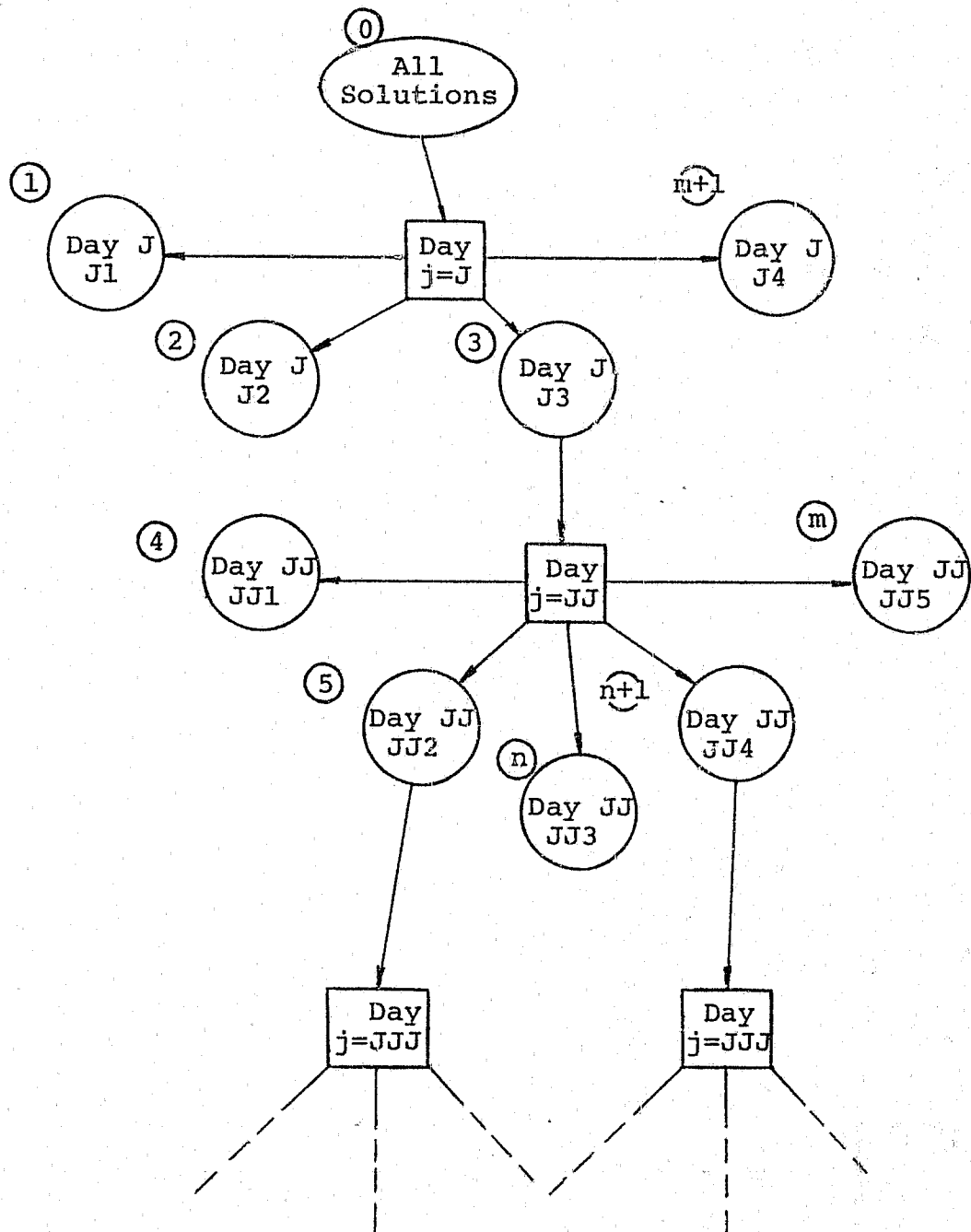


Figure 5.4

Schematic Representation of the Branching Process for  
the Cyclic Graph Enumeration Algorithm

J (i.e., assigned to  $S_J^1$  of the S matrix). The absence of any branches from node 1 indicates that an infeasibility condition exists and that node 1 contains no feasible solutions. Infeasible conditions are defined below. Node 2 is created by assigning the next start arrangement to day J; again no feasible solutions are found. When start arrangement J3 is assigned, creating node 3, however, no infeasibility conditions are encountered and this subset is examined further. The next day is selected (JJ may be any day of the week other than J), and each distinct start arrangement for day JJ (there are five) are examined. Nodes 4, n and m produce subsets with no feasible solutions; nodes 5 and n+1, however, are examined further. A new day (JJJ) is selected for each node (the JJJ day from node 5 need not be the same day selected for the JJJ day for node n+1) and the branching process continues (first from node 5 and then node n+1) until either an infeasibility condition is encountered or a feasible solution is obtained. The node number for the JJ3 start arrangement for day JJ is specified with the literal "n" since the algorithm will first explore an as yet to be determined number of paths from node 5 before returning to node n.

It is important to recognize that each node in the branching diagram represents a subset of cyclic graphs

which satisfy each of the constraints which lie on the path from that node back to node 0; e.g., all cyclic graphs defined by node 5 satisfy two constraints: day JJ with start arrangement JJ2 and day J with start arrangement J3. An acceptable cyclic graph is determined each time a path from node 0 has been constructed which specifies a start arrangement for each day of the week. The branching process terminates when the last start arrangement for day J has been examined.

### 5.3 BASIC CONCEPTS OF THE CYCLIC GRAPH ENUMERATION ALGORITHM

This section introduces three concepts which are fundamental to the cyclic graph enumeration algorithm.

#### 5.3.1 The Number of Recreation Period Starts for Each Day of the Week

Given any star diagram and set of recreation periods to be assigned, it is possible to establish upper and lower bounds on the total number of recreation periods that can start from each day of the week (i.e., limits on the sum  $\sum_{i=1}^k s_{ij}$  for each  $j$ ). Let  $T$  be the total number of periods to be assigned and let  $\phi_j$  be the number of recreation days allocated to day  $j$  (i.e. the number of nodes on ray  $j$  of the star diagram). Further, let  $B_j \max$  and  $B_j \min$  represent the maximum and minimum number of period starts on day  $j$ . It is easily shown that

$$B_j \text{ max} = \phi_j \quad j = 1, 2, \dots, 7 \quad (5.2)$$

and

$$B_j \text{ min} = \max\{0, \phi_j - \phi_{j-1}\} \quad j = 1, 2, \dots, 7 \quad (5.3)$$

$$\phi_0 \equiv \phi_7$$

Equation (5.2) sets the maximum number of period starts from day  $j$  equal to the number of recreation days allocated to day  $j$  ( $\phi_j$ ).<sup>\*</sup> The minimum number of starts for day  $j$  is zero unless there are more recreation days allocated to day  $j$  than to day  $j-1$ ; i.e., unless  $\phi_j - \phi_{j-1} > 0$ .

As an example, the number of recreation days allocated to each day of the week, and the maximum and minimum starts for each day in the star diagram in figure 5.1 based on four period starts,  $T = 4$ , are presented in table 5.3. The maximum number of starts for each day equals the number of recreation days allocated to that day. For Monday and Wednesday, the minimum number of starts is one since for both days the number of nodes on the preceding day is one less than the number of nodes on Monday and Wednesday each.

### 5.3.2 Enumeration of All Start Arrangements for Each Day of the Week

As each day of the week is selected in the enumeration scheme, all distinct values (or start arrangements) which can be assigned to that day must be examined. Each start

---

<sup>\*</sup> Equation (5.2) is applicable if recreation period lengths do not exceed the number of rays in a star diagram. If period lengths are unrestricted, equation (5.2) becomes  $B_j \text{ max} = \min\{T, \phi_j\}$ ,  $j = 1, 2, \dots, 7$ .

Table 5.3

Maximum and Minimum Number of Starts for  
Each Day of the Week for the Star Diagram  
in Figure 5.1\*

j=	Mon. 1	Tue. 2	Wed. 3	Thu. 4	Fri. 5	Sat. 6	Sun. 7
Number of recreation days on node i $\phi_j =$	2	1	2	2	2	1	1
Maximum number of period starts, $B_j \text{ max} =$	2	1	2	2	2	1	1
Minimum number of period starts, $B_j \text{ min} =$	1	0	1	0	0	0	0

\*Four recreation periods to be assigned ( $T=4$ ).

arrangement can be found with the following information:

- (1) the upper and lower limits on the total number of period starts which can be assigned to that day, and
- (2) the total number of starts to be assigned for each recreation period type.

As an example, suppose that Wednesday is selected as the first day (i.e.,  $J = 3$ ) to be examined from the star diagram in figure 5.1. From the bounds calculated above (see table 5.3), it is known that either one or two recreation periods must begin on Wednesday. Consider first the assignment of exactly two period starts. (There are four recreation periods available: one 2-day period and

three 3-day periods, denoted by  $L = (3,2)$  and  $N = (3,1)$ .) Only two start arrangements are possible; either one 2-day period and one 3-day period ( $S_J = (1,1)$ ) or two 3-day periods ( $S_J = (2,0)$ ). A start arrangement consisting of two 2-day periods ( $S_J = (0,2)$ ) is not included since only one 2-day period is available for assignment. If only one period start is assigned to Wednesday, there are two possibilities: one 2-day period ( $S_J = (0,1)$ ) or one 3-day period ( $S_J = (1,0)$ ). The four start arrangements for Wednesday are summarized in table 5.4.

Table 5.4

Distinct Start Arrangements for Wednesday  
Selected as the First Day from the  
Star Diagram in Figure 5.1

Start Arrangement Number	Number of Three-Day Starts	Number of Two-Day Starts	$S_3$
1.	1	1	(1,1)
2.	2	0	(2,0)
3.	1	0	(1,0)
4.	0	1	(0,1)

The enumeration of all start arrangements for each day of the week can be accomplished with the partitioning algorithm described in chapter 4. Let  $T_{js}$  represent the set of start arrangements for day  $j$  with exactly  $s$  starts. The set of all start arrangements for day  $j$ ,  $T_j$ , can be represented as

$$T_j = \{T_{js} | s = B_{j \max}, B_{j \max}-1, \dots, B_{j \min}\}$$

The partitioning algorithm can be used to enumerate all of the start arrangements within each set  $T_{js}$  with the following constraints:

- (1) the total number of days to be partitioned ( $R$ ) equals the total number of starts; i.e.,  $R = s$  and
- (2) the unity vector with  $k$  elements is used in place of the  $k$ -dimensional length vector  $L$ ; i.e.,  $L = 1$ .

As an example, the constraints for the use of the partitioning algorithm for enumeration of the start arrangements shown in table 5.4 are  $k = 2$  (the number of recreation period types),  $L = (1,1)$  (the two-dimensional unity length vector), and for  $T_{32}$ ,  $R = 2$ , and for  $T_{31}$ ,  $R = 1$  (the number of period starts to be assigned). Beginning with  $R = 2$ , the algorithm produces the three permutations shown in table 5.5. For  $R = 1$ , the partitioning algorithm produces the two permutations shown in table 5.6. Permutations 2 and 3 in table 5.5, and 1 and 2 in table 5.6 correspond to

Table 5.5

Start Arrangements for Two Recreation Periods  
Enumerated by the Partitioning Algorithm  
for Wednesday of the Star Diagram  
in Figure 5.1

Permutation Number	Partial Permutation	Full Permutation	Feasible Start Arrangement
1.	(.,.)	(2,0)	Yes
2.	(1,.)	(1,1)	Yes
3.	(0,.)	(0,2)	No

Table 5.6

Start Arrangements for One Recreation Period  
Enumerated by the Partitioning Algorithm  
for Wednesday of the Star Diagram  
in Figure 5.1

Permutation Number	Partial Permutation	Full Permutation	Feasible Start Arrangement
1.	(.,.)	(1,0)	Yes
2.	(0,.)	(0,1)	Yes

the four start arrangements shown in table 5.4. Permutation 3 in table 5.5 is not a feasible start arrangement since it requires two 2-day periods. The screening of the permutations generated by the partitioning algorithm to identify feasible start arrangements is based on the  $N$  and  $S_j$  vectors. Let  $N = (n_1, n_2, \dots, n_k)$  represent the vector of the numbers

of recreation periods of each length, (where  $n_i$  is the number of periods of length  $l_i$ ) and let  $S_j = (s_{1j}, s_{2j}, \dots, s_{kj})$  represent a specific start arrangement generated by the partitioning algorithm for day  $j$  ( $s_{ij}$  represents the number of periods of length  $l_i$  assigned to start on day  $j$ ). The start arrangement  $S_j$  is a feasible start arrangement if  $S_j \leq N$ ; i.e.,  $\{s_{ij} \leq n_i | i = 1, 2, \dots, k\}$ . This condition insures that the number of assigned periods of each length ( $s_{ij}$ ) does not exceed the number of periods of that length available for assignment ( $n_i$ ). In the example illustrated in table 5.5,  $N = (3, 1)$  and for permutation 3,  $S_j = (0, 2)$ ; since  $S_j \not\leq N$ , the permutation is not feasible.

If a lower bound of zero starts had been indicated for Wednesday in table 5.3, one additional feasible start arrangement would have been generated in the example above:  $S_3 = (0, 0)$  (i.e., the assignment of no period starts to Wednesday).

In summary, all start arrangements for each day of the week can be enumerated based on the following information:

- (1) the upper and lower bounds on the total number of starts that can be assigned to each day, and
- (2) the number and lengths of the recreation periods to be assigned.

The start arrangements for each day can be partitioned into mutually exclusive sets based on the number of starts

in each arrangement. The number of sets is determined directly from the upper and lower limits ( $B_{j \max}$  and  $B_{j \min}$ ) on the number of starts that can be assigned to each day. Beginning with the set containing arrangements with the greatest number of starts (i.e.,  $B_{j \max}$ ) the partitioning algorithm can be used to enumerate each feasible start arrangement. After all feasible arrangements for the first set have been determined, the set of arrangements with one less start is examined. The process continues until all arrangements within each set have been enumerated.

### 5.3.3 Reduced Star Diagram

Particularly useful to understanding the enumeration algorithm described in this chapter is the concept of a reduced star diagram. As indicated above, each start arrangement creates a new subset of cyclic graphs. Associated with each subset is a reduced star diagram which contains only those nodes (recreation days) which have not been included in any assigned recreation period.

As an example, assume that Wednesday is the first day examined for the star diagram in figure 5.1; and that the first start assignment considered is  $S_1 = (0,1)$  (i.e., one 2-day period, arrangement 4 in table 5.4); the recreation period uses one node on Wednesday and one node on Thursday. The resulting reduced star diagram is shown in figure 5.5;

one node has been removed from both the Wednesday and Thursday rays of the diagram in figure 5.1.

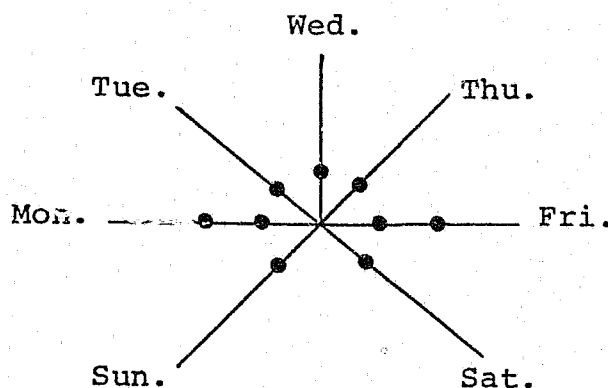


Figure 5.5

Reduced Star Diagram, One Wednesday Node and  
One Thursday Node Removed from the  
Star Diagram in Figure 5.1

This reduced star diagram and the reduced number of recreation periods to be assigned can be treated as a new subproblem with the additional constraint that no additional period starts can be assigned to Wednesday. The subproblem is defined by:

- (1) the reduced star diagram in figure 5.5,
- (2) the reduced set of unassigned recreation periods,  $N_J$ . ( $N_J$  represents the set of unassigned periods after day J has been specified.) In the example above,  $N_J = N - S_J = (3,1) - (0,1) = (3,0)$  indicating that three 3-day periods must still be assigned; and

- (3) the constraint that Wednesday cannot be assigned additional recreation period starts. This condition does not prohibit the use of Wednesday in recreation periods which begin on other days. In fact, as evident in figure 5.5, the Wednesday ray contains one node which will have to be included in a period beginning on a different day of the week.

#### 5.4 CYCLIC GRAPH ENUMERATION ALGORITHM LOGIC

This section draws upon the concepts introduced above to describe the procedures and logic of the cyclic graph enumeration algorithm. After describing the basic iterative pattern of the algorithm, a detailed example is presented which illustrates the branching process.

##### 5.4.1 Iterative Process

As each new node is examined in the branching process, the cyclic graph algorithm repeats the steps outlined in figure 5.6. The value  $j = 0$  represents the set of solutions that can be obtained by enumeration of all start arrangements for all seven days of the week. The information associated with  $j = 0$  is the original star diagram, denoted by  $SD_0$ , and the complete set of recreation periods to be assigned to the cyclic graph, represented by the vector  $N_0$ . For every other value of  $j$  the information available consists of a reduced star diagram  $SD_j$  and a reduced vector of unassigned recreation periods  $N_j$ . The four basic steps in the process of moving from one value of  $j$  (day) to another are:

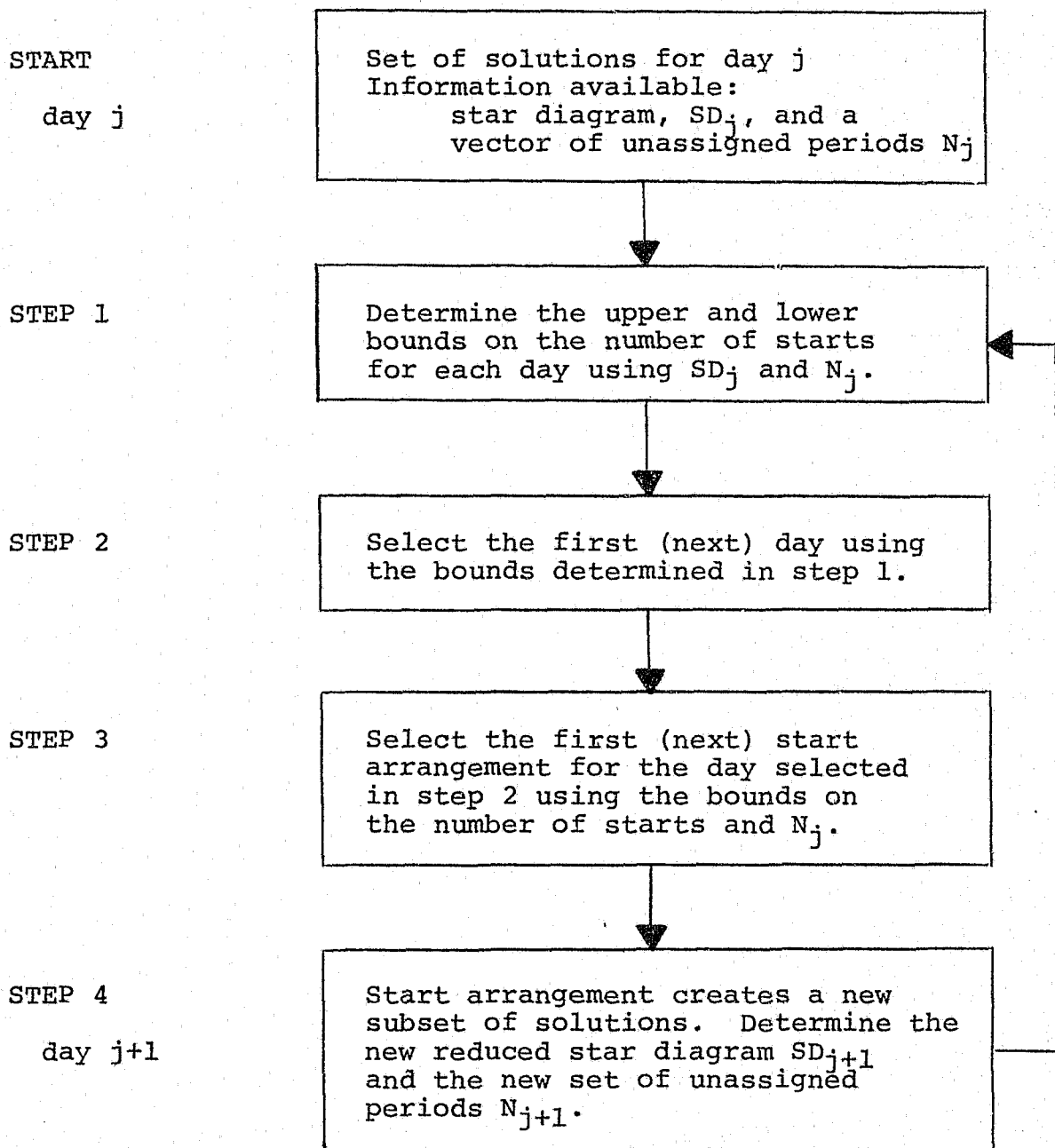


Figure 5.6

Basic Logic Flow for the Cyclic Graph  
Enumeration Algorithm

Step 1. Calculation of the minimum and maximum number of period starts ( $B_j$  min and  $B_j$  max) for each day of the week.

To select the first day to be examined from  $j = 0$  the upper and lower bounds on the number of period starts for each day of the week are calculated based on the complete star diagram denoted by  $SD_0$ . When  $j > 0$ , the upper and lower bounds are calculated based on the reduced star diagram  $SD_j$  associated with day  $j$ .

Step 2. Selection of the first (next) day to be assigned a specific start arrangement.

Although any sequence of the seven days of the week can be used in the enumeration algorithm, careful selection of the next day to be examined can reduce the computational effort required to find all solutions. The day selection criterion utilized in this work is motivated by the fact that to specify a complete graph it is only necessary to indicate which days of the week have one or more period starts since once all of the periods in  $N_0$  have been assigned, any remaining days of the week must have zero starts.

Hence, if days with non-zero start arrangements are examined as early as possible in the branching process, subsets of solutions requiring examination of fewer than seven days can be obtained for which no recreation periods

remain to be assigned. For each such subset, no further branching is necessary since all of the remaining days must receive zero starts.\*

In this thesis, the "next" day selected  $j'$  from each node is the day with the greatest number of required starts; i.e.,

$$\text{day } j' = \{\text{day } j \mid \max_j B_j \text{ min for all } j \in J\}$$

where  $J =$  (set of  $j$  indices associated with unassigned days of the week).

If two or more days have the same maximum number of required starts, the day with the lowest  $j$  value is selected.

Step 3. Selection of the first (next) start arrangement for the current day under consideration.

As described in section 5.3.2, the upper and lower bounds on the number of starts, and the current set of unassigned recreation periods associated with each selected day are used as input data for the partitioning algorithm to enumerate all possible start arrangements for that day. Beginning with the set of start arrangements with the maximum number of starts,  $B_{j \text{ max}}$ , the partitioning algorithm enumerates each arrangement. After all start arrangements

---

\* Each subset completed in this manner produces one cyclic graph.

in the initial set have been found, the algorithm is used to enumerate all arrangements with one less period start, i.e.,  $B_j \text{ max} - 1$ . This process continues until all of the arrangements from the set with the minimum number of required starts  $B_j \text{ min}$  have been enumerated.

If the minimum number of required starts  $B_j \text{ min}$  for the selected day is zero, the last start arrangement examined for that day consists of zero starts and the reduced star diagram and set of unassigned periods remain unchanged.

Step 4. Construction of the reduced star diagram and vector of unassigned periods.

Once the first (next) start arrangement has been assigned, a new reduced star diagram can be constructed. This is done by removing the nodes from the current star diagram which are used in the periods just assigned. The new vector of unassigned periods equals  $N_{j+1} = N_j - S_{j+1}$ .

The new star diagram and reduced vector now serve as part of the initial information for another iteration of the algorithm.

#### 5.4.2 Two Example Problems

Repeated application of the four-step cycle described above will lead eventually to either the enumeration of a feasible cyclic graph or to one of several infeasibility conditions which indicate that the current subset of solutions does not contain any feasible graphs. Two examples

of the algorithm are presented in this section to illustrate the iterative process described above and also to identify conditions which indicate when:

- (1) the current subset contains only infeasible solutions,
- (2) a feasible cyclic graph has been obtained, and
- (3) the enumeration process should be terminated.

To simplify the example to be presented, consider a four-day "week" consisting of days A, B, C, and D for which a three-week (i.e., 12 day) schedule is to be designed which contains seven recreation days. Further, assume that the seven recreation days have been allocated to the four days of the week as illustrated in the star diagram in figure 5.7, and that for this example, cyclic graphs will be enumerated for a partition of the recreation days consisting of one 3-day period and two 2-day periods; i.e.,  $L = (3,2)$  and  $N_0 = (1,2)$ .

To determine the first day to be examined, the upper and lower bounds on each day of the week must be determined based on the star diagram in figure 5.7 and the total number of periods to be assigned; these bounds, based on formulas (5.2) and (5.3), are shown in table 5.7. Day C with the highest lower bound is selected as the first day to be examined. The partitioning algorithm is used first to

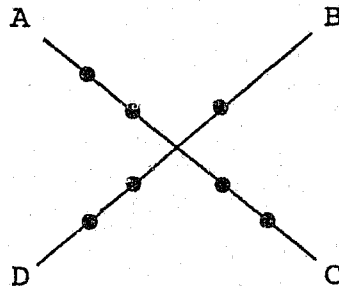


Figure 5.7

Initial Star Diagram for the Cyclic  
Graph Enumeration Example

Table 5.7

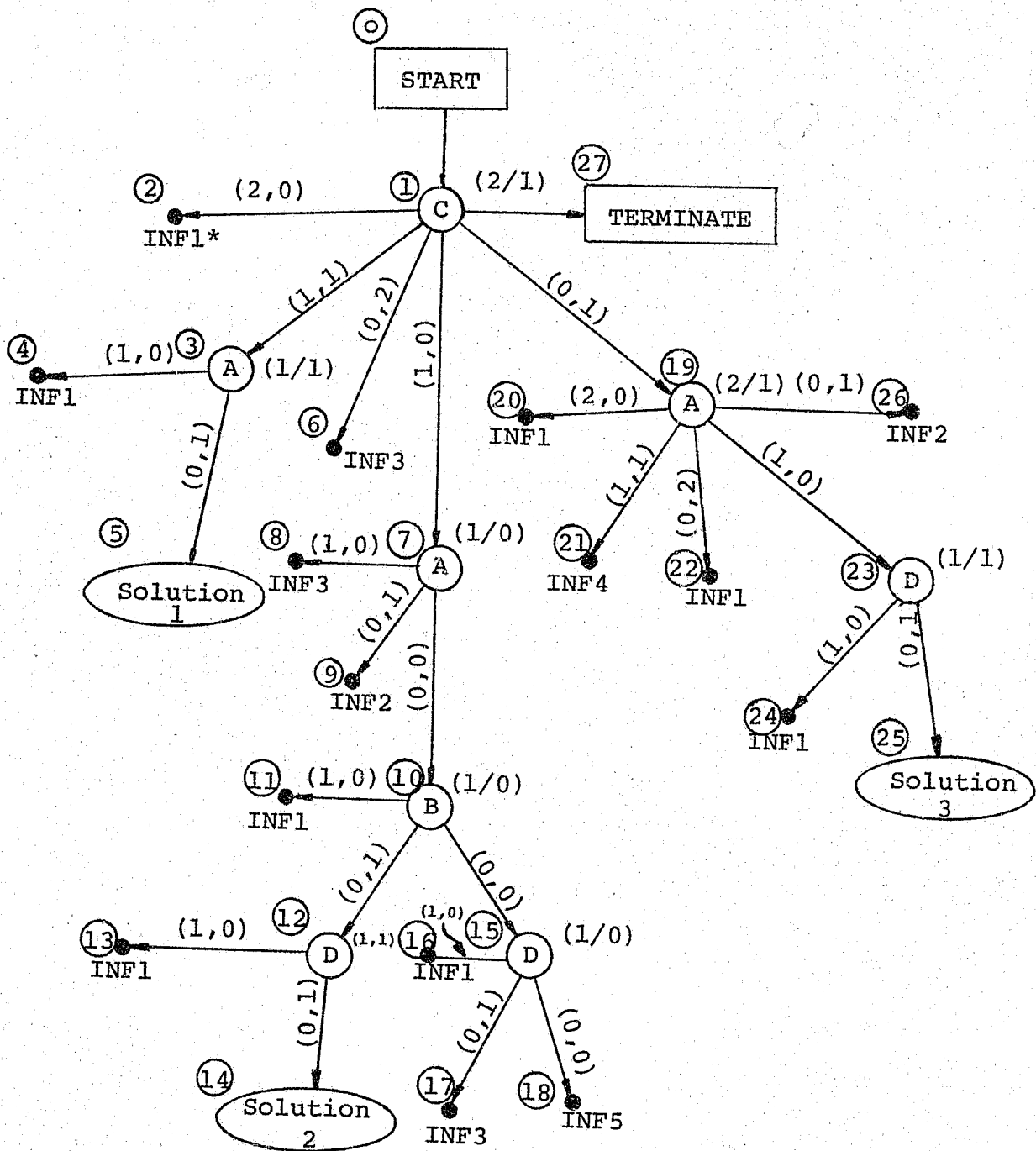
Upper and Lower Bounds on the Number of Period  
Starts for Each Day Based on the Star  
Diagram in Figure 5.7

	Day of the Week			
	A	B	C	D
Number of nodes	2	1	2	2
Maximum number of period starts, $B_j$ max	2	1	2	2
Minimum number of period starts, $B_j$ min	0	0	1	0

enumerate all start arrangements for day C that contain exactly two period starts (the maximum number of starts for C); after all two-start arrangements have been enumerated, all one-start arrangements are generated.

The tree structure generated by the branching algorithm for this example is shown in figure 5.8. The circled numbers at each node indicates the sequence of steps taken by the algorithm -- from step 0 (START) to step 27 (TERMINATE). From START, the first day selected is C; the right superscript on the C node: (2/1), indicates the upper and lower bounds on the number of starts that can be assigned to day C. Each branch from this node represents a specific start arrangement for C; the corresponding  $S_C$  vector associated with each arrangement is shown on each branch in the diagram. The initial start arrangement assigned to each day is represented by a horizontal branch directed to the left of each node; subsequent branches for each day are read in a counter-clockwise direction about the node. For example, the first arrangement assigned to day C is  $S_C = (2,0)$ , i.e., two 3-day periods and no 2-day periods. The last arrangement assigned to day C is  $S_C = (0,1)$ . Following the last start assignment for C, the branching process is terminated.

With the assignment of each specific start arrangement, one of two possibilities can occur, either



\*INF<sub>i</sub> indicates infeasibility condition i

Figure 5.8

Tree Structure for the Cyclic Graph  
Enumeration Example

- (1) one or more infeasibility conditions are encountered which indicate that the subset of solutions created with this arrangement does not contain any feasible graphs; or
- (2) no infeasibility conditions are encountered which indicates that the new subset may contain feasible graphs.

If no infeasibility conditions are encountered, the new subset contains exactly one feasible graph if

- (1) one start arrangement has been specified for each day of the week, and
- (2) all of the periods in  $N_0$  have been assigned.

Each of these possibilities is discussed below with the development of the example.

As noted above, the first start arrangement for day C is  $S_C = (2,0)$ . This assignment leads to infeasibility condition 1 (INF1 in figure 5.8) since the number of recreation periods assigned exceeds the number of periods available for at least one of the period types; i.e.,  $S_C \not\leq N((2,0) \not\leq (1,2))$ . The next start arrangement for C is  $S_C = (1,1)$ , one 3-day period and one 2-day period. Since  $S_C$  satisfies  $S_C \leq N_0 ((1,1) \leq (1,2))$ , the arrangement is feasible and the reduced star diagram and reduced vector of unassigned periods can be constructed. The reduced diagram,  $SD_C$ , shown in figure 5.9, is obtained by removing the nodes which are used in the periods for  $S_C = (1,1)$  from the original star diagram. The reduced set of unassigned

periods is given by  $N_C = N_0 - S_C = (1,2) - (1,1) = (0,1)$ ;  
i.e., only one 2-day period remains to be assigned.

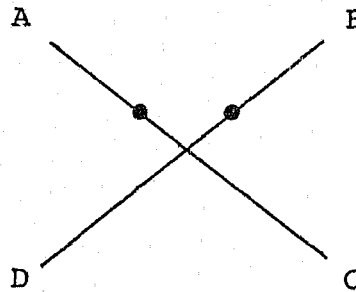


Figure 5.9

Reduced Star Diagram Corresponding to the  
Assignment of Two Period Starts (One  
Three-Day Period and One Two-Day  
Period) to Day C

With the creation of the reduced star diagram and the reduced vector of unassigned periods, a new subprogram is defined and another iteration of the four-step cycle described in section 5.4.1 is initiated. The upper and lower bounds on the number of period starts for each day are calculated and used to select the next day (see table 5.8). Day A is selected since it has the greatest number  $(B_j \text{ min})$  of required starts.

Table 5.8

Upper and Lower Bounds on the Number of Period Starts for Each Day Based on the Reduced Star Diagram in Figure 5.9

	Day of the Week			
	A	B	C	D
Number of nodes	1	1	0	0
Maximum number of period starts, $B_j$ max	1	1	0	0
Minimum number of period starts, $B_j$ min	1	0	0	0

Two infeasibility conditions can arise when upper and lower bounds on the number of period starts are calculated. The first, identified as INF2 in figure 5.8, only occurs when bounds are calculated based on a reduced star diagram. The existence of a reduced diagram implies that one or more days have already been assigned specific start arrangements and as a result, cannot be assigned further period starts in subsequent iterations of the algorithm along its current branch. Consequently, the lower bound on the number of period starts for each previously selected days must be zero (day C satisfies this requirement in table 5.8). If

the lower bound for a previously selected day is greater than zero, all solutions in the current subset are infeasible. The INF2 infeasibility condition occurs twice in the example problem -- at steps 9 and 26.

The second infeasibility condition (identified as INF3 in figure 5.8), which can also be recognized when the upper and lower bounds on the number of period starts are determined, occurs when the number of required starts for a day exceeds the maximum number of starts for the same day; i.e., when  $B_{j \min} > B_{j \max}$ . This condition can arise when a disproportionate number of nodes exist on one ray of either an initial or reduced star diagram. This infeasibility condition occurs twice in the example problem -- at steps 6 and 17 in figure 5.8.

Since neither infeasibility condition 2 or 3 exists in table 5.8, the algorithm continues with the enumeration of specific start arrangements for day A. Only two start arrangements are possible: (1,0) and (0,1). The first arrangement is rejected because  $S_A \nmid N_C$ , i.e., (1,0)  $\nmid$  (0,1) (infeasibility condition 1 at step 4). The second arrangement, (0,1) is feasible and one 2-day period is assigned to start on day A.

The new reduced star diagram  $SD_A$  is obtained by removing the two nodes corresponding to the 2-day period beginning

on day A from the star diagram in figure 5.9. The resulting null star diagram (i.e., zero nodes on all the rays) indicates that all of the allocated recreation days have been used in the three periods assigned to start on days A and C; and hence, the subset of solutions at step 5 in figure 5.8 contains a feasible cyclic graph. Each of the remaining days that have not been examined (i.e., B and D) are assigned zero starts.

The first solution is shown in tabular form in table 5.9 and as a cyclic graph in figure 5.10; the columns under days A and C in table 5.9 identify the start arrangement vectors indicated on the branches for those days in figure 5.8.

Table 5.9

Tabular Representation of the First Solution for the  
Cyclic Graph Enumeration Example  
(Step 5 in Figure 5.8)

Period Length (Days)	Number of Periods	Number of Starts			
		A	B	C	D
3	1	0	0	1	0
2	2	1	0	1	0

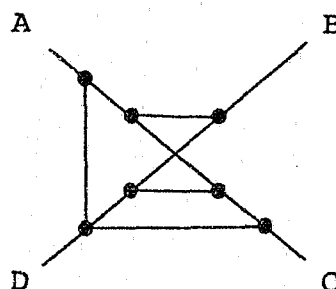


Figure 5.10

First Cyclic Graph Solution  
(Step 5 in Figure 5.8)

With the determination of a feasible graph, the algorithm continues by seeking the next start arrangement for day A. Since  $(0,1)$  is the last start arrangement possible for day A, the algorithm backtracks to day C to determine the next start arrangement (if any) for C. The next arrangement for day C is  $(0,2)$ ; i.e., no 3-day periods and two 2-day periods. Since  $S_C \leq N_0((0,2) \leq (1,2))$ , the arrangement is feasible and the reduced star diagram (see figure 5.11) and reduced vector of unassigned periods can be determined. The vector of unassigned periods becomes  $N_C = N_0 - S_C = (1,2) - (0,2) = (1,0)$ ;

i.e., only one 3-day period remains unassigned. The upper and lower bounds for each day based on  $N_C$  and the star diagram in figure 5.11 are shown in table 5.10. The bounds for day A indicate infeasibility condition 3 (INF3 in figure 5.8); i.e., the lower bound on the number of starts on day A exceeds its upper bound. As a consequence, the (0,2) start arrangement produces an infeasible subset of solutions.

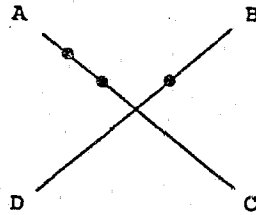


Figure 5.11

Reduced Star Diagram Corresponding to the Assignment of Two 2-day Recreation Periods to Day C

Table 5.10

Upper and Lower Bounds on the Number of Period Starts for Each Day Based on the Reduced Star Diagram in Figure 5.11

	Day of the Week			
	A	B	C	D
Number of nodes	2	1	0	0
Maximum number of period starts, $B_j \max$	1	1	0	0
Minimum number of period starts, $B_j \min$	2	0	0	0

The  $(0,2)$  arrangement for day C is the last arrangement with two period starts. Next, all arrangements with one start are enumerated beginning with the  $(1,0)$  arrangement (step 7 in figure 5.8). The arrangement is feasible, and based upon the upper and lower bounds on the number of period starts for each day determined from the reduced star diagram and vector of unassigned periods, day A is selected next. Based on upper and lower limits of one and zero starts respectively, three start arrangements are examined:  $(1,0)$ ,  $(0,1)$  and  $(0,0)$ . The  $(1,0)$  and  $(0,1)$  arrangements each produce infeasibility conditions (steps 8 and 9 in figure 5.8).

No infeasibility conditions, however, are encountered with the assignment of the  $(0,0)$  arrangement in step 10. Construction of the reduced star diagram  $SD_A$  and the vector of unassigned periods  $N_A$  is quite simple since no nodes are removed from the current star diagram  $SD_C$  and no periods are used from the set  $N_C$ . Hence, the same bounds on the number of starts for each day that were used to select day A can be used again with only one exception: in selecting the day for step 10, both days C and A are excluded since both have already been assigned a specific start arrangement.

Day B is selected as the next day and a second solution is eventually found at step 14 (see table 5.11 and figure 5.12). In the second solution, the 3-day period begins on day C and the two 2-day periods begin on

Table 5.11

Tabular Representation of the Second Solution for the  
Cyclic Graph Enumeration Example  
(Step 14 in Figure 5.8)

Period Length (Days)	Number of Periods	Number of Starts			
		A	B	C	D
3	1	0	0	1	0
2	2	0	1	0	1

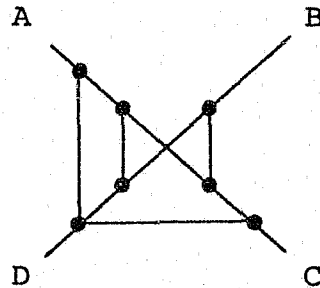


Figure 5.12

Second Cyclic Graph Solution  
(Step 14 in Figure 5.8)

days B and D. The third and final cyclic graph for this example, found at step 25, is presented in table 5.12 and figure 5.13. The complete branching process terminates at step 27 after the examination of the last start arrangement for day C.

This sample problem also illustrates two additional infeasibility conditions which can be used to recognize subsets of solutions which do not contain feasible cyclic graphs. Both conditions are discussed below.

At step 21 in figure 5.8, the (1,1) start arrangement is assigned to day A. Since the  $S_A = (1,1)$  arrangement appears to be feasible, the algorithm proceeds to the construction of the reduced star diagram  $SD_A$ . In removing the two periods in the (1,1) arrangement from  $SD_C$  (see figure 5.11), however, an infeasibility condition arises on ray B; both periods assigned to start on day A use a recreation node on day B, and hence, to construct the reduced diagram associated with the (1,1) arrangement, more nodes must be removed from the B ray than exist on  $SD_C$ . Since the number of nodes on any ray of the reduced star diagram cannot become less than zero, the (1,1) start arrangement in step 21 is not feasible (identified as INF4 in figure 5.8).

The final infeasibility condition occurs at step 18. After day D is assigned the (0,0) start arrangement, all four days of the week have been assigned specific arrangements.

Table 5.12

Tabular Representation of the Third Solution for the  
Cyclic Graph Enumeration Example  
(Step 25 in Figure 5.8)

Period Length (Days)	Number of Periods	Number of Starts			
		A	B	C	D
3	1	1	0	0	0
2	2	0	0	1	1

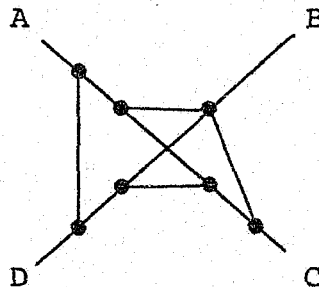


Figure 5.13

Third Cyclic Graph Solution  
(Step 25 in Figure 5.8)

Despite this, there are still two unassigned recreation periods: i.e., at step 18,  $N_D = N_0 - S_C - S_A - S_B - S_D = (1,2) - (1,0) - (0,0) - (0,0) - (0,0) = (0,2)$ . The examination of all days of the week without the assignment of all recreation periods is infeasibility condition 5 (INF5 in figure 5.8).

As a second example of the cyclic graph enumeration algorithm, consider the star diagram in figure 5.14. This diagram contains 12 recreation days distributed over the seven days of the week. Consider the problem of determining all cyclic graphs for this diagram if the 12 recreation days are partitioned into five clusters consisting of two 3-day

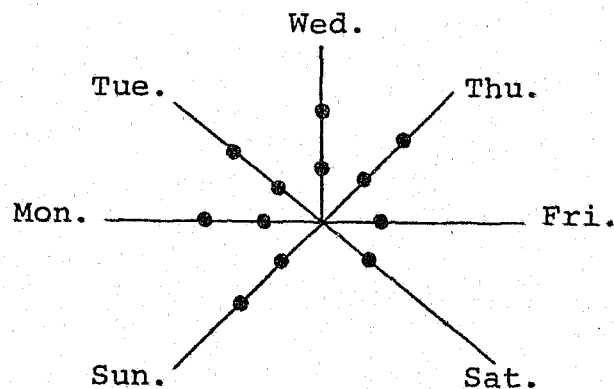


Figure 5.14

Star Diagram with Twelve Nodes

periods and three 2-day periods; i.e., if  $L = (3,2)$  and  $N_0 = (2,3)$ . The cyclic graph algorithm enumerates the 13 solutions to this problem (see figure 5.15) in only 126 steps. Each solution represents a unique distribution of the five recreation periods over the 12 recreation nodes.

#### 5.4.3 Enumeration Algorithm Performance Characteristics

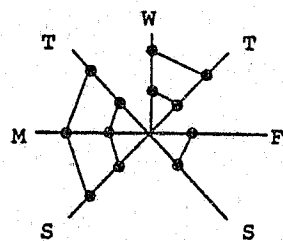
The performance characteristics of the cyclic graph algorithm for several sample problems are presented in tables 5.13 and 5.14.\* In each table, three star diagrams containing 8, 16, and 26 recreation days respectively are examined. In table 5.13, the recreation days are allocated as uniformly as possible over all seven days of the week; in table 5.14, the recreation days are distributed as if Friday and Saturday are busy days requiring more men on duty (and conversely permitting fewer men on recreation). Several recreation period partitions are used with each star diagram to provide comparative performance statistics.

Several observations are suggested by the data in tables 5.13 and 5.14:

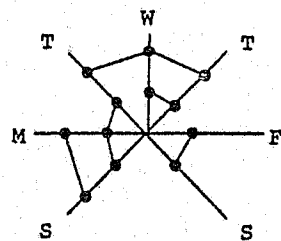
- (1) The amount of computer execution time required for a problem is proportional to the number of cyclic graphs to be enumerated.

---

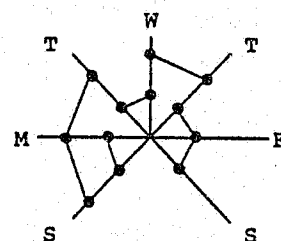
\*The data presented in columns 5 and 6 refer to the number and distribution of weekend recreation periods in schedules constructed from the cyclic graphs. A discussion of these properties is presented below in section 5.5.



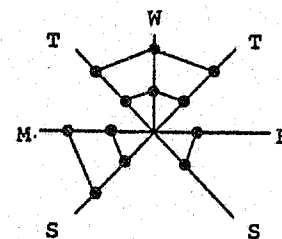
Solution 1



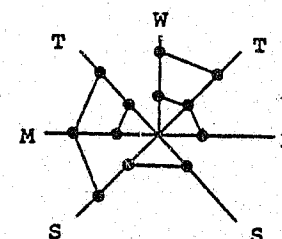
Solution 2



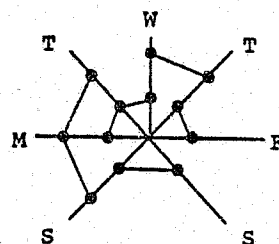
Solution 3



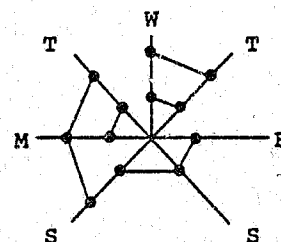
Solution 4



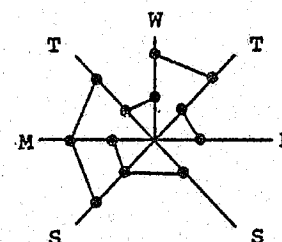
Solution 5



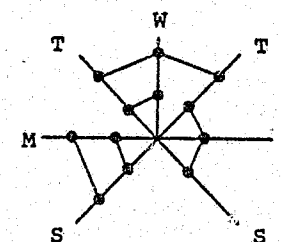
Solution 6



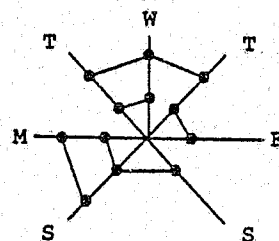
Solution 7



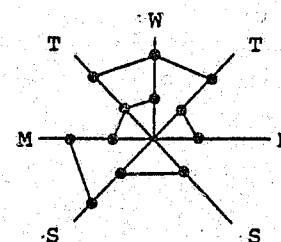
Solution 8



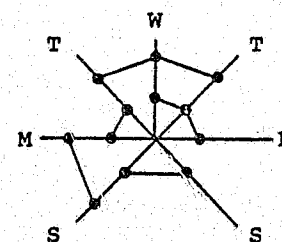
Solution 9



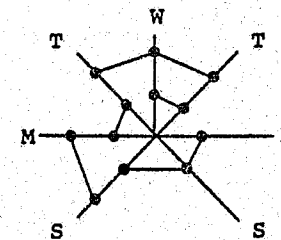
Solution 10



Solution 11



Solution 12



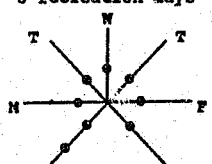
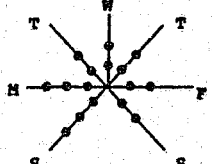
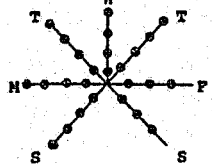
Solution 13

Figure 5.15

Thirteen Cyclic Graphs Enumerated for the Star Diagram  
in Figure 5.14 ( $L=(3,2)$  and  $P=(2,3)$ )

Table 5.13

Performance Characteristics of the Cyclic Graph Enumeration Algorithm for  
Ten Sample Problems with Uniform Recreation Day Distributions

Star Diagram	Problem No.	Recreation Period Length Vector L=	Number of Recreation Periods P=	Number of Cyclic Graphs	Computer CPU Time (Secs) *	Maximum No. of Weekend Recreation Periods	No. of Solutions with Max. No. of Weekend Rec. Periods
8 recreation days 	1.	(3,2)	(2,1)	3	1	1	3
	2.	(4,2)	(1,2)	3	1	1	3
16 recreation days 	3.	(3,2)	(4,2)	13	1	2	7
	4.	(3,2)	(2,5)	28	2	2	10
	5.	(4,3,2)	(1,2,3)	66	3	2	30
	6.	(4,3,2)	(2,2,1)	44	2	2	27
26 recreation days 	7.	(3,2)	(6,4)	77	3	3	19
	8.	(3,2)	(4,7)	140	6	3	40
	9.	(4,3,2)	(2,4,3)	393	20	3	191
	10.	(4,3,2)	(3,2,4)	454	25	3	184

\*Execution time only, based on a computer code written in FORTRAN IV and compiled with a FORTRAN G level compiler on an IBM 360/65 system.

Table 5.14

Performance Characteristics of the Cyclic Graph Enumeration Algorithm for  
Ten Sample Problems with Non-Uniform Recreation Day Distributions

Star Diagram	Problem No.	Recreation Period Length Vector $L=$	Number of Recreation Periods $P=$	Number of Cyclic Graphs	Computer CPU Time (Secs) *	Maximum No. of Weekend Recreation Periods	No. of Solutions with Max. No. of Weekend Rec. Periods
8 recreation days 	1.	(3,2)	(2,1)	2	1	0	2
	2.	(4,2)	(1,2)	0	1	0	0
16 recreation days 	3.	(3,2)	(4,2)	7	1	1	7
	4.	(3,2)	(2,5)	13	1	1	8
	5.	(4,3,2)	(1,2,3)	29	1	1	22
	6.	(4,3,2)	(2,2,1)	16	1	1	12
26 recreation days 	7.	(3,2)	(6,4)	31	1	2	17
	8.	(3,2)	(4,7)	70	3	2	30
	9.	(4,3,2)	(2,4,3)	133	5	2	2
	10.	(4,3,2)	(3,2,4)	109	4	2	79

\*Execution time only, based on a computer code written in FORTRAN IV and compiled with a FORTRAN G level compiler on an IBM 360/65 system.

- (2) The number of graphs found for a star diagram is dependent on the nature of both the length vector  $L$  and the partition vector  $N$ .
- (3) The number of graphs for a star diagram and a given length vector  $L$  increases as the number of recreation periods increases (i.e., as the average period length decreases).
- (4) The number of cyclic graphs for a given length vector  $L$  increases as the number of recreation days  $R$  increases or as the total number of recreation periods increases.
- (5) For a given number of recreation days  $R$ , the number of graphs decreases as the days become less uniformly distributed over the days of the week.

#### 5.4.4 Enumeration Algorithm Logic Summary

This section presents a summary of the algorithm to enumerate all cyclic graphs for a given star diagram and set of recreation periods. The following notation is utilized:

- $\{j\}$  - set of indices which identify each day of the week;  $j = 1, 2, \dots, 7$  (e.g.,  $j = 1$  for Monday,  $j = 2$  for Tuesday, ...,  $j = 7$  for Sunday)
- $\{i\}$  - set of indices which identify the sequence in which the  $d$  days are examined in order to find each graph;  $i = 1, 2, \dots, 7$ .

$D_i$  - name of the  $i^{\text{th}}$  selected day (e.g., if Tuesday is the first day examined  $D_1 = 2$ )

$A$  - set of days  $D_i$  which have been assigned specific start arrangements; initially set  $A$  is empty (i.e.,  $A = \phi$ ).

$U$  - set of days  $D_i$  which have not been assigned specific start arrangements

$SD_j$  - reduced star diagram obtained when day  $j$  is assigned a specific arrangement (original star diagram is  $SD_0$ )

$S_j = (s_{1j}, s_{2j}, \dots, s_{ij}, \dots, s_{kj})$  - the  $s_{ij}$  component in the  $S_j$  vector identifies the number of periods of length  $i$  which have been assigned to start on day  $j$ . The  $S_j$  vector identifies a specific start arrangement for day  $j$

$N_j = (n_{1j}, n_{2j}, \dots, n_{ij}, \dots, n_{kj})$  - the  $n_{ij}$  component in the  $N_j$  vector identifies the number of periods of length  $i$  which remain unassigned after a specific arrangement has been assigned to day  $j$ .  $N_0$  is the vector of the original set of recreation periods to be assigned,

$$N_j = N_0 - \sum_{\text{all } j=D_i \in A} S_j.$$

$B_j \max, B_j \min$  - upper and lower bounds on the total number of period starts which can be assigned to day  $j$ .

The enumeration algorithm consists of the following steps:

1. Initialization: Set  $i = 0, j = 0, A = \phi,$   
 $U = \{j | \text{all } j\}$
2. Determine limits on the number of starts for each day. Calculate  $\{B_j \max, B_j \min | j = 1, 2, \dots, 7\}$  using  $SD_j$  and  $N_j$

- (i) If  $B_{j \min} > B_{j \max}$  for any  $j$ , go to 3 (infeasibility condition 3).
- (ii) If  $B_{j \min} > 0$  for any  $j \in A$ , go to 3 (infeasibility condition 2).
- (iii) Go to 4.

3. Backtrack.

- (i) If  $i = 0$  or 1, terminate.
- (ii) If  $i > 1$ , move  $j = D_i$  from  $A$  to  $U$ , let  $i = i-1$ , go to 5.

4. Select day for step  $i$  in the sequence. Select day  $j'$  using  $\{j' | \max_j \{B_{j \min}\}, j = D_i \in U\}$

- (i) If  $U$  is empty, go to 3 (infeasibility condition 5).
- (ii) Let  $i = i+1$ ,  $D_i = j'$ , move  $j = D_i$  from  $U$  to  $A$ , go to 6.

5. Determine next start arrangement for day  $j$ .

Determine next  $S_j$  for  $D_i$  using partitioning algorithm (note  $j = D_i$ ).

- (i) If no  $S_j$  exists, go to 3.
- (ii) If  $S_j \not\subseteq N_{j-1}$ , go to 5 (infeasibility condition 1).
- (iii) Go to 6.

6. Determine reduced star diagram and unassigned period vector for day  $j$ . Determine  $SD_j$  and  $N_j$

(Use  $N_j = N_{j-1} - S_j$ ).

- (i) If  $SD_j = \phi$  (no nodes), go to 7.
- (ii) If  $SD_j$  requires negative number of nodes, go to 5 (infeasibility condition 4).
- (iii) Go to 2.

7. Record solution. Complete solution obtained.

Set  $\{S_j = 0 \mid \text{all } j \in U\}$ , record solution,  
go to 5.

#### 5.5 WEEKEND RECREATION PERIODS

The cyclic graph enumeration algorithm described above can be used to determine all cyclic graphs for a given star diagram and set of recreation periods. Among the graphs obtained, however, some may be considerably more desirable than others for use in the design of schedules. Although no formula is known for calculating the exact number of schedules that can be designed from each cyclic graph, it is possible to identify specific properties which every schedule derived from the same graph will possess. These properties can be used to identify cyclic graphs which yield more desirable schedules.

The computer design models described in this thesis use two preference measures which can be determined directly from each cyclic graph; these are (1) the number of weekend recreation periods, and (2) the number of recreation periods of each length that begin on each day of the week. The following discussion of these properties is presented in two parts. The first examines the relationship between the number of weekend recreation periods in a graph, and the distribution of recreation days on the Saturday and Sunday rays of the cyclic graph; the second describes the use of

the number of weekend recreation periods to accelerate the schedule design process.

#### 5.5.1 Weekend Classification of Recreation Periods

An important schedule property which is governed by the distribution of recreation periods over the days of the week is the number of weekend recreation periods. Each recreation period in a seven-day cyclic graph can be classified as one of three types:

- full weekend recreation period (f) - a recreation period which includes consecutive recreation days on Saturday and Sunday;
- partial weekend recreation period (p) - a recreation period which is not a full weekend period, and either begins on Sunday or ends on Saturday; or a
- non-weekend recreation period (n) - a recreation period which is not a full or partial weekend period (i.e., contains neither a Saturday or a Sunday).

The weekend-related characteristics of recreation periods having lengths of from one to six days are shown in table 5.15.

The number of full and partial weekend recreation periods which can be constructed from a star diagram is determined by the number of recreation days which have been allocated to Saturday and Sunday; i.e., by the number of nodes on the Saturday and Sunday rays,  $n_6$  and  $n_7$ . It

Table 5.15

Weekend Classification for Recreation Periods from  
One to Six Days in Length and Beginning on  
Each Day of the Week

Period Length (Day)	Start Day of the Period						
	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.
1	n	n	n	n	n	p	p
2	n	n	n	n	p	f	p
3	n	n	n	p	f	f	p
4	n	n	p	f	f	f	p
5	n	p	f	f	f	f	p
6	p	f	f	f	f	f	p

Note: f=full weekend recreation period, p=partial weekend recreation period, and n=non-weekend recreation period.

follows directly from the definitions above that the maximum number of full weekend periods,  $W_f$ , for any star diagram is given by:

$$\max W_f = \min\{n_6, n_7\}. \quad (5.4)$$

The maximum numbers of full weekend recreation periods possible for the sample problems illustrated in tables 5.13 and 5.14 are listed in column 5 of each table. It is important to note that the maximum number of full weekend periods is dependent only on the number of the recreation days

allocated to Saturday and Sunday, and is independent of the partition (i.e., vectors L and P) of recreation periods used to enumerate the cyclic graph. As a result, the maximum number of full weekend periods which a cyclic graph can possess can be "read" directly from the original allocation of recreation days. Hence, if the number of full weekend recreation periods is a valued schedule attribute, a schedule designer may decide to reallocate recreation days in order to increase the potential number of full weekend periods that can be scheduled, even at the expense of achieving a less "optimal" manpower allocation.

To illustrate, consider the manpower allocation for a 10-week schedule in table 5.16. Despite the presence of five recreation days on Sunday ( $n_7 = 5$ ), a maximum of only one full weekend recreation period can be scheduled during the entire 10 weeks because of the single recreation day allocated to Saturday ( $n_6 = 1$ ). Table 5.17 shows a reallocation of the recreation watches (one recreation day has been shifted from Sunday to Saturday) which allows up to two full weekend periods to be scheduled. Other redistributions of the recreation days in table 5.16 could also be used which would permit more than two full weekend periods to be constructed. Any of these reallocations, however, would be achieved at the expense of a slight departure from the original "optimal" manpower allocation. It should also

Table 5.16

Manpower Allocation for a Ten-Week PR Schedule  
with a Maximum of One Full Weekend  
Recreation Period per Rotation Period

Number of Men	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Total
On Duty	6	7	7	8	8	9	5	50
On Recreation	4	3	3	2	2	1	5	20

Table 5.17

Manpower Allocation for a Ten-Week PR Schedule  
with a Maximum of Two Full Weekend  
Recreation Periods per Rotation Period

Number of Men	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Total
On Duty	6	7	7	8	8	8	6	50
On Recreation	4	3	3	2	2	2	4	20

be noted that formula (5.4) indicates only the theoretical upper limit on the number of full weekend periods; the actual number of weekend periods for which feasible schedules can be designed may be less.

Although the number of full weekend recreation periods is an important schedule attribute, it cannot by itself be used successfully to provide an adequate ranking for the

cyclic graphs which can be enumerated from a single star diagram. As an example, consider the 20 sample cases presented in tables 5.13 and 5.14; column 6 in each table indicates the number of cyclic graphs enumerated that possess the maximum number of full weekend periods; in some cases well over half of all of the graphs obtained possess the maximum number of weekend periods.

The number of partial weekend recreation periods,  $W_p$ , does not provide additional evaluative information about each graph once the number of full weekend recreation periods,  $W_f$ , is known because the number of full and partial weekend periods are not independent measures. That is, for any seven-ray cyclic graph containing only recreation periods which are less than seven days in length, the following identity relating  $W_f$  and  $W_p$  holds:

$$2W_f + W_p = n_6 + n_7 \quad W_f, W_p = 0, 1, 2, \dots \quad (5.5)$$

Result (5.5) is easily shown. Every full weekend period must by definition use two weekend recreation days and since each period is less than seven days in length, no more than two weekend days are used. All recreation days on Saturday and Sunday which are not used in a full weekend period must by definition be part of a partial weekend period. Since every partial weekend period is also less than seven days long, each uses exactly one weekend recreation day and formula (5.5) follows.

Since both  $n_6$  and  $n_7$  are known from the original star diagram, once  $W_f$  has been specified,  $W_p$  is determined. As a result all cyclic graphs which have the maximum number of full weekend recreation periods also have identical numbers of partial weekend recreation periods. Result (5.5) also indicates that if the maximum number of full and partial weekend periods is desired the schedule should contain no full weekend periods; i.e.,

$$W_f = 0 \text{ and } W_p = n_6 + n_7.$$

Formulas (5.4) and (5.5) also suggest that the number of classes of cyclic graphs, when classified in terms of the number of full and partial weekend recreation periods, produced from any star diagram is relatively small. As an example, consider the daily recreation allocation in table 5.17. The maximum number of full weekend periods for any cyclic graph based on this allocation is two ( $W_f = \min\{2,4\} = 2$ ); and every such graph also possesses two partial weekend periods ( $W_p = n_6 + n_7 - 2W_f = 2 + 4 - 2(2) = 2$ ). Only two other distinct weekend-related classes of cyclic graphs are possible for this allocation: one class consisting of graphs with one full weekend period and four partial weekend periods ( $2W_f + W_p = 2(1) + 4 = 2 + 4$ ), and another class consisting of

cyclic graphs with no full weekend periods and six partial weekend periods. As a result, every cyclic graph produced for the allocation in table 5.17 belongs to one of only three distinct weekend-related classes. This example also illustrates that the number of weekend-related classifications,  $C_W$ , is given by,

$$C_W = 1 + \max W_f = 1 + \min\{n_6, n_7\}. \quad (5.6)$$

The ability to classify graphs into  $C_W$  classes according to their  $W_f$  values is the basis for the acceleration techniques discussed below.

#### 5.5.2 Acceleration of the Design Process Based on the Number of Full Weekend Recreation Periods

Schedule properties which can be identified in cyclic graphs can be used to accelerate the enumeration procedures used to design optimal one-shift PR schedules in two ways: first by identifying cyclic graphs that can not yield optimal schedules, and as a result, need not be used in subsequent steps of the design process, and second, by determining the most efficient order in which the remaining graphs should be examined (i.e., by identifying graphs which are most likely to produce an optimal schedule).

The usefulness of the cyclic graph properties discussed in earlier sections of this chapter in accelerating the schedule design process is dependent upon the relative

importance assigned to each property (i.e., the higher the preference rating given to a schedule property, the more effective is the acceleration procedures that utilizes that property).

All of the manpower schedules described in this thesis were designed using the number of full weekend recreation periods (hereafter referred to as WRP) as the most important schedule attribute. The other schedule attribute that can be obtained from cyclic graphs (i.e., the number of recreation periods of each length that begin on each day of the week) was used in the lexicographic decision model only to distinguish between schedules which had identical values for (1) the number of weekend recreation periods, (2) the maximum number of consecutive working weekends, (3) the maximum length work period, and (4) the number of maximum length work periods (see section 2.3).

The remainder of this section describes the use of the number of WRPs to accelerate the schedule design procedure. Two techniques are presented. The first uses the following logic: as each cyclic graph is enumerated, it is used to construct a separation matrix. This matrix is then used with a branch-and-bound algorithm to enumerate feasible one-shift PR schedules. The design logic does not enumerate another cyclic graph until all feasible schedules associated with the first graph have been found. This process continues until all cyclic graphs have been examined.

As each schedule is enumerated, it is compared with the current optimal (CO) schedule using the lexicographic decision model. The first attribute for comparison between each schedule and the CO is the number of WRPs. Let  $W_f^{CO}$  and  $W_f$  represent the number of WRPs for the CO and latest schedule respectively. If  $W_f^{CO} > W_f$ , the latest schedule is rejected; if  $W_f^{CO} = W_f$ , other schedule attributes must be compared; and finally, if  $W_f > W_f^{CO}$ , the latest schedule becomes the new current optimal schedule.

The usefulness of this procedure lies in the fact that the  $W_f$  value for a schedule can be determined directly from its generating cyclic graph without the necessity of constructing the separation matrix and using the branch-and-bound algorithm. To illustrate, if a cyclic graph is produced with a  $W_f$  value which is strictly less than  $W_f^{CO}$ , then none of the schedules which can be generated from this graph can replace the CO schedule, and the graph need not be examined further. Hence, the  $W_f^{CO}$  value can be used to screen all schedules generated from the same cyclic graph merely by comparing  $W_f^{CO}$  with the  $W_f$  value of the graph. Stated in another way, a cyclic graph is examined further if and only if its  $W_f$  satisfies the inequality:  $W_f \geq W_f^{CO}$ .

In summary, the acceleration procedure consists of the following steps:

1. Set  $W_f^{CO} = -1$
2. Generate cyclic graph C:
  - a. if none exist, stop.
  - b. go to step 3.
3. Examine  $W_f^{CO}$ :
  - a. if  $W_f^{CO} \geq 0$ , go to step 4.
  - b. go to step 5.
4. Compare  $W_f$  and  $W_f^{CO}$ :
  - a. if  $W_f < W_f^{CO}$ , go to step 2.
  - b. if  $W_f \geq W_f^{CO}$ , go to step 5.
5. Construct the separation matrix for C;  
use the branch-and-bound algorithm, and  
for each schedule produced compare  $W_f$  with  $W_f^{CO}$ :
  - a. if  $W_f < W_f^{CO}$ , retain  $W_f^{CO}$ ,
  - b. if  $W_f = W_f^{CO}$ , compare current  
schedule and  $W_f^{CO}$  schedule on other  
schedule attributes,
  - c. if  $W_f > W_f^{CO}$ , replace  $W_f^{CO}$  schedule  
with current schedule.
6. Go to step 2.

This acceleration technique can be further enhanced if the order in which the cyclic graphs are examined is manipulated so that a schedule with a high  $W_f$  value is likely to be found early in the design process. This strategy has been tested in the following manner. Instead of examining each cyclic graph as it is produced, all of the graphs are first enumerated and stored.\* As each graph is produced, it is assigned to

---

\*This is not as difficult as it may appear. Star diagrams with 20 or fewer nodes usually produce fewer than 200 graphs (assuming all recreation periods are at least two days long).

one of  $C_W$  classes based on its  $W_f$  value ( $C_W = 1 + \min\{n_6, n_7\}$ ). The classes are then rank ordered beginning with the highest  $W_f$  value (i.e., all classes are represented by the set  $\{C_1, C_2, \dots, C_{C_W}\}$  where  $C_i$  represents the  $i^{\text{th}}$  ranked set of graphs).

The graphs are now examined in sequence according to their class until the first schedule is found (i.e., all graphs in class  $C_1$  are examined first -- if a schedule is found, stop; if no schedule is found, all graphs in  $C_2$  are examined and so on). Enumeration of the first schedule eliminates the need for examining graphs in any of the lower ranking classes since every schedule based on those graphs must, by definition, have  $W_f < W_f^{\text{CO}}$ . Hence, if a schedule is found from a graph in  $C_1$ , then only the remaining graphs in  $C_1$  must be examined since they may also produce schedules with exactly  $W_f^{\text{CO}}$  weekend recreation periods. Each such schedule must then be compared with the current optimal schedule on the basis of other preference measures.

In summary, this procedure consists of the following steps:

1. Enumerate all cycle graphs and categorize each according to its  $W_f$  value. Define  $C_1$  as the set of all graphs with maximum  $W_f$  values,  $C_2$  as the set of all graphs with  $W_f - 1$  weekend periods, etc.
2. Set  $i = 1$ .

3. Examine all graphs in  $C_i$ :
  - a. if one or more schedules are found, determine the best schedule in  $C_i$  using the lexicographic decision model. Stop.
  - b. if no schedule is found, set  $i = i+1$ , go to step 4.
4. Compare  $i$  with  $C_W$ :
  - a. if  $i \leq C_W$ , go to step 3.
  - b. if  $i > C_W$ , no schedules exist, stop.

## 6. THE ENUMERATION OF ONE-SHIFT CYCLIC PR SCHEDULES

### 6.1 INTRODUCTION

This chapter describes an implicit enumeration algorithm to generate cyclic one-shift PR schedules based on the properties of recreation periods in a cyclic graph. The relationship between work period lengths and each pair of ordered recreation periods within a graph is utilized to introduce the concept of a separation matrix. This matrix serves as a screening device to eliminate schedules which contain one or more work periods of unacceptable length. The enumeration algorithm uses a series of modified separation matrices to generate preferable one-shift schedules. Several modifications to the enumeration procedure are also described which can, under certain circumstances, significantly reduce the computational effort required.

The remainder of this chapter is divided into four parts. Section 6.2 briefly examines the number of feasible schedules that can be enumerated from a single cyclic graph. The discussion illustrates the impossibility of using manual methods for all but the smallest problems (i.e., shift tours that are only three or four weeks long), and the need for an efficient enumerating algorithm. The concepts and use of separation matrices are described in sections 6.3 through 6.5. The enumeration algorithm itself is presented in section 6.6 and illustrated with a detailed sample

problem. Three acceleration techniques are described in section 6.7.

## 6.2 NUMBER OF FEASIBLE SCHEDULES FROM EACH CYCLIC GRAPH

Each cyclic graph enumerated for a set of recreation periods represents a unique allocation of the periods for the daily distribution of recreation days represented in a star diagram. To obtain feasible one-shift schedules, the recreation periods within the cyclic graph must be "arranged" in a feasible manner over the weeks of the shift schedule. As an example, consider the cyclic graph in figure 6.1.

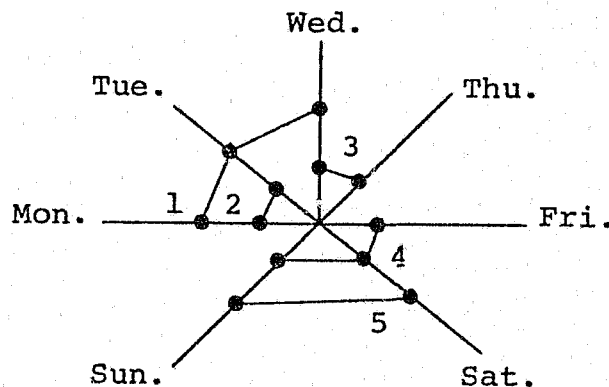


Figure 6.1

Cyclic Graph with Five Recreation Periods

The graph contains 12 recreation days which are joined together into five recreation periods. Assume that these five periods are to be distributed over a six-week shift schedule containing 30 work days and 12 recreation days. One feasible arrangement of the five recreation periods in a six-week schedule is shown in figure 6.2. Recreation Period 1, which begins on Monday in the cyclic graph, is assigned to week 1. Period 2, which also begins on Monday, is assigned to week 2; and periods 3, 4, and 5 are assigned

	M	T	W	T	F	S	S
1	<sup>1</sup> R	R	R				
2	<sup>2</sup> R	R					
3			<sup>3</sup> R	R			
4					<sup>4</sup> R	R	R
5						<sup>5</sup> R	R
6							

Figure 6.2

PR Schedule Based on the Five Recreation  
Periods in Figure 6.1

respectively to weeks 3, 4, and 5 in the schedule. It is important to note that this schedule satisfies the allocation and period characteristics defined by the cyclic graph in figure 6.1; i.e., the number of recreation days on each day of the week in the schedule equals the number of nodes on the corresponding ray of the cyclic graph, and the length and start day of each period in the schedule coincide with a period of the same length and start day in the cyclic graph. Hence, the schedule in figure 6.2 contains all of the information needed to reconstruct the cyclic graph in figure 6.1. This example illustrates that each one-shift PR schedule uniquely defines one and only one cyclic graph. The inverse of this statement, however, is generally not true. Frequently, large numbers of schedules can be produced from a single cyclic graph. It can also occur that no feasible schedules can be derived from a graph.

As an example of an alternate schedule derived from a given graph, figure 6.3 illustrates a second one-shift schedule generated from the cyclic graph in figure 6.1. Although this schedule differs from the schedule shown in figure 6.2, it retains all of the basic properties imposed by the generating cyclic graph; i.e., the number of recreation days on each day of the week matches the daily allocation of recreation days in the graph, and the length and start day characteristics for each of the five recreation periods in figure 6.3 are identical to those in the generating graph.

	M	T	W	T	F	S	S
1	<sup>1</sup> R	R	R				
2			<sup>3</sup> R	R		<sup>5</sup> R	R
3							
4					<sup>4</sup> R	R	R
5							
6	<sup>2</sup> R	R					

Week

Figure 6.3

PR Schedule Based on the Five Recreation  
Periods in Figure 6.1

The number of feasible arrangements of the recreation periods (where each arrangement corresponds to a distinct schedule) is related to the number of recreation periods  $n$  in the graph, and the number of weeks  $w$  in the schedule. As an example, in the six-week schedules shown in figures 6.2 and 6.3, the start days for recreation periods 1 and 2 could have been placed in any two of the six Mondays within the schedule. Similarly, periods 3, 4, and 5 each had six

possible locations for their respective start days. Assuming that the position of each period within the schedule is independent of the placement of the other periods, an upper bound on the total number of distinct, one-shift schedules (i.e., arrangements of the periods)  $\hat{N}$  that can be generated for a w-week schedule with n recreation periods is given by the product of the total number of locations for each period; i.e.,  $\hat{N} = \prod_{i=1}^n W = W^n$ . For cyclic one-shift schedules, it can be assumed without loss of generality that period 1 is always assigned to week 1 in the schedule, leaving only n-1 periods to be assigned; consequently  $\hat{N}_C = W^{n-1}$ . These results indicate that the five recreation periods in the cyclic graph in figure 6.1 can produce, at the most, a total of  $\hat{N}_C = 6^{5-1} = 1,296$  one-shift cyclic schedules.

Since the cyclic graph in figure 6.1 has a pair of period disjoint rays\*, (the graph has two pairs: (Sunday, Monday) and (Thursday, Friday)) the exact number of distinct non-cyclic schedules N can be calculated using the equation derived in appendix 10.2; i.e.,

$$N = \prod_{i=1}^7 \frac{(W-R_i+n_{i.})!}{(W-R_i)! \prod_{j=1}^k (n_{ij}!)} \quad \begin{array}{l} w, R_i, n_{ij} \text{ integer} \\ 0 \leq n_{i.} \leq W-R_i \end{array} \quad (6.1)$$

---

\* Adjacent rays in a cyclic graph not joined by a period line (see appendix 10.2).

where

$R_i$  = number of nodes on ray  $i$

$n_{ij}$  = number of recreation periods of type  $j$  (length) that start on day  $i$ ;  $j = 1, 2, \dots, k$  and  $i = 1, 2, \dots, 7$ .

$n_{i.}$  = total number of periods that start in day  $i$ ;

$$n_{i.} = \sum_j n_{ij}.$$

Equation (6.1) indicates that the five periods in the cyclic graph in figure 6.1 can be used to produce 4,500 distinct, non-cyclic one-shift schedules; and 750 cyclic schedules ( $N_c = N/W$ ).

As a second example, consider the cyclic graph in figure 6.4. This graph contains 20 recreation days combined into eight recreation periods: four 3-day periods and four 2-day periods; and Tuesday and Wednesday are period disjoint rays. Using  $R = 20$  and  $n = 8$  in equation (6.1) indicates that 6,531,840 distinct non-cyclic and 653,184 distinct cyclic 10-week schedules can be created.

These examples illustrate the exponential-like growth in the number of distinct schedules that can be produced from a cyclic graph as the number of weeks and recreation periods increase; and the enormous number of schedules that can exist for even relatively small cyclic graphs. Although equation (6.1) is not applicable for cyclic graphs without a pair of period disjoint rays, it is reasonable to believe

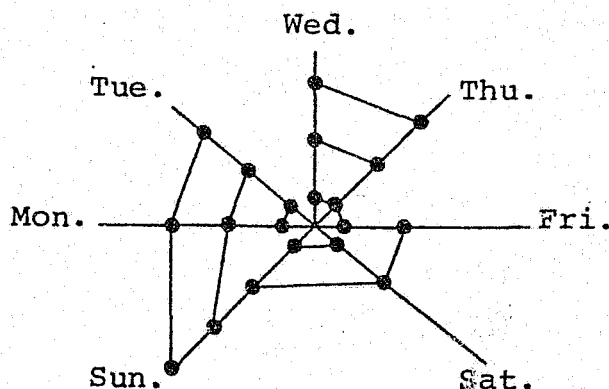


Figure 6.4

Cyclic Graph with Eight Recreation Periods

that these graphs will also contain approximately the same numbers of schedules as has been illustrated above. This conjecture is strengthened in section 6.5.3 where the relationship between the number of distinct schedules associated with a cyclic graph (whether containing period disjoint rays or not) and the number of tours for a modified version of the travelling salesman problem is established.

### 6.3 WORK PERIODS DEFINED BY THE SEQUENCE OF RECREATION PERIODS

On the surface, the enormous number of possible schedules that can be produced for even relatively small cyclic graphs

presents a discouraging picture. Complete enumeration, even done implicitly, can be computationally unreasonable when hundreds of thousands of schedules must be examined. In addition, the qualitative nature of many of the variables or characteristics that are important in selecting one schedule over another make the use of algorithmic schemes which utilize well-defined quantitative objective functions impractical. Both of these difficulties suggest that if desirable schedules are to be algorithmically determined, methods must be found which significantly reduce the universe of schedules to be examined, and that these reduction methods should be based on constraints which reflect desirable schedule qualities.

One of the most useful and important schedule characteristics which can be used to compare and discard large subsets of schedules is the set of lengths of the individual work periods associated with each schedule. Individual work periods in a schedule may be conveniently defined in terms of the recreation periods which they separate. As an example, in figure 6.2 the work period in week 1 separates recreation periods 1 and 2. The length of the work period, four days, is the number of work days separating the two recreation periods. Hence, the work period is defined by the pair of recreation periods (1,2). The lengths and defining recreation periods for each of the work periods in figure 6.2 are

summarized in table 6.1. The work periods for the schedule in figure 6.3 are similarly summarized in table 6.2.

Table 6.1

Work Period Lengths for the  
Schedule in Figure 6.2

Work Period	Defining Recreation Periods	Number of Days Between the Recreation Periods (Work Period Length)
1.	(1,2)	4
2.	(2,3)	7
3.	(3,4)	7
4.	(4,5)	5
5.	(5,1)	7
Total Number of Work Days		30

Both schedules contain the same five recreation periods and a total of 30 work days. The five work periods in each schedule, however, are seen to vary considerably in length. In figure 6.2, the work periods range from four to seven days in length; three of the periods are exactly seven days long. In figure 6.3, however, the work period lengths display more variance, ranging in length from a minimum of one day to a maximum of 11 days. On the basis of work

Table 6.2

Work Period Lengths for the  
Schedule in Figure 6.3

Work Period	Defining Recreation Periods	Number of Days Between the Recreation Periods (Work Period Length)
1.	(1,3)	6
2.	(3,5)	1
3.	(5,4)	11
4.	(4,2)	7
5.	(2,1)	5
Total Number of Work Days		30

period lengths alone, the second schedule would be less desirable (and probably unacceptable) to most schedule designers.

The process by which work period lengths are used to screen out large subsets of schedules, and also to measure schedules that are produced is central to the remainder of this chapter. In the sections that follow, the concept, construction, and use of a unique separation matrix

associated with each cyclic graph are described. This matrix is used to identify schedules which satisfy upper and lower limits on work period lengths, and as the basis for the implicit enumeration algorithm to determine desirable schedules.

#### 6.4 SEPARATION MATRIX

Associated with every cyclic graph with  $n$  recreation periods is a  $n \times n$  matrix  $S$ , a separation matrix. Each entry,  $s_{ij}$ , in the matrix represents the number of days in the work period defined by the ordered pair of recreation periods  $(i,j)$ : i.e., the entry  $s_{ij}$  is the number of consecutive work days that separate the end of period  $i$  from the beginning of period  $j$ . As an example, in the cyclic graph in figure 6.1, period 1 ends on Wednesday and period 2 begins on Monday. The  $s_{12}$  entry for the work period defined by the recreation period pair  $(1,2)$  equals four since four work days separate the end of period 1 from the beginning of period 2. Figure 6.5 shows the complete separation matrix associated with the cyclic graph in figure 6.1.

To construct the matrix shown in figure 6.5, the minimum separation values were used for each entry, (i.e., the minimum number of days separating the end of period  $i$  from the beginning of period  $j$ ). Diagonal entries,  $s_{ii}$ , are not defined, and are represented by a dash by figure 6.5. A separation matrix in which minimum separating values are

End of Recreation Period	Beginning of Recreation Period				
	1	2	3	4	5
1	-	4	6	1	2
2	5	-	0	2	3
3	3	3	-	0	1
4	0	0	2	-	5
5	0	0	2	4	-

Figure 6.5

Elementary Separation Matrix for the  
Cyclic Graph in Figure 6.1

used for each pair of ordered recreation periods in graph C is said to be the elementary separation matrix for C. By construction, each graph has only one elementary separation matrix. Each entry  $s_{ij}$  in an elementary matrix must lie in the range

$$0 \leq s_{ij} \leq N_R - 1, \quad i \neq j$$

where  $N_R$  equals the number of rays in the graph. Hence, for graphs based on a seven-day week,  $0 \leq s_{ij} \leq 6$  for all  $i \neq j$ .

Each entry in the elementary separation matrix represents the minimum number of consecutive work days that can exist

between each ordered pair of recreation periods. For most cyclic graphs, several additional values are possible for each entry due to the fact that two recreation periods can be separated by the minimum number of days plus one, two, or more weeks. As an example, figure 6.6 indicates the minimum length work period that can exist if recreation period 1 is immediately followed by period 2. As indicated in the matrix in figure 6.5, the minimum  $s_{1,2}$  value is four days. This minimum separation value can only occur if

	M	T	W	T	F	S	S
1	<sup>1</sup> R	R	R	(4 work days)			
2	<sup>2</sup> R	R					
3							
4							
5							
6							

Week

Figure 6.6

Placement of Recreation Periods 1  
and 2 for Minimum Separation



**CONTINUED**

**3 OF 6**

periods 1 and 2 appear in consecutive weeks of a schedule. An alternate value for  $s_{12}$  can be obtained if periods 1 and 2 are positioned as shown in figure 6.7. In this alternate configuration, the recreation periods are now separated by 11 consecutive work days; i.e., they are separated by the minimum separation value (4 days) plus one week (7 days) and, hence  $s_{12} = 4+7 = 11$ . A third value for  $s_{12}$  is possible if recreation period 2 is placed in week 4; then  $s_{12}$  becomes 18 (i.e.,  $s_{12} = 4+7+7 = 18$ ). In general, the values for each entry in a separation matrix can be

	M	T	W	T	F	S	S
1	1 R	R	R	(4 work days &			
2			7 work days )				
3	2 R	R					
4							
5							
6							

Figure 6.7

Placement of Recreation Periods 1 and 2  
for Minimum Separation Plus One Week

represented by

$$s_{ij} = s'_{ij} + kN_R, \quad k = 0, 1, 2, \dots$$

where  $s'_{ij}$  indicates the value in the elementary separation matrix. The number of multiples ( $k$ ) of  $N_R$  that can be added to the  $s'_{ij}$  value depends on the number of weeks in the schedule, the number of recreation periods, and the entry value itself.

Thus, in addition to the unique elementary separation matrix for each cyclic graph, there also exist numerous other separation matrices, each of which contains one or more entries which are not equal to their elementary matrix values. The total number of distinct separation matrices that can be generated from a single cyclic graph can be quite large; if each entry in an  $n \times n$  separation matrix (there are  $n(n-1)$  non-diagonal entries in each matrix) has two possible values, there are  $2^{n(n-1)}$  distinct matrices. Hence, a cyclic graph with only five recreation periods can have over one million distinct separation matrices associated with it.\* Despite the large number of matrices that may exist for a given graph, it can be shown that all acceptable schedules can, in fact, be enumerated from only one matrix. The selection and use of that matrix is the subject of the following sections.

---

\*Assuming exactly two alternative values for each non-diagonal entry in the  $5 \times 5$  matrix produces  $2^{5 \times 4} = 2^{20} = 1,048,576$ .

## 6.5 USE OF THE SEPARATION MATRIX TO CONSTRUCT ONE-SHIFT, CYCLIC SCHEDULES

In this section, a sample work schedule is used to illustrate how all of the information required to construct PR schedules can be summarized by designating selected entries in a separation matrix. The sample schedule is also used to identify properties of the designated entries which are used in the enumeration algorithm.

The schedule, based on the cyclic graph in figure 6.1, is shown in figure 6.8. The length and defining recreation periods for each work period within this schedule are identified in table 6.3. The separation matrix to be discussed, shown in figure 6.9, was obtained by adding seven days to selected entries in the elementary matrix introduced

	M	T	W	T	F	S	S
1	1 R	R	R				
2			3 R	R			
3					4 R	R	R
4						5 R	R
5							
6	2 R	R					

Figure 6.8

PR Schedule Based on the Cyclic  
Graph in Figure 6.1

Table 6.3

Work Period Lengths for the  
Schedule in Figure 6.8

Work Period	Defining Recreation Periods	Number of Days Between the Recreation Periods (Work Period Lengths)
1.	(1,3)	6
2.	(3,4)	7
3.	(4,5)	5
4.	(5,2)	7
5.	(2,1)	5
Total Number of Work Days		30

Beginning of  
Recreation Period

		1	2	3	4	5
End of Recreation Period	1	-	4	(6)	* 8	* 9
	2	(5)	-	* 7	* 9	*10
	3	*10	*10	-	(7)	* 8
	4	* 7	* 7	* 9	-	(5)
	5	* 7	(7)	* 9	4	-

Figure 6.9

Modified Separation Matrix Obtained from the  
Elementary Matrix in Figure 6.5

in figure 6.5. The altered entries are identified with an asterisk in the upper left-hand corner of their cells. The selection of this particular separation matrix is discussed below.

All of the information contained in table 6.3 can be indicated by designating certain entries in the separation matrix shown in figure 6.9. As an example, the first work period in table 6.3 is six days long and is defined by the ordered pair of recreation periods (1,3). This information can be indicated in the matrix by circling the (1,3) entry. In a similar manner, the information associated with each of the remaining work periods can also be recorded in the matrix by circling the appropriate entries. After the five matrix entries have been designated, the separation matrix in figure 6.9 and cyclic graph in figure 6.1 contain all of the information that is needed to construct the work schedule in figure 6.8; the cyclic graph specifies the start day and length for each recreation period, and the circled entries in the separation matrix describe the sequence of recreation periods and the number of consecutive work days between each pair of adjacent recreation periods.

The significance of this example is that it illustrates the fact that in determining alternate schedules from a given cyclic graph, the only schedule properties that vary from schedule to schedule are (1) the sequence of recreation periods, and (2) the lengths of the separating work periods;

and, as noted above, both of these properties can be described by a set of designated entries in a separation matrix constructed from the given cyclic graph. Hence, the enumeration of all feasible schedules for a given cyclic graph with  $n$  recreation periods is equivalent to the enumeration of all possible sets of  $n$  entries from all of the separation matrices that can be produced from the graph. Quite obviously, however, if no means were available to limit the scope of this equivalent enumeration process, no particular advantage would have been gained. Fortunately, it is not necessary to examine all separation matrices that can be constructed for a given cyclic graph. In fact, it is easily shown that all acceptable schedules can be enumerated from only one matrix. Selection of that matrix is discussed below.

#### 6.5.1 Selection of the Appropriate Separation Matrix

The number of separation matrices to be examined for each cyclic graph can be limited by setting upper ( $U_W$ ) and lower ( $L_W$ ) limits on work period lengths within each schedule. Setting limits on work period lengths is equivalent to specifying upper and lower limits on the value of each separation matrix entry  $s_{ij}$ , (i.e.,  $L_W \leq s_{ij} \leq U_W$  for all  $i, j$ ,  $i \neq j$ ). As indicated in table 6.4, the range of the work period lengths ( $\text{range} = U_W - L_W$ ) determines the number of

distinct values that may exist for each entry in matrices constructed for cyclic graphs with  $N_R$  rays.

Table 6.4

Number of Distinct Values for  
Each Separation Matrix Entry

Work Period Range ( $U_W - L_W$ )	Number of Distinct Values for Each Separation Matrix Entry
$0 \leq U_W - L_W < N_R - 1^*$	One or none
$U_W - L_W = N_R - 1$	One
$U_W - L_W > N_R - 1$	One or more

\* $N_R$ =number of rays in the cyclic graph.

For all of the applications presented in this paper,  $N_R = 7$ ; thus for work period ranges up to six days no more than one acceptable value can exist for each matrix entry, and, as a result, only one separation matrix will exist for each cyclic graph.

As an example, the separation matrix in figure 6.9 is obtained from the elementary matrix in figure 6.5 if the upper limit on work period lengths is ten days ( $U_W = 10$ ) and the lower limit is four days ( $L_W = 4$ ). Since the range is exactly six days, only one acceptable value is obtained for each  $S_{ij}$  entry. As a result, this one matrix contains all of the information necessary to enumerate all acceptable schedules (i.e., all schedules based on the cyclic graph in figure 6.1 which contain work periods that are between four and ten days in length).

If a work period range of less than six days is used, there may not be an acceptable value for every matrix entry. For example, if the work period limits had been set at  $U_W = 9$  days and  $L_W = 4$  days above, no acceptable values would have been found for entries (2,5), (3,1) and (3,2). In such a case, these entries could not be used in the construction of a schedule and the number of schedules (i.e., sequences of recreation periods) based on this matrix would be correspondingly reduced.

#### 6.5.2 Characteristics of the Designated Matrix Entries Corresponding to Feasible Schedules

In addition to limiting the search for acceptable schedules to a single separation matrix by designating upper and lower limits on work period lengths, the enumeration of all schedules for a given cyclic graph can be accelerated by utilizing characteristics which are common

to every set of designated matrix entries which describes a feasible schedule. These characteristics are:

- (1) the sum of the values of the designated entries in the matrix must equal the number of work days in the schedule; and
- (2) the separation matrix must contain one and only one designated entry in each row and column.

The first characteristic is a quantitative statement of the correspondence which must exist between the work period lengths in a feasible schedule and the corresponding set of designated matrix entries. The second characteristic reflects the requirement that each recreation period of the cyclic graph be used once and only once in a feasible schedule. As an example, consider the schedule shown in figure 6.8, and the corresponding set of designated matrix entries shown in figure 6.9. This set of entries possesses both of the characteristics identified above: the values of the five entries sum to 30 work days, and one and only one designated entry appears in each row and column of the matrix. The first characteristic insures that the five work periods and five recreation periods produce a six-week schedule (42 days long). Since the five recreation period lengths sum to 12 days, the five work periods must contribute exactly 30 days. The appearance of one designated entry in each row and column of the matrix insures that each recreation period appears once and only once in the schedule.

This example illustrates that both characteristics are necessary properties (i.e., no feasible schedules will be found for any set of entries which does not possess both properties). Although necessary, the properties are not sufficient (i.e., not every set of matrix entries which possesses both properties will define a feasible schedule).

In the following section, these two feasibility properties are used to develop an analogy between feasible tours for the travelling salesman problem and feasible solutions to the one-shift scheduling problem. This analogy is used to identify an additional feasibility requirement for each set of matrix entries.

#### 6.5.3 Analogy of the One-Shift Scheduling Problem to the Travelling Salesman Problem

In the simplest statement of the travelling salesman problem, a salesman is required to visit  $n$  cities, beginning from his home in city 1 and visiting each of the other  $n-1$  cities once and only once before returning home. The salesman's problem is to select the sequence of cities (tour) which will minimize the total distance travelled.

Associated with each  $n$ -city travelling salesman problem is a  $n \times n$  distance matrix,  $D$ , in which each matrix entry,  $\{d_{ij}\}$ , equals the travel distance between two cities ( $d_{ij}$  equals the distance from city  $i$  to city  $j$ ). It is not necessarily true that  $d_{ji} = d_{ij}$ ; i.e., the travel distance between two cities may differ with travel direction.

Associated with each visit sequence to the  $n$  cities is a unique set of  $n$  distance matrix entries which are defined by the travel sequence. The total travel distance of each sequence equals the sum of these  $n$  entries, and the optimal travel sequence defines the set of  $n$   $d_{ij}$ 's with the lowest sum.

As an example, figure 6.10 contains a distance matrix for a five-city problem. The matrix contains one row and column for each city and displays the travel distances

		City				
Home		1	2	3	4	5
City	Home 1	-	4	(6)	8	9
	2	(5)	-	7	9	10
	3	10	10	-	(7)	8
	4	7	7	9	-	(5)
	5	7	(7)	9	4	-

Figure 6.10

Distance Matrix for a Five-City  
Travelling Salesman Problem

between each pair of cities for both directions of travel (the entries on the main diagonal are not used). For this particular problem, the minimum (optimal) travel distance is 30, which can be achieved with four distinct travel sequences. Using city 1 as the home city, the four sequences are:

- 1 {1,2,3,4,5,1}
- 2 {1,2,3,5,4,1}
- 3 {1,3,4,5,2,1}
- 4 {1,3,5,4,2,1}

The five matrix entries associated with the number 3 sequence are circled in the distance matrix.

In general, an  $n$ -city travelling salesman problem has  $(n-1)!$  distinct visit sequences that must be examined. To date no integer programming algorithm has been devised which is capable of finding the optimal travel sequence for the general  $n$ -city problem. A variety of iterative schemes have been attempted, but all eventually succumb to the enormous number of sequences that exist as the number of cities increases (95).

The analogy between the travelling salesman problem and the work scheduling problem is easily established with recognition of the equivalence between distance and separation matrices. Each separation matrix associated with an  $n$ -recreation period cyclic graph can also be considered as

a distance matrix for an n-city travelling salesman problem if the following correspondences are made:

- (1) each recreation period in the separation matrix corresponds to a city in the distance matrix; and
- (2) each work period length in the separation matrix corresponds to a travel distance between cities in the distance matrix.

With these correspondences, use of a separation matrix to find work schedules in which each recreation period appears once and only once is equivalent to using a distance matrix to find a travel sequence in which each city is visited once and only once.

As an example, if the two correspondences identified above are applied to the separation matrix in figure 6.9, the matrix becomes identical in both structure and interpretation to the distance matrix in figure 6.10. In figure 6.9, the five designated entries define a feasible work schedule (figure 6.8), and the five designated entries in figure 6.10 define a feasible travel sequence (sequence 3). The identical structure of the designated entries in both matrices illustrates that the properties identified in the proceeding section for each set of separation matrix entries corresponding to a feasible schedule also apply, in an analogous manner, to each set of designated entries in a distance matrix which correspond to a feasible travel sequence. In terms of the travelling salesman problem, these characteristics are:

- (1) one and only one designated entry appears in each row and column (each city must be visited once and only once); and
- (2) the sum of the designated entries equals the total travel distance of the sequence.

The analogy between the work scheduling and travelling salesman problem indicates that the task of determining feasible work schedules from a separation matrix is equivalent to finding solutions for a special kind of travelling salesman problem. As illustrated above, solutions to both problems are structurally identical; i.e., each feasible schedule (travel sequence) must use (visit) each recreation period (city) once and only once with a return to the initiating period (home city). In the travelling salesman problem, the usual objective is to find the travel sequence which minimizes the total distance travelled. For the work scheduling problem, however, the objective is to enumerate schedules (travel sequences) which contain (equal) a given number of work days (travel distance).

Recognition of the equivalence between the scheduling and travelling salesman problems suggests another feasibility constraint for the set of designated matrix entries corresponding to a feasible schedule. As in the travelling salesman problem, disconnected sequences (subtours) of the recreation periods (cities) do not produce feasible schedules (visit sequences), i.e., the sequence of recreation periods

(cities) must not "return" to the initiating recreation period (home city) until each recreation period (city) has been used (visited) once. As an example, the recreation period sequence described in table 6.3 is  $\{(1,3), (3,4), (4,5), (5,2), (2,1)\}$ ; each two-number pair  $(i,j)$  defines a work period of the schedule (or leg of the travel sequence), and the sequence of recreation periods is  $\{1,3,4,5,2,1\}$ . Each period appears only once in the sequence before period 1 appears again to complete the schedule (tour).

To illustrate an invalid set of entries, consider the sequence of entries  $\{(1,3), (3,2), (2,1), (4,5), (5,4)\}$  from the same separation matrix (figure 6.9). Although this sequence uses one and only one entry from each row and column of the matrix, and the sum of the entries equals 30, the required amount, the entries do not define a feasible schedule (or travel sequence) because the sequence is disconnected: the sequence  $\{1,3,2,1,4,5,4\}$  returns to period 1 before including periods 4 and 5.

The following section describes an algorithm to enumerate sets of matrix entries from a given separation matrix which define feasible one-shift schedules. The algorithm utilizes the feasibility requirements developed in this and the preceding section.

## 6.6 AN ALGORITHM FOR THE ENUMERATION OF ONE-SHIFT CYCLIC SCHEDULES

This section describes an implicit enumeration scheme to obtain all feasible work schedules for a given separation matrix by constructing the unique set of matrix entries associated with each schedule. The efficient enumeration of feasible sets of separation matrix entries is achieved by utilizing the characteristics (feasibility constraints) identified in section 6.5. Summarizing briefly, for each set of designated matrix entries corresponding to a feasible work schedule:

- (1) one and only one designated entry appears in each row and column of the separation matrix (i.e., each recreation period is used once and only once);
- (2) the sum of the values of the designated entries equals the total number of work days in the schedule; and
- (3) the sequence of matrix entries is connected (i.e., each recreation period appears once in the sequence before the sequence returns to the initiating period).

The logic used to enumerate feasible schedules is a branch-and-bound procedure (93,94). All of the basic elements of the algorithm are presented in this section. The branching rule is described first. This rule is derived from the requirements that feasible sets of entries must be connected, and must use each recreation period once. This is followed by a discussion of the algorithm's bounding procedure, which is based on the requirement that every

feasible schedule possess a specified number of work days.

The algorithm has been used to find one-shift schedules with up to 10 recreation periods. The primary difficulty encountered in use of the algorithm on matrices which are greater than  $10 \times 10$  in size lies not with an inefficiency of the algorithm itself, but rather with the large number of feasible schedules which frequently must be enumerated. Some acceleration techniques which focus the algorithm on more desirable schedules are presented in section 6.7.

#### 6.6.1 Enumeration of Feasible Sequences of Recreation Periods - The Branching Process

As indicated above, the determination of all distinct one-shift schedules from an  $n \times n$  separation matrix is equivalent to determination of all sequenced set of matrix entries

$$\{(i,j), (j,k), \dots, (m,n), (n,i)\}$$

which possess the following characteristics:

- (1) the set contains exactly  $n$  entries;
- (2) the set forms a connected sequence of entries; and
- (3) the sum of the entries equals the total number of work days required in the schedule.

The branching process is used to construct, entry by entry, all sets which satisfy properties (1) and (2) above. The process of constructing feasible sets is based on the

observation that each set of  $n$  connected matrix entries can be alternately described by a unique sequence of recreation periods containing  $n+1$  components (i.e., the sequence of connected matrix entries  $\{(i,j), (j,k), \dots, (m,n), (n,i)\}$  is uniquely described by the sequence of recreation periods  $\{i,j,k,\dots,m,n,i\}$ ). Each sequence of  $n+1$  recreation periods which describes a set of  $n$  matrix entries satisfying properties (1) and (2) above, possesses the following characteristics:

- (1) the sequence contains exactly  $n$  distinct recreation periods (the first  $n$  periods in the sequence); and
- (2) the sequence begins and ends with the same period.

To illustrate, consider the set of five entries indicated in the separation matrix in figure 6.9:  $\{(1,3), (3,4), (4,5), (5,2), (2,1)\}$ . This set of entries which defines the feasible work schedule shown in figure 6.8, can be uniquely represented by the six-component sequence of recreation periods:

$\{1,3,4,5,2,1\}$ . (The sequence contains five distinct recreation periods, and begins and ends with the same period.)

Recalling the travelling salesman problem, the sequence of recreation periods is the analogue to the sequence of cities visited. The remainder of this section describes the branching process in terms of the enumeration of all feasible sequences of recreation periods.

To describe the process the following terms will be used:

- node - a specified subset of the set of all feasible sequences (node 0 represents the set of all sequences)
- branch - a directed line segment connecting two nodes
- tree of solutions - a graph of all the nodes and branches for a particular problem
- path to node k - a sequence of nodes and branches from node 0 to node k
- terminal node - a node representing one feasible sequence.

To illustrate the process, consider the enumeration of all feasible sequences  $\{r_1, r_2, r_3, r_4\}$  from a  $3 \times 3$  separation matrix. There are a total of six feasible sequences to be found:  $\{1, 2, 3, 1\}$ ,  $\{1, 3, 2, 1\}$ ,  $\{2, 1, 3, 2\}$ ,  $\{2, 3, 1, 2\}$ ,  $\{3, 1, 2, 3\}$ , and  $\{3, 2, 1, 3\}$ . The tree of solutions for this problem is shown in figure 6.11. The circles represent the nodes and the superscript for each node indicates the sequence in which the nodes are formed. The equation within each node is the constraint imposed at that node; e.g., at node 1,  $r_1 = 1$  indicates that node 1 represents the set of all sequences with  $r_1 = 1$ . A sequence at any node must satisfy all of the constraints imposed by every node which lies on the path from node 0 to that node. Node 14, for example, represents the set of all sequences with  $r_1 = 2$  and  $r_2 = 1$ . The six feasible sequences for this problem are represented by terminal nodes 7, 11, 18, 22, 29, and 33.



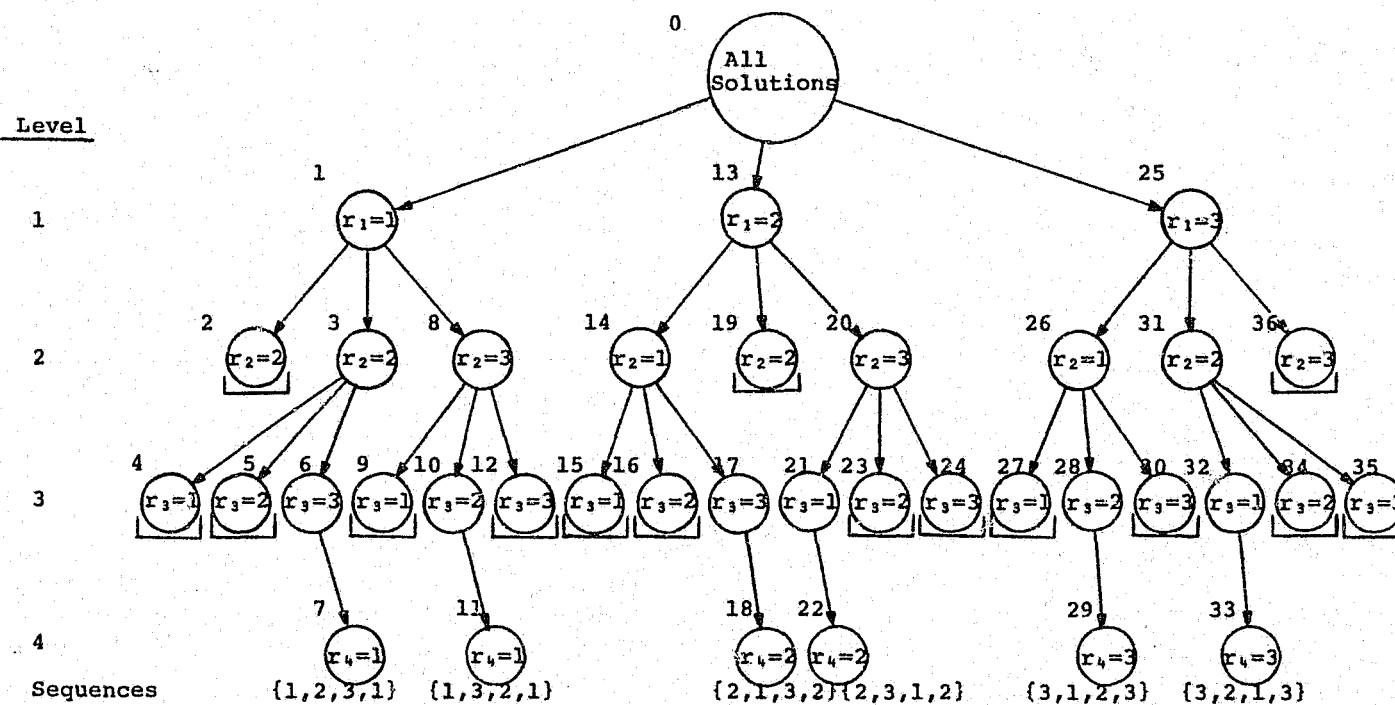


Figure 6.11

Tree of Solutions for the Enumeration of Feasible Sequences  
of Recreation Periods for a 3x3 Separation Matrix

To enumerate these six feasible sequences and the tree of solutions in figure 6.11, the process begins at node 0. The set of all sequences, represented by node 0, is divided into three mutually exclusive subsets: the set of all solutions with  $r_1 = 1$  (represented by node 1), the set of all solutions with  $r_1 = 2$  (represented by node 13), and, the set of all solutions with  $r_1 = 3$  (represented by node 25).

Next, each of these three nodes is subdivided into three mutually exclusive sets. From node 13, for example, three sets of solutions are produced: the set of all solutions with  $r_2 = 1$  and  $r_1 = 2$  (represented by node 14), the set of all solutions with  $r_2 = 2$  and  $r_1 = 2$  (represented by node 19), and the set of all solutions with  $r_2 = 3$  and  $r_1 = 2$  (represented by node 20).

If each of the level 2 nodes\* were now subdivided into three subsets, and each of these subsets further divided into three sets, each of the resulting 81 nodes ( $3^4 = 81$ ) would represent one sequence, defined by the four constraints prescribed on the path from each level 4 node to node 0. Only six of these 81 solutions, however, is feasible.

---

\*The level of a node is defined to be the number of constraints that apply to the set of sequences represented by that node (e.g., in figure 6.11, node 13 is a level 1 node, node 20 is a level 2 node, node 21 is a level 3 node, and node 22 is a level 4 node). In a tree of solutions for an  $n \times n$  separation matrix, all terminal nodes are  $n+1$  level nodes.

The enumeration of all 81 level 4 nodes or even all 27 level 3 nodes ( $3^3 = 27$ ) is not necessary however if the constraints identified above for each feasible sequence of recreation periods are used. In deriving feasible sequences for an  $n \times n$  separation matrix, these constraints are:

- (1) the first  $n$  recreation periods in each sequence must be distinct; and
- (2) the  $n+1$ st period in each sequence must be identical to its initiating (level 1) period.

Use of these constraints is also illustrated in figure 6.11. The first constraint is used at levels 2 and 3 to weed out subsets of solutions which use the same recreation period more than once. Three such subsets occur at level 2 (nodes 2, 19, and 36). For each of these nodes, marked with a  $\square$  sign, the recreation period assigned to  $r_2$  of the sequence is identical to the period assigned to  $r_1$  of the sequence. As a result, only six of the level 2 nodes are used to generate nodes at level 3. Of the 18 nodes formed at level 3, 12 also violate this feasibility constraint by assigning a recreation period to  $r_3$  that has already been assigned to  $r_1$  or  $r_2$  of the sequence. Only nodes 6, 10, 17, 21, 28, and 32 satisfy this constraint.

In general, the branching process is effectively completed at level  $n$  of the tree of solutions (level 3 in figure 6.11) because the final period in each feasible sequence (assigned at level  $n+1$ ) must be identical to its

initiating period. Consequently, as shown in figure 6.11, only one branch descends from each of the six active nodes at level 3, and the period assigned at level 4 is identical to the initiating period for each sequence.

To illustrate the construction of the unique work schedule associated with each feasible sequence of recreation days assume that the recreation periods to be used in the six sequences found in the example above come from the cyclic graph shown in figure 6.12. The graph contains seven recreation days which are clustered into three recreation periods: period 1 which is two days long and begins on

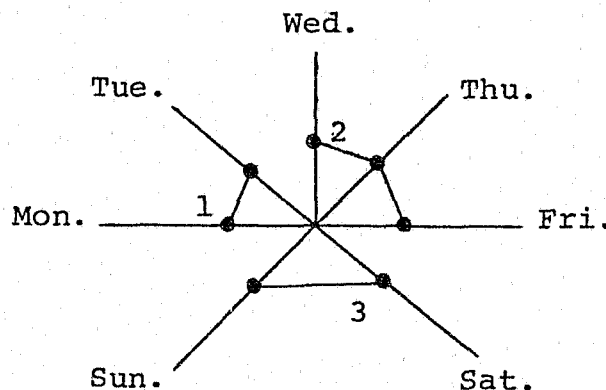


Figure 6.12

Sample Cyclic Graph with Three  
Recreation Periods

Monday, period 2 which is three days long and begins on Wednesday, and period 3 which is two days long and begins on Saturday. If upper and lower limits of 10 and 4 days respectively are used for work period lengths, the modified separation matrix in figure 6.13 is formed.

With the information contained in the cyclic graph in figure 6.12 and the separation matrix in figure 6.13, each of the six feasible sequences of recreation periods derived above can be used to construct a one-shift PR schedule. The convention used in this thesis, without loss of generality, is that the initiating recreation period of a feasible

		Recreation Period		
		1	2	3
Recreation Period	1	-	7	10
	2	9	-	7
	3	7	9	7

Figure 6.13

Separation Matrix for the Cyclic Graph in Figure 6.12

sequence is always placed in week 1 of the schedule.\* As an example, consider the sequence of periods represented by node 7 in figure 6.11:  $\{1,2,3,1\}$  which defines the following sequence of matrix entries:  $\{(1,2), (2,3), (3,1)\}$ . The values for these entries indicate the lengths of the work periods used to separate each pair of recreation periods in the schedule. The four-week cyclic schedule based on this sequence is shown in figure 6.14.

Period 1 is placed in week 1 of the schedule. Once the position of the first recreation period is determined, the remainder of the schedule can be constructed by laying

	M	T	W	T	F	S	S
Week	1 R	R					
2			2 R	R	R		
3						3 R	R
4							

Figure 6.14

PR Schedule Constructed with the  $\{1,2,3,1\}$  Sequence of Recreation Periods and the Separation Matrix in Figure 6.13

---

\*This convention is not applicable for the construction of non-cyclic schedules (see chapter 7).

out the work and recreation periods in the order specified in the feasible sequence. Since recreation period 1 begins on Monday and is two days long, the work period following it must begin on Wednesday. Since this work period separates recreation periods 1 and 2, its length is given by matrix entry  $(1,2)$ : seven days. Hence the work period begins on Wednesday of the first week and ends seven days later on Tuesday of the second week. The second recreation period, period 2, begins on Wednesday of the second week and ends three days later on Friday. The next work period which separates recreation periods 2 and 3 begins on Saturday of the second week and ends seven days later ( $\text{matrix entry } (2,3) = 7$ ) on Friday of the third week. The third recreation period is placed on Saturday and Sunday of the third week and the final work period which separates recreation periods 3 and 1 runs from Monday through Sunday of the fourth week ( $\text{matrix entry } (3,1) = 7$ ). Each work schedule associated with the other feasible sequences found in figure 6.11 and based on the separation matrix in figure 6.13 are constructed in a similar manner (see figures 6.15-6.19).

These six schedules also illustrate two characteristics which can be used to reduce the computational effort associated with the enumeration of acceptable cyclic schedules. These

	M	T	W	T	F	S	S
1			<sup>2</sup> R	R	R		
2						<sup>3</sup> R	R
3							
4	<sup>1</sup> R	R					

Figure 6.15

PR Schedule Constructed with the {2,3,1,2} Sequence of Recreation Periods and the Separation Matrix in Figure 6.13

	M	T	W	T	F	S	S
1						<sup>3</sup> R	R
2							
3	<sup>1</sup> R	R					
4			<sup>2</sup> R	R	R		

Figure 6.16

PR Schedule Constructed with the {3,1,2,3} Sequence of Recreation Periods and the Separation Matrix in Figure 6.13

	M	T	W	T	F	S	S
1	1 R	R					
2						<sup>3</sup> R	R
3							
4			<sup>2</sup> R	R	R		
5							

Figure 6.17

PR Schedule Constructed with the {1,3,2,1} Sequence of Recreation Periods and the Separation Matrix in Figure 6.13

	M	T	W	T	F	S	S
1						<sup>3</sup> R	R
2							
3			<sup>2</sup> R	R	R		
4							
5	<sup>1</sup> R	R					

Figure 6.18

PR Schedule Constructed with the {3,2,1,3} Sequence of Recreation Periods and the Separation Matrix in Figure 6.13

	M	T	W	T	F	S	S
1			<sup>2</sup> R	R	R		
2							
3	<sup>1</sup> R	R					
4						<sup>3</sup> R	R
5							

Figure 6.19

PR Schedule Constructed with the {2,1,3,2} Sequence of Recreation Periods and the Separation Matrix in Figure 6.13

characteristics are:

- (1) each feasible sequence produced by the branching process described above does not produce a distinct cyclic schedule, and
- (2) all distinct cyclic schedules produced by the branching process do not have the same period length.

The use of each of these characteristics to accelerate the enumeration of all distinct schedules with a prespecified schedule period length is discussed below.

#### 6.6.1.1 Distinct Cyclic Schedules

The branching process described above is based on the observation that each feasible sequence of  $n$  matrix entries can be characterized as an unique linear permutation of  $n$  recreation periods (i.e., all feasible sequences can be found by enumerating all linear permutations of the  $n$  periods and adding an  $n+1$ st component which is identical to the first component of the permutation). As a result, the branching process described above produces as many feasible sequences as there are linear permutations of  $n$  periods (i.e., for  $n$  recreation periods, exactly  $n!$  feasible sequences will be found\*). As an example, in the three-period problem above,  $n! = 3! = 6$  feasible sequences were found.

The work schedules constructed from the six sequences derived in that example illustrate that each feasible sequence of recreation periods does not necessarily produce a distinct cyclic schedule. The six cyclic schedules shown above actually represent only two distinct schedules, each shown in three different ways. The three schedules shown in figure 6.14, 6.15, and 6.16 are equivalent representations of the same cyclic schedule, and the three schedules shown in figures 6.17, 6.18, and 6.19 are equivalent representations of a second distinct cyclic schedule. The work schedules

---

\* This result assumes that the  $n$  periods are distinct. Modification of the branching process when two or more periods are identical (i.e., have the same start day and length) is discussed in section 6.7.

shown in figures 6.14, 6.15, and 6.16 are equivalent cyclic schedules in the sense that each schedule can be obtained from either of the other two by rotating the schedule brackets (e.g., the schedule in figure 6.14 can be obtained from the schedule in figure 6.15 by rotating each bracket "down" one week and moving the bracket for week 4 back to week 1). Conversely, distinct cyclic schedules have the property that they can not be obtained from one another by rotating schedule brackets.

The identification of sequences of recreation periods which yield equivalent cyclic schedules is quite simple. If the underlying (or generating) permutations\* for any two sequences are themselves cyclic permutations of each other, the resulting schedules from the sequences will be equivalent. As an example, the underlying permutations for the sequences used to construct the equivalent schedules in figures 6.14, 6.15, and 6.16 are  $\{1,2,3\}$ ,  $\{2,3,1\}$ , and  $\{3,1,2\}$ ; each is a cyclic permutation of the other two. Similarly, the permutations for the sequences used to construct the equivalent schedules in figure 6.17, 6.18, and 6.19 are  $\{1,3,2\}$ ,  $\{3,2,1\}$  and  $\{2,1,3\}$ ; again each is a cyclic permutation of the other two.

---

\*The underlying permutation for any feasible sequence of  $n+1$  periods is the first  $n$  components of the sequence (e.g., the underlying permutation for the sequence  $\{1,2,3,1\}$  is  $\{1,2,3\}$ ).

The problem of determining the number of distinct cyclic schedules that can be produced from an  $n$  recreation period matrix is analogous to the problem of determining the total number of distinct seating arrangements for  $n$  persons around a table. It has been shown by many authors that there are exactly  $(n-1)!$  distinct arrangements for  $n$  distinguishable persons and  $n$  cyclic permutations for each arrangement. The distinguishing feature of each seating arrangement is the relative position of each person at the table with respect to every other person rather than the absolute location of each person at the table itself.

Similarly, feasible sequences based on  $n$  distinct recreation periods can be used to create  $(n-1)!$  distinct cyclic schedules, and each distinct schedule can be represented in  $n$  different ways corresponding to  $n$  cyclic permutations of the underlying permutation associated with each feasible sequence. Hence, there are  $(n-1)! \cdot n = n!$  feasible sequences for an  $n$  recreation period matrix -- a result noted above. As an example, consider the three recreation period problem ( $n=3$ ) discussed above: six feasible sequences were found ( $n! = 3! = 6$ ), two distinct cyclic schedules were constructed ( $(n-1)! = (3-1)! = 2$ ), and three equivalent sequences were found for each schedule ( $n=3$ ).

Enumeration of a set of  $(n-1)!$  feasible sequences, each corresponding to a distinct cyclic schedule can be accomplished with the branching process described above

by adding the requirement that the same recreation period be used as the initiating (or level 1) period in every sequence (i.e., requiring  $r_1 = c$  for all sequences). With this additional constraint, the branching process will produce  $(n-1)!$  feasible sequences (for an  $n$ -period problem) which can be used to construct  $(n-1)!$  distinct cyclic schedules.

The validity of the  $r_1$  constraint is based on the observation that each of the  $n$  underlying permutations which correspond to each distinct schedule can be characterized by the recreation period which occupies the  $r_1$  component (i.e., each of the  $n$  underlying permutations begins with a different recreation period). Since there are  $n$  distinct periods in each underlying permutation, each period is used as the  $r_1$  component for one cyclic permutation. Hence, specification of a constant  $r_1$  component for all sequences has the effect of eliminating exactly  $n-1$  feasible sequences for each distinct schedule.

As an example, for the  $n = 3$  case, the underlying permutations for the two distinct schedules are (1)  $\{1,2,3\}$ ,  $\{3,1,2\}$ , and  $\{2,3,1\}$ , and (2)  $\{1,3,2\}$ ,  $\{3,2,1\}$ , and  $\{2,1,3\}$ . Specification of the  $r_1$  component to one value, (e.g.,  $r_1 = 1$ ), eliminates all but one permutation for each schedule.

Computationally, adding the requirement that  $r_1 = c$  (any of the  $n$  periods can be used for  $c$ ) eliminates all but one of the level 1 nodes from the tree of solutions. For example, if the constraint  $r_1 = 1$  had been applied to the tree of solutions in figure 6.11 only that portion of the tree descending from node 1 would have been generated and only two feasible sequences would have been enumerated:  $\{1,2,3,1\}$  and  $\{1,3,2,1\}$ . These sequences, based on non-cyclic permutations, produce the distinct cyclic schedules shown in figures 6.14 and 6.17 respectively.

#### 6.6.1.2 Cyclic Schedule Length

The rotation period length of a cyclic schedule is the number of weeks  $P_W$  (or days  $P_D = 7P_W$ ) required to rotate once through all schedule brackets. The six work schedules in figure 6.14 through 6.19 illustrate that the period length of distinct schedules based on the same set of recreation periods and separation matrix may be different. The discussion in this section identifies the relationship between the rotation period length of a schedule and the total number of work days  $S_W$  defined by the sequence of matrix entries associated with that schedule.

The rotation period length of a schedule in days equals the sum of the recreation days  $R$  defined by the cyclic graph, and the work days  $S_W$  defined by the sequence of matrix entries; i.e.,

$$P_D = R + S_W . \quad (6.2)$$

Since  $R$  is constant for every schedule enumerated from the same separation matrix, schedules with different  $P_D$  values can only occur because of differences in their respective  $S_W$  values. Conversely, specification of a  $P_D$  value for a schedule also identifies its  $S_W$  value; i.e.,

$$S_W(P_D, R) = P_D - R \quad (6.3)$$

where  $S_W(P_D, R)$  indicates that  $S_W$  is a function of both  $P_D$  and  $R$ .

As an example, both schedules shown in figures 6.14 and 6.17 use ten recreation days ( $R = 10$ ). The four-week schedule ( $P_D=28$ ) in figure 6.14, however, only requires  $S_W(28,10) = 18$  work days while the five-week schedule ( $P_D=35$ ) in figure 6.17 requires  $S_W(35,10) = 25$  work days.

The algorithm described in this chapter uses result (6.3) to produce schedules with a specific rotation period length by enumerating only sequences of matrix entries which sum to a  $S_W$  value specified by  $S_W(P_D, R)$ . The use of this restriction as a bounding procedure is described in section 6.6.2.

It is interesting to note that despite the large number of feasible sequences of recreation periods that can be enumerated from a separation matrix ( $(n-1)!$  sequences from an  $n \times n$  matrix), the number of distinct  $S_W$  values associated with the sequence of matrix entries is usually quite small.

Let  $A_W$  represent the set of allowable integer values for  $S_W$  for a given set of recreation periods and a specific separation matrix. Each value in set  $A_W$  must satisfy the following constraints:

$$(1) \quad nL_W \leq S_W \leq nU_W \quad (6.4)$$

$$(2) \quad (R+S_W)_{\text{mod } 7} = 0^* \quad (6.5)$$

In the first constraint,  $U_W$  and  $L_W$  are the upper and lower limits respectively on work period lengths, and  $n$  equals the number of work periods in each schedule. (In a cyclic schedule, the number of work periods equals the number of recreation periods.†) The second constraint limits  $S_W$  to values which make the sum  $R+S_W$  equal to a multiple of seven (i.e., values which make the schedule period length equal to an integral number of weeks).

Both constraints are imposed on  $S_W$  by the manner in which the separation matrix is constructed. The first constraint is easily shown. Since each entry in the matrix  $\{s_{ij}\}$  satisfies the constraint:

$$L_W \leq \{s_{ij}\} \leq U_W,$$

---

\*  $A_{\text{mod } x} = A - x \left[ \frac{A}{x} \right]$  where  $[ ]$  indicates the greatest integer less than or equal to  $A/x$  (e.g.,  $10_{\text{mod } 7} = 3 - 10 - 7 \left[ \frac{10}{7} \right] = 3$ ).

† Except in the trivial case when either  $R$  or  $S_W$  are zero (i.e., a schedule containing no work or recreation days).

the sum of any  $n$  entries from the matrix must satisfy the first constraint identified above. (Every  $S_W$  represents the sum of exactly  $n$  matrix entries.)

To verify the second constraint, the following geometric interpretation of schedule period length can be used. Each cyclic schedule consists of an unique alternating sequence of work and recreation periods. Each sequence can be used to define a circular path on the generating cyclic graph of the schedule. The path is defined to begin on the start ray of the first recreation period in the feasible sequence and advances ray by ray around the graph in a clockwise direction for each recreation and work period in the schedule until the ray corresponding to the last day of the final work period is reached. For each period, the path is advanced by the number of rays that equals the number of days in the period. Hence, the total number of rays covered by the entire path equals the period length of the schedule in days.\*

The length of the schedule rotation period in weeks can be obtained by counting the number of times the path forms a complete cycle around the seven rays of the graph. Each cycle (or week) begins on the start ray of the first recreation period and ends seven rays later. The period

---

\* Since the path may extend around the graph multiple times, each ray in the graph may be covered or crossed several times; each crossing is counted as one day.

length of a schedule will equal an integral number of weeks if the path ends on the ray immediately preceeding the start ray of the first recreation period. The last work period for each feasible sequence obtained from the branching process described above satisfies the requirement for a complete cycle, that is, for sequences based on  $n$  recreation periods, the last work period in the schedule is defined by the ordered pair of recreation periods  $(r_n, r_1)$ . Hence, the last work period extends the path to the ray immediately preceeding the start ray for period  $r_1$ , thus completing the last cycle of the path and producing a schedule period length equal to an integral number of weeks.

To illustrate how the two constraints limit the number of values that can exist in set  $A_W$ , consider a problem in which cyclic schedules are enumerated using a cyclic graph with ten recreation days ( $R=10$ ), and a five period ( $n=5$ ) separation matrix with  $U_W = 8$  and  $L_W = 4$ . The (6.4) constraint sets the following upper and lower limits on  $S_W$ ; i.e.,

$$\begin{aligned} (5)(4) &\leq S_W \leq (5)(8) \\ 20 &\leq S_W \leq 40 . \end{aligned} \quad (6.6)$$

The allowable values for  $S_W$  imposed by the constraint (6.5) are more easily determined if the constraint is rewritten as:

$$S_W = \begin{cases} 7k & k = 0, 1, 2, \dots, \text{if } R_{\text{mod } 7} = 0 \\ 7k + (7 - R_{\text{mod } 7}) & k = 0, 1, 2, \dots, \text{if } R_{\text{mod } 7} \neq 0 \end{cases} \quad (6.7)$$

Using (6.7), the allowable values for  $R = 10$  ( $10_{\text{mod } 7} = 3 \neq 0$ ) are given by:

$$S_W = 7k + 4 \quad k = 0, 1, 2, \dots$$

or

$$S_W = 4, 11, 18, 25, 32, 39, 46, \dots \quad (6.8)$$

Combining (6.6) and (6.8) yields the values for set  $A_W$ ; i.e.,

$$A_W = \{25, 32, 39\}.$$

The schedule period length (in weeks) for each  $S_W$  in  $A_W$  is obtained by adding the  $R$  value and dividing by seven (e.g.,  $S_W = 25$  produces a schedule with  $P_W = (25+10)/7 = 5$  weeks,  $S_W = 32$  produces a schedule with  $P_W = 6$  weeks, and  $S_W = 39$  produces a schedule with  $P_W = 7$  weeks).

Hence, in the example above, although 24 feasible sequences ( $n = 5$ ,  $(n-1)! = 24$ ) would be produced from the separation matrix (with each sequence representing a distinct schedule), the sum of the matrix entries associated with each sequence would sum to only one of three values: either 25, 32, or 39 days.

#### 6.6.2 Enumeration of Feasible Sequences of Recreation Periods - The Bounding Process

The branching procedure described in section 6.6.1 is used to enumerate all distinct cyclic schedules (regardless of period length) based on a given set of recreation periods and a unique separation matrix. The procedure enumerates one sequence of recreation periods for each schedule. The bounding

procedure described in this section is used to restrict the branching process to feasible sequences which correspond to schedules with a specific period length.

The bounding procedure is based on the calculation of upper and lower bounds on the total number of work days for all recreation period sequences which descend from each node in the branching process. These bounds, based on the unique partial sequence associated with each node, can be used to terminate exploration of tree branches prior to the final level based on the knowledge that the corresponding recreation period sequences will not produce schedules with the correct number of work days (i.e., the number of work days required to produce the desired schedule period length). With the use of these bounds the only sequences which are fully enumerated are those which are both feasible and correspond to schedules with a specific rotation period length.

Initial upper and lower bounds on the total number of work days for all feasible sequences are calculated using entry values from the separation matrix. As the tree of solutions grows, information based on the partial sequence of recreation periods associated with each node in the tree is used to improve the initial bounds (i.e., reduce the range between the upper and lower bounds). The bounds associated with each node are used to determine whether any acceptable schedules will be produced by any of the feasible

sequences that may descend from that node. If the bounds indicate that no acceptable schedules will be found, the node is terminated. The remainder of this section (1) describes how the initial bounds are calculated from the separation matrix, (2) describes how the bounds are improved at each node in the tree, and (3) illustrates the procedure with a simple example.

Calculation of the initial bounds for all feasible sequences is based on the observation that (1) the total number of work days associated with each feasible sequence is given by the sum of the matrix entries defined by the sequence, and (2) that each set of entries consists of one and only one entry from each row and column of the matrix. As a result, the total number of work days  $S_W$  associated with each set of entries can be described either as the sum of  $n$  "row" entries (i.e., one entry from each row of the matrix), denoted as

$$S_W = r_1 + r_2 + \dots + r_n \quad (6.9)$$

where  $r_i$  equals the matrix entry for row  $i$ , or equivalently, as the sum of  $n$  "column" entries (i.e., one entry from each column of the matrix), denoted as

$$S_W = c_1 + c_2 + \dots + c_n \quad (6.10)$$

where  $c_j$  equals the matrix entry for column  $j$ . Upper and lower bounds for  $S_W$  are calculated by using upper and lower

bounds on the individual terms  $r_i$  and  $c_j$  in equations (6.9) and (6.10).

Specifically, let  $\ell_{r_i}$  and  $\ell_{c_j}$  represent lower bounds on the matrix entries in each row  $i$  and column  $j$  of the separation matrix; i.e.,

$$\ell_{r_i} = \min_j s_{ij}, \quad i = 1, 2, \dots, n \quad (6.11)$$

and

$$\ell_{c_j} = \min_i s_{ij}, \quad j = 1, 2, \dots, n \quad (6.12)$$

Similarly, let  $u_{r_i}$  and  $u_{c_j}$  represent upper bounds on the matrix entries in each row  $i$  and column  $j$ ; i.e.,

$$u_{r_i} = \max_j s_{ij}, \quad i = 1, 2, \dots, n \quad (6.13)$$

and

$$u_{c_j} = \max_i s_{ij}, \quad j = 1, 2, \dots, n \quad (6.14)$$

Using the lower bounds given in (6.11) and (6.12) on the values of individual matrix entries, the following results can be obtained. Since entries from each row  $i$  must satisfy  $r_i \geq \ell_{r_i}$ , the sum of the  $n$  entries in each feasible sequence must satisfy

$$\sum_{i=1}^n r_i \geq \sum_{i=1}^n \ell_{r_i} \quad (6.15)$$

Similarly, since entries from each column  $j$  must satisfy  $c_j \geq \ell_{c_j}$ , the sum of the  $n$  entries in each feasible sequence

must satisfy

$$\sum_{j=1}^n c_j \geq \sum_{j=1}^n \ell_{c_j} \quad (6.16)$$

Since the left-hand sides of (6.15) and (6.16) both equal  $S_W$  (equations (6.9) and (6.10)), a lower bound  $L$  on the total number of work days in any feasible sequence of  $n$  matrix entries is given by

$$L = \max \left\{ \sum_{i=1}^n \ell_{r_i}, \sum_{j=1}^n \ell_{c_j} \right\} \quad (6.17)$$

An upper bound  $U$  on the total number of work days can be derived in a similar manner. Since  $r_i \leq u_{r_i}$  holds for every row  $i$ , the sum of  $n$  row entries in a feasible sequence is bounded from above by

$$\sum_{i=1}^n r_i \leq \sum_{i=1}^n u_{r_i} \quad (6.18)$$

Similarly, since  $c_j \leq u_{c_j}$  holds for every column  $j$ , the sum of  $n$  column entries in a feasible sequence is bounded from above by

$$\sum_{j=1}^n c_j \leq \sum_{j=1}^n u_{c_j} \quad (6.19)$$

Hence, an upper bound  $U$  on the total number of work days in any feasible sequence of  $n$  matrix entries is given by

$$U = \min \left\{ \sum_{i=1}^n u_{r_i}, \sum_{j=1}^n u_{c_j} \right\}. \quad (6.20)$$

To illustrate the computation of these upper and lower bounds, consider the 4x4 separation matrix shown in figure 6.20. The matrix is based on the cyclic graph containing ten recreation days shown in figure 6.21, and is to be used to enumerate one-shift schedules with a five-week rotation period. The upper and lower bounds on the matrix entries within each row and column of the separation matrix are indicated to the right of each row and below each column (e.g., the upper and lower bounds for the entries in row 2 are ten and seven days respectively). The U and L values for the matrix are circled in the lower right-hand corner of the figure ( $23 \leq S_W \leq 37$ ).

If the total number of required work days  $S_W$  (i.e., the specific value of  $S_W$  determined from  $P_D$  and  $R$ ), does not fall within the range  $[L, U]$ , no sequences with the correct number of work days will be enumerated from this matrix. If, however,  $S_W$  is within the range  $[L, U]$ , the matrix may contain one or more feasible sets of entries which sum to  $S_W$ . Since the rotation period for the schedule sought in the example above is five weeks (i.e.,  $P_D = 35$ ) and there are ten recreation days per rotation, each feasible sequence must include a total of 25 work days. The bounds  $[23, 37]$  indicate that one or more feasible sequences may exist.

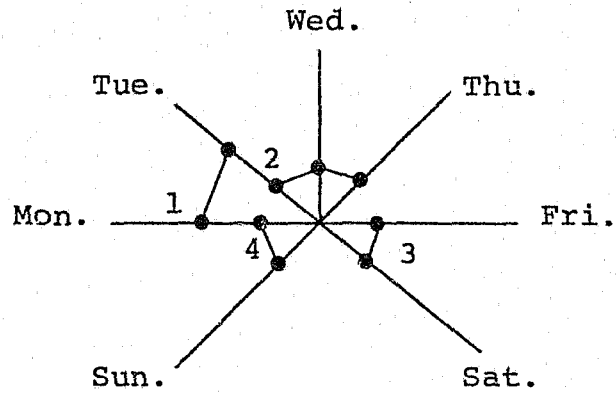


Figure 6.20

Cyclic Graph with Four Recreation Periods

		Recreation Period				Row min/max
		1	2	3	4	
Recreation Period	1	-	6	9	4	4/9
	2	10	-	7	9	7/10
	3	7	8	-	6	6/8
	4	6	7	10	-	6/10
Column min/max		6/10	6/8	7/10	6/9	23/37

Figure 6.21

Separation Matrix Based on the  
Cyclic Graph in Figure 6.20

In the discussion to follow, it will be shown that equations (6.17) and (6.20) can also be used to determine upper and lower bounds on the number of work days for all feasible sequences descending from any given node within the tree structure. These bounds are used to identify nodes from which no further branching is necessary.

The improved bounds for each node in the tree structure are based upon a series of modified separation matrices. These matrices are derived from the original separation matrix using information related to the partial sequence associated with each node. The partial sequence of recreation periods corresponding to a given node defines one or more matrix entries. These entries will be contained in every set of entries for sequences generated from branches descending from that node and are said to be "dedicated" entries for that node. These dedicated entries can be used to identify other matrix entries which cannot be used in any feasible sequences derived from that node. The entries which are identified as unacceptable for future use are said to be "voided". If voided entries are deleted from the separation matrix, the result is called a "reduced" separation matrix; this reduced array is used to obtain improved estimates of the upper and lower bounds  $U$  and  $L$ .

Dedicated matrix entries can be used to void other entries in the matrix because of the requirement that each feasible sequence of recreation periods must be a connected sequence and must use each recreation period once and only once. These requirements are used to void entries in two ways:

- (1) since one dedicated matrix entry can appear in each row and column, if entry  $(i,j)$  is dedicated, all other entries in both row  $i$  and column  $j$  can be voided; and
- (2) since the set of dedicated entries must correspond to a connected sequence, if period  $j$  is added to the partial sequence  $\{a,b,c,d,i,\dots\}$ , entry  $(i,j)$  becomes dedicated and entries  $(j,i)$ ,  $(j,d)$ ,  $(j,c)$ ,  $(j,b)$ , and  $(j,a)$  can be voided.\*

The second voiding procedure cited above is analogous to the requirement in the travelling salesman problem that each city in a feasible solution be visited exactly once before returning to the home city. The only exception to this rule occurs when the  $j^{\text{th}}$  period is the last period selected for a sequence. In that case, the  $(j,a)$  entry (period  $a$  is the first period in the sequence) must not be voided since it is used in level  $n+1$  to connect the last period of the sequence with period  $a$ .

Two reduced matrices are illustrated in figure 6.22 for partial sequences  $\{1,3,\dots\}$  and  $\{1,3,2,\dots\}$  based on the original separation matrix of figure 6.21. Dedicated entries

---

\* Additional voiding procedures are discussed in section 6.7.3.

		Recreation Period				Row min/max
		1	2	3	4	
Recreation Period	1	-	6	9	4	4/9
	2	10	-	7	9	7/10
	3	7	8	-	6	6/8
	4	6	7	10	-	6/10
Column min/max		6/10	6/8	7/10	4/9	23/37

(a) Initial Separation Matrix

		Recreation Period				Row min/max	
		1	2	3	4		
Recreation Period	1	-	-	9		9/9	Dedicated entries: (1,3)
	2	10	-	-	9	9/10	
	3	-	8	-	6	6/8	Voided entries: (1,2), (1,4), (2,3), (3,1), (4,3)
	4	6	7	-	-	6/7	
Column min/max		6/10	7/8	9/9	6/9	30/34	

(b) Reduced Separation Matrix Associated with the Partial Sequence {1,3, . . .}

		Recreation Period				Row min/max	
		1	2	3	4		
Recreation Period	1	-	-	9	-	9/9	Dedicated entries: (3,2)
	2	-	-	-	9	9/9	
	3	-	8	-	-	8/8	Voided entries: (2,1), (3,4), (4,2)
	4	6	-	-	-	6/6	
Column min/max		6/6	8/8	9/9	9/9	34/34	

(c) Reduced Separation Matrix Associated with the Partial Sequence {1,3,2 . . .}

Figure 6.22

Reduced Separation Matrices for Partial Sequences {1,3, . . .} and {1,3,2, . . .} Based on the Initial Separation Matrix in Figure 6.21

are circled and voided entries are indicated by a dash. The voided entries are also identified to the right of each matrix. For both reduced matrices, individual row and column limits, and U and L bounds are shown.

The value of computing these U and L bounds at each node is summarized in the following observation: if  $L_k$  and  $U_k$  denote the bounding limits based on the reduced matrix for node k, then for node k+1 which descends from node k, it can be shown that

$$L_{k+1} \geq L_k \quad (6.21)$$

and

$$U_{k+1} \leq U_k \quad (6.22)$$

Verification of inequality (6.21) is easily established by noting that the relationship

$$\ell_{r_i}^{k+1} \geq \ell_{r_i}^k \quad (6.23)$$

is valid for each row i in the reduced matrices for the k and k+1 nodes. This result follows from the observation that the lower limit  $\ell_{r_i}^{k+1}$  for each row i is based on either all or a subset of the matrix entries used to determine  $\ell_{r_i}^k$ . Consider each possibility. If the set of entries in row i are identical for both reduced matrices, then  $\ell_{r_i}^{k+1} = \ell_{r_i}^k$ . If, however, one or more entries are voided in row i of the reduced matrix for node k in order to produce the reduced matrix for node k+1, then either  $\ell_{r_i}^{k+1} = \ell_{r_i}^k$  or

$\ell_{r_i}^{k+1} > \ell_{r_i}^k$  depending on which entries are voided. Combining the results of both possibilities produces inequality (6.23).

Since the inequality in (6.23) holds for every row in the matrix, it follows that

$$\sum_{i=1}^n \ell_{r_i}^{k+1} \geq \sum_{i=1}^n \ell_{r_i}^k .$$

Repeating the same argument for the lower limits on the entries in each column in the reduced matrices for node  $k$  and each descendent node  $k+1$  leads to the parallel observation that

$$\sum_{j=1}^n \ell_{c_j}^{k+1} \geq \sum_{j=1}^n \ell_{c_j}^k .$$

It follows directly from equation (6.17) that  $L_{L+1} \geq L_k$ . The same argument can be used to confirm the upper bound result stated in (6.22).

Results (6.21) and (6.22) state that as each descendent node is generated, the  $U$  and  $L$  bounds calculated from the reduced matrix will be "as tight or tighter" than the corresponding bounds for the parent node. This useful result is illustrated in figure 6.22. In only two steps, the  $U$  and  $L$  bounds narrow from  $[23,37]$  for level 1 to  $[30,34]$  for level 2, and to  $[34,34]$  for level 3. Although shown for illustration purposes the level 3 matrix in figure 6.22 would not be generated by the algorithm since the  $[30,34]$  bounds indicate

that no schedule with the correct number of work days (i.e.,  $S_W = 25$ ) can be found for any sequence descending from this node.

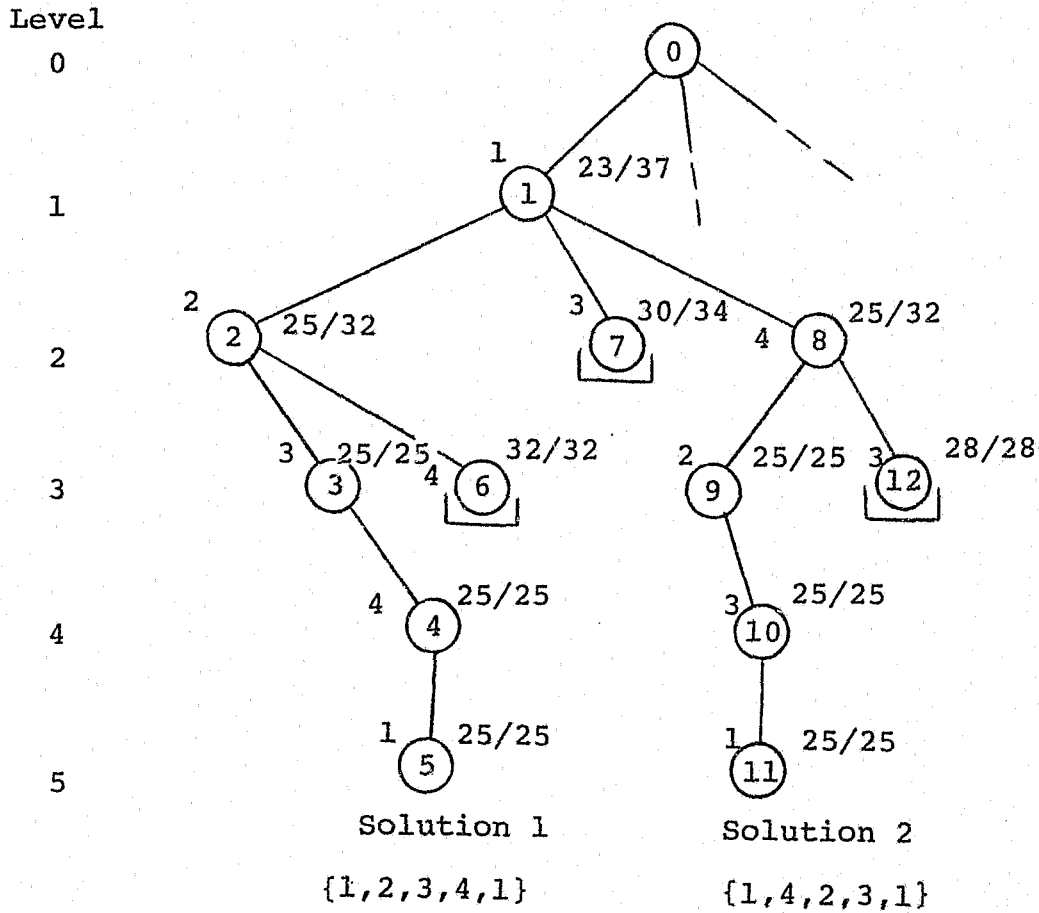
### 6.6.3 The Branch-and-Bound Algorithm - A Sample Problem

The branch-and-bound procedures described above represent the basic components of the algorithm used in this thesis to generate one-shift cyclic schedule for a given set of recreation periods and an  $n \times n$  separation matrix. The algorithm constructs, period by period, feasible sequences consisting of  $n+1$  recreation periods. Associated with each sequence is an unique set of  $n$  separation matrix entries which define a one-shift cyclic schedule with the required schedule period length. The basic elements of the algorithm are:

- (1) a branching process based on the requirement that the first  $n$  recreation periods in each feasible sequence must be distinct, and
- (2) a bounding procedure based on the requirement that each feasible sequence must define a set of matrix entries which sum to a given value.

Like other branch-and-bound algorithms, this procedure is equivalent to the enumeration of all possible recreation period sequences, but achieves its efficiency through implicit rather than explicit enumeration.

An illustration of the use of the algorithm to find schedules for the cyclic graph and separation matrix of figure 6.20 and 6.21 is presented below. The complete tree structure for the problem is shown in figure 6.23. The sequence in which the nodes were created is indicated by the



Note: Total number of required work days is  $W_R=25$ .

Figure 6,23

Partial Tree Structure for the Enumeration of  
One-Shift Schedules Based on the  
Separation Matrix in Figure 6.21

number within the node; the right-hand superscript indicates the upper and lower bounds on the total number of work days for sequences descending from the node. The left-hand superscript indicates the recreation period to which the node corresponds. The tree structure does not indicate nodes created at level 2 by attempting to utilize period 1 again at that point in the sequence, and so on for subsequent levels of the tree (this has been done to simplify the presentation in the figure). The complete tree structure indicates that two distinct schedules exist; they are shown in figure 6.24 and 6.25. The reduced matrices used to calculate the upper and lower bounds for each of the 12 nodes in the tree structure are shown in figure 6.26.

#### 6.6.4 Summary of the Enumeration Algorithm

The implicit enumeration algorithm described in sections 6.6.1 and 6.6.2 can be summarized with the use of the following notation:

- $L_k$  - lower bound on the number of work days for each sequence enumerated from the reduced separation matrix at level  $k$ .
- $U_k$  - upper bound on the number of work days for each sequence enumerated from the reduced separation matrix at level  $k$ .
- $W$  - required number of work days in each feasible sequence
- $V$  -  $n+1$  component sequence vector  $(v_1, v_2, \dots, v_{n+1})$

	M	T	W	T	F	S	S
1	<sup>1</sup> R	R					
2		<sup>2</sup> R	R	R			
3					<sup>3</sup> R	R	R
4							<sup>4</sup> R
5	R						

Figure 6.24

Five-Week PR Schedule Enumerated as  
Solution 1 in Figure 6.23

	M	T	W	T	F	S	S
1	<sup>1</sup> R	R					<sup>4</sup> R
2	R						
3		<sup>2</sup> R	R	R			
4					<sup>3</sup> R	R	R
5							

Figure 6.25

Five-Week PR Schedule Enumerated as  
Solution 2 in Figure 6.23

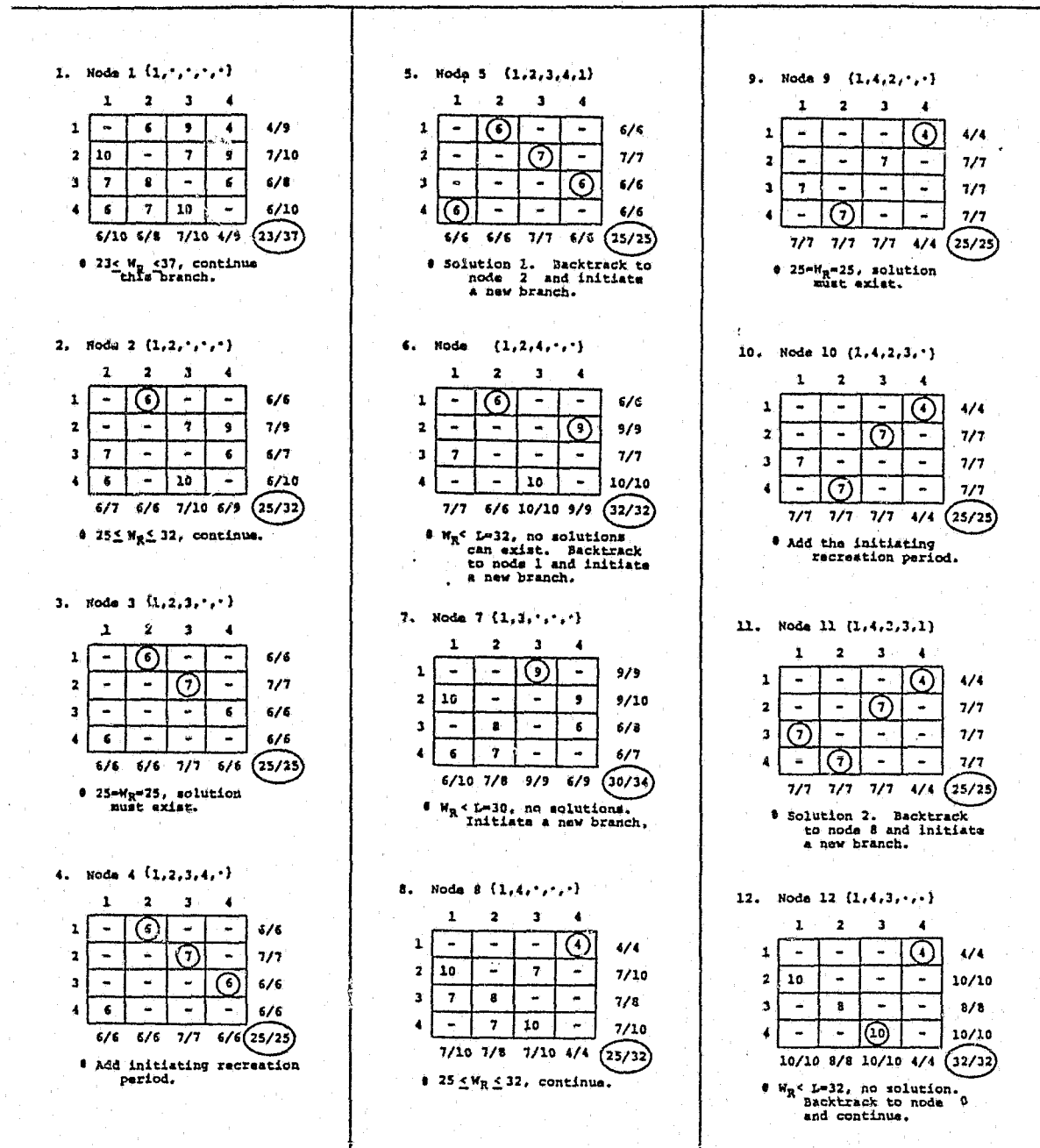


Figure 6.26

Reduced Separation Matrices Associated with Each Node in the Partial Tree Structure in Figure 6.23

$M_k$  - reduced  $n \times n$  separation matrix at level  $k$  ( $M_1$  is the original  $n \times n$  matrix)

$L$  - level indicator

$P$  - set of  $n$  recreation periods labeled from 1 to  $n$ .

Algorithm:

Step 0: Initialization:

- a)  $V = 0$ .
- b)  $L = 1$ , go to step 1.

Step 1: Find  $L$  and  $U$  for  $M$ :

- a) if  $W \notin [L_1, U_1]$ , stop, no solutions.
- b) if  $W_R \in [L_1, U_1]$ , set  $v_1 = 1$ , (i.e., select period 1 as the first period in every sequence), go to step 2.  
(note: if  $L_1 = W = U_1$ , all feasible sequences will produce acceptable schedules)

Step 2: Increment  $L$  by 1:

- a) if  $L = n+1$ , set  $v_L = v_1$ , store feasible sequence, set  $L = n$ , go to step 3.
- b) if  $L < n+1$ , go to step 3.

Step 3: Examine  $v_L$ :

- a) if  $v_L = 0$ , set  $v_L = p'$  where  $\{p' = \min p \mid \text{all } p \text{ not in } v\}$ , go to step 5; if none exist, go to step 4.

- b) if  $v_L = j$ ,  $j \neq n$ , set  
 $v_L = p'$  where  $\{p' = \min p \mid \text{all } p > j \text{ and not in } v\}$   
go to step 5; if none exist, go  
to step 4.
- c) if  $v_L = n$ , go to step 4.

Step 4: Set  $v_L = 0$ ,  $L = L-1$ :

- a) if  $L = 1$ , stop.
- b) if  $L > 1$ , go to step 3.

Step 5: Construct  $M_L$  based on  $V$  and  $M_{L-1}$ , go  
to step 6.

Step 6: Find  $L_L$  and  $U_L$  from  $M_L$

- a) if  $W \notin [L_L, U_L]$  go to step 3.
- b) if  $W \in [L_L, U_L]$  go to step 2.

(note: if  $L_L = W = U_L$ , all feasible  
sequences from the current node will  
produce acceptable schedules

## 6.7 ACCELERATION TECHNIQUES FOR THE ENUMERATION ON ONE-SHIFT CYCLIC SCHEDULES

This section describes three procedures for improving the efficiency of the enumeration algorithm described above. These procedures were not included in the description of the basic algorithm in section 6.6 because the usefulness of each depends upon the presence of a special feature or characteristic of either the input data or the enumeration process itself. The first procedure is useful whenever two

or more identical recreation periods are present in the cyclic graph used to construct the separation matrix, the second procedure is used when uniformity of work period lengths is a desired schedule attribute, and the third procedure utilizes special features within reduced separation matrices to maximize the number of voided entries.

#### 6.7.1 Identical Recreation Periods

In the description of the branch-and-bound algorithm in section 6.6, it was shown that for separation matrices based on  $n$  recreation periods, a maximum of  $(n-1)!$  sequences of periods can be produced such that each sequence produces a distinct cyclic schedule. The maximum number of sequences and distinct schedules is obtained only when each recreation period is distinct.\* (Non-distinct or identical periods begin on the same day of the week and have the same period length.) When two or more identical periods exist within a cyclic graph, the number of distinct schedules is reduced (i.e., some of the  $(n-1)!$  sequences produce equivalent cyclic schedules).

This section describes a modification to the branch-and-bound algorithm which insures that each distinct schedule is enumerated only once. To introduce this modification, consider the cyclic graph in figure 6.27. The graph contains

---

\*This discussion ignores the fact that some of the  $(n-1)!$  distinct schedules may have different schedule period lengths.

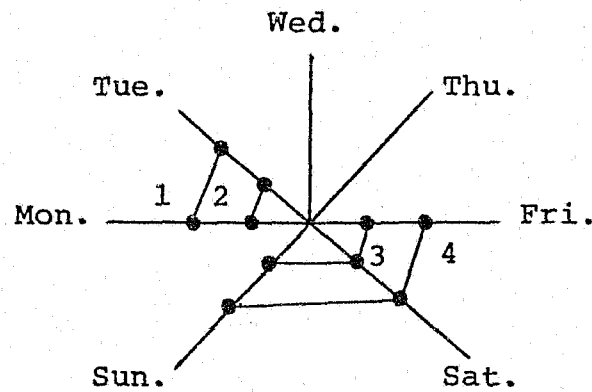


Figure 6.27

Cyclic Graph with Two Pairs of  
Identical Recreation Periods

two pairs of identical recreation periods: periods 1 and 2, and periods 3 and 4. A separation matrix based on this graph is shown in figure 6.28; the entry values in the matrix are restricted to a range of from four to nine days. Two cyclic schedules, based on the sequences  $\{1,2,3,4,1\}$  and  $\{1,2,4,3,1\}$  are shown in figures 6.29 and 6.30. The sequences are non-cyclic permutations of each other and both would be produced by the enumeration algorithm described in section 6.6. The schedules produced by these sequences, however, are equivalent because of the presence

		Recreation Period			
		1	2	3	4
Recreation Period	1	-	5	9	9
	2	5	-	9	9
	3	7	7	-	4
	4	7	7	4	-

Figure 6.28

Separation Matrix Based on the Cyclic Graph in Figure 6.27

		M	T	W	T	F	S	S
Week	1	1R	R					
	2	2R	R					
	3					3R	R	R
	4					4R	R	R
	5							

Figure 6.29

PR Schedule Based on the {1,2,3,4,1} Sequence of Recreation Periods and the Separation Matrix in Figure 6.28

	M	T	W	T	F	S	S
Week 1	1 R	R					
Week 2	2 R	R					
Week 3					4 R	R	R
Week 4					3 R	R	R
Week 5							

Figure 6.30

PR Schedule Based on the {1,2,4,3,1} Sequence of  
Recreation Periods and the Separation  
Matrix in Figure 6.28

of identical recreation periods. The acceleration procedure described in this section modifies the algorithm so that only one sequence is fully enumerated for each distinct schedule. (Hence, in the example above, the modified algorithm would enumerate either sequence {1,2,3,4,1} or sequence {1,2,4,3,1}, but not both.)

To understand the motivation for the acceleration procedure, it is useful to examine the tree structure for

the sequences discussed above. A partial tree diagram illustrating the enumeration of these sequences is shown in figure 6.31. Using the recreation period labels assigned in figure 6.29, and recalling that the algorithm uses the lowest numbered period eligible for each descendant node, the first sequence enumerated is  $\{1,2,3,4,1\}$ . After storing this result, the algorithm backtracks to the second level of the tree and selects a recreation period for the next descendant branch from node 2. Since the previous descendant node used period 3, the next eligible period is period 4. Adding this period to the sequence, and continuing the path from node 6 produces the sequence  $\{1,2,4,3,1\}$ . These sequences produce equivalent schedules because the recreation periods used for the  $r_3$  (or level 3) components in both sequences are identical (i.e., nodes 3 and 6 in the tree diagram do not represent distinct partial sequences). The partial sequence at node 3:  $\{1,2,3,\dots\}$  is equivalent to the partial sequence at node 6:  $\{1,2,4,\dots\}$ .

The acceleration procedure described in this section eliminates the enumeration of identical partial sequence nodes in the tree structure. This is done by assigning a second label to each recreation period based on its type (i.e., on its start day and length). Identical recreation periods are assigned identical labels. The additional label is incorporated into the selection criteria used to

determine the next recreation period for each descendant node. The modified selection rule insures that the set of descendant nodes enumerated from each parent node in the tree structure defines a set of distinct partial sequences. The modified rule permits each recreation period type to be used in only one descendant node from each parent node in the tree structure.

The modified selection rule can be illustrated using the tree diagram in figure 6.31. After the sequence {1,2,3,4,1} has been produced at node 5 and the algorithm has backtracked to node 2, a new recreation period must be selected for the next descendant node (i.e., node 6). Application of the branching rules described in section 6.6 eliminate periods 1, 2 and 3 as candidates. Application of the modified selection rule also eliminates period 4 because a period is not eligible if any previous descendant node has used a recreation period of the same type. With this constraint, period 4 is ineligible because it is identical to recreation period 3 which was used for descendant node 3.

The extent of the reduction in the number of nodes produced in the tree structure with the elimination of identical partial sequences is suggested by the following results. If all  $n$  recreation periods are distinct, the number of descendant nodes  $d(L)$  from a parent node at level  $L$  is given by

$$d(L) = \max\{n-L, 1\}. \quad (6.24)$$

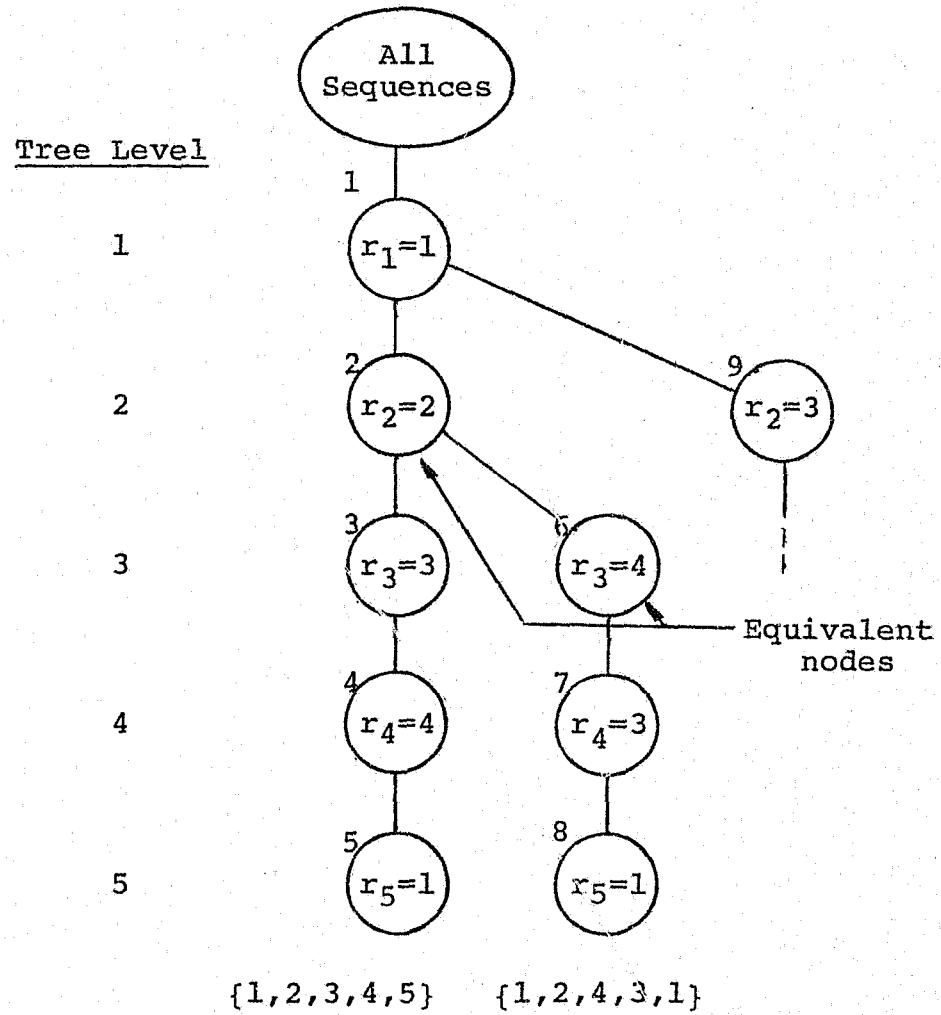


Figure 6.31

Partial Tree Diagram for the Enumeration of the Schedules in Figures 6.29 and 6.30

Equation (6.24) indicates that as the level of the tree structure increases, more recreation periods are used in the definition of each partial sequence and, as a result, fewer periods are available for descendant nodes. Let  $t$  equal the number of distinct recreation period types. The modified selection rule for enumerating descendant nodes has the effect of replacing  $n$  in equation (6.24) with  $t$ ; i.e.,

$$d_t(L) = \max\{t-L, 1\} \quad (6.25)$$

If every recreation period is distinct, then  $t = n$  and  $d_t(L) = d(L)$ . If two or more periods are identical, however,  $t < n$  and  $d_t(L) < d(L)$ . Summarizing these results, since

$$1 \leq t \leq n$$

it follows from equations (6.24) and (6.25) that

$$1 \leq d_t(L) \leq d(L)^*.$$

The additional selection constraint introduced above will eliminate the enumeration of all equivalent schedules except those based on sequences in which the initial period in the

---

\* If  $t = 1$ , all of the recreation periods are identical; in that event,  $d_t(L) = 1$  for each node and only one sequence is produced.

sequence is identical to one or more other periods in the cyclic graph. The three sequences produced in the tree diagram in figure 6.32 illustrate this exception. The first sequence:  $\{1,2,3,4,1\}$ , yields the schedule shown in figure 6.29 and the second and third sequences:  $\{1,3,2,4,1\}$  and  $\{1,3,4,2,1\}$ , yield the schedules shown in figures 6.33 and 6.34. Despite the use of the additional selection constraint, two equivalent schedules are enumerated (the schedules in figures 6.29 and 6.34). These schedules are equivalent because the initial recreation period used in the sequence for each: period 1, is identical to period 2. The sequences for these schedules illustrate that equivalent schedules will still be enumerated if a recreation period, identical to the initial period in the sequence, is placed either immediately after the initial period (as in sequence  $\{1,2,3,4,1\}$ ), or immediately before the initial period (as in sequence  $\{1,3,4,2,1\}$ ).

The enumeration of these equivalent schedules can be eliminated in two ways by adding the restriction that recreation periods that are identical to the initial period in the sequence cannot be used either in level 2 of the tree structure or cannot be used in level  $n$  of the tree structure. The first method (i.e., prohibiting identical periods from appearing in level 2) is preferred since it eliminates larger portions of the tree structure. As an example,

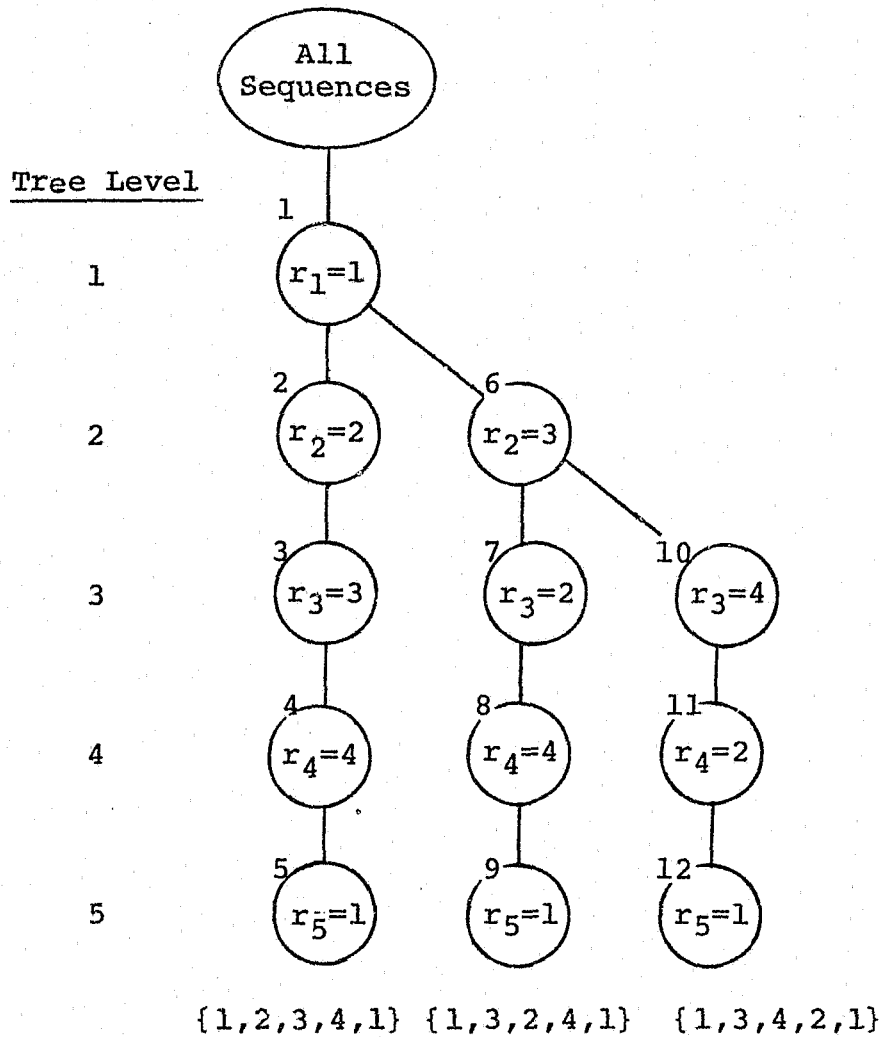


Figure 6.32

Tree Diagram for the Enumeration of Recreation  
Period Sequences Based on the Cyclic  
Graph in Figure 6.27

	M	T	W	T	F	S	S
1	<sup>1</sup> R	R					
2					<sup>3</sup> R	R	R
3							
4	<sup>2</sup> R	R					
5							
6					<sup>4</sup> R	R	R

Figure 6.33

PR Schedule Based on the {1,3,2,4,1} Sequence of  
Recreation Periods and the Separation  
Matrix in Figure 6.28

	M	T	W	T	F	S	S
1	<sup>1</sup> R	R					
2					<sup>3</sup> R	R	R
3					<sup>4</sup> R	R	R
4							
5	<sup>2</sup> R	R					

Figure 6.34

PR Schedule Based on the {1,3,4,2,1} Sequence of  
Recreation Periods and the Separation  
Matrix in Figure 6.28

if period 2 had been prohibited from appearing at node 2 in the tree diagram in figure 6.32, nodes 2, 3, 4, and 5 would not have been created and only two sequences, each corresponding to a distinct schedule, would have been enumerated.

#### 6.7.2 Work Period Lengths

The acceleration technique presented in this section is based on a strategy of dividing the computational effort required to enumerate all feasible schedules from a given separation matrix with the branch-and-bound algorithm into several reduced efforts based on the enumeration of subsets of schedules from a series of reduced matrices. The computational advantage of this strategy lies in the fact that if preferred schedules can be found early in the enumeration process, large numbers of less preferred schedules can be implicitly discarded without use of the branch-and-bound algorithm.

The procedure is based on the observation that all feasible schedules that can be enumerated from a separation matrix can be partitioned into mutually exclusive subsets on the basis of their maximum and minimum work period lengths. Let the two-number pair  $(a, b)$  represent the set of all schedules with minimum work period length  $W_{\min} = a$ , and maximum work period length  $W_{\max} = b$ . Every schedule in the set  $(W_{\min} = a, W_{\max} = b)$  has a work period range equal to

$R = b - a$ , and associated with each set of schedules is an unique reduced separation matrix. (Derivation of this matrix is described below.)

The number of partitioning sets that exist for a given original separation matrix is a function of:

- (1) the average work period length ( $A_W = W/n$ ); and
- (2) the minimum ( $L_W$ ) and maximum ( $U_W$ ) work period lengths in the matrix.

The  $W_{\max}$  values for individual schedules enumerated from the original matrix are bounded by

$$A_W \leq W_{\max} \leq U_W, \quad W_{\max} \text{ integer} \quad (6.26)$$

The number of distinct values for  $W_{\max}$  depends on the integrality of  $A_W$ ; i.e.,

$$N_{W_{\max}} = \begin{cases} U_W - A_W + 1, & \text{if } A_W \text{ is an integer} \\ U_W - [A_W], & \text{otherwise} \end{cases}$$

The values for  $W_{\min}$  for individual schedules are bounded by

$$L_W \leq W_{\min} \leq A_W, \quad W_{\min} \text{ integer} \quad (6.27)$$

and the number of distinct values for  $W_{\min}$  equals  $N_{W_{\min}} = [A_W] - L_W + 1$ . The total number of distinct partitioning sets that can be formed equals  $N_S = (N_{W_{\min}}) \cdot (N_{W_{\max}})$ .

It is easily shown that for any work period range  $R = U_W - L_W$ , the maximum value for  $N_S$  occurs when  $A_W = [(U_W + L_W)/2]$  which yields

$$\max N_S(R) = \begin{cases} \left(\frac{R+2}{2}\right)^2, & \text{if } R \text{ is even} \\ \left(\frac{R+1}{2}\right)\left(\frac{R+3}{2}\right), & \text{if } R \text{ is odd.} \end{cases}$$

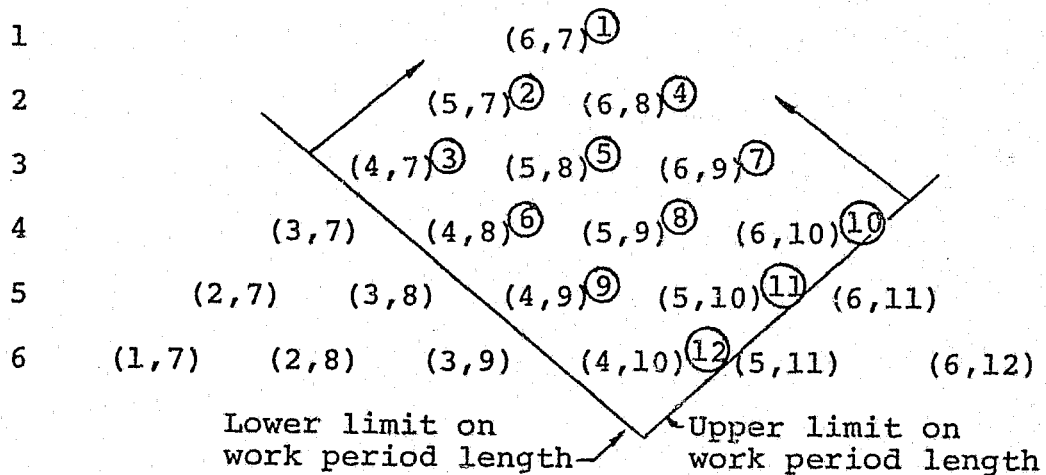
Since  $R = 6$  is the maximum range used in this study, the maximum number of sets equals  $N_S(6) = 16$ .

Results (6.26) and (6.27) can be used to identify all partitioning sets for a given separation matrix. As an example, the 12 sets that exist when  $U_W = 10$ ,  $L_W = 4$ , and  $A_W$  is not an integer (e.g.,  $A_W = 6.25$ ) are shown in figure 6.35. The sets are arranged vertically in the figure by length of the work period range (top to bottom), and horizontally by the size of the maximum work period length (left to right). The work period ranges vary in length from one to six days, and the maximum work period lengths vary from seven to ten days.

The partitioning sets for a given separation matrix can be used to reduce the computational effort of the branch-and-bound algorithm in the following ways:

- (1) it may be possible to determine an optimal schedule by examining only a small number of the partitioning sets; and
- (2) the reduced separation matrix associated with each set may contain a member of void entries which may reduce the computational effort required to enumerate all schedules from the matrix.

Work Period  
Range  
(Days)



Note: Average work period length,  $A_W=6.25$  days, and the limits on work period length are  $U_W=10$  and  $L_W=4$ .

Figure 6.35

Partitioning Sets for a Non-Integer Valued Average  
Work Period Length

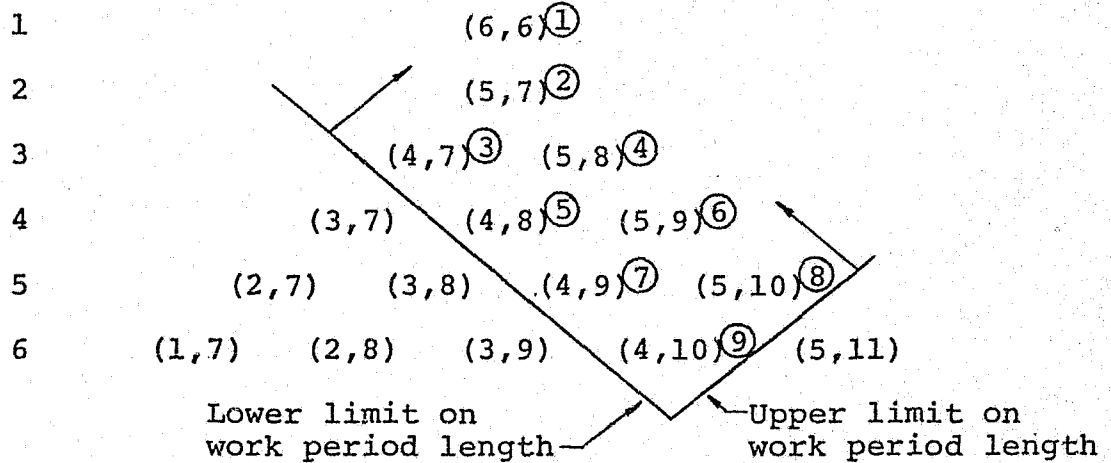
Since work period range and maximum work period length can be used to distinguish alternative schedules enumerated from the same separation matrix, if the partitioning sets are ranked correctly, every schedule in a given set will be superior to or dominate (i.e., in terms of their preference

vectors) every schedule in all lower ranking sets. To illustrate, consider the 12 sets shown in figure 6.35. The ranking for each set, indicated by a circled right superscript, is determined in two steps: first by comparing maximum work period lengths (i.e., the  $W_{\max}$  value) and second by comparing work period ranges (i.e.,  $W_{\max} - W_{\min}$ ). Hence, sets (6,7), (5,7) and (4,7) are the highest ranking sets because each contains only schedules with  $W_{\max} = 7$ , the lowest value possible. Every schedule in the nine other sets contains at least one work period that is longer than seven days. The tie among the three sets: (6,7), (5,7) and (4,7) is broken by comparing work period ranges. Set (6,7) with a one-day work period range, is ranked first; set (5,7) with a two-day work period range, is ranked second; and set (4,7) is ranked third.

As a second example, the nine partitioning sets for a separation matrix with  $U_W = 10$ ,  $L_W = 4$ , and an integer  $A_W$  value (e.g.,  $A_W = 6$ ) are shown in figure 6.36. The ranking for each set is indicated by the circled right superscript.

Although the sets identified in figures 6.35 and 6.36 are both based on  $U_W = 10$  and  $L_W = 4$ , fewer sets are shown in figure 6.36 because of the integrality of  $A_W$ . The reduction in the number of sets occurs because no sets are included in figure 6.36 which contain either  $W_{\min} = A_W$  or  $W_{\max} = A_W$ . These sets are excluded because any set of the form  $(W_{\min} = A_W, W_{\max} = A_W)$

Work Period  
Range  
(Days)



Note: Average work period length,  $A_W=6.0$  days, and the limits on work period length are  $U_W=10$  and  $L_W=4$ .

Figure 6.36

Partitioning Sets for an Integer Valued Average Work Period Length

or  $(W_{\min} < A_W, W_{\max} = A_W)$  can not contain any schedules. This is easily seen by noting that if  $W_{\min} = A_W$ , then every work period in the schedule must equal  $A_W$ . Hence, for every schedule with  $W_{\min} = A_W$ ,  $W_{\max}$  will equal  $A_W$  and the set

$(W_{\min} = A_W, W_{\max} > A_W)$  will be empty. A parallel argument can be used to show that the set  $(W_{\min} < A_W, W_{\max} = A_W)$  must also be empty. These same arguments can also be used to verify that no schedules can exist with an integer value for  $A_W$  and a work period range of exactly one day.

The ability to rank all feasible schedules according to their partitioning set permits the search for the optimal schedule to be done in a step-wise manner by applying the branch-and-bound algorithm to the reduced matrix for each set of schedules. The algorithm is applied to each set, in order of their rankings, until a feasible schedule is enumerated. Once a schedule is found, all lower ranking sets are discarded since they contain, by definition, only lower ranking schedules.\*

In addition to accelerating the process of finding the optimal schedule, the use of the partitioning sets may also reduce the computational effort required to enumerate the schedules for each set. This reduction in effort occurs because the branch-and-bound algorithm is applied to a unique reduced separation matrix associated with each set. The reduced matrix for each partitioning set (a,b) is constructed from the original separation matrix in the following manner. Each entry  $s_{ij}$

---

\* If multiple schedules are found within a set, the optimal schedule is determined from among them by comparing other schedule attributes.

in the reduced matrix for set (a,b) is determined according to the following rule:

$$s'_{ij} = \begin{cases} s_{ij}, & \text{if } a \leq s_{ij} \leq b \\ \text{voided}, & \text{otherwise} \end{cases}$$

where  $s_{ij}$  represents the (i,j) entry in the original separation matrix.

As an example, an original separation matrix and the reduced matrix obtained from it for the (6,7) partitioning set are shown in figure 6.37. The reduced matrix contains

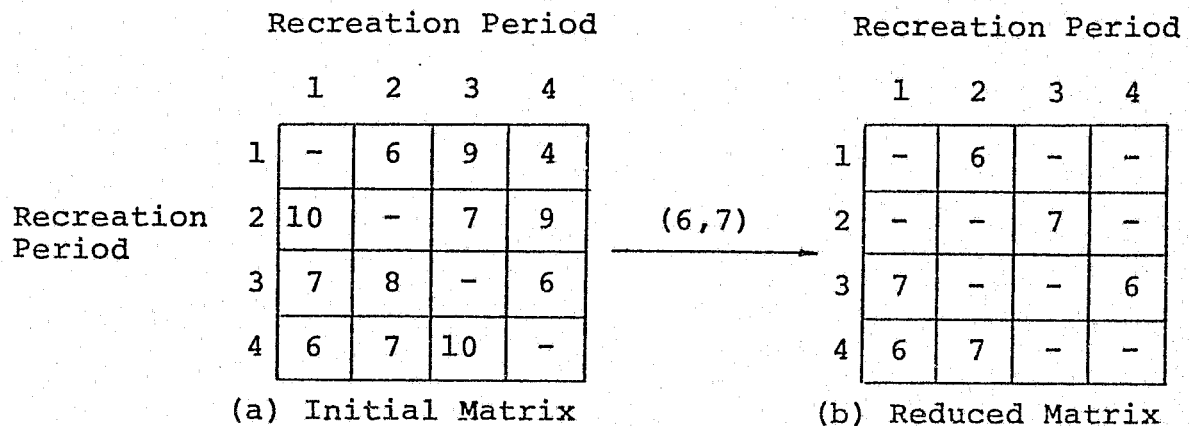


Figure 6.37

Original 4x4 Separation Matrix and  
Reduced (6,7) Matrix

all entries from the original separation matrix necessary to enumerate all feasible schedules with a maximum work period length of seven days and a work period range of only one day. The reduced matrix, with only six valid entries, represents a much simpler problem. One solution from the reduced matrix is the sequence of periods  $\{1,2,3,4,1\}$  which defines the entries  $\{(1,2), (2,3), (3,4), (4,1)\}$ ; these entries define a schedule with 25 work days. The enumeration of at least one schedule for this set indicates that all lower ranking sets can be discarded. If no other schedules are found for the (6,7) set, the sequence identified above would represent the optimal solution.

To summarize, the following procedures can be used to reduce the computational effort required to enumerate desirable schedules from a given separation matrix:

- Step 1: Partition the set of all feasible schedules into mutually exclusive sets based on maximum and minimum work period lengths ( $W_{\max}$  and  $W_{\min}$ ).
- Step 2: Use  $W_{\max}$  and  $R = W_{\max} - W_{\min}$  to rank order the partitioning sets.
- Step 3: Construct the reduced matrix for the (next) highest ranking set.

Step 4: Use the matrix conditioning procedures (see section 6.7.3) to minimize the number of valid matrix entries.

Step 5: Use the branch-and-bound algorithm to enumerate all feasible schedules from the reduced matrix:

- i) if schedules are found, determine the optimal schedule from among them, stop.
- ii) if no schedules are found, return to step 3.

### 6.7.3 Matrix Conditioning

The computational effort associated with use of the branch-and-bound algorithm to enumerate feasible schedules from a separation matrix can often be significantly reduced or even eliminated with several procedures collectively described in this section as matrix conditioning. All of the procedures involve the examination of individual separation matrix entries. Some procedures are used only to determine whether the separation matrix contains any feasible solutions (i.e., whether the branch-and-bound algorithm should be used), while others are used to maximize the number of voided entries in a matrix prior to using the enumeration algorithm. The four procedures discussed in this section are:

- (1) the examination of the matrix for the presence of at least one valid entry in each row and column;
- (2) the dedication of each matrix entry identified as the only valid entry in either a row or column of the matrix;
- (3) the examination of the reduced matrix associated with each partitioning set for the presence of at least one entry equal to  $W_{\min}$  and at least one entry equal to  $W_{\max}$ ; and
- (4) the dedication of an entry if it is either the only entry in the reduced matrix equal to  $W_{\min}$ , or the only entry in the reduced  $W_{\max}$  matrix equal to  $W_{\max}$ .

The rationale and use of each of the procedures are discussed below.

Procedures 1 and 2 are based on the requirement that the entry set for each feasible sequence of periods must use one and only one entry from each row and column of the separation matrix. For procedure 1, the matrix is examined to determine whether at least one valid entry exists in each row and column of the matrix. If a row or column without a valid entry is found no feasible schedules can be enumerated from the matrix, and the branch-and-bound algorithm is not used.

Procedure 2, only applied to separation matrices not rejected by procedure 1, is used to identify matrix entries which cannot appear in any feasible sequence of periods. The procedure is based on the identification of matrix entries with the property that each is the only valid entry

in either a row or column of the matrix (there may be several such entries in the same matrix). Since each of these entries must, by definition, belong to the entry set for every feasible sequence enumerated from the matrix, these entries are said to be dedicated. The value in identifying dedicated entries is that each such entry can be used to identify other matrix entries that cannot appear in any feasible sequences, and hence, can be voided. As each dedicated entry is identified, the following rules are used to identify other entries that can be voided.

1. Since each dedicated entry  $(i,j)$  must be used in every feasible solution, every other valid matrix entry in row  $i$  and column  $j$  of the matrix will not appear in any feasible sequence and, as a result, can be voided.
2. Since every feasible solution is represented by a connected entry set, if dedicated entry  $(i,j)$  appears in every solution, then matrix entry  $(j,i)$  cannot not appear in any solution, and can be voided.\*

If any matrix entries are voided because of procedure 2, procedures 1 and 2 (if necessary) are applied again. The

---

\*This second rule applies only to schedules with three or more work periods.

procedures are repeated because each time one or more entries are voided, the matrix is modified and must be examined again to determine: (1) whether any feasible sequences can still be enumerated from the matrix, i.e., whether each row and column in the matrix still contains valid entries, and (2) whether any additional entries are now dedicated. The iterative use of both procedures continues until either of the following occurs:

- (1) the matrix is rejected by procedure 1 because of the absence of valid entries in a row or column or
- (2) no additional entries are voided by procedure 2.

To illustrate the use of these procedures, consider the reduced matrix for the partitioning set (6,7) shown in figure 6.37. Examination of the matrix for procedure 1 indicates that at least one valid entry exists in each row and column, and hence the matrix may contain one or more feasible sequences. Examining the matrix for procedure 2 reveals that three of the matrix entries: (1,2), (2,3), and (3,4), are the only valid entries in either their respective rows or columns. Hence, each entry is, by definition, dedicated and can be used to void other entries in the matrix (e.g., dedicated entry (1,2) can be used to eliminate entry (4,2) and dedicated entry (3,4) can be used to eliminate entry (3,1)).

The original reduced matrix has now been "conditioned" down to the four entries shown in figure 6.38. (The (4,1)

		Recreation Period			
		1	2	3	4
Recreation Period	1	-	6	-	-
	2	-	-	7	-
	3	-	-	-	6
	4	6	-	-	-

Figure 6.38

Conditioned Reduced Matrix from Figure 6.37

entry is now also dedicated because of the voiding of entries (3,1) and (4,2)). Each valid entry remaining in the matrix is dedicated and, by observation, sequence {1,2,3,4,1} is the only feasible sequence that can be enumerated from the matrix. The conditioned matrix in figure 6.38 corresponds to the reduced matrix associated with node 5 in the tree diagram in figure 6.23 (the schedule corresponding to that node and to sequence {1,2,3,4,1} is shown in figure 6.24).

This example illustrates the computational benefits that can be realized when the acceleration procedures based on work period length (described in section 6.7.2) and matrix conditioning are used. Beginning with the original matrix in figure 6.37, the optimal schedule was determined, first by creating the reduced matrix for the partitioning set (6,7), and second by using the matrix conditioning rules to eliminate all but four of the original matrix entries. These four entries identified the only feasible schedule in set (6,7). If set (6,7) was the top ranking set, then the schedule found would represent the optimal solution -- "enumerated" without use of the branch-and-bound algorithm.

Matrix conditioning procedures 3 and 4 are used exclusively on reduced separation matrices associated with partitioning sets. The conditioning procedures are motivated by following observation: although the reduced matrix for each partitioning set (a,b) is constructed in a manner which insures the inclusion of all schedules which belong to set (a,b), the reduced matrix may also contain schedules which belong to other sets. As an example, the reduced matrix for the partitioning set (6,8) shown in figure 6.39 contains valid entries (3,2) and (4,2), and the four entries contained in the reduced (and conditioned) matrix for partitioning set (6,7) shown in figure 6.38. As

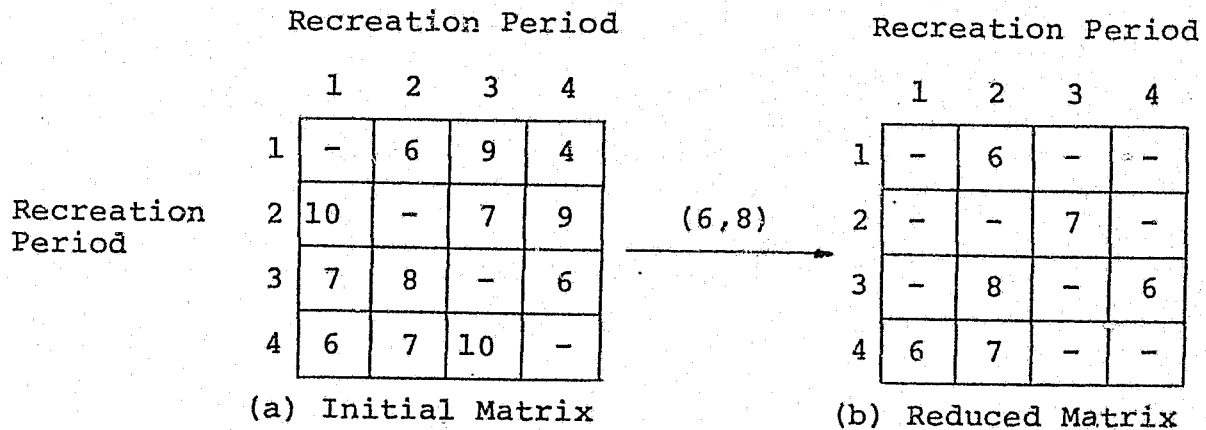


Figure 6.39

Original 4x4 Separation Matrix and  
Reduced (6,8) Matrix

a result, the reduced matrix for the (6,8) set contains all feasible schedules belonging to set (6,8) (i.e., all feasible schedules with  $W_{\min} = 6$  and  $W_{\max} = 8$ ), and all feasible schedules belonging to set (6,7)\*. Conditioning procedures 1 and 2 are not used to eliminate "spurious"

\* In general, the reduced matrix for partitioning set (a,b) contains all feasible schedules contained in sets (c,d) where  $a \leq c$  and  $d \leq b$ .

schedules from a reduced matrix; rather they are used to determine the presence of and to accelerate the enumeration of all feasible schedules within a matrix. In contrast, conditioning procedures 3 and 4 are used to determine the presence of and to accelerate the enumeration of feasible schedules contained in partitioning set  $(a,b)$ .

Procedures 3 and 4 are based on the observation that each feasible schedule in the partitioning set  $(a,b)$  must have  $W_{\min} = a$  and  $W_{\max} = b$  (i.e., the entry set defined by each feasible sequence must contain at least one entry equal to  $W_{\min}$  and at least one entry equal to  $W_{\max}$ ). For procedure 3, the reduced matrix is examined to insure that it contains entries (at least one of each) equal to  $W_{\min}$  and  $W_{\max}$ . If either condition fails, the reduced matrix contains no feasible schedules from set  $(W_{\min}, W_{\max})$ , and the matrix is discarded without use of the branch-and-bound algorithm.

Procedure 4, only applied to matrices which satisfy the conditions of procedure 3, is based on the observation that if only one valid entry in the reduced matrix equals either  $W_{\min}$  or  $W_{\max}$ , that entry (or entries if one entry is found for  $W_{\min}$  and another for  $W_{\max}$ ) must appear in the entry set for every feasible schedule belonging to the partitioning set  $(W_{\min}, W_{\max})$ . As a result, each such entry (there can only be two in any matrix) can be dedicated, and the voiding rules identified in procedure 2 can be applied.

Procedures 3 and 4 can be illustrated using the reduced matrix for the partitioning set (6,8) shown in figure 6.39. The two values of interest for the examination of the matrix for procedure 3 are  $W_{\min} = 6$  and  $W_{\max} = 8$ . Both values are present in the reduced matrix. Since the (3,2) entry is the only matrix entry equal to  $W_{\max} = 8$ , it can be dedicated and used to void entries (1,2), (3,4), and (4,2). Since matrix entries have been voided, all four conditioning procedures are applied again. The conditioned matrix is now rejected by procedure 1 because of the absence of any valid entries in either row 1 or column 4. Hence, the matrix contains no schedules for set (6,8).

## 7. THE DESIGN OF MULTISHIFT PR SCHEDULES

### 7.1 INTRODUCTION

This chapter describes a methodology for constructing multishift PR schedules using the schedule attributes identified in chapter 2 and the enumeration algorithms described in chapters 4, 5, and 6. The methodology is presented in four parts. First, schedule properties associated with shift changeovers (i.e., points in a multishift PR schedule when officers change shift assignments) are reviewed, and sample schedules are examined to illustrate that the use of optimal one-shift PR schedules does not insure the formation of preferable (or even acceptable) multishift schedules.

Next, shift changeover properties are quantified, and used to classify non-cyclic, one-shift schedules for each shift tour to be used within the multishift schedule. This classification scheme is used to partition all feasible non-cyclic schedules for each tour into a small number of sets such that the schedules within each set are identical in terms of their shift changeover properties.

The next section describes the enumeration of a set of optimal non-cyclic schedules for each shift tour. The algorithms developed in chapters 4, 5, and 6, although originally designed for the enumeration of cyclic schedules,

are used for this task by identifying a 1-1 relationship between each non-cyclic schedule to be examined and a corresponding cyclic schedule.

The collection of optimal non-cyclic schedules enumerated for each tour are used to construct multishift schedules by selecting and placing in proper sequence one non-cyclic schedule for each tour. In the final section of this chapter, a simple implicit enumeration algorithm is described for examining all combinations of the one-shift schedules, and identifying the most preferable multishift schedules than can be constructed from them.

## 7.2 MULTISHIFT SCHEDULE PROPERTIES

Multishift schedules are cyclic PR schedules in which each officer rotates through two or more shift assignments during each rotation period of the complete schedule. As an example, the multishift schedule in figure 7.1 requires each officer to spend time on three shifts (afternoon, day, and night). The schedule has a 13-week period consisting of five weeks on the afternoon shift, four weeks on the day shift, and four weeks on the night shift.\*

All of the schedule measures identified in chapter 2 for the determination of optimal one-shift PR schedules

---

\*The schedule in figure 7.1 is a three-shift schedule with one tour on each shift (three tours per period). Multishift schedules can include multiple tours on each shift during each schedule period. A multishift schedule with six tours during each period (i.e., two tours on each of three shifts) is shown in figure 1.5.

Shift		M	T	W	T	F	S	S
Afternoon	1	R	R	R				
	2			R	R			
	3					R	R	R
	4							
	5	R	R					
Day	6						R	R
	7	R						
	8		R	R				
	9				R	R		
Night	10	R	R					
	11			R	R			
	12					R	R	
	13							

Figure 7.1

Sample Three-Shift PR Schedule

(e.g., the number and frequency of weekend recreation periods, the lengths of work and recreation periods, etc.) can also be used to assess the desirability of multishift schedules. In addition, the discussion in chapter 2 also points out that multishift schedules should be evaluated in terms of their shift changeover properties.

To review briefly, in a multishift schedule each point in the schedule at which shift assignments are changed is called a shift changeover point. There are three changeover points in the schedule in figure 7.1: at the end of weeks 5, 9, and 13.\* The attributes of a multishift schedule at each shift changeover point are important because unless the schedule has been carefully designed, several undesirable features can appear. Before describing these features, it is first convenient to introduce the following definition: a changeover recreation period is a recreation period which includes either the last day (Sunday) of a shift assignment, the first day (Monday) of a shift assignment, or both of those days. There are two changeover periods in the schedule in figure 7.1: the three-day period in week 1 (which contains the first day of the afternoon shift) and the two-day period in week 10 (which contains the first day of the night shift). There cannot, by definition, be more than one

---

\* Since all of the PR schedules discussed in this study begin on Monday and end on Sunday, every shift changeover occurs from Sunday to Monday.

changeover recreation period at each shift changeover point in a schedule.

The absence of a changeover recreation period at a shift changeover point in a multishift schedule is usually considered undesirable because an officer changing shift assignments may experience:

(1) a short off-duty period between the last work day on the shift tour just completed and the first work day on the shift tour just initiated;\* and

(2) a long and fatiguing work period that begins on one shift and ends on the following shift.

It is important to recognize that these undesirable features can arise from the absence of a changeover recreation period even when the schedules used for each shift tour are considered satisfactory as one-shift schedules. As an example, the day and afternoon schedules in figure 7.1, if used as cyclic schedules (see figure 7.2) may be quite acceptable; yet when placed in sequence in a multishift schedule they produce a shift changeover point without a changeover recreation period. The absence of the changeover recreation period results in a 10-day work period that begins on Wednesday of the last week of the afternoon shift and ends on Friday of the first week of the day shift, and an

---

\* Without a recreation changeover period it is possible for an officer to have as few as eight hours between successive work assignments. As a result, the officer may be required to work as many as 16 hours in a single 24-hour period. (See tables 2.4 and 2.5.)

	M	T	W	T	F	S	S
1						R	R
2	R						
3		R	R				
4				R	R		

(a) Day Shift

	M	T	W	T	F	S	S
1	R	R	R				
2			R	R			
3					R	R	R
4							
5	R	R					

(b) Afternoon Shift

	M	T	W	T	F	S	S
1	R	R					
2			R	R			
3					R	R	R
4							

(c) Night Shift

Figure 7.2

Three One-Shift Schedules Used in the Multishift Schedule in Figure 7.1

8-hour off-duty period\* between the last work day on the afternoon shift and the first work day on the day shift.

Both of the undesirable properties associated with the afternoon-day shift changeover in figure 7.1 would be eliminated if a recreation period was scheduled at the changeover point. Such a recreation period would insure that each officer would receive some days off between shift assignments, thereby eliminating the 8-hour break between successive work days and the long work period extending over two shifts. As an example, consider the multishift schedule in figure 7.3. Both the afternoon and night shifts in this schedule are identical to those shown in figure 7.1; only the day shift schedule is different. Each shift changeover point in figure 7.3 contains a changeover recreation period (a two-day period at the afternoon-day changeover, a three-day period at the day-night changeover, and another three-day period at the night-afternoon changeover). The recreation period at the afternoon-day changeover eliminates both of the undesirable features identified in the schedule of figure 7.1.

As indicated in chapter 2, the multishift schedule design methods developed for this thesis are limited to the construction of multishift PR schedules with changeover

---

\*Assuming the night shift is eight hours long.

Shift		M	T	W	T	F	S	S
Afternoon	1	R	R	R				
	2			R	R			
	3					R	R	R
	4							
	5	R	R					
Day	6	R	R					
	7			R	R			
	8					R	R	
	9							R
Night	10	R	R					
	11			R	R			
	12					R	R	R
	13							

Note: Multishift schedule is based on the same manpower allocation used for the schedule in figure 7.1

Figure 7.3

Sample Three-Shift PR Schedule



**CONTINUED**

**4 OF 6**

recreation periods at every changeover point.\*

Although the presence of a changeover recreation period eliminates short breaks between successive work assignments and long intershift work periods, other undesirable (or unacceptable) changeover conditions can arise. As an example, although the day-night changeover point in the multishift schedule shown in figure 7.1 has a changeover period (i.e., the two-day period beginning on Monday of week 10), the work period at the end of the day shift is only two days long. If the day shift schedule were used as a one-shift cyclic schedule, the two work days at the end of week 9 would be joined with the five work days at the beginning of week 6 to form a work period of seven days.† When used in the multishift schedule, however, the two work days at the end of the day shift become a work period defined by the two-day recreation period in week 9 and the changeover recreation period in week 10. If the same lower limit of four days, used to design the three shift schedules in figure 7.1, were applied to work periods in the multishift schedule, then the schedule shown in figure 7.1 would be unacceptable.

---

\* Computational experience indicates that this requirement is not very restrictive. Empirical evidence suggests that for many practical problems, if any multishift schedules can be found ignoring the changeover period requirement, then schedules satisfying the changeover requirement also exist.

† Each of the three shift schedules shown in figure 7.1 were designed as one-shift cyclic schedules using upper and lower limits on work period lengths of eight and four days respectively.

Another design difficulty that can arise when a changeover recreation period is present is illustrated by the day-night changeover point in figure 7.3. The interesting feature of this changeover point is the fact that the three-day changeover recreation period extends over two shifts (i.e., its total length is dependent upon recreation days contributed by both shifts). The design difficulty introduced by this construction is that if each shift schedule is designed with an upper limit of  $U_R$  on the length its recreation periods, it becomes possible, if both schedules should contribute  $U_R$  recreation days, for changeover periods to be  $2U_R$  days long. Hence, if the  $U_R$  upper limit is also applied to changeover recreation period lengths, it is possible that two one-shift schedules, each containing only periods of acceptable lengths may produce an excessively long changeover recreation period.

The resolution of the design difficulties associated with shift changeover points is the central focus of this chapter. The algorithms described below were designed to meet two objectives:

- (1) the placement of a changeover recreation period of acceptable length at every shift changeover point; and
- (2) the design of work periods of acceptable length to precede and follow each changeover recreation period.

The remaining three sections of this chapter describe how each objective was achieved.

### 7.3 CLASSIFICATION OF NON-CYCLIC, ONE-SHIFT SCHEDULES

The sample schedules presented in the previous section illustrate that the use of optimal cyclic schedules for each shift tour does not insure the design of acceptable multishift schedules. These examples indicate that the primary difficulties associated with the design of multishift schedules are related to the placement of recreation periods at each shift changeover point. Fundamental to these difficulties is the fact that each shift schedule, designed initially as a one-shift cyclic schedule, is non-cyclic when used in a multishift schedule. Although a multishift schedule is cyclic over its entire rotation period (e.g., in figure 7.3, an officer rotates from week 13 back to week 1), each component shift schedule is non-cyclic because after working for the specified number of weeks on each shift officers move to a new shift.

It is important to note that the terms cyclic and non-cyclic describe how an one-shift schedule is used rather than its structure. For example, when used as cyclic schedules, one-shift schedules have no specific beginning or ending brackets, and the labelling of each bracket in the schedule is arbitrary; only the sequence or order of the brackets is significant. Used as non-cyclic schedules, however, one-shift schedules are characterized, not only by the sequence in which the brackets are worked, but also by which bracket is designated as the initial bracket to be worked during each tour on that shift.

It is convenient to classify the properties of non-cyclic schedules as either,

- (1) internal properties - Schedule properties that are independent of which bracket is designated as the initial week; or
- (2) changeover properties - Schedule properties which are dependent upon the particular bracket that is designated as the first week of the non-cyclic schedule.

The optimality of the cyclic schedules derived in chapter 6 was determined exclusively on the basis of their internal properties. When a cyclic schedule is used in a multishift schedule, however, the closed cycle of brackets is "broken" and a non-cyclic schedule is produced with a specific set of changeover properties. Consequently, the design of multishift PR schedules must be based upon a methodology for constructing non-cyclic shift schedules which accounts for both internal and changeover properties.

It is useful to distinguish three types of recreation periods which a non-cyclic schedule can possess:

- (1) a beginning period - a recreation period which precedes the first work period in the schedule;
- (2) an ending period - a recreation period which follows the final work period in the schedule; and
- (3) interior periods - any recreation period which appears between the first and last work periods of the schedule.

Beginning recreation periods always start on the first day of the shift schedule (Monday), and ending periods always

end on the last day of the shift schedule (Sunday). By definition, every non-cyclic schedule has one beginning recreation period and one ending recreation period (each may have length zero).<sup>\*</sup> A non-cyclic schedule may have any number of interior periods; there may be none or several. (Every recreation period in a one-shift cyclic schedule is defined to be an interior period.) The lengths of the beginning and ending recreation periods for a non-cyclic schedule indicate the number of recreation days that are contributed to the changeover recreation periods at the beginning and end of the shift tour. Specifically, let the two-number pair  $(b,e)$  represent the set of all non-cyclic schedules with a beginning period of length  $b$  and an ending period of length  $e$ . To illustrate, the three non-cyclic shift schedules in figure 7.3 belong to sets  $(3,0)$ ,  $(2,1)$ , and  $(2,0)$  respectively.

The requirement that a changeover recreation period of acceptable length be placed at each shift changeover point in a multishift schedule is equivalent to the requirement that:

$$L_{CR} \leq l_i \leq U_{CR} \quad i = 1, 2, \dots, N \quad (7.1)$$

---

<sup>\*</sup> If the first day of a shift schedule is a work day, the beginning period is defined to have length zero. Similarly, if a shift schedule ends with a work day, the ending period is defined to have length zero.

where  $\ell_i$  is the length of the changeover recreation period between shifts  $i$  and  $i+1$ , and  $L_{CR}$  and  $U_{CR}$  are the lower and upper limits respectively on the lengths of changeover recreation periods.\* The length of each changeover period  $\ell_i$  equals the sum of the lengths of the ending period of shift  $i$  and the beginning period of shift  $i+1$ . Hence, (7.1) can be written:

$$L_{CR} \leq e_i + b_{i+1} \leq U_{CR} \quad i = 1, 2, 3, \dots, N \quad (7.2)$$

$$(N+1 \equiv 1)$$

From (7.2) it may be seen that:

$$0 \leq e_i, b_{i+1} \leq U_{CR} \quad i = 1, 2, 3, \dots, N$$

$$(N+1 \equiv 1)$$

which indicates that the beginning and ending periods for each non-cyclic shift schedule used in a multishift schedule must satisfy the limits:

$$0 \leq b_i, e_i \leq U_{CR} \quad i = 1, 2, \dots, N \quad (7.3)$$

Result (7.3) implies that the only sets of non-cyclic schedules from which acceptable schedules for each shift tour can be obtained are:

$$\{(0,0), (0,1), \dots, (0, U_{CR}), (1,0), (1,1), \dots, (U_{CR}, U_{CR}-1), (U_{CR}, U_{CR})\}$$

---

\* For convenience, it is assumed that the same limits ( $L_{CR}, U_{CR}$ ) apply to every changeover period in the multishift schedule; and further, that these limits are identical to those used to design the interior recreation periods for each non-cyclic schedules (i.e.,  $U_{CR} = U_R$  and  $L_{CR} = L_R$ ).

or more concisely, the collection  $S$  of all sets

$$\{(b,e) \mid 0 \leq b,e \leq U_{CR}\}.$$

The number of sets in  $S$  equals  $(1+U_{CR})^2$ . Hence, if  $U_{CR} = 4$ , there are 25 distinct sets of non-cyclic schedules (ranging from  $(0,0)$  to  $(4,4)$ ) which contain schedules which satisfy condition (7.3).

Because the shift changeover properties depend on the changeover recreation periods, it is not possible to design optimal multishift schedules merely by using an optimal non-cyclic schedule for each component shift derived independently of the other shifts. However, the  $(b,e)$  classification of non-cyclic schedules into (a relatively small number of) sets may be used to deal with this problem while preserving the independence of the design process for each shift. Within each set  $(b,e)$ , all of the non-cyclic schedules have the same changeover properties (i.e., each schedule begins and ends with the same number of recreation days). As a result, differences in the preferability of schedules within the set  $(b,e)$  can only be based on differences in their internal properties; and hence, it is possible to determine an optimal non-cyclic schedule for each set  $(b,e)$  using the algorithms developed in chapters 4, 5, and 6.

In this manner, it is possible to construct for each shift,  $(1+U_{CR})^2$  dominating non-cyclic schedules, corresponding to the optimal schedule for each set  $(0,0)$ ,  $(0,1)$ , ...,  $(U_{CR}, U_{CR})$ . Multishift schedules can then be formed by selecting

one dominating schedule for each shift, and examining the resulting shift changeover properties in the multishift schedule (these properties are immediately measurable since they are determined by the values of  $b$  and  $e$  associated with the schedules selected for each shift). Since the internal properties of the component schedules for each shift are optimal for any value of  $b$  and  $e$  selected, the resulting multishift schedule will be optimal if the optimal shift changeover conditions can be found.

The next section describes how the set of optimal non-cyclic schedules can be constructed for each shift.

#### 7.4 DESIGN AND CONSTRUCTION OF DOMINATING NON-CYCLIC SCHEDULES

##### 7.4.1 Introduction

This section describes how the enumeration algorithms developed in chapters 4, 5, and 6 can be used to generate a set of optimal non-cyclic schedules for each shift tour in a multishift schedule.

The design procedures described in this section are based on the observation that corresponding to each set of non-cyclic schedules  $(b,e)$ , for which an optimal schedule is sought, is a set of cyclic schedules  $(b,e)^C$ \* with the following properties:

---

\*The label  $(b,e)^C$  is used to identify the set of cyclic schedules which correspond to the set  $(b,e)$  of non-cyclic schedules; the  $b$  and  $e$  values are used merely for identification since cyclic schedules have no beginning or ending periods.

- (1) a 1-1 correspondence exists between each non-cyclic schedule in the set  $(b,e)$  and a cyclic schedule in the set  $(b,e)^C$ ; and
- (2) the optimal cyclic schedule in set  $(b,e)^C$  corresponds to the optimal non-cyclic schedule in the set  $(b,e)$ .

Using the corresponding set of cyclic schedules, the procedure for the design of the dominating non-cyclic schedule for set  $(b,e)$  consists of the following steps:

1. Construction of the alternate set of cyclic schedules  $(b,e)^C$ .
2. Use of the algorithms from chapters 4, 5, and 6 to determine the optimal cyclic schedule for the set  $(b,e)^C$ .
3. Use of the optimal cyclic schedule to identify the optimal non-cyclic schedule for set  $(b,e)$ .

The basic concepts and procedures for each step are discussed below.

#### 7.4.2 Artificial Recreation Periods

Fundamental to the construction of  $(b,e)^C$  is the concept of an artificial recreation period. When designing cyclic schedules for a given shift, an artificial period is used to account for the time spent on other shifts (including both work and recreation periods). To illustrate, consider a supervisor who is permanently assigned to shift A in the two-shift schedule shown in figure 7.4. From the supervisor's point of view, each officer working the two-shift schedule is assigned to shift A for five weeks and is on recreation for the following four-week period (see figure 7.5). In reality, of course, each

	M	T	W	T	F	S	S
1	Five-Week Schedule for Shift A						
2							
3							
4							
5							
6	Four-Week Schedule for Shift B						
7							
8							
9							

Figure 7.4

Actual Nine-Week  
Two-Shift Schedule

	M	T	W	T	F	S	S
1	Five-Week Schedule for Shift A						
2							
3							
4							
5							
6	Four-Week Recreation Period						
7							
8							
9							

Figure 7.5

Perception of the Nine-Week  
Schedule in Figure 7.4 by  
a Supervisor Permanently  
Assigned to Shift A

officer spends his four-week "artificial recreation period" assigned to shift B. However, for purposes of designing the schedule for shift A, time spent on other shifts may be treated as a single recreation period beginning on a Monday and ending on a Sunday.

Each cyclic schedule in the set  $(b,e)^C$  is formed by attaching an artificial recreation period to a non-cyclic schedule in the set  $(b,e)$  to represent the "off-shift" time between the end of the last week and the beginning of the first week of the non-cyclic schedule. The properties required of the artificial period include the following:

- (1) it has length zero (i.e., the period includes no recreation days);
- (2) it begins on the Monday following the last Sunday of the non-cyclic schedule; and
- (3) it ends on the Sunday preceding the first Monday of the non-cyclic schedule.

As an example, consider the cyclic graph in figure 7.6 from which a four-week, non-cyclic schedule must be designed. Figure 7.7 shows the same cyclic graph with an artificial recreation period added (dashed line); note that the period begins on Monday (property 2), ends on Sunday (property 3), and passes through none of the recreation day nodes in the graph (property 1).

The elementary separation matrices for both cyclic graphs are shown in figures 7.8 and 7.9. The matrix for

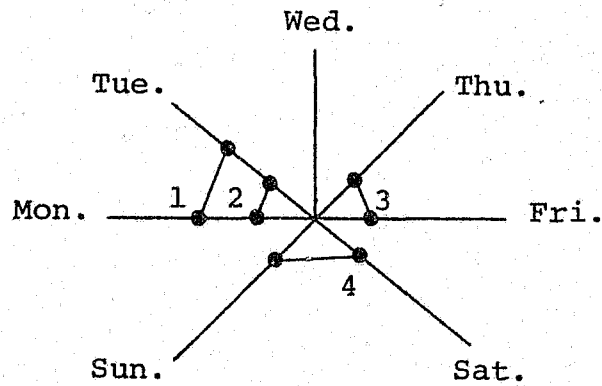


Figure 7.6

Sample Cyclic Graph with  
Four Recreation Periods

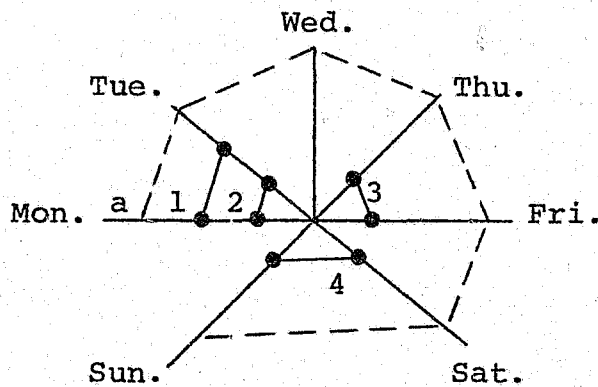


Figure 7.7

Sample Cyclic Graph with an  
Artificial Recreation Period

	Recreation Period			
	1	2	3	4
1	-	5	1	4
2	5	-	1	3
3	2	2	-	0
4	0	0	3	-

Figure 7.8

Elementary Separation Matrix for the  
Cyclic Graph in Figure 7.6

	Recreation Period				
	1	2	3	4	a
1	-	5	1	3	5
2	5	-	1	3	5
3	2	2	-	0	2
4	0	0	3	-	0
a	0	0	3	5	-

Figure 7.9

Expanded Elementary Separation Matrix for  
the Cyclic Graph in Figure 7.7

the cyclic graph with the artificial period has one additional row and column. The matrix entries in row "a" and column "a" represent work period lengths defined by the artificial period and each of the real recreation periods. Each entry  $(a, j)$  in row a indicates the length of the work period that will appear at the start of week 1 of the non-cyclic schedule if period j is used as the first recreation period in the schedule; for example: if period 3 is placed in week 1, the schedule would begin with a three-day work period, as seen from matrix entry  $(a, 3)$ . The three days represent the number of days between the end of the artificial period on Sunday and the beginning of period 3, on Thursday. Any zero-valued matrix entry in row a indicates a recreation period that can be used as a beginning recreation period (i.e., an initial work period of length zero); there are two such periods indicated in figure 7.9: periods 1 and 2. Each entry  $(i, a)$  in column a indicates the length of the work period that will end the last week of the shift if recreation period i is used as the last period in the schedule. For example: if recreation period 2 is used as the last period, the shift schedule would end with a five-day work period, as seen from matrix entry  $(2, a)$ . The five-day work period separates the last day of period 2 (Tuesday) from the first day of the artificial period, which is the following Monday. Any zero-valued entry in

column a indicates a recreation period that ends on Sunday and can be used as an ending recreation period for the schedule; only one such period exists in figure 7.1: period 4.

The use of artificial recreation periods is indicated by the structure of the expanded separation matrix. By using certain entries in row a and column a, real recreation periods can be "placed" adjacent to the artificial recreation period. A real period placed immediately after the artificial period (i.e., using period j with  $(a,j) = 0$ ) can be used as a beginning period for the schedule; a period placed immediately in front of the artificial period (i.e., using period i with  $(i,a) = 0$ ) can be used as an ending period for the schedule. Hence the expanded separation matrix provides a mechanism for the design of cyclic schedules which correspond to non-cyclic schedules with specific types of beginning and ending recreation periods (i.e., the matrix provides a mechanism for the control of the changeover properties of the non-cyclic schedules). Procedures for modifying the expanded elementary separation matrix to insure correct placement of beginning and ending recreation periods to design non-cyclic schedules for the set (b,e) are discussed below.

#### 7.4.3 Use of the Expanded Separation Matrix

The construction of the modified, elementary separation matrix for the design of non-cyclic schedules involves three steps:

- (1) the adjustment of interior work period lengths (i.e., the adjustment of matrix entries in rows and columns other than row a and column a);\*
- (2) the adjustment of entries in row a; and
- (3) the adjustment of entries in column a.

The procedure for adjusting interior work period lengths is identical to the modification procedures described in section 6.5.1. The only information required is the upper and lower limits on work period lengths (i.e.,  $U_W$  and  $L_W$ ). Briefly, the modification procedure for each matrix entry (i,j) is:

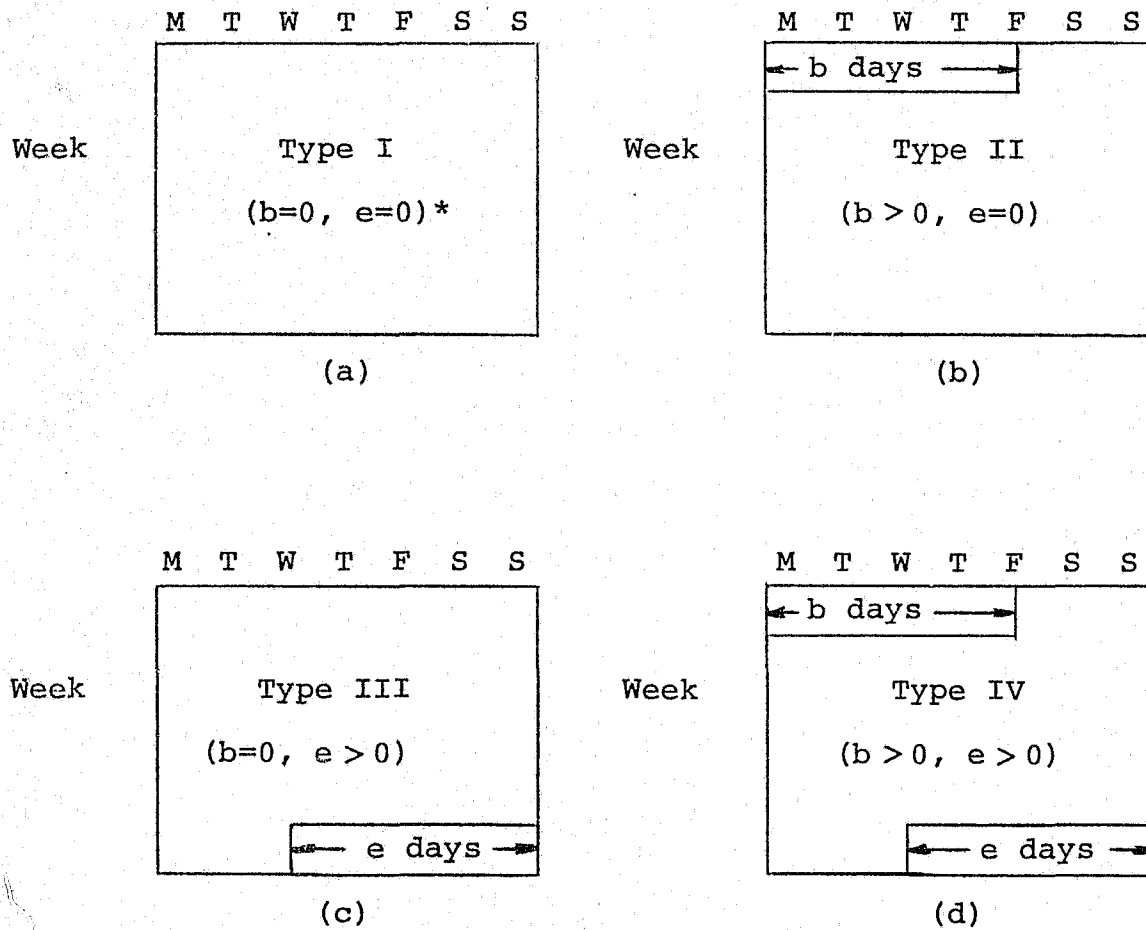
Step 1 Interior Matrix Entries ( $i \neq a, j \neq a$ )

- (1) if  $L_W \leq s_{ij} \leq U_W$ , entry (i,j) is unchanged, ( $s_{ij}$  is the value of the (i,j) entry);
- (2) if  $s_{ij} > U_W$ , void the (i,j) entry;
- (3) if  $s_{ij} < L_W$ , let  $s'_{ij} = s_{ij} + 7k$ , increment k ( $k = 1, 2, 3, \dots$ ) until  $L_W \leq s'_{ij} \leq U_W$  and replace  $s_{ij}$  with  $s'_{ij}$ ; if an acceptable  $s'_{ij}$  value cannot be found, void the (i,j) entry.

The procedures for modification of entry values in row a and column a are dependent upon the required length of the beginning and ending recreation periods. There are four types of non-cyclic schedules to consider; these are (see figure 7.10):

---

\* These entries represent the number of work days separating interior recreation periods.



\* $b$  equals the length (in days) of the beginning recreation period, and  $e$  equals the length (in days) of the ending recreation period.

Figure 7.10

Four Schedule Types for the Placement of  
Beginning and Ending Recreation Periods  
in Non-Cyclic Schedules

- (1) Type I  $(0,0)$  - both the beginning and ending recreation periods have length zero;
- (2) Type II  $(b,0)$  - the beginning period has length  $b$ , ( $b>0$ ) and the ending period has length zero;
- (3) Type III  $(0,e)$  - the beginning period has length zero and the ending period has length  $e$ , ( $e>0$ ); and
- (4) Type IV  $(b,e)$  - both the beginning and ending period have lengths greater than zero.

Consider first, the design of beginning recreation periods. Only two possibilities exist: either the beginning period has length zero (schedule types I and III), or the beginning period has a length greater than zero (schedule types II and IV). The latter possibility is discussed first.

The design of a non-cyclic schedule with a beginning period of length  $b$  days, is equivalent to "placing" a recreation period,  $b$  days long, adjacent to (and following) the artificial period in the schedule, (i.e., the matrix entry corresponding to the work period between the artificial period  $a$  and the designated beginning period  $j$  must equal zero). Hence, the design of a schedule with a beginning recreation period with length greater than zero requires:

- (1) the presence of a zero-valued matrix entry  $(a,j)$  in the expanded separation matrix (where  $j$  corresponds to a real recreation period of length  $b$ ), and
- (2) the use of entry  $(a,j)$  in a feasible sequence of recreation periods.

These requirements dictate the following matrix modification rules for row  $a$  for the design of non-cyclic schedules with a beginning period with length  $b$  ( $b > 0$ ):

1. Examine row  $a$  and select one zero-valued entry  $(a, j)$  (where recreation period  $j$  has length  $b$ ); if no entries are found, the matrix cannot produce any acceptable schedules.
2. Dedicate entry  $(a, j)$  by voiding all other entries in row  $a$  and column  $j$  (this insures the use of entry  $(a, j)$  in every feasible sequence).

To design a non-cyclic schedule with a beginning period of length zero, a different matrix modification procedure is used. Since the beginning period has length zero, no recreation days are present at the beginning of the schedule (i.e., the shift actually begins with a work period that separates the end of the artificial period from the beginning of the first interior recreation period). Any interior recreation period can be used so long as it forms a work period of acceptable length with the artificial period. Hence, the entry values in row  $a$  of the expanded elementary matrix are modified using the same procedures that are used to modify matrix entries which correspond to interior work periods (i.e., the procedures described in step 1 above).

Summarizing, the matrix modification procedures for row  $a$  of an expanded, elementary separation matrix are:

Step 2: Row a Entries ( $i = a$ )

- (1) If  $b = 0$ , modify each entry in row a using the rules in step 1.
- (2) If  $b > 0$ ,
  - (a) select one zero-valued  $(a,j)$  entry in row a corresponding to a recreation period with length b, and
  - (b) dedicate the  $(a,j)$  entry by voiding all other entries in row a and column j.

The matrix modification procedures for the entry values in column a of the elementary matrix parallel those described for row a. Two procedures exist: one for the design of ending recreation periods with length zero (schedule types III and IV), and another for the design of ending periods with lengths greater than zero (schedule types I and II). If the length of the ending period is greater than zero (i.e.,  $e > 0$ ), a real recreation period with length e must be placed adjacent to the beginning of the artificial period (i.e., one entry in column a must equal zero). Once a zero-valued  $(i,a)$  entry is identified (where i corresponds to a real recreation period of length e), it is dedicated to insure its use in every sequence enumerated from the matrix.

If a zero-length ending period is desired, the shift schedule ends with a work period that separates the last interior recreation period from the beginning of the artificial period. For this kind of schedule, column a entry values are modified according to the same rules that are used to modify

matrix entries corresponding to interior work periods (i.e., step 1 above).

Summarizing, the matrix modification rules for column a of an expanded elementary separation matrix are:

Step 3: Column a Entries ( $i = a$ )

- (1) If  $e = 0$ , modify each entry in column a using the rules in step 1.
- (2) If  $e > 0$ ,
  - (a) select one zero-valued  $(i,a)$  entry in column a corresponding to a recreation period with length  $e$ , and
  - (b) dedicate the  $(i,a)$  entry by voiding all other entries in row  $i$  and column a.

Several examples illustrating the use of the matrix modification procedures are presented in the following section.

#### 7.4.4 Modified Matrix Examples

This section presents several examples of the matrix modification procedures described above. Using the expanded, elementary separation matrix in figure 7.9, the modified matrix for each of the schedule types illustrated in figure 7.10 are derived. The four modified matrices, each based on work period limits:  $L_W = 4$  and  $U_W = 8$ , are shown in figures 7.11 through 7.14.

The modified separation matrix for a type I schedule shown in figure 7.11 was formed by using the step 1 procedure for all of the matrix entries, (including those in row a and column a); the seven voided entries (excluding the diagonal terms) represent work period lengths which could not be modified to fit the interval  $[4,8]$ .

		Recreation Period					Row min/max
		1	2	3	4	a	
Recreation Period	1	-	5	8	-	5	5/8
	2	5	-	8	-	5	5/8
	3	-	-	-	7	-	7/7
	4	7	7	-	-	7	7/7
	a	7	7	-	5	-	5/7
Column min/max		5/7	5/7	8/8	5/7	5/7	28/36

Figure 7.11

Modified Separation Matrix for Type I  
Schedule (0,0) Based on the Elementary  
Separation Matrix in Figure 7.9

		Recreation Period					Row min/max
		1	2	3	4	a	
Recreation Period	1	-	5	8	-	-	5/8
	2	-	-	8	-	5	5/8
	3	-	-	-	7	-	7/7
	4	-	7	-	-	7	7/7
	a	0	-	-	-	-	0/0
Column min/max		0/0	5/7	8/8	7/7	5/7	25/29

Figure 7.12

Modified Separation Matrix for Type II  
Schedule (b,0) Based on the Elementary  
Separation Matrix in Figure 7.9

		Recreation Period					Row
		1	2	3	4	a	min/max
Recreation Period	1	-	5	8	-	-	5/8
	2	5	-	8	-	-	5/8
	3	-	-	-	7	-	7/7
	4	-	-	-	-	0	0/0
	a	7	7	-	-	-	7/7
Column min/max		5/7	5/7	8/8	8/8	0/0	25/29

Figure 7.13

Modified Separation Matrix for Type III  
Schedule (0,e) Based on the Elementary  
Separation Matrix in Figure 7.9

		Recreation Period					Row
		1	2	3	4	5	min/max
Recreation Period	1	-	5	8	-	-	5/8
	2	-	-	8	-	-	8/8
	3	-	-	-	7	-	7/7
	4	-	-	-	-	0	0/0
	5	0	-	-	-	-	0/0
Column min/max		0/0	5/5	8/8	7/7	0/0	20/20

Figure 7.14

Modified Separation Matrix for Type IV  
Schedule (b,e) Based on the Elementary  
Separation Matrix in Figure 7.9

In the matrices for both type II and IV schedules (figures 7.12 and 7.14), the number 1 recreation period has been selected as the beginning period; this is indicated by the fact that the (a,1) entry has value zero and all of the other entries in row a and column 1 are voided.\* Similarly, the (4,a) entry is dedicated in the matrices for the type III and IV schedules (figures 7.13 and 7.14) to insure that recreation period 4 appears as the ending period in every schedule generated from either matrix. The matrix for the type IV schedule in figure 7.14 contains two dedicated entries: (a,1) to insure that recreation period 1 is the beginning period, and (4,a) to insure that period 4 is the ending period.

The four modified matrices illustrate that the use of an artificial recreation period and an expanded separation matrix does not necessarily increase the size of the matrix problem (i.e., increase the number of valid matrix entries). Whenever a beginning or ending period with  $b > 0$  or  $e > 0$  is required, the resulting dedicated entry can be used to eliminate all other valid entries in one row and one column of the matrix.

---

\*The (1,a) entry is also voided in both matrices because (a,1) is a dedicated entry (see section 6.7.3).

Once the expanded, elementary separation matrix has been modified to insure the enumeration of schedules for set (b,e) the branch-and-bound algorithm and acceleration techniques described in chapter 6 can be used. The only additional rule required is the convention that the artificial period is always used as the first period in constructing each feasible sequence. By following this rule, the second period in each sequence always represents the first real recreation period in the corresponding schedule, and the next-to-last period in each sequence always represents the last real recreation period in each schedule.

As an example, consider the enumeration of schedules from the modified matrices in figures 7.11 through 7.14 assuming each schedule must contain exactly 27 work days ( $W = 27$ ). The upper and lower bounds on the total number of work days contained in any feasible sequence derived from each matrix are circled in the lower right-hand corner of each figure. These limits indicate that schedules with  $W = 27$  cannot be obtained from the matrix in figure 7.11 or the matrix in figure 7.14; the limits on the other two matrices (figures 7.12 and 7.13) include the required value and may produce acceptable schedules. Three non-cyclic schedules can be enumerated from these matrices: two from the matrix in figure 7.12 (see figure 7.15), and one from the matrix in figure 7.13 (see figure 7.16).

	M	T	W	T	F	S	S
1	1 R	R					
2	2 R	R					
3				3 R	R		
4						4 R	R
5							

(a)

	M	T	W	T	F	S	S
1	1 R	R					
2				3 R	R		
3						4 R	R
4							
5	2 R	R					

(b)

Figure 7.15

Two Type II Non-Cyclic Schedules Based on the  
Modified Separation Matrix in Figure 7.12

	M	T	W	T	F	S	S
1							
2	1 R	R					
3	2 R	R					
4				3 R	R		
5						4 R	R

Figure 7.16

A Type III Non-Cyclic Schedule Based on the  
Modified Separation Matrix in Figure 7.13

The type II schedules derived from the matrix in figure 7.12 use recreation period 1 as the beginning period; both schedules belong to the  $(2,0)$  set of non-cyclic schedules. The sequence of recreation periods for the schedules are  $\{a,1,2,3,4,a\}$  for schedule (a) and  $\{a,1,3,4,2,a\}$  for schedule (b). The type III schedule derived from the matrix in figure 7.13 uses period 4 as the ending period and belongs to the set  $(0,2)$ ; the solution sequence is  $\{a,1,2,3,4,a\}$ . Although the sequences for schedule (a) in figure 7.15 and the schedule in figure 7.16 are identical, the schedules are different because they were derived from different modified separation matrices.

#### 7.4.5 Verification of the Properties of the Corresponding Set of Cyclic Schedules

In section 7.4.1, the concept of using a corresponding set of cyclic schedules to derive an optimal or dominating non-cyclic schedule for each set  $(b,e)$  was introduced. Two essential properties were identified for each corresponding set of cyclic schedules:

- (1) that a 1-1 correspondence exists between each non-cyclic schedule in the set  $(b,e)$  and a cyclic schedule in the corresponding set  $(b,e)^C$ ; and
- (2) that the optimal cyclic schedule in set  $(b,e)^C$  corresponds to the dominating non-cyclic schedule in set  $(b,e)$ .

The 1-1 correspondence between non-cyclic and cyclic schedules is easily seen by examining the solution sequences for the three schedules shown in figures 7.15 and 7.16. The sequence of recreation periods used for each cyclic schedule and its corresponding non-cyclic schedule are:

	<u>Cyclic Schedule</u>		<u>Non-Cyclic Schedule</u>
Figure 7.15a	{a,1,2,3,4,a}	→	{1,2,3,4}
Figure 7.15b	{a,1,3,4,2,a}	→	{1,3,4,2}
Figure 7.16	{a,1,2,3,4,a}	→	{1,2,3,4}

The sequence for each non-cyclic schedule in set  $(b,e)$  is formed by deleting the artificial period from the corresponding sequence for each cyclic schedule in set  $(b,e)^C$ . Hence, for every distinct schedule in set  $(b,e)^C$ , there is a unique and corresponding schedule in set  $(b,e)$ , and conversely, for every distinct schedule in the set  $(b,e)$ , there is a unique and corresponding schedule in the set  $(b,e)^C$ . The strict 1-1 correspondence between schedules establishes that the sets must contain equal numbers of schedules. This observation, and the fact that (1) the branch-and-bound algorithm implicitly enumerates every schedule in set  $(b,e)^C$ , and (2) the matrix modification rules insure that the non-cyclic schedule associated with every cyclic schedule enumerated from the modified matrix belongs to set  $(b,e)$ , verifies that every non-cyclic schedule in set  $(b,e)$  will be implicitly enumerated when the branch-and-bound algorithm is applied to the modified separation matrix.

The second required property, that the optimal cyclic schedule in  $(b,e)^C$  correspond to the dominating non-cyclic schedule in  $(b,e)$  is achieved by using preference measures applicable to non-cyclic schedules to determine the optimal schedule in set  $(b,e)^C$ . A list of "non-cyclic" preference measures used to rank cyclic schedules in set  $(b,e)^C$  is shown in table 7.1. The table also identifies schedule properties which are measured differently for cyclic and non-cyclic schedules.

#### 7.4.6 The Design of Optimal Non-Cyclic Schedules

Using the concepts introduced above, this section describes the algorithm for determining the set of dominating non-cyclic schedules for each shift tour to be used in a multishift schedule. The design procedure is outlined in figure 7.17 as a sequence of 11 steps; a brief discussion of each step follows.

To illustrate each step in the algorithm as it is discussed, consider the design of a dominating set of non-cyclic schedules for shift i with the following daily allocation of recreation days:

Mon.	Tue.	Wed.	Thu.	Fri.	Sat	Sun.	Total
2	2	0	1	1	1	1	8



Table 7.1

## Preference Measures Used to Rank Non-Cyclic Schedules

Schedule Measures Used to Rank Cyclic Schedules	Modifications for Use with Non-Cyclic Schedules	Measure Values for the Non-Cyclic Schedule in Figure 7.15
1. Number of weekend recreation periods	1. None	1
2. Maximum number of consecutive working weekends	2. Cannot measure across artificial period (e.g. cannot combine weeks 4 and 5 with weeks 7 and 2 in figure 7.15)	2
3. Maximum length work period (days)	3. For (0,0) non-cyclic schedules, cannot combine the first and last work period lengths	8
4. Number of maximum length work periods	4. None	1
5. Recreation period measure	5. For (b,e) non-cyclic schedules with $b > 0$ and $e > 0$ , cannot combine the beginning and ending recreation period lengths	*
6. Work period range	6. None	3
7. Maximum number of days in consecutive work periods	7. First and last work periods are not treated as consecutive work periods	15
8. Number of maximum length consecutive work periods	8. None	1
9. Standard deviation of the work to recreation period length ratios	9. If $b > 0$ , cannot use beginning recreation period; if $e = 0$ , cannot use final work period	0.236

\*See table 2.9 for the preference rankings used to measure recreation period sets.

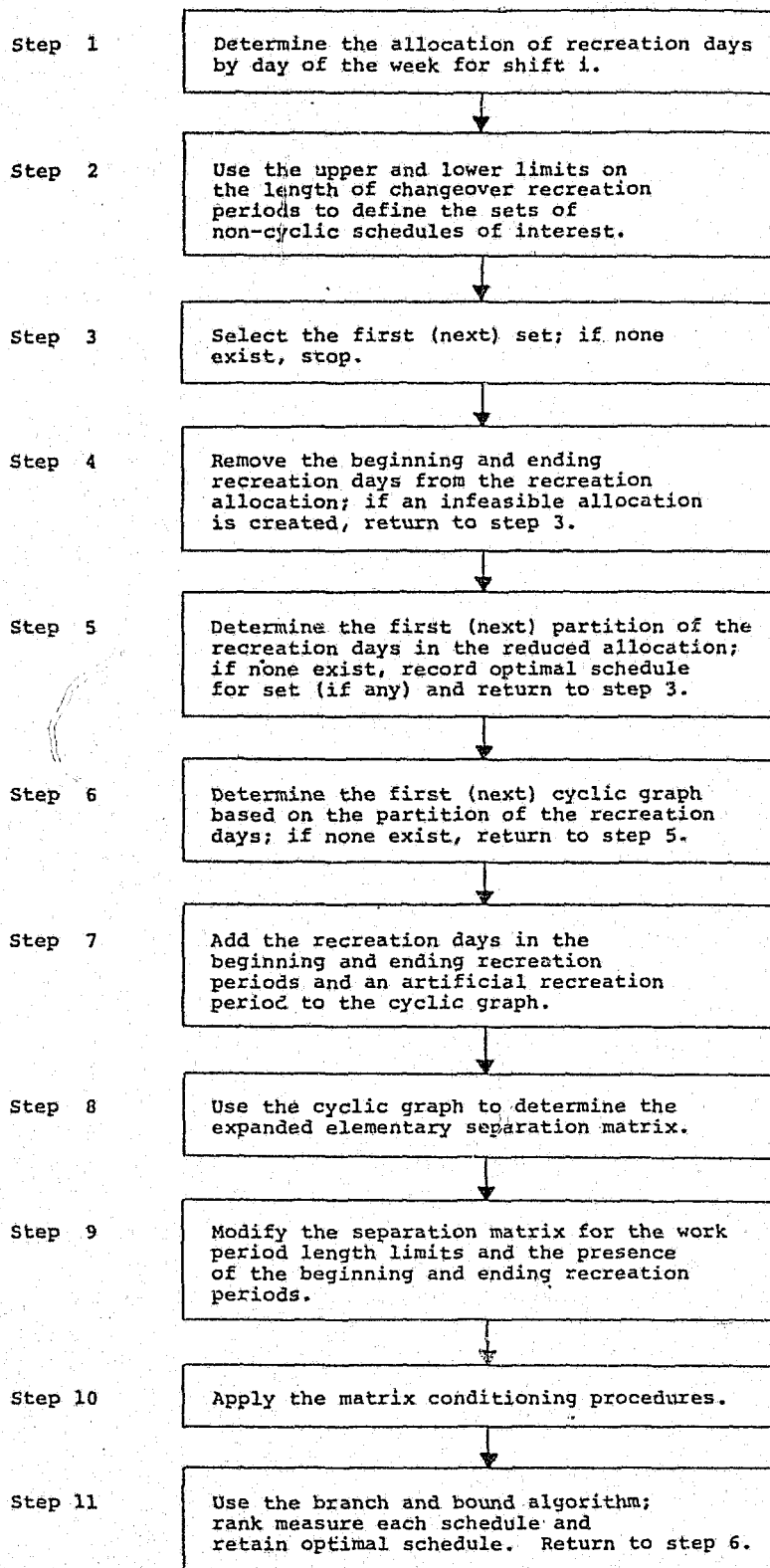


Figure 7.17

Eleven-Step Procedure for the Algorithmic Construction of Dominating Non-Cyclic Schedules

The upper and lower limits on the length of each changeover recreation period are  $U_{CR} = 4$  and  $L_{CR} = 2$  respectively; as a result, there are  $(U_{CR}+1)^2 = (4+1)^2 = 25$  sets of schedules to be examined. The 25 sets are  $\{(0,0), (0,1), (0,2), (0,3), (0,4), (1,0), (1,1), \dots, (4,3), (4,4)\}$ . A dominating non-cyclic schedule must be found for each set. Obtaining the daily allocation of recreation days, and defining and ordering the sets of schedules to be examined completes steps 1 and 2.

From this point on in the algorithm (steps 3 through 11), each set of schedules is examined individually. If any schedules exist for the set, the optimal schedule is found and retained. To illustrate this process, consider finding the dominating schedule for the (0,2) set; every non-cyclic schedule in this set must have an ending recreation period that is exactly two days long.

Since the exact lengths of the beginning and ending recreation periods are known, they can be used to reduce the computational effort of both the partitioning and cyclic graph algorithms. For example: since the length and position of the ending period is known (the period is two days long and covers Saturday and Sunday), the period can be temporarily removed from the daily allocation of recreation days to obtain a reduced allocation based on only six recreation days. The reduced allocation for the (0,2) set is obtained by subtracting one recreation day from each day of the week covered by the period, in this case Saturday and Sunday; the new allocation is:

Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Total
2	2	0	1	1	0	0	6

It may happen that the specified beginning and ending recreation periods are not compatible with the daily allocation of recreation days. For example, each schedule from the set (3,4) has a beginning period three days long and an ending period four days long. Removing both of these periods from the original daily allocation produces the following result:

Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Total
1	1	-1	0	0	0	0	2

The -1 value for Wednesday occurs because the beginning period requires a recreation day on Wednesday that does not exist in the initial allocation; as a result, no non-cyclic schedules can be designed for the (3,4) case. In fact, since there are no recreation days allocated to Wednesday, no schedules can be designed with beginning periods that are more than two days long (i.e.,  $b > 2$ ); hence, no schedules can exist for the sets  $\{(3,0), (3,1), \dots, (4,4)\}$ .

The reduced allocation of recreation days for the (0,2) case is feasible (i.e., there are no negative values) and Step 4 is completed. The reduced daily allocation is now used in the algorithms described in chapters 4 and 5 to find partitions

and cyclic graphs (steps 5 and 6). The removal of the beginning and ending periods serves two purposes: (1) the reduced allocation array produces a smaller problem requiring less computational effort; and (2) the beginning and ending period can be added to each cyclic graph produced thereby insuring that only schedules from the (0,2) set will be produced.

The six recreation days in the reduced allocation for the (0,2) case produce only three partitions which satisfy the constraints on the lengths of interior recreation periods (i.e.,  $L_R = 2$  and  $U_R = 4$ ). The partitions are:

Partition	Number of		
	2-day periods	3-day periods	4-day periods
1	3	0	0
2	0	2	0
3	1	0	1

Each partition, in turn, is used to produce cyclic graphs on the star diagram corresponding to the reduced allocation; the reduced star diagram for the (0,2) case is shown in figure 7.18. Because of the small number of nodes in the star diagram, it is possible to determine by inspection that partitions 2 and 3 produce no graphs. (The star diagram in figure 7.18 can only be used for one or two-day recreation periods.) The remaining partition which consists of three

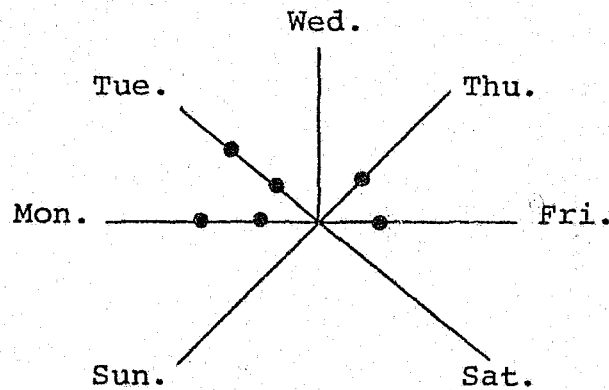


Figure 7.18

Reduced Star Diagram

two-day periods produces only one cyclic graph (see figure 7.19). Formulation of the graph completes step 6.

To construct the complete cyclic graph used to produce the expanded elementary separation matrix, the beginning and ending recreation periods, and an artificial period must be added to the reduced graph. For the (0,2) case, a two-day ending period covering Saturday and Sunday, and an artificial

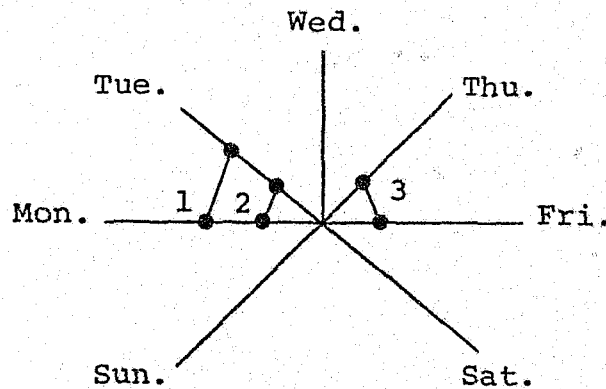


Figure 7.19

Reduced Cyclic Graph

recreation period extending from Monday through Sunday are added to the cyclic graph in figure 7.19. The resulting cyclic graph, used in an example earlier in this chapter, is shown in figure 7.7. Construction of the complete cyclic graph concludes step 7.

The expanded, elementary separation matrix for this cyclic graph is shown in figure 7.9 (step 8), and the

modified matrix for the (0,2) case (a type III schedule) is shown in figure 7.13 (step 9). Since recreation period 4 is used as the ending period, matrix entry (4,a) is dedicated by setting the entry (4,a) equal to zero and voiding all of the other entries in row 4 and column a. No dedicated entry exists in row a because the beginning period has length zero. The only matrix conditioning rule (step 10) that can be applied is the voiding of the (a,4) entry; since (4,a) is dedicated, it must appear in every feasible sequence and (a,4) will appear in none.

Only one schedule (see figure 7.16) can be enumerated from the modified matrix in figure 7.13 (step 11). This schedule represents the dominating schedule for the (0,2) set and joins the collection of other dominating schedules for shift i that will be used to design the optimal multi-shift schedule.

## 7.5 THE ENUMERATION OF OPTIMAL MULTISHIFT SCHEDULES

### 7.5.1 Introduction

This section describes an implicit enumeration scheme to find optimal multishift schedules based on the set of dominating non-cyclic schedules. The enumeration strategy is quite simple: a candidate non-cyclic schedule is selected from the set of dominating schedules for one shift, the changeover recreation period defined by this candidate schedule and the non-cyclic schedule selected for the

preceding shift is examined, and if the length of the change-over period is acceptable, the candidate schedule is retained and the algorithm moves to the selection of a non-cyclic schedule for the next shift tour in the multishift schedule. This step-wise process is continued until a schedule has been selected for each shift tour. Each complete multishift schedule is then given a preference rating which is used to determine the optimal schedule.

This section describes the enumeration procedures used to efficiently examine all of the combinations of the dominating non-cyclic schedules from each shift. In addition to the acceptability requirement associated with each changeover recreation period, several acceleration techniques are described which reduce the computational effort required to find optimal and near-optimal schedules.

#### 7.5.2 The Enumeration Process

The implicit enumeration scheme to determine optimal multishift schedules is schematically represented in figure 7.20; the diagram is based on a three-shift schedule. Each node in the tree diagram represents a unique sequence of one-shift, non-cyclic schedules included in the sequence (e.g., each level 2 node represents a unique sequence of two non-cyclic schedules). Each level in the tree corresponds to a specific shift tour in the multishift schedule, and the sequence of levels corresponds to the sequence of shifts

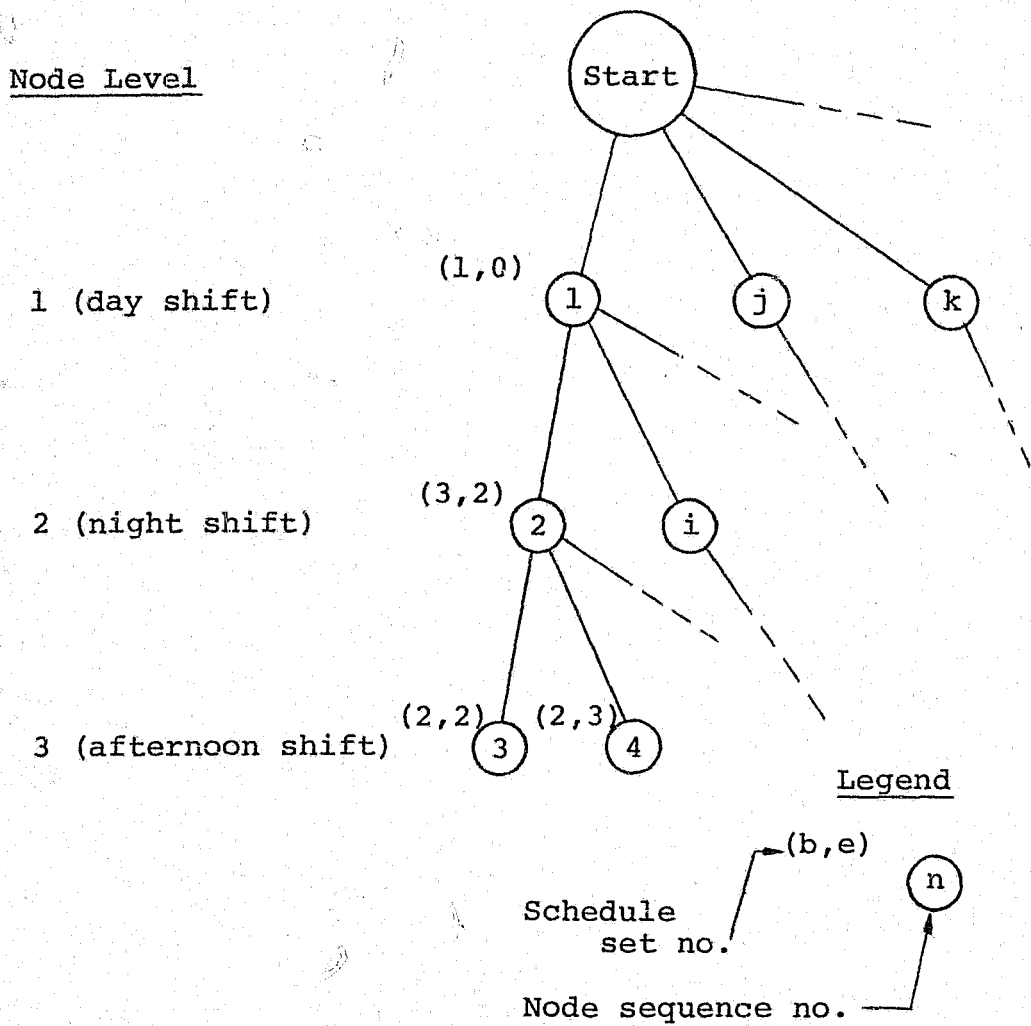


Figure 7.20

Schematic Diagram of the Tree Search Procedure for  
Constructing Multishift Schedules

required in the multishift schedule (e.g., the sequence of levels in figure 7.20 correspond to multishift schedules with a shift sequence of day, night, afternoon. Which shift is used in level 1 is arbitrary, but once determined, the shift assignment for each subsequent level must correspond to the required shift sequence in the multishift schedule.

The number within each node identifies the order in which the nodes are created; the left superscript for each node indicates the set of the non-cyclic schedule added to the sequence at that level (e.g., the dominating schedule for set (3,2) was added to the sequence at node 2). The entire sequence represented by each node can be determined by noting the schedule set numbers on the path from START to each node (e.g., the number 4 node represents the sequence  $\{(1,0), (3,2), (2,3)\}$ ; that is, the day shift schedule is from set (1,0), the night shift schedule is from set (3,2), and the afternoon shift schedule is from set (2,3).

As each candidate shift schedule is selected, the length of the changeover recreation period between the candidate schedule and the schedule selected for the preceding shift must be examined to determine whether it is acceptable; that is, whether it satisfies the condition:

$$L_{CR} \leq e_i + b_{i+1} \leq U_{CR} \quad i = 1, 2, \dots, N$$
$$b_{N+1} \equiv b_1$$

where  $N$  equals the number of shift tours. Each level  $N$  candidate schedule must satisfy the changeover period requirement for two changeover points:  $i = N-1$  and  $i = N$ ; that is, the candidate schedule at level  $N$  must be compatible (in terms of the changeover recreation period) with the shift schedule at level  $N-1$  and with the shift schedule at level 1 since shift  $N$  "precedes" shift 1 in a multishift schedule.

### 7.5.3 A Sample Problem

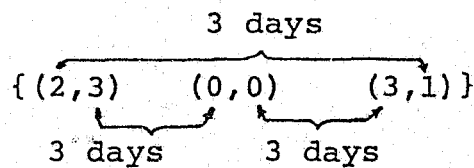
To illustrate the enumeration algorithm, consider the construction of an optimal multishift schedule using the following sets of dominating schedules: for shift 1, schedules exist for sets (2,3) and (4,1); for shift 2, schedules exist for sets (1,1), (1,4), and (3,1); and for shift 3, schedules exist for sets (0,0), (1,3) and (3,4). (This information is summarized in table 7.2.)

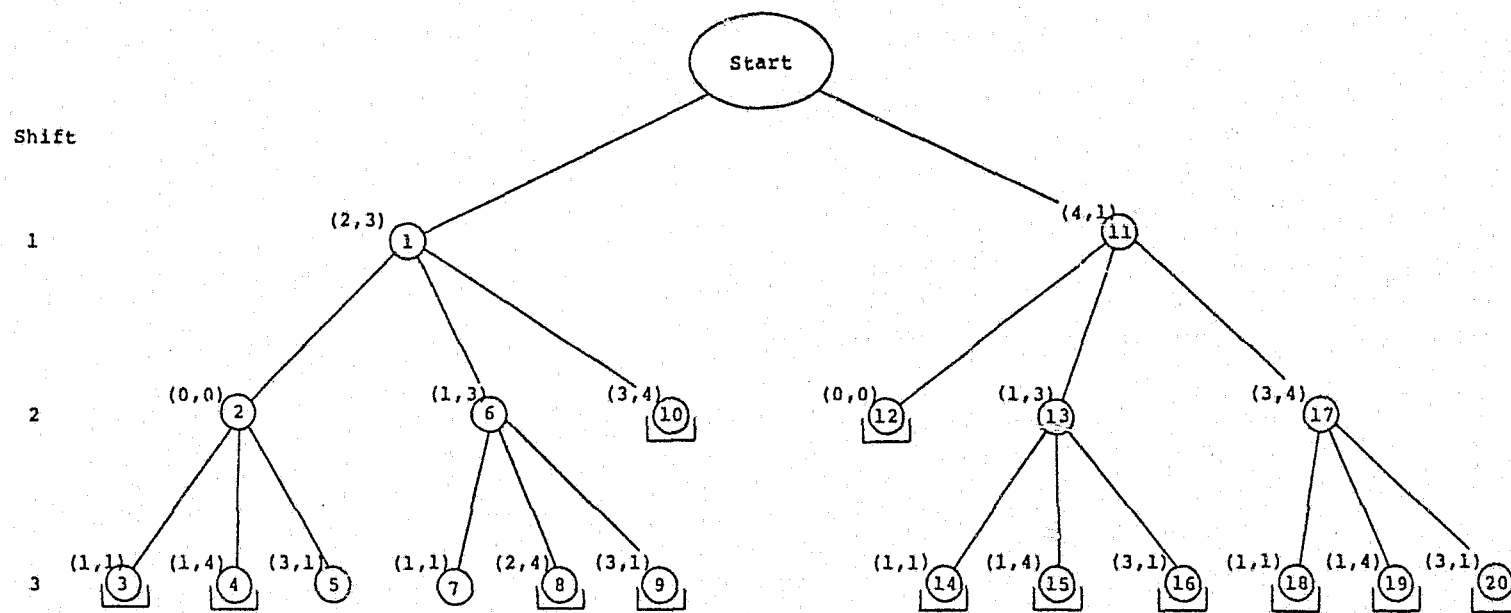
The tree diagram for enumerating multishift schedules based on these eight shift schedules is shown in figure 7.21. The upper and lower limits on the lengths of changeover recreation periods are four and two days respectively, (i.e.,  $U_{CR} = 4$  and  $L_{CR} = 2$ ). Two acceptable multishift schedules are found: one at node 5 corresponding to the sequence  $\{(2,3), (0,0), (3,1)\}$ , and a second at node 7 corresponding to the sequence  $\{(2,3), (1,3), (1,1)\}$ . Adding the lengths of the ending and beginning recreation periods for each adjacent pair of non-cyclic schedules in each sequence

Table 7.2  
Dominating Schedule Sets for the  
Three-Shift Sample Problem

Shift	Schedules Found for the Following Sets
1.	(2,3), (4,1)
2.	(0,0), (1,3), (3,4)
3.	(1,1), (1,4), (3,1)

verifies that each changeover point in the multishift schedule has a changeover recreation period of acceptable length. As an example, for the sequence associated with node 5, each changeover recreation period is three days long; i.e.,





\* Acceptable multishift schedule, a changeover recreation period of acceptable length exists at each shift changeover point.

Figure 7.21

Tree Diagram for the Three-Shift Search Example

The  $\square$  symbol in the tree diagram indicates nodes in which the candidate shift schedule failed to produce a changeover recreation period of acceptable length.

#### 7.5.4 Acceleration Techniques

Although the enumeration procedure described above is quite satisfactory for relatively small scheduling problems, as the number of shifts, and dominating schedules per shift increases, the computational effort required for the enumeration of all acceptable schedules also increases very rapidly. This section examines the growth characteristics of the enumeration process, and describes two acceleration procedures for improving the effectiveness of the enumeration algorithm. In the concluding portion of this section, a modified procedure is described for enumerating both optimal and near-optimal schedules.

##### 7.5.4.1 Growth Characteristics of the Enumeration Process

The number of acceptable multishift schedules and the size of the tree structure (i.e., the number of nodes) that exist for a given problem depend upon the number of shifts,  $N$ , and the limits imposed on the lengths of the changeover recreation periods. As derived earlier in this chapter, each shift can have a maximum of  $(U_{CR}+1)^2$  dominating schedules. Hence the first level in the tree structure can have as many as  $(U_{CR}+1)^2$  nodes.\* Although  $(U_{CR}+1)^2$

---

\* For convenience, it is assumed in the remainder of this section that there are exactly  $(U_{CR}+1)^2$  dominating schedules for each shift.

schedules are available for branching (i.e., as descendent nodes) from each node at level 1, only a limited number of the schedules assigned to each level 2 node will satisfy the changeover criterion,  $L_{CR} \leq e_1 + b_2 \leq U_{CR}$ . The number of active nodes (i.e., nodes that satisfy the changeover constraint) at level 2,  $N_2$ , equals

$$N_2 = \sum_{k=0}^{U_{CR}} (n_k) \cdot (m_k)$$

where  $n_k$  equals the number of level 1 schedules with  $e_1 = k$ , and  $m_k$  equals the number of level 2 schedules that are compatible with  $e_1 = k$  (i.e., level 2 schedules with  $b_2$  values that satisfy  $L_{CR} \leq k + b_2 \leq U_{CR}$ ). The number of level 1 schedules with  $e_1 = k$  depends on the number of distinct values for  $b_1$ ; since  $0 \leq b_1 \leq U_{CR}$  or  $b_1 = 0, 1, 2, \dots, U_{CR}$ , there are exactly  $U_{CR} + 1$  schedules with  $e_1 = k$ .

Hence,

$$N_2 = \sum_{k=0}^{U_{CR}} (U_{CR} + 1) m_k . \quad (7.4)$$

To determine  $m_k$ , the number of beginning period lengths in schedules at level 2 that are compatible with each value of  $k$ , it is convenient to use the following listing:

end period length, $e = k$	number of beginning period lengths that are compatible with $k$
$U_{CR}$	1
$U_{CR} - 1$	2
$\vdots$	$\vdots$
$L_{CR} + 1$	$U_{CR} - L_{CR}$
$L_{CR}$	$U_{CR} - L_{CR} + 1$
$L_{CR} - 1$	$U_{CR} - L_{CR} + 1$
$\vdots$	$\vdots$
1	$U_{CR} - L_{CR} + 1$
0	$U_{CR} - L_{CR} + 1$

For values of  $k$  from  $U_{CR}$  to  $L_{CR}+1$ ,  $m_k$  values equals  $U_{CR}+1-k$ ; for values of  $k$  less than  $L_{CR}+1$ ,  $m_k$  is independent of  $k$  and equals  $U_{CR}-L_{CR}+1$ . As an example for  $k = 0$ ,  $U_{CR} = 4$  and  $L_{CR} = 2$ , there are three beginning period lengths ( $U_{CR}-L_{CR}+1 = 4-2+1 = 3$ ): 2, 3, and 4. Since there are  $(U_{CR}+1)$  schedules at each level 2 node that begin with each value, there are  $M_0 = 3(U_{CR}+1) = 15$  schedules that are compatible with  $k = 0$ . Hence (7.4) becomes

$$N_2 = \sum_{k=0}^{U_{CR}} (U_{CR}+1)^2 \cdot (U_{CR}-L_{CR}+1) + \sum_{k=L_{CR}+1}^{U_{CR}} (U_{CR}+1)^2 \cdot (U_{CR}+1-k)$$

which becomes:

$$N_2 = (U_{CR}+1)^2 \cdot \left\{ \frac{(U_{CR}+2)(U_{CR}+1)}{2} - \frac{L_{CR}(L_{CR}+1)}{2} \right\} \quad (7.5)$$

Now let

$$F = \left\{ \frac{(U_{CR}+2)(U_{CR}+1)}{2} - \frac{L_{CR}(L_{CR}+1)}{2} \right\}$$

then (7.5) becomes

$$N_2 = (U_{CR}+1)^2 F$$

which indicates that  $F$  nodes are generated at level 2 for each level 1 node. Hence, in general, the total number of active nodes for level  $i$  is given by

$$N_i = (U_{CR}+1)^2 F^{i-1}$$

and the total number of nodes  $N_T$  for  $N$  shifts equals

$$N_T = \sum_{i=1}^N N_i = (U_{CR}+1)^2 \left( \frac{F^N - 1}{F - 1} \right) .$$

One measure of the screening capability of the changeover period length criterion is the fraction of nodes that are rejected (terminated) at each level in the tree structure. Define the screening factor,  $f$ , as

$$\begin{aligned}
 f &= \frac{\text{number of nodes rejected at level } i}{\text{total number of nodes possible at level } i} \\
 &= \frac{\text{total number of nodes possible at level } i - \text{total number of active nodes at level } i}{\text{total number of nodes possible at level } i} \\
 &= \frac{N_{i-1}(U_{CR}+1)^2 - N_{i-1}F}{N_{i-1}(U_{CR}+1)^2} \\
 f &= 1 - \frac{F}{(U_{CR}+1)^2} .
 \end{aligned}$$

For  $L_{CR} = 2$  and  $U_{CR} = 4$ ,  $F$  equals 12 and the screening factor  $f$  equals  $13/25$  or .52. Hence the changeover length requirement for these values of  $L_{CR}$  and  $U_{CR}$  only eliminates 52 percent of the nodes.

This screening factor can also be used to determine the total number of acceptable schedules  $N_S$  that exist for a given set of initial conditions. Acceptable multishift schedules must come from active level  $N$  nodes which are not eliminated when the changeover period length criterion is applied to the changeover period between the level 1 and level  $N$  schedules; therefore

$$\begin{aligned}
 N_S &= N_N \cdot (1-f) = \left( (U_{CR}+1)^2 F^{N-1} \right) \cdot \left( \frac{F}{(U_{CR}+1)^2} \right) \\
 N_S &= F^N
 \end{aligned}$$

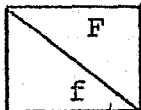
Values of  $F$  and screening factors for a variety of changeover period limits are shown in figure 7.22. The values in the table indicate that as the range of the

Lower Limit in Days on the Length of  
the Changeover Recreation Period,  $L_{CR}$

Upper Limit  
in Days  
on the  
Length of the  
Changeover  
Recreation  
Period,  
 $U_{CR}$

	1	2	3	4	5	6
1	2 .5					
2	5 .44	3 .67				
3	9 .438	7 .563	4 .75			
4	14 .44	12 .52	9 .64	5 .80		
5	20 .444	18 .500	15 .583	11 .694	6 .83	
6	27 .449	25 .490	22 .551	18 .633	13 .735	7 .857

$$F^* = \frac{(U_{CR}+2)(U_{CR}+1)}{2} - \frac{(L_{CR}+1)L_{CR}}{2}$$



$$f^* = 1 - \frac{F}{(U_{CR}+1)^2}$$

\*Assumes that the maximum number of dominating  
schedules,  $(U_{CR}+1)^2$ , are found for each shift.

Figure 7.22

F Values and Screening Factors (f) for the Multishift  
Schedule Enumeration Procedure as a Function of the Upper  
and Lower Limits on the Length of Changeover Recreation  
Periods

limits (i.e.,  $U_{CR}-L_{CR}$ ) becomes larger, F values increase and screening factors fall to approximately 0.5. Even for the smallest range possible (i.e.,  $U_{CR}=L_{CR}$ ), the screening factor is less than 0.86 for  $N \leq 6$ . The total number of nodes and schedules that exist for the three sets of limits are shown in table 7.3. These results indicate how quickly the total number of acceptable schedules and nodes increase as the range of the changeover period limits increases. They also indicate that unless additional constraints are used to screen out acceptable schedules, application of the enumeration procedure is quite limited.

#### 7.5.4.2 Selection of the Level 1 Shift

One method for accelerating the enumeration algorithm is the selection of a shift tour for level 1 which minimizes the total number of active nodes produced in the tree structure. This decision can be based on the number of dominating schedules  $n_i$  that exist for each shift. The number of active nodes  $N_i$  at each level  $i$  in the tree structure can be estimated by

$$N_i = n_1, n_2, \dots, n_i (P_a)^i = \prod_{i=1}^i n_i (P_a)^i$$

where  $P_a$  is the probability that each sum  $e_{i-1}+b_i$  will fall in the range  $(L_{CR}, U_{CR})$ . This acceptance probability can be estimated by  $\hat{P}_a = (1-f)$  where  $f$  is the screening factor



Table 7.3

Number of Schedules and Nodes Enumerated for Different  
Changeover Period Limits and Numbers of Shifts\*

Number of Shifts N	Changeover Recreation Period Limits					
	$U_{CR}=2, L_{CR}=2$		$U_{CR}=4, L_{CR}=2$		$U_{CR}=5, L_{CR}=1$	
	Number of Schedules F	Number of Nodes $N_T$	Number of Schedules F	Number of Nodes $N_T$	Number of Schedules F	Number of Nodes $N_T$
2	9	36	144	325	400	756
3	27	117	1,728	3,925	8,000	15,156
4	81	360	20,736	47,125	160,000	303,156
5	243	1,089	248,832	565,525	3,200,000	6,063,156
6	729	3,276	2,985,984	6,786,325	64,000,000	121,263,156

\*Calculations are based on a maximum of  $(U_{CR} + 1)^2$  dominating schedules for each shift.

derived above. The validity of this estimate for  $P_a$  is based on the assumption that the dominating schedules found for each shift represent a random selection of the  $(U_{CR}+1)^2$  schedules that could exist. Using  $(1-f)$  in place of  $P_a$ , the total number of nodes becomes

$$N_T = \sum_{i=1}^N N_i = \sum_{i=1}^N \left( \prod_{k=1}^N n_k (1-f)^i \right) \quad (7.6)$$

Equation (7.6) is more easily computed if the terms are rearranged and written in the form:

$$N_T = n_1 p (1 + n_2 p (1 + \dots (1 + n_N p) \dots)) \quad (7.7)$$

where  $p = (1-k)$ . Result (7.7) can be used to estimate the total number of nodes that will be generated in the tree structure using each of the  $N$  shifts in level 1 (or equivalently, using each cyclic permutation of the  $N$ -shift sequence). The estimating equation is used prior to the enumeration algorithm to determine which level 1 shift minimizes  $N_T$ .

As an example, consider the enumeration of all multi-shift schedules with six shift tours based on the following sets of dominating schedules:

Shift Label	Number of Schedules, $n_i$ =		
	Set 1	Set 2	Set 3
a	15	10	25
b	15	18	1
c	15	14	25
d	15	20	1
e	15	12	25
f	15	16	13

Each set contains 90 schedules.

The  $N_T$  values for the six cyclic permutations of each set of schedules are presented in table 7.4. The  $N_T$  values for each set indicate that selecting an appropriate level 1 shift becomes more important as the variability in the number of dominating schedules per shift increases. As expected, no differences in  $N_T$  values occurs when all of the  $n_i$  values are equal (set 1). In set 3, however, the variability in the number of schedules produces  $N_T$  values that vary by a factor of 3 (e.g., the  $N_T$  value for sequence 6 is less than one-third of the  $N_T$  value for sequence 3).

#### 7.5.4.3 Ordering the Dominating Schedules for Each Shift

The following definitions and notation are needed to describe the second acceleration procedure. Let  $W_{ij}$  equal the number of weekend recreation periods for the  $j^{\text{th}}$  schedule for shift  $i$ ; and let  $W_i$  equal the maximum  $W_{ij}$  value for shift  $i$ ; i.e.,

Table 7.4

Estimated Number of Active Nodes for Cyclic Permutations of the  
Shift Sequence for Three Sample Problems

Permu- tation Number	Schedule Set 1		Schedule Set 2		Schedule Set 3	
	Shift Sequence	Est. No. of Active Nodes $N_T^*$	Shift Sequence	Est. No. of Active Nodes $N_T^*$	Shift Sequence	Est. No. of Active Nodes $N_T^*$
1.	(15,15,15,15,15,15)	205,358	(10,18,14,20,12,16)	173,615	(25,1,25,1,25,13)	3,798
2.	↑ ↓	↑ ↓	(18,14,20,12,16,10)	179,328	(1,25,1,25,13,25)	4,064
3.			(14,20,12,16,10,18)	168,774	(25,1,25,13,25,1)	11,139
4.			(20,12,16,10,18,14)	175,750	(1,25,13,25,1,25)	3,983
5.			(12,16,10,18,14,20)	171,857	(25,13,25,1,25,1)	10,126
6.	(15,15,15,15,15,15)	205,358	(16,10,18,14,20,12)	185,922	(13,25,1,25,1,25)	3,477

\*Equation (7.7).

$$W_i = \max_j W_{ij}, \quad i = 1, 2, \dots, N$$

Using the  $W_i$  value for each shift, an upper limit on the total number of weekend recreation periods  $W$  in each multi-shift schedule is given by

$$W = \sum_{i=1}^N W_i .$$

The  $W_{ij}$  and  $W_i$  values can be used to define a  $d_{ij}$  value for each dominating schedule. The  $d_{ij}$  value for the  $j^{\text{th}}$  schedule in shift  $i$  is given by:

$$d_{ij} = W_i - W_{ij} .$$

Each  $d_{ij}$  value indicates the difference between the maximum number of weekend periods possible from shift  $i$  and the number of weekend periods in schedule  $j$ . Let  $D$  equal the sum of the  $d_{ij}$ 's for each multishift schedule; i.e.,

$$D = \sum_{i=1}^N \sum_{\{j\}} d_{ij}$$

where  $\{j\}$  represents the set of  $j$  values corresponding to the  $N$  shift schedules used in a multishift schedule. The  $D$  value for each schedule indicates the difference between the maximum number of weekend periods possible in the entire schedule and the actual number included over all shift tours; i.e.,

$$D = W - \sum_{i=1}^N \sum_{\{J\}} W_{ij}$$

As a result, a zero value for D indicates a multishift schedule that contains the maximum number of weekend periods.

The  $d_{ij}$  value assigned to each dominating schedule can be used to provide an ordinal ranking of the candidate schedules within each shift to accelerate the enumeration procedure. Let  $J_{ik}$  equal the number of schedules for shift  $i$  which have  $d_{ij} = k$  ( $J_{i0} \geq 1$  since at least one schedule must have  $d_{ij} = 0$ ). The sum of the  $J_{ik}$ 's equals  $n_i$ : the total number of schedules from shift  $i$ ; i.e.,

$$\sum_{k=0}^{W_i} J_{ik} = n_i$$

The ordinal ranking for each schedule in shift  $i$  is obtained in the following manner. Schedules with  $d_{ij} = 0$  values are ranked from 1 to  $J_{i0}$  (the order within the group of  $d_{ij} = 0$  schedules is arbitrary); next, schedules with  $d_{ij} = 1$  values are ranked from  $J_{i0}+1$  to  $J_{i0}+J_{i1}$ . This process continues until all of the schedules within shift  $i$  are labelled. The process is repeated for each shift.

With the schedules labelled in this manner, two modifications can be made in the enumeration procedure. The first involves the following two steps:

Step 1. Use the enumeration procedure described above to find an acceptable multishift schedule; let  $D$  equal the sum of the  $d_{ij}$ 's for the  $N$  shift schedules, and let  $C$  equal the  $d_{ij}$  maximum number of consecutive working weekends in the schedule.

Step 2. For each node examined subsequent to finding the first schedule, use the following test based on the sum  $D_i$  of the  $d_{ij}$ 's for the shift schedules in the partial sequence at level  $i$ , and the maximum number  $C_i$  of consecutive working weekends in the first  $i$  shift schedules to determine the acceptability of the candidate schedule; this test is in addition to the determination of an acceptable changeover recreation period:

- (a) if  $D_i > D$ , or
- (b) if  $D_i = D$  and  $C_i > C$

reject the candidate schedule.

The additional tests described in step 2 are designed to identify (as early as possible in the enumeration process) groups of schedules which will be less preferable than the current optimal schedule. The  $D_i$  value indicates the maximum number of weekend recreation periods that any schedule produced from node  $i$  can have (the maximum number equals  $W - D_i$ ). Since each  $d_{ij}$  value is non-negative, the sequence of  $D_i$  values for  $i = 1, 2, \dots, N$  is monotonically nondecreasing; hence if  $D_i > D$ , no schedule will be produced from node  $i$  with as many weekend periods as the current optimal solution.

If  $D_i = D$  at a level  $i$  node, there may be some acceptable multishift schedules from node  $i$  which possess as many weekend periods as the current optimal schedule. In that event, the second most important preference criterion, the maximum number of consecutive working weekends, is used to identify nodes which must yield less preferable schedules. Like  $D_i$ , the  $C_i$  values for  $i = 1, 2, \dots, N$  form a monotonically nondecreasing sequence. Hence if  $D_i = D$  and  $C_i > C$ , the

preference measure of every schedule completed from node  $i$  will be less than the preference measure of the best schedule already found.

The second change in the enumeration procedure, based on the ordering of the dominating schedules for each shift, comes into play once the first acceptable multishift schedule is found by the enumeration algorithm. From that point on, the  $D_{i-1}$  value for each node can be used to screen out candidate schedules for shift  $i$ . The criterion is based on the observation that in order for a level  $i$  node to be active, its  $D_i$  value must satisfy the condition  $D_i \leq D$ . This inequality, in turn, yields the following result about the  $d_{ij}$  values of the dominating schedules that can be added at level  $i$ ; i.e., if

$$D_i \leq D$$

then

$$D_{i-1} + d_{ij} \leq D$$

$$d_{ij} \leq D - D_{i-1}$$

and

$$\max_j d_{ij} = D - D_{i-1} \quad (7.7)$$

Result (7.7) states that in order for the node at level  $i$  to be active, the  $d_{ij}$  value of the candidate schedule for shift  $i$  must not exceed  $D - D_{i-1}$ . As a result, only the first

$J_{i0} + J_i + \dots + J_{im}$  dominating schedules have to be examined from shift  $i$ . (There must be at least one schedule since  $\min D - D_{i-1} = 0$  and  $J_{i0} \geq 1$ .) Hence as the  $D$  value decreases as new multishift schedules are found with more weekend recreation periods, the number of dominating schedules that have to be examined from each shift decreases.

#### 7.5.4.4 Enumeration of Optimal and Near-Optimal Multishift Schedules

One difficulty with the modifications described above is the amount of computational effort that may be required to find the first acceptable schedule. In addition, determination of only the optimal multishift schedule may not be a practical design approach. Although one multishift schedule can be identified as optimal in terms of the specific schedule measures used in the design algorithms, computational experience with the enumeration procedure indicates that the differences that distinguish the optimal schedule from the second, tenth, or even fifteenth ranked schedule are frequently very small, and that such small differences may not be of major importance to the schedule designer. He may, in fact, place more value on the opportunity to select from among a group of optimal and near-optimal alternatives, a schedule possessing schedule features not explicitly controlled in the design algorithm.

To limit the effort expended in finding the first acceptable schedule, and to produce both optimal and near-optimal schedules, a modified enumeration strategy can be used. The procedure uses the  $W$  value derived from the examination of the set of dominating schedules for each shift. The altered strategy assumes that some acceptable multishift schedules exist with exactly  $W$  weekend recreation periods, and limits the enumeration process to finding only those schedules. (The optimal multishift schedule must, by definition, be included in the collection of schedules with  $W$  weekend periods.) To limit the search process,  $D$  is set equal to zero at the beginning of the enumeration. (This has the effect of indicating that a schedule with  $W$  weekend periods has already been found.) As a result, only shift schedules with  $d_{ij} = 0$  are examined, and every acceptable multishift schedule found has  $D_N = 0$  and  $\sum_{i=1}^N \sum_{j \in J_i} W_{ij} = W$ . Since the search is limited to only the first  $J_{i0}$  schedules from each shift, the enumeration is accelerated, and the collection of feasible schedules found includes both optimal and near-optimal schedules. If no schedules (or very few) are found with  $D = 0$ , the  $D$  value can be incremented to  $D = 1$  and the enumeration process can be initiated again. This procedure can be continued until either the optimal plus a sufficient number of near-optimal schedules are found or the  $D$  value exceeds a user specified limit.

#### 7.5.5 Computer Code

To test the enumeration algorithm and acceleration techniques described above for designing optimal and preferable multishift schedules, a computer code, entitled MERGE, was written in FORTRAN IV, and successfully implemented on an IBM 370-55 system. The program has been utilized to construct schedules for several police agencies (see section 8.4). Multishift schedules have been constructed with as many as six shift tours extending over 19 weeks. Execution times have consistently remained below one minute of CPU time. Printouts of the MERGE program are presented in section 8.2.

## 8. CONCLUSIONS

### 8.1 INTRODUCTION

This concluding chapter reviews the major results of this thesis, describes the application of these results to the design of manpower schedules for two units within the St. Louis Metropolitan Police Department, and finally, outlines several areas for future work.

### 8.2 MAJOR RESULTS

#### 8.2.1 Measures of Manpower Schedules

The primary objective of this thesis was the development of a system for the design of optimal and preferable multishift PR schedules. The development and integration of a design capacity into the schedule enumeration algorithms consisted of the following steps: (1) identification of relevant and specific manpower schedule attributes and properties, and (2) specification of one or more quantitative measures for each attribute and property. In the procedures described in this thesis, these measures are applied to each candidate schedule to determine the relative preferability of alternate schedules, and ultimately to determine an optimal or most preferred schedule for a given set of initial conditions.

The schedule measures used in this work were applied in two ways. Some were used to screen out infeasible and unacceptable schedules; these measures represented minimum

standards to be satisfied by every acceptable schedule.

The minimum requirements imposed on all schedules in this thesis included:

- (1) preserving a specified manpower allocation by shift and day of the week;
- (2) satisfying upper and lower limits on all work period lengths;
- (3) satisfying upper and lower limits on all interior recreation period lengths; and
- (4) including a changeover recreation period of acceptable length at every shift changeover point (only applicable to multishift schedules).

Other schedule measures were used as preference measures to assess the merit of alternative acceptable schedules; these measures, in order of their importance (as used in this thesis) were:

- (1) the number of weekend recreation periods;
- (2) the maximum number of consecutive working weekends;
- (3) the maximum work period length;
- (4) the number of maximum length work periods;
- (5) the recreation period measure;\*
- (6) the work period range (i.e., the difference between the maximum and minimum length work periods);

---

\* Based on the number and kinds of recreation periods in each schedule. See table 2.9 for the preference rankings used for individual recreation periods in this thesis.

- (7) the maximum number of days in consecutive work periods;
- (8) the number of consecutive work periods with maximum length; and
- (9) the standard deviation of the ratios of work period to recreation period lengths.

Optimal PR schedules were derived from the set of acceptable schedules using these nine preference measures in a sequential lexicographic decision model.

#### 8.2.2 Construction of Optimal One-Shift PR Schedules

Three algorithms were developed to utilize the sequential construction process for one-shift PR schedules first described by Heller (16). The three algorithms are:

- (1) partitioning algorithm - to partition the recreation days allocated to each shift into recreation periods of acceptable lengths;
- (2) cyclic graph algorithm - to enumerate for each set of recreation periods produced by the partitioning algorithm, all distinct arrangements of these periods over the days of the week which preserve the required daily manpower allocation; and
- (3) separation matrix algorithm - to generate from each separation matrix (one matrix is selected for each cyclic graph on the basis of upper and lower limits on work period lengths), all acceptable manpower schedules (i.e., all sequences of alternating work and recreation periods which have the correct number of work and recreation days for the entire shift).

Each algorithm was developed and tested separately. The three were then combined into one computer code entitled EXEC for the design of both cyclic and non-cyclic one-shift PR schedules.

### 8.2.2.1 One-Shift Cyclic Schedules

The EXEC code has been used successfully to determine optimal cyclic schedules for a variety of daily manpower allocations and problem sizes. Figures 8.1 through 8.3 illustrate the printout from the EXEC code for a nine-week cyclic schedule designed to match the daily manpower allocation shown in table 8.1. Figure 8.1 indicates the initial allocation data by day of the week, and the limits on both work and recreation period lengths. The optimal schedule based on the initial data is shown in figure 8.2 (a work day is indicated by an asterisk and a recreation day by an R). Quantitative measures for the schedule are

Table 8.1

Daily Manpower Allocation for a Nine-Week Schedule

Number of Officers								Total
	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	
On Duty	6	6	6	7	7	7	6	45
On Recreation	3	3	3	2	2	2	3	18
Total	9	9	9	9	9	9	9	63

INITIAL DATA

TOTAL NUMBER OF WORK DAYS = 45

TOTAL NUMBER OF RECREATION DAYS = 18

DAILY DISTRIBUTION OF THE RECREATION AND WORK PERIODS

	MON	TUE	WED	THU	FRI	SAT	SUN	TOTAL
DUTY	6	6	6	7	7	7	6	45
RECREATION	3	3	3	2	2	2	3	18

TOTAL NUMBER OF WEEKS = 9

RECREATION PERIODS (WATCH INTERIOR)

MINIMUM LENGTH = 3  
MAXIMUM LENGTH = 4

WORK PERIODS

MINIMUM LENGTH = 4  
MAXIMUM LENGTH = 8

Figure 8.1

Initial Data for a Nine-Week Cyclic Schedule,  
EXEC Code Printout

\*\*\* BEST WATCH SCHEDULE \*\*\*

WORK DAYS		REC DAYS				
8	3					
8	3					
8	3					
5	3					
8	3					
0	1					
BREAK		9 WEEKS				
MON	TUE	WED	THU	FRI	SAT	SUN
1	*	*	*	*	*	*
2	*	R	R	*	*	*
3	*	*	*	*	R	R
4	R	*	*	*	*	*
5	*	*	R	R	*	*
6	*	*	*	*	*	R
7	R	R	*	*	*	*
8	R	R	*	*	*	*
9	*	*	*	R	R	R
BREAK						

Figure 8.2

Nine-Week Cyclic Schedule,  
EXEC Code Printout

shown in figure 8.3. Computer CPU time to find this schedule was approximately 40 seconds on an IBM 360/65 system.

Computational experience to date indicates that, under most circumstances, optimal one-shift schedules up to nine or ten weeks long, can be found with the EXEC code with CPU times of five minutes or less. In general, CPU times increase as:

- (1) the total number of recreation days increases;
- (2) the range of upper and lower limits on recreation period lengths increases;
- (3) the lower limit on recreation period lengths decreases;
- (4) the daily allocation of recreation days becomes more uniform across the days of the week; and
- (5) the range of upper and lower limits on work period lengths increases.

It should be noted that nine or ten-bracket PR schedules are adequate for many scheduling applications; for example, few multishift schedules are designed with more than an eight-week (or two-month) tour on each shift, and very good matches between allocated manpower and reported workload for each day of the week can be achieved for all but the most unusual workload distributions.

To illustrate the latter point, let  $f$  represent the average number of duty tours expected from each officer per week, and let  $N$  equal the total number of men available. The product  $Nf$  equals the total number of duty tours

MEASURES

WEEKEND RECREATION PERIODS		
NUMBER OF	=	2
AVERAGE NUMBER OF CONSECUTIVE		
WORKING WEEKENDS	=	3.50
MAXIMUM NUMBER OF CONSECUTIVE		
WORKING WEEKENDS	=	5

WORK PERIODS		
MAXIMUM LENGTH	=	8
MINIMUM LENGTH	=	5
AVERAGE LENGTH	=	7.50
RANGE	=	3
STANDARD DEVIATION	=	1.12
NUMBER OF MAXIMUM LENGTH	=	5

WORK PERIOD PAIR SUMS		
MAXIMUM LENGTH	=	16
MINIMUM LENGTH	=	13
AVERAGE LENGTH	=	14.80
RANGE	=	3
STANDARD DEVIATION	=	1.47

WORK PERIOD/RECREATION PERIOD RATIO		
AVERAGE RATIO	=	2.50
STANDARD DEVIATION	=	0.37

RANK SCORE = 2 508 2000110000100 30216 + 0.37

Figure 8.3

Preference Measures for the Nine-Week Cyclic  
Schedule in Figure 8.2, EXEC Code Printout

available for distribution each week. The fraction of  $Nf$  duty tours that can be scheduled on any one day is given by  $k/Nf$ ,  $k = 0, 1, 2, \dots$ ; hence, since the maximum number of duty tours that can be allocated to any day is  $k = N$ , the maximum fraction of duty tours that can be scheduled on one day is  $1/f$ .

The increment  $\Delta$  between successive fractions of duty tours equals

$$\Delta = \frac{k+1}{Nf} - \frac{k}{Nf} = \frac{1}{Nf} \quad (8.1)$$

As a result, for daily workload requirements that do not exceed  $1/f$ , the maximum difference between the fraction of workload for a given day, and the fraction of manpower assigned to that day equals  $1/Nf$ .\* Hence, for a 10-bracket schedule with one man assigned to each bracket,  $\Delta = 1/10f$ , and assuming a value of  $f = 4.5$  days per week, the maximum difference between the fractions of allocated manpower and reported workload for any day becomes  $\Delta = 1/45$  or only 2.2 percent.

#### 8.2.2.2 One-Shift, Non-Cyclic Schedules

To design preferable multishift PR schedules, the EXEC code can also be used to determine sets of optimal or dominating non-cyclic schedules for each shift tour used in a multishift schedule. Each dominating schedule is the

---

\* If the number of duty tours is always rounded to the nearest integral value, the maximum difference becomes  $1/2Nf$ .

optimal one-shift, non-cyclic PR schedule for a unique set of schedules characterized by the lengths of their beginning and ending recreation periods. The concept of an artificial recreation period is used to establish a 1-1 correspondence between each set of non-cyclic schedules from which an optimal schedule must be found, and a corresponding set of cyclic schedules; the optimal schedule for the set of cyclic schedules is used to identify the dominating non-cyclic schedule of interest.

Figures 8.4 through 8.7 illustrate the EXEC printout for two sets of non-cyclic schedules designed to match the six-man daily allocation (identified as the day watch) in table 8.2. The printout of the daily allocation, the limits on the lengths of the beginning, ending and interior recreation

Table 8.2

Daily Manpower Allocation for the Day Watch

Number of Officers	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Total
On Duty	5	4	4	4	5	4	4	30
On Recreation	1	2	2	2	1	2	2	12
Total	6	6	6	6	6	6	6	42

INITIAL DATA      DAY    WATCH  
TOTAL NUMBER OF WORK DAYS      = 30  
TOTAL NUMBER OF RECREATION DAYS = 12

DAILY DISTRIBUTION OF THE RECREATION AND WORK PERIODS

	MON	TUE	WED	THU	FRI	SAT	SUN	TOTAL
DUTY	5	4	4	4	5	4	4	30
RECREATION	1	2	2	2	1	2	2	12

TOTAL NUMBER OF WEEKS = 6

RECREATION PERIODS (START OF WATCH)  
MINIMUM LENGTH = 0  
MAXIMUM LENGTH = 4

RECREATION PERIODS (END OF WATCH)  
MINIMUM LENGTH = 0  
MAXIMUM LENGTH = 4

RECREATION PERIODS (WATCH INTERIOR)  
MINIMUM LENGTH = 2  
MAXIMUM LENGTH = 4

WORK PERIODS  
MINIMUM LENGTH = 4  
MAXIMUM LENGTH = 8

Figure 8.4

Initial Data, Day Watch,  
EXEC Code Printout

periods, and the limits imposed on the lengths of the work periods are shown in figure 8.4. As indicated in the printout, the upper limit on the lengths of the beginning and ending recreation periods is four days; as a result, 25 dominating schedules are sought for this shift, one for each set from (0,0) to (4,4). Acceptable schedules may not exist for all 25 sets.

Figure 8.5 summarizes the search statistics to enumerate the dominating schedule for the (0,2) case (i.e., schedules with a zero length beginning period and a two-day ending period). Five partitions of the 12 recreation days are found with four to six periods each. A total of 23 cyclic

PARTITIONS FOR DAY WATCH, CASE ( 0, 2)						
TOTAL NO OF PERIODS	PERIOD LENGTHS					NO OF CYCLIC GRAPHS
	1	2	3	4	5	
6	0	6	0	0	0	1
5	0	4	0	1	0	4
5	0	3	2	0	0	8
4	0	2	0	2	0	3
4	0	1	2	1	0	7
TOTAL NUMBER OF PARTITIONS						= 5
TOTAL NUMBER OF CYCLIC GRAPHS						= 23

Figure 8.5

Number of Partitions and Cyclic Graphs  
Enumerated for the (0,2) Set of  
Schedules for the Day Watch,  
EXEC Code Printout

graphs are enumerated from the five partitions (the number of cyclic graphs found for each partition ranges from one to eight). The optimal non-cyclic schedule for the (0,2) set is shown in figure 8.6. As required the schedule begins with a work period (producing a zero length beginning recreation period) and ends with a two-day recreation period. Preference measure values for this schedule are also shown in figure 8.6.

Figure 8.7 indicates the search statistics and results of the enumeration for the (4,2) set. Despite finding three partitions and four cyclic graphs, no acceptable schedules were found. The results for each of the 25 sets examined for the allocation in table 8.2 are summarized in table 8.3; dominating schedules were found for 19 of the 25 sets. The results in table 8.3 suggest that, in general, as more recreation days are dedicated to beginning and ending recreation periods (i.e., as the sum of their lengths,  $b+e$  increases), the numbers of partitions and cyclic graphs that exist decreases. For example, over half of the 115 cyclic graphs found for the 25 sets were generated for only three cases: (0,0), (0,1), and (0,2), and schedules were found for all 10 sets with  $b+e \leq 3$ . In contrast, schedules were found for only two of the six sets with  $b+e \geq 6$ : (2,4) and (4,3).

\*\*\* BEST WATCH SCHEDULE \*\*\*

DAY WATCH

CASE ( 0, 2)

LEG	WORK DAYS	REC DAYS
5 - 1	8	3
1 - 3	7	4
3 - 2	7	3
2 - 4	8	2
4 - 5	0	-

BREAK 6 WEEKS

	MON	TUE	WED	THU	FRI	SAT	SUN
1	*	*	*	*	*	*	*
2	*	R	R	R	*	*	*
3	*	*	*	*	R	R	R
4	R	*	*	*	*	*	*
5	*	R	R	R	*	*	*
6	*	*	*	*	*	R	R

BREAK

MEASURES

WEEKEND RECREATION PERIODS	=	2
NUMBER OF		
AVERAGE NUMBER OF CONSECUTIVE	=	2.00
WORKING WEEKENDS		
MAXIMUM NUMBER OF CONSECUTIVE	=	2
WORKING WEEKENDS		
NUMBER OF CONSECUTIVE WORKING	=	2
WEEKENDS(BEGINNING OF WATCH)		
NUMBER OF CONSECUTIVE WORKING	=	0
WEEKENDS(END OF WATCH)		
WORK PERIODS	=	8
MAXIMUM LENGTH	=	7
MINIMUM LENGTH	=	7.50
AVERAGE LENGTH	=	1
RANGE	=	0.50
STANDARD DEVIATION	=	2
NUMBER OF MAXIMUM LENGTH		
WORK PERIOD PAIR SUMS	=	15
MAXIMUM LENGTH	=	14
MINIMUM LENGTH	=	14.67
AVERAGE LENGTH	=	1
RANGE	=	0.47
STANDARD DEVIATION		
WORK PERIOD/RECREATION PERIOD RATIO	=	2.69
AVERAGE RATIO	=	0.83
STANDARD DEVIATION		

RAVX SCORE = 2 208 2100 1000000 10215 + 0.83

Figure 8.6

Dominant Schedule and Preference Measures  
for the (0,2) Set of Schedules for the Day Watch,  
EXEC Code Printout

PARTITIONS FOR DAY WATCH, CASE ( 4, 2)

TOTAL NO OF PERIODS	PERIOD LENGTHS					NO OF CYCLIC GRAPHS
	1	2	3	4	5	
5	0	4	0	1	0	1
4	0	2	0	2	0	2
4	0	1	2	1	0	1

TOTAL NUMBER OF PARTITIONS = 3  
TOTAL NUMBER OF CYCLIC GRAPHS = 4

\*\*\* BEST WATCH SCHEDULE \*\*\*

DAY WATCH

CASE ( 4, 2)

NO SCHEDULE FOUND FOR THIS CASE

Figure 8.7

Enumeration Statistics and Result for the  
(4,2) Set of Schedules for the Day Watch,  
EXEC Code Printout

Table 8.3

Number of Partitions, Cyclic Graphs, and  
Dominating Schedules Enumerated for the  
Day Watch Data in Figure 8.4

Set	Number of Partitions	Number of Cyclic Graphs	Schedule Found
(0,0)	7	27	Yes
(0,1)	4	11	Yes
(0,2)	5	23	Yes
(0,3)	3	4	Yes
(0,4)	4	5	Yes
(1,0)	4	4	Yes
(1,1)	5	2	Yes
(1,2)	3	4	Yes
(1,3)	4	1	Yes
(1,4)	2	1	Yes
(2,0)	5	5	Yes
(2,1)	3	3	Yes
(2,2)	4	5	Yes
(2,3)	2	1	Yes
(2,4)	3	1	Yes
(3,0)	3	2	Yes
(3,1)	4	1	No
(3,2)	2	2	No
(3,3)	3	0	No
(3,4)	1	1	No
(4,0)	4	4	Yes
(4,1)	2	2	Yes
(4,2)	3	4	No
(4,3)	1	1	Yes
(4,4)	2	1	No
Total (25 Sets)	83	115	19 6

To illustrate how the computational effort of the EXEC code varies with the size and nature of daily shift allocations, consider the allocations shown in figures 8.8 and 8.9; the first (identified as the afternoon watch) requires a seven-bracket schedule and the second (identified as the night watch) requires a six-bracket schedule. The computational efforts for these allocations and the day shift described above are summarized in table 8.4. Notice that the increased uniformity of the afternoon and night shift manpower allocations produces a significant increase in the number of cyclic graphs enumerated for these shifts. The dominating schedules for all three shifts were produced from one run of the EXEC code; construction of the 68 schedules (of a possible 75), 319 partitions, and 1,619 cyclic graphs required 267 seconds of CPU time.

#### 8.2.3 Enumeration of Optimal and Near-Optimal Multishift PR Schedules

An enumeration algorithm was developed for this thesis to construct optimal and preferable multishift PR schedules with a changeover recreation period at every shift changeover point. The algorithm uses the sets of dominating non-cyclic schedules constructed for each shift tour to construct acceptable multishift schedules. The enumeration algorithm, incorporated into a computer code called MERGE, has been used to construct multishift schedules that are 20 weeks long, and contain six shift tours.

INITIAL DATA            AFT    WATCH  
TOTAL NUMBER OF WORK DAYS        = 34  
TOTAL NUMBER OF RECREATION DAYS = 15

DAILY DISTRIBUTION OF THE RECREATION AND WORK PERIODS

	MON	TUE	WED	THU	FRI	SAT	SUN	TOTAL
DUTY	5	5	5	5	5	5	4	34
RECREATION	2	2	2	2	2	2	3	15

TOTAL NUMBER OF WEEKS = 7

RECREATION PERIODS (START OF WATCH)

MINIMUM LENGTH = 0  
MAXIMUM LENGTH = 4

RECREATION PERIODS (END OF WATCH)

MINIMUM LENGTH = 0  
MAXIMUM LENGTH = 4

RECREATION PERIODS (WATCH INTERIOR)

MINIMUM LENGTH = 2  
MAXIMUM LENGTH = 4

WORK PERIODS

MINIMUM LENGTH = 4  
MAXIMUM LENGTH = 8

Figure 8.8

Initial Data for the Afternoon Watch,  
EXEC Code Printout

INITIAL DATA	NITE WATCH
TOTAL NUMBER OF WORK DAYS	= 28
TOTAL NUMBER OF RECREATION DAYS	= 14

DAILY DISTRIBUTION OF THE RECREATION AND WORK PERIODS

	MON	TUE	WED	THU	FRI	SAT	SUN	TOTAL
DUTY	4	4	4	4	4	4	4	28
RECREATION	2	2	2	2	2	2	2	14

TOTAL NUMBER OF WEEKS = 6

RECREATION PERIODS (START OF WATCH)

MINIMUM LENGTH = 0  
MAXIMUM LENGTH = 4

RECREATION PERIODS (END OF WATCH)

MINIMUM LENGTH = 0  
MAXIMUM LENGTH = 4

RECREATION PERIODS (WATCH INTERIOR)

MINIMUM LENGTH = 2  
MAXIMUM LENGTH = 4

WORK PERIODS

MINIMUM LENGTH = 4  
MAXIMUM LENGTH = 8

Figure 8.9

Initial Data for the Night Watch,  
EXEC Code Printout

Table 8.4

Number of Partitions, Cyclic Graphs, and Dominating Schedules  
Enumerated for the Watch Data in Figures 8.4, 8.8, and 8.9

Watch	No. of Sets	No. of Partitions Enumerated	Set With the Most Partitions, No. of/Set	No. of Cyclic Graphs Enumerated	Set With the Most Cyclic Graphs, No. of/Set	No. of Dominating Schedules Enumerated	CPU Time (Secs)
Day	25	83	7/(0,0)	115	27/(0,0)	19	27
Afternoon	25	125	8/(0,1)	720	148/(0,0)	24	117
Night	25	111	8/(0,0)	780	148/(0,1)	25	123
Total	75	319	-	1,619	-	68	267

To illustrate the MERGE code, figures 8.10 through 8.13 summarize the MERGE printout based on the data and three sets of dominating shift schedules described in the previous section. Figure 8.10 summarizes the initial data printout from the MERGE program which identifies the daily allocation, number of weeks, and maximum number of weekend recreation periods for each of the three shifts, and identifies the upper and lower limits on the lengths of the changeover recreation periods (i.e., four and two days). The line entitled "SPECIAL RECREATION PERIOD LENGTH..." identifies a special user option for the inclusion of one long changeover period or "mini-vacation" during each rotation period of the multishift schedule. The MERGE printout in figure 8.10 indicates that the user has requested a seven-day mini-vacation. The line "MIN MAX NUMBER OF ..." identifies the minimum value that can be obtained for the maximum number of consecutive working weekends; this min-max value is based on the number of weekend recreation periods and the number of weeks in the multishift schedule.

Figure 8.11 summarizes the preference measure for the top ten ranked multishift schedules enumerated for the initial data shown in figure 8.10. All of the schedules listed have six weekend recreation periods and a maximum of three consecutive working weekends. The characteristics for the optimal schedule (rank number 1) are listed first; shown

# INITIAL DATA

NUMBER OF WATCHES = 3  
 NUMBER OF CASES/WATCH = 25  
 MINIMUM RECREATION PERIOD LENGTH = 2  
 MAXIMUM RECREATION PERIOD LENGTH = 4  
 SPECIAL RECREATION PERIOD LENGTH = 7

TOTAL SCHEDULE PERIOD (NO OF WEEKS) = 19  
 NITE WATCH = 6  
 AFT WATCH = 7  
 DAY WATCH = 6

MAXIMUM NUMBER OF WEEKEND RECREATION PERIODS = 6  
 NITE WATCH = 2  
 AFT WATCH = 2  
 DAY WATCH = 2

MIN MAX NUMBER OF CONSECUTIVE WORKING WEEKENDS = 3

## DAILY MANPOWER ALLOCATION

WATCH	MON	TUE	WED	THU	FRI	SAT	SUN	TOTAL
NITE	4	4	4	4	4	4	4	28
AFT	5	5	5	5	5	5	4	34
DAY	5	4	4	4	5	4	4	30
TOTAL	14	13	13	13	14	13	12	92

## DAILY RECREATION ALLOCATION

WATCH	MON	TUE	WED	THU	FRI	SAT	SUN	TOTAL
NITE	2	2	2	2	2	2	2	14
AFT	2	2	2	2	2	2	3	15
DAY	1	2	2	2	1	2	2	12
TOTAL	5	6	6	6	5	6	7	41

Figure 8.10

Initial Data for the Three-Watch Example,  
 MERGE Code Printout

# RANKING SUMMARY FOR

## 6 WEEKEND RECREATION PERIODS 3 CONSECUTIVE WORKING WEEKENDS

RANK	SOLUTION	RANK	MEASURE	CASE	NUMBERS	
				NITE	AFT	DAY
1	154	6030802300	2000100 21601 + 0.90	(4,2)	(0,3)	(1,3)
2	19	6030802220	1000100 21601 + 0.95	(2,4)	(0,4)	(3,0)
3	122	6030802220	1000100 21601 + 0.95	(4,0)	(2,1)	(2,3)
4	14	6030802220	1000100 21601 + 1.00	(2,3)	(1,3)	(4,0)
5	18	6030802220	1000100 21601 + 1.00	(2,4)	(0,3)	(4,0)
6	121	6030802220	1000001 21601 + 0.91	(4,0)	(2,1)	(1,3)
7	156	6030802211	1000100 21601 + 0.89	(4,2)	(0,4)	(0,3)
8	171	6030802211	1000100 21601 + 0.95	(4,2)	(2,1)	(2,3)
9	96	6030802211	1000100 21601 + 1.00	(3,3)	(1,1)	(2,4)
10	101	6030802211	1000100 21601 + 1.00	(3,3)	(1,3)	(1,4)

No. of weekend recreation periods

Max. no. of consecutive working weekends

Maximum work period length (days)

No. of maximum length work periods

Recreation period measure (no. of each type of recreation period)

Work period range (days)

Max. length, consecutive work periods

No. of max. length work period pairs

Standard deviation of the work to recreation period length ratios

Set identifier for the night shift schedule

Set identifier for the afternoon shift schedule

Set identifier for the day shift schedule

Figure 8.11

Preference Measures for the Top Ten Ranked Multishift Schedules with Six Weekend Recreation Periods and Three Consecutive Working Weekends, MERGE Code Printout

are the solution number (the optimal schedule was the 154th schedule found), the preference vector, and the specific sets from which each of the non-cyclic shift schedules were drawn for each shift. (The seven-day mini-vacation occurs between the day and night shifts.) Following the optimal schedule, the characteristics of the other schedules are listed in order of their preferability (up to 100 schedules can be listed).

The optimal schedule is shown in figure 8.12. (Notice that the beginning and ending period lengths for each shift correspond to the three sets identified at the far right in figure 8.11.) The preference measures for this schedule are listed in figure 8.13.

To illustrate how small the differences may be between an optimal and a lower ranking schedule, the 10th ranked schedule and its preference measures based on the same data used to derive the optimal schedule above are shown in figures 8.14 and 8.15 respectively. Although this second schedule uses a different non-cyclic shift schedule for each tour, both this schedule and the optimal schedule in figure 8.12 have the same values for the number of weekend recreation periods (6), the maximum number of consecutive working weekends (3), maximum work period length (8), the number of maximum length work periods (2), work

RANK NO = 1

SOLUTION NO = 154

MON TUE WED THU FRI SAT SUN

1	R	R	R	R	*	*	*
2	*	*	*	*	R	R	R
3	R	*	*	*	*	*	*
4	*	R	R	*	*	*	*
5	*	*	*	R	R	*	*
6	*	*	*	*	*	R	R

NITE / AFT

7	*	*	*	*	*	*	R
8	R	R	*	*	*	*	*
9	*	*	R	R	*	*	*
10	*	*	*	*	R	R	R
11	R	*	*	*	*	*	*
12	*	R	R	R	*	*	*
13	*	*	*	*	R	R	R

AFT / DAY

14	R	*	*	*	*	*	*
15	*	R	R	R	*	*	*
16	*	*	*	*	*	R	R
17	*	*	*	*	*	*	*
18	*	R	R	R	*	*	*
19	*	*	*	*	R	R	R

DAY / NITE

RANK MEASURE = 6030802 300 2000100 21601 + 0.90

Figure 8.12

Top Ranked Multishift Schedule,  
MERGE Code Printout

NUMBER OF WEEKEND RECREATION PERIODS = 6  
 MAXIMUM NUMBER OF CONSECUTIVE WORKING WEEKENDS = 3  
 MAXIMUM LENGTH WORK PERIOD = 8  
 NUMBER OF MAXIMUM LENGTH WORK PERIODS = 2

RECREATION PERIOD COMPOSITION

RANK	LENGTH/START	NUMBER
1	4 / FRI	3
2	4 / THU	0
3	4 / SAT	0
4	3 / FRI	0
5	3 / SAT	0
6	2 / SAT	2
7	4 / WED	0
8	3 / THU	0
9	4 / SUN	0
10	3 / SUN	1
11	2 / FRI	0
12	2 / SUN	0

WORK PERIOD RANGE = 2  
 MAXIMUM - TWO CONSECUTIVE WORK PERIODS = 16  
 NUMBER OF MAXIMUM - TWO CONSECUTIVE WORK PERIODS = 1  
 STANDARD DEVIATION - WORK/REC RATIOS = 0.90

WORK PERIOD LENGTHS	7	7	7	7	6	7	7	7	7	7	8	8	7
REC PERIOD LENGTHS	4	2	2	2	3	2	4	3	4	3	2	3	7

Figure 8.13

Preference Measures for the Multishift Schedule  
 in Figure 8.12, MERGE Code Printout

RANK NO = 10

SOLUTION NO = 101

	MON	TUE	WED	THU	FRI	SAT	SUN
1	R	R	R	*	*	*	*
2	*	*	*	R	R	*	*
3	*	*	*	*	*	R	R
4	R	R	*	*	*	*	*
5	*	*	R	R	*	*	*
6	*	*	*	*	R	R	R
	NITE / AFT						
7	R	*	*	*	*	*	*
8	*	R	R	*	*	*	*
9	*	*	*	R	R	R	R
10	*	*	*	*	*	*	R
11	R	R	*	*	*	*	*
12	*	*	R	R	*	*	*
13	*	*	*	*	R	R	R
	AFT / DAY						
14	R	*	*	*	*	*	*
15	*	R	R	R	*	*	*
16	*	*	*	*	*	R	R
17	*	*	*	*	*	*	*
18	*	R	R	*	*	*	*
19	*	*	*	R	R	R	R
	DAY / NITE						

RANK MEASURE = 603C802 211 1000100 21601 + 1.00

Figure 8.14

Tenth Ranked Multishift Schedule,  
MERGE Code Printout



**CONTINUED**

**5 OF 6**

NUMBER OF WEEKEND RECREATION PERIODS = 6  
 MAXIMUM NUMBER OF CONSECUTIVE WORKING WEEKENDS = 3  
 MAXIMUM LENGTH WORK PERIOD = 8  
 NUMBER OF MAXIMUM LENGTH WORK PERIODS = 2

RECREATION PERIOD COMPOSITION

RANK	LENGTH/START	NUMBER
1	4 / FRI	2
2	4 / THU	1
3	4 / SAT	1
4	3 / FRI	0
5	3 / SAT	0
6	2 / SAT	1
7	4 / WED	0
8	3 / THU	0
9	4 / SUN	0
10	3 / SUN	1
11	2 / FRI	0
12	2 / SUN	0

WORK PERIOD RANGE = 2  
 MAXIMUM - TWO CONSECUTIVE WORK PERIODS = 16  
 NUMBER OF MAXIMUM - TWO CONSECUTIVE WORK PERIODS = 1  
 STANDARD DEVIATION - WORK/REC RATIOS = 1.00

WORK PERIOD LENGTHS	7	7	7	7	7	7	6	7	7	7	8	8	7
REC PERIOD LENGTHS	2	4	2	4	2	4	3	2	4	3	2	2	7

Figure 8.15

Preference Measures for the Multishift Schedule  
 in Figure 8.14, MERGE Code Printout

period range (2), maximum length for two consecutive work periods (16), and the number of maximum length consecutive work periods (1). In fact, the only distinguishing preference characteristic between these schedules is the distribution of their recreation periods: the optimal schedule has more four-day recreation periods beginning on Friday than the 10<sup>th</sup> ranked schedule.

### 8.3 APPLICATIONS OF COMPUTERIZED MANPOWER SCHEDULING

This section describes the use of the computerized manpower scheduling programs discussed above to design work schedules for two units in the St. Louis Metropolitan Police Department.

#### 8.3.1 Evidence Technician Unit, St. Louis Police Department

The Evidence Technician Unit (ETU) of the St. Louis Metropolitan Police Department is a 19-man component of the Department's Laboratory Division. It operates three evidence collection vans 24 hours per day, one in each of the city's three police field operations areas. The vans, whose personnel perform preventive patrol activities between assignments, are dispatched to the scenes of crimes on the request of the beat patrol officers responding to the incident, where they collect evidence, search for fingerprints, and take photographs of the crime scene.

Since 1970, the unit has utilized a three-shift PR schedule; prior to 1973, the schedules were designed

manually by Nelson Heller. Since that time, however, the unit has used PR schedules designed by the EXEC and MERGE programs. The basic steps in the design of schedules for the ETU are:

- (1) determination of the unit workload by shift and day of the week;
- (2) allocation of the unit manpower by shift and day of the week in proportion to the workload;
- (3) designation of the manpower allocation for evidence technicians who will rotate shift assignments (some technicians have permanent shift assignments); and
- (4) design of an optimal multishift PR schedule for the rotating technicians.

Each of these steps is discussed below.

The workload for the 1973 ETU schedule was based on the number of radio assignments received by the unit from January 1, 1972 through October 10, 1972, a total of 8,487 assignments. The distribution of assignments by shift and day of the week is shown in table 8.5.\* The manpower distribution for the 19 men (92 duty tours per week based on  $f = 4.81$  work days per technician per week) based on this workload distribution is shown in table 8.6. (For a more complete description of how the manpower allocation for this schedule and the other applications in this section were derived, see reference 17.) Two of the 19 technicians

---

\* The ETU works three eight-hour shifts per day: the day shift (7 A.M. to 3 P.M.), the afternoon shift (3 P.M. to 11 P.M.), and the night shift (11 P.M. to 7 A.M.).



Table 8.5

Number of ETU Radio Assignments by Shift and Day of the Week,  
January 1, 1972-October 10, 1972

Shift	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Total (%)	
Day	452	416	398	347	405	417	306	2,741	(32.3)
Afternoon	462	440	431	453	430	358	395	2,969	(35.0)
Night	385	351	369	405	428	446	393	2,777	(32.7)
Total (%)	1,299 (15.3)	1,207 (14.2)	1,198 (14.1)	1,205 (14.2)	1,263 (14.9)	1,221 (14.4)	1,094 (12.9)	8,484 (100.0)	(100.0)

Table 8.6

Manpower Allocation by Shift and Day of the Week for the  
1973 Evidence Technician Unit Schedule

Shift	Number of Men Assigned	Number of Officers on Duty							Total On Duty	Percent Manpower	Percent Workload**
		Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.			
Day	6	5	4	4	4	4	4	3	28	30.4%	32.3%
Afternoon	7	5	5	5	5	5	5	4	34	37.0	35.0
Night	6	4	4	4	4	5	5	4	30	32.6	32.7
Total	19	14	13	13	13	13	14	11	92		
Percent Manpower*	100.0%	15.2%	14.1	14.1	14.1	15.2	15.2	12.0		100.0%	
Percent Workload**	-	15.3%	14.2	14.1	14.2	14.9	14.4	12.9			100.0%

\*Daily manpower percentages do not sum to 100.0 because of roundoff.

\*\*Based on the workload distribution shown in table 8.5.

were not included in the multishift rotating schedule: one was permanently assigned to the afternoon shift and another was permanently assigned to the night shift -- both men received Sunday and Monday off each week. Removing these two technicians from the 19-man allocation shown in table 8.6 produced a 17-man allocation (table 8.7) that was used as the basis for the multishift PR schedule.

Since the technicians disliked long assignments on the same shift, the decision was made to split each shift into two tours; the day and afternoon assignments were divided into two three-week tours, and the night assignment was divided into one three-week tour and one two-week tour. A set of dominating schedules was found for each shift tour using the EXEC code, and several six-tour multishift schedules were generated using the MERGE program; the number one ranked schedule (and the one implemented by the ETU) is shown in figure 8.16. Preference measures for the schedule are listed in Figure 8.17.

This schedule possesses the following useful properties:

- (1) manpower is proportional to the workload by shift and day of the week;
- (2) the number of weekend recreation periods is maximized;
- (3) the weekend recreation periods are distributed so that no tour contains more than one, and the maximum number of consecutive working weekends is four;



Table 8.7

Manpower Allocation by Shift and Day of the Week for the  
Seventeen Men Assigned to the 1973 ETU PR Schedule

Shift	Number of Men Assigned	Number of Men On Duty							On Duty Total
		Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	
Day	6	5	4	4	4	4	4	3	28
Afternoon	6	5	4	4	4	4	4	4	29
Night	5	4	3	3	3	4	4	4	25
Total	17	14	11	11	11	12	12	11	82

RANK NO = 1

SOLUTION NO = 27

	MON	TUE	WED	THU	FRI	SAT	SUN
1	R	*	*	*	*	*	*
2	*	R	R	*	*	*	*
3	*	*	*	R	R	R	R
NITE A/ AFT B							
4	*	*	*	*	*	*	*
5	*	R	R	*	*	*	*
6	*	*	*	R	R	R	R
AFT B/ DAY B							
7	*	*	*	*	*	*	*
8	*	R	R	*	*	*	*
9	*	*	*	R	R	R	R
DAY B/ NITE B							
10	*	*	*	*	*	*	*
11	*	R	R	R	*	*	*
NITE B/ AFT A							
12	R	R	*	*	*	*	*
13	*	*	R	R	*	*	*
14	*	*	*	*	R	R	R
AFT A/ DAY A							
15	R	R	R	R	*	*	*
16	*	*	*	*	R	R	R
17	*	*	*	*	*	*	R
DAY A/ NITE A							

RANK MEASURE = 5040803 30 1C0C00001 51505 + 0.99

Figure 8.16

1973 Multishift PR Schedule for the Evidence  
Technician Unit, St. Louis Metropolitan  
Police Department, MERGE Code Printout

NUMBER OF WEEKEND RECREATION PERIODS	=	5
MAXIMUM NUMBER OF CONSECUTIVE WORKING WEEKENDS	=	4
MAXIMUM LENGTH WORK PERIOD	=	8
NUMBER OF MAXIMUM LENGTH WORK PERIODS	=	3
RECREATION PERIOD COMPOSITION		
RANK	LENGTH/START	NUMBER
1	4 / FRI	0
2	4 / THU	3
3	4 / SAT	0
4	3 / FRI	1
5	3 / SAT	0
6	2 / SAT	0
7	4 / WED	0
8	3 / THU	0
9	4 / SUN	0
10	3 / SUN	0
11	2 / FRI	0
12	2 / SUN	1
WORK PERIOD RANGE	=	5
MAXIMUM - TWO CONSECUTIVE WORK PERIODS	=	15
NUMBER OF MAXIMUM - TWO CONSECUTIVE WORK PERIODS	=	5
STANDARD DEVIATION - WORK/REC RATIOS	=	0.99
WORK PERIOD LENGTHS	7 7 8 7 8 7 8 3 7 7 7 6	
REC PERIOD LENGTHS	2 4 2 4 2 4 3 2 2 7 3 2	

Figure 8.17

Preference Measures for the 1973 ETU Schedule  
in Figure 8.16, MERGE Code Printout

- (4) all interior and changeover recreation periods (except one by design) are two, three, or four days long, with most of the three- and four-day periods covering both Saturday and Sunday;
- (5) one seven-day, mini-vacation changeover recreation period has been designed into the schedule;
- (6) a changeover recreation period has been placed at every shift changeover point;
- (7) except for one three-day period, all of the work periods are six, seven, or eight days long;
- (8) technicians change shift assignments every two or three weeks; and
- (9) the 17 technicians using the multishift PR schedule have identical schedules.

A new PR schedule was designed for the ETU in 1974.

The workload data used for the new schedule is shown in table 8.8 and the manpower allocation to match that work distribution is shown in table 8.9. The optimal PR schedule designed and implemented to match this manpower allocation is shown in figure 8.18 (all 19 technicians were assigned to the rotating schedule that year). The preference measures are listed in figure 8.19.\*

#### 8.3.2 Traffic Safety Unit, St. Louis Police Department

The Traffic Safety Unit of the St. Louis Metropolitan Police Department consists of 46 officers whose primary responsibilities are traffic law enforcement and traffic control. The Unit is composed of three subunits, the

---

\* The slight changes in workload distribution in 1974 plus the generally favorable response by ETU technicians to the properties of the 1974 schedule lead to the decision by the Unit supervisor to use the same schedule in 1975.



Table 8.8

Number of ETU Radio Assignments by Shift and Day of the Week,  
September 1, 1972-June 30, 1973

Shift	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Total	(%)
Day	546	513	500	518	539	464	353	3,433	(32.4)
Afternoon	663	643	713	643	711	529	448	4,350	(41.1)
Night	385	366	346	395	431	466	422	2,811	(26.5)
Total (%)	1,594 (15.0)	1,522 (14.4)	1,599 (14.7)	1,556 (14.7)	1,681 (15.9)	1,459 (13.8)	1,223 (11.5)	10,594 (100.0)	(100.0)

Table 8.9

Manpower Allocation by Shift and Day of the Week for the  
1974 Evidence Technician Unit Schedule

Shift	Number of Men Assigned	Number of Officers on Duty							Total On Duty	Percent Manpower	Percent Workload**
		Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.			
Day	6	5	4	4	4	5	4	4	30	32.6%	32.4%
Afternoon	7	5	5	5	5	5	5	4	34	37.0	41.1
Night	6	4	4	4	4	4	4	4	28	30.4	26.5
Total	19	14	13	13	13	14	13	12	92		
Percent Manpower*	100.0%	15.2%	14.1	14.1	14.1	15.2	14.1	13.0		100.0%	
Percent Workload**	-	15.0%	14.4	14.7	14.7	15.9	13.8	11.5			100.0%

\*Daily manpower percentages do not sum to 100.0 because of roundoff.

\*\*Based on the workload distribution shown in table 8.8.

RANK NO = 1

SOLUTION NO = 838

MON TUE WED THU FRI SAT SUN

1	R	R	R	*	*	*	*
2	*	*	*	R	R	*	*
3	*	*	*	*	*	R	R
NITE A/ AFT B							
4	*	*	*	*	*	*	R
5	R	*	*	*	*	*	*
6	*	R	R	*	*	*	*
7	*	*	*	R	R	R	R
AFT B/ DAY A							
8	*	*	*	*	*	*	*
9	*	R	R	R	*	*	*
10	*	*	*	*	*	R	R
DAY A/ NITE B							
11	R	R	*	*	*	*	*
12	*	*	R	R	*	*	*
13	*	*	*	*	R	R	R
NITE B/ AFT A							
14	R	*	*	*	*	*	*
15	*	R	R	R	*	*	*
16	*	*	*	*	R	R	R
AFT A/ DAY B							
17	R	*	*	*	*	*	*
18	*	R	R	*	*	*	*
19	*	*	*	R	R	R	R
DAY B/ NITE A							

RANK MEASURE = 603C802 211 1000001 21601 + 0.85

Figure 8.18

1974 ETU Multishift PR Schedule,  
MERGE Code Printout

NUMBER OF WEEKEND RECREATION PERIODS = 6  
 MAXIMUM NUMBER OF CONSECUTIVE WORKING WEEKENDS = 3  
 MAXIMUM LENGTH WORK PERIOD = 8  
 NUMBER OF MAXIMUM LENGTH WORK PERIODS = 2

RECREATION PERIOD COMPOSITION

RANK	LENGTH/START	NUMBER
1	4 / FRI	2
2	4 / THU	1
3	4 / SAT	1
4	3 / FRI	0
5	3 / SAT	0
6	2 / SAT	1
7	4 / WED	0
8	3 / THU	0
9	4 / SUN	0
10	3 / SUN	0
11	2 / FRI	0
12	2 / SUN	1

WORK PERIOD RANGE = 2  
 MAXIMUM - TWO CONSECUTIVE WORK PERIODS = 16  
 NUMBER OF MAXIMUM - TWO CONSECUTIVE WORK PERIODS = 1  
 STANDARD DEVIATION - WORK/REC RATIOS = 0.85

WORK PERIOD LENGTHS	7	7	6	7	7	8	8	7	7	7	7	7
REC PERIOD LENGTHS	2	2	2	2	4	3	4	2	4	3	4	2

Figure 8.19

Preference Measures for the 1974 ETU Schedule  
 in Figure 8.18, MERGE Code Printout

largest of which is the Radar-Vascar Unit. It is manned by 20 officers whose responsibilities include speeding enforcement and general traffic patrol throughout the city. The second largest section is the Motorcycle Unit, manned by 14 officers assigned to patrol high accident and heavy traffic areas. The smallest subunit is the Highway Unit, manned by six two-man teams assigned to patrol the city's 29.6 miles of expressways. Each subunit operates on two eight-hour shifts (days and afternoon), seven days a week.

In 1973, the Unit commander requested assistance with the design of manpower schedules for Radar-Vascar and Motorcycle units. The workload measure he suggested for both units was the total number of reported accidents for the entire city by shift and day of the week (see table 8.10).

The manpower allocation used for the 20-manpower Radar-Vascar Unit is shown in table 8.11; equal manning levels on both shifts were requested by the Unit commander to preserve the manning levels used in previous schedules, despite the disparity in workload between the two shifts. To avoid long assignments on the same shift, each 10-week shift assignment was divided into three tours creating a six-tour multishift schedule. The optimal schedule designed for the Radar-Vascar unit is shown in figure 8.20 and the preference measures are listed in figure 8.21.



Table 8.10

Number of Reported Accidents by Shift and Day of the Week  
Covered by the Radar-Vascar and Motorcycle Units,  
Traffic Safety Unit, City of St. Louis, 1971

Shift	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Total	(%)
Day	1,104	1,270	1,130	1,184	1,294	1,732	478	8,192	(42.1)
Afternoon	1,440	1,373	1,373	1,546	2,974*	1,874	696	11,276	(57.9)
Total (%)	2,544 (13.1)	2,643 (13.6)	2,503 (12.9)	2,730 (14.0)	4,268 (21.9)	3,606 (18.5)	1,174 (6.0)	19,468 (100.0)	(100.0)

\*The Friday afternoon shift represents 11 hours of coverage. The officers assigned are divided into two groups; the first group works from 3 P.M. to 11 P.M., and the second group works from 6 P.M. to 2 A.M.

Table 8.11

Manpower Allocation by Shift and Day of the Week for the  
1973 Radar-Vascar Unit Schedule

Shift	Number of Men Assigned	Number of Officers On Duty							Total On Duty	Percent Manpower	Percent Workload**
		Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.			
Day	10	7	8	7	8	8	7	5	50	50.0%	42.1%
Afternoon	10	7	7	7	7	9	8	5	50	50.0	57.9
Total	20	14	15	14	15	17	15	10	100*		
Percent Manpower	100.0%	14.0%	15.0	14.0	15.0	17.0	15.0	10.0		100.0%	
Percent Workload**	-	13.1%	13.6	12.9	14.0	21.9	18.5	6.0			100.0%

\*Total number of on-duty shifts includes time off for paid holidays.

\*\*Based on the workload distribution shown in table 8.10.

RANK NO 1

SOLUTION NC = 10

MON TUE WED THU FRI SAT SUN

1	R	*	*	*	*	*	R
2	R	*	*	*	*	R	R
3	*	*	*	*	*	*	R

AFT A/ DAY B

4	R	*	*	*	*	*	*
5	*	*	R	R	*	*	*
6	*	*	*	*	R	R	R

DAY B/ AFT B

7	*	*	*	*	*	*	*
8	*	R	R	R	*	*	*
9	*	R	R	R	*	*	*

AFT B/ DAY A

10	R	R	R	*	*	*	*
11	*	*	*	*	*	R	R
12	*	*	*	*	*	*	R

DAY A/ AFT C

13	R	*	*	*	*	*	*
14	*	R	R	*	*	*	*
15	*	*	*	R	R	R	R
16	*	*	*	*	*	*	R

AFT C/ DAY C

17	R	*	*	*	*	*	*
18	*	R	R	*	*	*	*
19	*	*	*	R	R	R	R
20	*	*	*	*	*	*	R

DAY C/ AFT A

RANK MEASURE = 5040901 20 102C00005 61503 + 0.93

Figure 8.20

Multishift PR Schedule for the Radar-Vascar Unit,  
St. Louis Metropolitan Police Department,  
MERGE Code Printout



NUMBER OF WEEKEND RECREATION PERIODS = 5  
 MAXIMUM NUMBER OF CONSECUTIVE WORKING WEEKENDS = 4  
 MAXIMUM LENGTH WORK PERIOD = 9  
 NUMBER OF MAXIMUM LENGTH WORK PERIODS = 1

RECREATION PERIOD COMPOSITION

RANK	LENGTH/START	NUMBER
1	4 / FRI	0
2	4 / THU	2
3	4 / SAT	0
4	3 / FRI	1
5	3 / SAT	0
6	2 / SAT	2
7	4 / WED	0
8	3 / THU	0
9	4 / SUN	0
10	3 / SUN	0
11	2 / FRI	0
12	2 / SUN	5

WORK PERIOD RANGE = 6  
 MAXIMUM - TWO CONSECUTIVE WORK PERIODS = 15  
 NUMBER OF MAXIMUM - TWO CONSECUTIVE WORK PERIODS = 3  
 STANDARD DEVIATION - WORK/REC RATIOS = 0.93

WORK PERIOD LENGTHS	5	4	6	8	7	8	4	3	9	6	7	7	6	7	7	6
REC PERIOD LENGTHS	2	2	2	2	3	3	3	3	2	2	2	4	2	2	4	2

Figure 8.21

Preference Measures for the Radar-Vascar Unit Schedule  
 in Figure 8.20, MERGE Code Printout

A comparison of the computer-designed schedule in figure 8.20 with the manually-designed schedules used by the Radar-Vascar Unit in 1972 and 1973 is presented in table 8.12. This comparison indicates that of the three schedules, the computer-designed schedule has the following superior attributes:

- (1) the highest number of weekend recreation periods;
- (2) the lowest maximum number of consecutive working weekends;
- (3) the lowest maximum number of days worked in any two consecutive work periods;
- (4) the lowest number of one-day recreation periods; and
- (5) the lowest number of shift changeover points not covered by a changeover recreation period.

The manpower allocation for the 14-man Motorcycle Unit is shown in table 8.13. The optimal four-tour schedule designed for this allocation is shown in figure 8.22 and the preference measures for the schedule are listed in figure 8.23. A comparison of the multishift PR schedule in figure 8.22 with the Motorcycle Units manually-designed schedules for 1972 and 1973 is presented in table 8.14. The computer-designed schedule has the following advantages:

- (1) a reduction in the maximum number of consecutive working weekends;
- (2) a reduction in the maximum work period length;
- (3) a reduction in the maximum number of days in consecutive work periods;

Table 8.12

Comparison of the Computer-Designed Radar-Vascar Unit  
Schedule for 1973 with the Unit's Manually Designed  
Schedules for 1972 and 1973

Schedule Attributes*	Manually-Designed		Computer-Designed
	1972	1973	1973
Number of weekend recreation periods	14	12	15
Maximum number of consecutive working weekends	6	6	4
Maximum length work period (days)	9	10	9
Minimum length work period (days)	4	4	3
Maximum days worked in consecutive work periods	18	19	15
Maximum length recreation period (days)	4	7	4
Minimum length recreation period (days)	1	1	2
Number of shift changeovers with no recreation days	8	8	0

\*Schedule attributes are measured over a 60-week period to allow an integral number of rotation periods for each schedule.

Table 8.13

Manpower Allocation by Shift and Day of the Week for the  
1973 Motorcycle Unit Schedule

Shift	Number of Men Assigned	Number of Officers On Duty							Total On Duty	Percent Manpower	Percent Workload**
		Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.			
Day	7	5	5	5	5	6	5	4	35	50.0%	42.1%
Afternoon	7	5	5	5	5	6	5	4	35	50.0	57.9
Total	14	10	10	10	10	12	10	8	70*		
Percent Manpower	100.0%	14.3%	14.3	14.3	14.3	17.1	14.3	11.4		100.0%	
Percent Workload**	-	13.1%	13.6	12.9	14.0	21.9	18.5	6.0			100.0%

\*Total number of on-duty shifts includes time off for paid holidays.

\*\*Based on the workload distribution shown in table 8.10.

RANK NO =	2	SOLUTION NO = 9					
		MON	TUE	WED	THU	FRI	SAT SUN
1	R	R	*	*	*	*	*
2	*	*	R	R	*	*	*
3	*	*	*	*	*	R	R
		AFT A/ DAY B					
4	*	*	*	*	*	*	R
5	R	*	*	*	*	*	*
6	*	R	R	*	*	*	*
7	*	*	*	R	R	R	R
		DAY B/ AFT B					
8	R	R	R	*	*	*	*
9	*	*	*	R	R	*	*
10	*	*	*	*	*	R	R
11	*	*	*	*	*	*	R
		AFT B/ DAY A					
12	R	R	*	*	*	*	*
13	*	*	R	R	*	*	*
14	*	*	*	*	*	R	R
		DAY A/ AFT A					

RANK MEASURE = 4030802 1 2000101 21503 + 0.91

Figure 8.22

Multishift PR Schedule for the Motorcycle Unit,  
St. Louis Metropolitan Police Department,  
MERGE Code Printout

NUMBER OF WEEKEND RECREATION PERIODS = 4  
 MAXIMUM NUMBER OF CONSECUTIVE WORKING WEEKENDS = 3  
 MAXIMUM LENGTH WORK PERIOD = 8  
 NUMBER OF MAXIMUM LENGTH WORK PERIODS = 2

RECREATION PERIOD COMPOSITION

RANK	LENGTH/START	NUMBER
1	4 / FRI	0
2	4 / THU	0
3	4 / SAT	1
4	3 / FRI	0
5	3 / SAT	0
6	2 / SAT	2
7	4 / WED	0
8	3 / THU	0
9	4 / SUN	0
10	3 / SUN	1
11	2 / FRI	0
12	2 / SUN	1

WORK PERIOD RANGE = 2  
 MAXIMUM - TWO CONSECUTIVE WORK PERIODS = 15  
 NUMBER OF MAXIMUM - TWO CONSECUTIVE WORK PERIODS = 3  
 STANDARD DEVIATION - WORK/REC RATIOS = 0.91

WORK PERIOD LENGTHS	7	8	6	7	7	7	7	6	7	8
REC PERIOD LENGTHS	2	2	2	2	7	2	2	3	2	4

Figure 8.23

Preference Measures for the Motorcycle Unit  
 Schedule in Figure 8.22,  
 MERGE Code Printout

Table 8.14

Comparison of the Computer-Designed Motorcycle Unit  
Schedule for 1973 with the Unit's Manually-Designed  
Schedules for 1972 and 1973

Schedule Attributes*	Manually-Designed		Computer-Designed
	1972	1973	1973
Number of weekend recreation periods	8	12	12
Maximum number of consecutive working weekends	9	4	3
Maximum length work period (days)	11	10	8
Minimum length work period (days)	2	6	6
Maximum days worked in consecutive work periods	19	19	15
Maximum length recreation period (days)	5	6	7
Minimum length recreation period (days)	1	2	2
Number of shift changeovers with no recreation days	8	8	0

\*Schedule attributes are measured over a 42-week period to allow an integral number of rotation periods for each schedule.

- (4) inclusion of a seven-day mini-vacation;
- (5) elimination of one-day recreation periods; and
- (6) a reduction in the number of shift changeover points not covered by a changeover recreation period.

#### 8.4 FUTURE WORK

Although the design procedures developed in this thesis represent an interesting expansion of scheduling technology, many questions remain unanswered. Some schedule attributes, not easily incorporated into PR schedules were not considered in this study, and scheduling problems related to schedule implementation and administration were outside the scope of this work. Some of the issues and unanswered questions relating to PR schedules identified during the course of this study include:

- (1) compensatory time-off - what procedures or guidelines should govern the selection of specific days for compensatory time-off in a PR schedule? Are certain days in a PR schedule better than others, and can such days be designed into schedules?
- (2) vacation scheduling - what PR schedule attributes constrain schedule administrators in the selection of vacation time-off? What procedures can be used to provide each employee with the vacation time of his choice?
- (3) pay period lengths - how can PR schedules be designed to insure that employees work the same number of work days during each pay period (particularly important for employees working under wage contracts)?

- (4) changing manpower levels - what rules should be used to minimize disruptions to a PR schedule when the number of brackets (men) is altered?
- (5) part-time/split-shifts - how can PR schedules be designed for part-time and split-shift employees?, and
- (6) PR schedule implementation - what procedure should be used to minimize employee and work disruptions when a PR schedule is implemented?

9. ACKNOWLEDGEMENTS

I would like to express my appreciation and gratitude to my advisor, Dr. Nelson Heller, not only for his invaluable guidance during the preparation of this dissertation, but also for his friendship, counsel, and encouragement when I doubted the completion of this work. I would also like to thank several persons who freely shared their knowledge about police operations and manpower scheduling with me. These include Dr. J. Thomas McEwen, former research analyst with the St. Louis Metropolitan Police Department; Mr. Grant Buby, Associate Director of the St. Louis Governmental Research Institute; and the following persons at the St. Louis Metropolitan Police Department: Mr. Frank Dohr of the Planning and Research Unit, Lt. William Armstrong and Ptn. Robert Dunsford of the Evidence Technician Unit, and Lt. John Carneghi of the Traffic Safety Unit. I am also grateful to Dr. Philip Zwart, who as a faculty member in the Department of Applied Mathematics and Computer Science at Washington University, provided me with invaluable advice about the theoretical aspects and implementation of the cyclic graph enumeration algorithm.

Grateful acknowledgement is made of the financial assistance provided by the Department of Applied Mathematics and Computer Science for the academic years 1971-1972 and 1975-1976, and by the Law Enforcement Assistance Administration,

United States Department of Justice, for the period 1972-1975.

A special thank you is extended to Ms. Victoria O'Dell and Ms. Gloria Horkits for their diligent work in the preparation of this manuscript.

Finally, I wish to confirm my very deep love for my wife, Elaine, and for our three children, Paul, David, and Anne; successful completion of this dissertation would not have been possible without their continual support and understanding. This work is dedicated to only one person: my son David; although he may never understand the meaning of these words, he has taught me to see, not the limitations, but the beauty in each person I meet.

10. APPENDICES

## APPENDIX 10.1

### Survey of Police Manpower Scheduling Practices

#### 10.1.1 Introduction

To obtain more information about the varieties of manpower schedules used by police agencies, a written survey was administered to representatives from 21 law enforcement agencies. The following sections describe the survey instrument, implementation, and results.

#### 10.1.2 Survey Instrument and Implementation

The survey instrument, shown in figure 10.1.1, consists of 10 questions designed to elicit information about each agency's workload and manpower distribution by shift, the type of manpower schedule used to meet that distribution, and the specific attributes and properties of the schedule used.

The survey was administered in November 1972 to 21 persons attending a short course at The Traffic Institute at Northwestern University, Evanston, Illinois.\*

The position and agency of the 21 respondents to the survey are shown in table 10.1.1; represented among the 21 agencies were seven state highway patrol units, three county police agencies, and eleven municipal police departments.

---

\*The author wishes to thank Dr. Nelson Heller for administering the survey to law enforcement personnel who were attending a seminar on manpower scheduling presented by him.

POLICE MANPOWER SCHEDULING

Department: \_\_\_\_\_ City: \_\_\_\_\_

Name: \_\_\_\_\_ Position: \_\_\_\_\_

Please answer the following questions on the schedule used by officers in your department. If some units have different schedules, base your answers on the schedule used by the called-for-service (radio motor patrol) units.

1. The watches begin at: 1st \_\_\_\_\_ 2nd \_\_\_\_\_ 3rd \_\_\_\_\_  
Any additional watches (please specify) \_\_\_\_\_
2. How many paid holidays do officers receive annually? \_\_\_\_\_
3. Are the schedules for all officers basically identical with respect to the patterns for days off, watch rotation, etc.?  
Yes \_\_\_\_\_ No \_\_\_\_\_
4. Are all officers permanently assigned to watches? Yes  
No If some, but not all of the officers work permanent watch assignments, please give the percent of men working each watch who are permanently assigned: 1st \_\_\_\_\_, 2nd \_\_\_\_\_, 3rd \_\_\_\_\_. Other watches (please specify) \_\_\_\_\_
5. Is any attempt made to schedule fewer days off on the busier days of the week? Yes \_\_\_\_\_ No \_\_\_\_\_
6. Average percent of calls and manpower on each watch:  
Calls: 1st \_\_\_\_\_, 2nd \_\_\_\_\_, 3rd \_\_\_\_\_  
Manpower: 1st \_\_\_\_\_, 2nd \_\_\_\_\_, 3rd \_\_\_\_\_
7. For officers working a rotating schedule.
  - a. Indicate the watch rotation sequence, e.g., 3-2-1: \_\_\_\_\_
  - b. What is the length (in days) of the assignment to each watch?  
1st \_\_\_\_\_, 2nd \_\_\_\_\_, 3rd \_\_\_\_\_
  - c. When officers change watches, on what day of the week is the change made? \_\_\_\_\_ Do all officers change at the same time? Yes \_\_\_\_\_ No \_\_\_\_\_
  - d. How frequently do officers experience 8 or 32 hour off-duty periods between working watches due to changing watches?  
\_\_\_\_\_
8. Indicate the longest and shortest work periods (consecutive work days without a day off) in the schedule:  
longest (days) \_\_\_\_\_ shortest (days) \_\_\_\_\_
9. Indicate the longest and shortest recreation periods (consecutive days off), not including vacations:  
longest (days) \_\_\_\_\_ shortest (days) \_\_\_\_\_
10. How frequently, on the average, do officers have weekends off (both Saturday and Sunday)? One weekend out of \_\_\_\_\_ What is the maximum number of working weekends separating weekends off? \_\_\_\_\_

Figure 10.1.1

Police Manpower Scheduling Survey Instrument

Table 10.1.1

Police Departments Surveyed, Traffic Institute, Northwestern University,  
June 1972

Name	Position	Police Agency	State
1. F. Bruce Baker	Sergeant, Planning & Research Division	Washington State Patrol	Washington
2. F. DeWayne Beggs	Patrolman	Norman Police Department	Oklahoma
3. Eugene Burdine III	Planning Analyst	Montgomery County Police Department	Maryland
4. Ron Cochran	Lieutenant	Fort Lauderdale Police Department	Florida
5. William Cooke	Planning and Research	Bureau of Police, Bethlehem	Pennsylvania
6. Ralph Davis	Captain	Bartlesville Police Department	Oklahoma
7. William Doster	Captain	Kalamazoo Police Department	Indiana
8. Lee Duggan	Chief	Ocean City Police Department	Maryland
9. Edward D. Flaherty	Director Research & Development Division	Waterbury Police Department	Connecticut
10. C. J. Gawronski	Patrolman	Cook County Sheriff's Department	Illinois

Table 10.1.1 - (continued)

Name	Position	Police Agency	State
11. Albert T. Jameson	Lieutenant	Maine State Police	Maine
12. Warren Keller	Analyst	Kansas City Police Department	Missouri
13. Michael Laski	Director, Planning & Research	St. Louis County Police	Missouri
14. William S. Lindsey	Lieutenant, Research & Development Section	Anne Arundel County Police	Maryland
15. James J. McJoratta	Captain	Connecticut State Police	Connecticut
16. Richard E. Patrick	Analyst	Louisiana State Police	Louisiana
17. John F. Roche	Operational Planning	Hartford Police Department	Connecticut
18. George R. Ryan	Planning Sergeant	Wisconsin State Patrol	Wisconsin
19. James B. Scrivner	Sergeant, Planning & Training Bureau	Madison Police Department	Wisconsin
20. Mark C. Thompson	Trooper	New Hampshire State Police	New Hampshire
21. Richard A. Wiberg	Patrolman	Minnesota Highway Patrol	Minnesota

### 10.1.3 Survey Results

The results of the survey are presented in four parts:

- (1) general schedule properties (questions 1, 2, and 3);
- (2) workload distribution and manpower allocation by shift (questions 5 and 6);
- (3) rotating schedule properties (questions 4 and 7); and
- (4) schedule preference attributes (questions 8, 9, and 10).

#### 10.1.3.1 General Schedule Properties

Table 10.1.2 summarizes the responses to survey questions 1, 2, and 3. The responses indicate that most departments provide the same work schedules for all officers, and use three eight-hour shifts per day with the morning or day shift usually beginning between 6 and 8 A.M. Five of the departments (23.8 percent) reported using work shifts that were longer than eight hours. Only one department indicated use of split shifts, but nine departments (42.9 percent) reported using some form of overlay shift to provide more manpower during the busier hours of the day. At the time of this survey, most of the departments were providing from 7 to 12 paid holidays annually -- the average number reported was slightly greater than 9 days per year.

#### 10.1.3.2 Workload Distribution and Manpower Allocation

Survey questions 5 and 6 were used to obtain information about the workload distribution by shift, and the manner in which available manpower was allocated by shift and day of the week; the responses to these questions are presented in table 10.1.3.

Table 10.1.2

General Schedule Properties Based on  
Survey Questions 1, 2, and 3

Property	Response	No. of Responses	Percentage*
1. All officers work the same schedule	Yes	15	71.4%
	No	5	23.8
	No response	1	4.8
2. Start hour, first shift	6 A.M.	4	19.0
	7 A.M.	5	23.8
	8 A.M.	8	38.1
	Other	2	9.5
	No set hours	2	9.5
3. Shift length	Eight hours	16	76.2
	Other	5	23.8
4. Split shifts	Yes	1	4.8
	No	20	95.2
5. Overlay shifts	Yes	9	42.9
	No	12	57.1
6. Number of paid holidays**	0-3	1	4.8
	4-6	1	4.8
	7-9	8	38.1
	10-12	8	38.1
	13-15	3	14.3

\*The sum of the percentages for each property may not equal 100.0 because of roundoff.

\*\*Average number of paid holidays equals 9.3 days.

Table 10.1.3

Workload Distribution and Manpower Allocation  
(Survey Questions 5 and 6)

Question 5			
	Response	Number of Responses	Percentage
Is any attempt made to schedule fewer days off on the busier days of the week?	Yes	15	71.4%
	No	6	28.6%
Question 6*			
Shift	Workload Distribution Percentage**	Manpower Allocation Percentage	
Day (7 A.M.-3 P.M.)***	33.1%	32.7%	
Afternoon (3 P.M.-11 P.M.)	43.3	41.0	
Night (11 P.M.-7 A.M.)	23.7	26.3	

\*Workload and manpower data based on responses from 18 departments.

\*\*The sum of the percentages does not sum to 100.0 because of roundoff.

\*\*\*Actual shift hours vary slightly by department, see table 10.1.2.

Over 70 percent of the departments indicated that an attempt was made to schedule fewer days off on busier days of the week. Table 10.1.3 presents the average calls for service distribution and manpower allocation for each shift based on responses from 18 of the 21 departments. The calls for service data clearly reveal the disproportionate amount of work on the afternoon shift; with 43.3 percent of all calls for service, the afternoon shift receives almost twice as many calls as the night shift.

The average manpower allocation by shift: 41 percent on the afternoon shift, 33 percent on the day shift, and 26 percent on the night shift, indicates that these police agencies were successful in scheduling available manpower in proportion to the workload distribution by shift.

#### 10.1.3.3 Shift Rotation Properties

Table 10.1.4 summarizes the information obtained about the number of departments which use shift rotating schedules (question 4) and the nature of those schedules (question 7). Sixteen departments (76.2 percent) reported using some form of a rotating schedule, although 7 of the 16 departments also indicated that some men were permanently assigned to one shift. Somewhat surprisingly, only 9 of the 16 departments indicated use of a night-afternoon-day or backward rotation sequence; this result suggests that departments which utilize other rotation sequences must

Table 10.1.4

Shift Rotation Properties Based on  
Survey Questions 4 and 7

Property	Response	No. of Responses	Percentage*
1. All officers are permanently assigned to one shift	Yes	5	23.8
	No	16	76.2
2. Some men are permanently assigned to one shift	Yes	7	43.8
	No	9	56.3
3. Rotation sequence	Day-aft-night	4	25.0
	Night-aft-day	9	56.3
	Other	2	12.5
	No response	1	6.3
4. Time assigned to each shift	Same on each shift	13	81.3
	Varies by shift	2	12.5
	No response	1	6.3
5. Length of time assigned to each shift	1 week	4	25.0
	2 weeks	3	18.8
	4 weeks	5	31.3
	8 weeks	1	6.3
	Other	2	12.5
	No response	1	6.3
6. Shift change day	Monday	5	31.3
	Sunday	2	12.5
	Varies	7	43.8
	Other	1	6.3
	No response	1	6.3
7. Frequency of 8- and 32-hour changeover recreation periods	Never	1	6.3
	Each week	2	12.5
	Every 2 weeks	5	31.3
	Every 4 weeks	1	6.3
	Other	4	25.0
	Unknown	3	18.8

Note: Responses for properties 2 through 7 are based on the 16 departments that responded "No" to property 1.

\*The sum of the percentages for each property may not equal 100.0 because of roundoff.

schedule recreation days at some shift changeover points to avoid consecutive work tours.

Interestingly, 81.3 percent of the departments using rotating schedules also reported that they use equal length shift tours. Although this result appears to conflict with the survey response which indicated that manpower allocations are proportional to the workload by shift, several scheduling procedures can be used to achieve proportional manpower distributions with equal length tours. These include:

- (1) permanently assigning some officers to busier shifts;
- (2) restricting time-off for vacations, paid holidays, and training to lighter workload shifts;
- (3) using overlay shifts to provide additional manpower during busiest hours of the day (8 of the 16 departments using rotating schedules also use an overlay shift); and
- (4) scheduling each officer for more shift tours on busier shifts (e.g., a rotation sequence of day-afternoon-night-afternoon would, if each shift tour was the same length, put approximately half of the available manpower on the afternoon shift).

The survey responses revealed considerable variation, among the 16 departments using rotating schedules, in the length of time officers are assigned to each shift. Shift lengths of one, two, and four weeks were the most frequently reported. Half of the 16 departments reported using the same day of the week for every shift changeover; Monday and

Sunday were the most frequently used days. Several departments reported that changeover days were determined by calendar date rather than by the day of week (e.g., changing shifts on the 15th and 30th of each month). Although most departments reported that 8- and 32-hours breaks between shift assignments occurred, there was considerable variation reported among the departments in the frequency of such breaks.

#### 10.1.3.4 Schedule Preference Measures

The survey responses to questions 8, 9, and 10, summarized in table 10.1.5, reveal the variations in both the lengths of work and recreation periods, and the number and frequency of weekend recreation periods in the manpower schedules used by the police agencies included in this survey. Maximum work period lengths ranged from 5 to 10 days (the average was 6.5), while minimum work period lengths varied from 1 to 7 days (the average minimum was 4.0); the average range (i.e., the difference between the maximum and minimum work period lengths) equaled 2.4 days. Recreation period lengths varied from 1 to 5 days with the average lengths for the maximum and minimum length periods equal to 3.5 and 2.0 days respectively.

Although most departments reported that a weekend recreation period was scheduled every three or four weeks, some departments indicated that weekend periods could only

Table 10.1.5

Schedule Preference Measures Based on  
Survey Questions 8, 9, and 10

Work and Recreation Period Lengths			
	Number of Days		
	Maximum	Minimum	Average
Work period length:			
Longest	10	5	6.5
Shortest	7	1	4.0
Range	9	0	2.4
Recreation period length:			
Longest	5	2	3.5
Shortest	5	1	2.0
Range	4	0	1.5
Weekend Recreation Periods			
	Response	No. of Responses	Percentage***
Frequency of weekend recreation periods*	One/3 weeks	4	19.0%
	One/4 weeks	7	33.3
	One/5 weeks	1	4.8
	One/6 weeks	2	9.5
	Other	5	23.8
	Unknown	2	9.5
Maximum number of consecutive working weekends**	3 weeks	2	9.5
	4 weeks	3	14.3
	5 weeks	3	14.3
	6 weeks	5	23.8
	Other	3	14.3
	Unknown	5	23.8

\*Average equals one weekend recreation period every 4.4 weeks.

\*\*Average equals five consecutive working weekends.

\*\*\*The sum of the percentages for each property may not equal 100.0 because of roundoff.

be scheduled once every five, six, or more weeks. The relatively small number of weekend recreation periods was also reflected in the number of consecutive working weekends; 9 of the 16 responding agencies reported using schedules which included a maximum of at least five consecutive working weekends.

## APPENDIX 10.2

### Derivation of the Equation for the Exact Number of Distinct, Feasible One-Shift PR Schedules for a Given Cyclic Graph

#### 10.2.1 Notation

The following notation is used in the discussion  
below:

- $N$  - total number of feasible schedules
- $\hat{N}$  - upper limit on the total number of feasible schedules
- $N_i$  - total number of distinct arrangements of the recreation periods that start on day  $i$ ,  
 $i = 1, 2, \dots, 7$
- $n_{ij}$  - total number of recreation periods of type  $j$   
(i.e.,  $j$  days long) that begin on day  $i$ ,  
 $i = 1, 2, \dots, 7$
- $n_{i.}$  - total number of recreation periods that start  
on day  $i$ ,  $n_{i.} = \sum_j n_{ij}$
- $n_{..}$  - total number of recreation periods,  $n_{..} = \sum_{i=1}^7 n_{i.}$
- $R_i$  - total number of recreation days allocated  
to day  $i$  (i.e., the number of nodes on ray  $i$   
of the star diagram)
- $W$  - total number of weeks in the schedule

#### 10.2.2 Recreation Period Interactions

In section 6.2, an upper bound on the total number of distinct, feasible schedules that can be enumerated from a given cyclic graph is derived based on the simplifying assumption that the placement of each recreation period is independent of the placement of all other periods in the graph. Two upper bounds are derived:

- (1) for cyclic schedules:

$$\hat{N}_c = W^{n..-1} \quad (10.2.1)$$

- (2) for non-cyclic schedules:

$$\hat{N} = W^{n..} \quad (10.2.2)$$

More accurate expressions for  $\hat{N}_c$  and  $\hat{N}$  are derived in this appendix. Improvements over expressions (10.2.1) and (10.2.2) are obtained by including the effects of the interactions between recreation periods within a cyclic graph created by their start day and length characteristics. Each of these characteristics is discussed below.

#### 10.2.2.1 Recreation Periods that Begin on the Same Day of the Week

Since the start day for each recreation period in a graph must occupy a unique position within a W-week schedule, the total number of arrangements for  $n_i$  distinct recreation periods (i.e., periods with different lengths), each of which begins on day  $i$ , is equivalent to the number of arrangements that are possible when  $n_i$  distinguishable balls are placed in W slots; i.e.,

$$N_i = \frac{W!}{(W-n_i)!}, n_i \text{ integer} \quad (10.2.3)$$
$$0 \leq n_i \leq W$$

As an example, the two recreation periods that begin on Monday in the cyclic graph in figure 10.2.1 can be arranged in  $6!/(6-2)! = 30$  distinct ways over a six-week schedule.

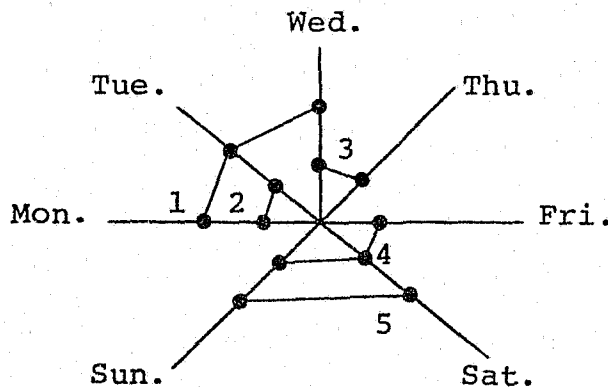


Figure 10.2.1

Cyclic Graph with Five Recreation Periods

Two of the arrangements are illustrated in figures 10.2.2 and 10.2.3.

#### 10.2.2.2 Recreation Periods That Begin on the Same Day of the Week and Have the Same Length

Since identical recreation periods (i.e., periods that have the same length and begin on the same day of the week) are indistinguishable, the number of distinct arrangements of  $n_i$  periods over a  $W$ -week schedule is reduced if one or more pairs of identical periods are present in  $n_i$ . The

reduced number of arrangements can be calculated by modifying equation (10.2.3) to include the number of each period type  $n_{ij}$  that starts on day  $i$ . Whenever  $n_{ij} > 1$  for any  $j$ , the number of arrangements for day  $i$  is reduced, the modified equation is:

$$N_i = \frac{W!}{(W-n_i)! \prod_j (n_{ij}!)} \quad \begin{array}{l} n_{ij} \text{ integer} \\ 0 \leq n_i \leq W \end{array} \quad (10.2.4)$$

If  $n_{ij} = 1$  for all  $j$ , equation (10.2.4) is equivalent to equation (10.2.3).

As an example, if the two recreation periods that begin on Monday in the cyclic graph in figure 10.2.1 have the same length (e.g., if  $n_{12} = 2$ ), they can be arranged in  $6!/(6-2)!2 = 15$  distinct ways over a six-week schedule.

### 10.2.2.3 Non-Start Recreation Days

The final interaction condition to be discussed is illustrated in the schedules shown in figures 10.2.2 and 10.2.3. On both Wednesday and Saturday, two recreation days appear over the six-week schedules. On both days, one recreation day, a start day, is used to begin a recreation period, and the other recreation day, a non-start day, is used as part of a recreation period that begins on another day of the week. The presence of non-start recreation days on day  $i$  reduces the number of weeks which

	M	T	W	T	F	S	S
1	<sup>1</sup> R	R	R				
2	<sup>2</sup> R	R					
3			<sup>3</sup> R	R			
4					<sup>4</sup> R	R	R
5						<sup>5</sup> R	R
6							

Figure 10.2.2

PR Schedule Based on the Five Recreation  
Periods in Figure 10.2.1

	M	T	W	T	F	S	S
1	<sup>1</sup> R	R	R				
2			<sup>3</sup> R	R		<sup>5</sup> R	R
3							
4					<sup>4</sup> R	R	R
5							
6	<sup>2</sup> R	R					

Figure 10.2.3

Second PR Schedule Based on the Five Recreation  
Periods in Figure 10.2.1

can be used for periods which begin on day  $i$  (e.g., in each schedule above, the periods that begin on Wednesday and Saturday are limited to five weeks because of the presence of one non-start day). The total number of non-start recreation days on day  $i$  is given by  $R_i - n_i$ , and the total number of weeks that can be used for  $n_i$  period starts on day  $i$  is reduced from  $W$  to  $W - (R_i - n_i)$ .

Replacing  $W$  in equation (10.2.4) with  $W - (R_i - n_i)$  yields the following modified expression for the total number of distinct arrangements for day  $i$ :

$$N_i = \frac{(W - R_i + n_i)!}{(W - R_i)! \prod_j (n_{ij}!)} \quad \begin{array}{l} n_{ij} \text{ integer} \\ 0 \leq n_{ij} \leq W - R_i \end{array} \quad (10.2.5)$$

It is important to note that the validity of equation (10.2.5) relies on the following assumptions:

- (1) the cyclic graph contains no recreation periods that are greater than seven days in length\*, and
- (2) each of the  $R_i - n_i$  non-start recreation days, have been assigned to specific locations (weeks) within the  $W$ -week schedule.

---

\* A more general statement of the assumption is that the length of every recreation period must be less than or equal to the number of rays in the cyclic graph.

### 10.2.3 Exact Number of Distinct Schedules for One Class of Cyclic Graphs

Although equation (10.2.5) can be used to determine the exact number of distinct arrangements for each day of the week, the product of the  $N_i$ 's; i.e.,

$$N = \prod_{i=1}^7 N_i = \prod_{i=1}^7 \frac{(W-R_i+n_i)!}{(W-R_i)! \prod_j (n_{ij}!)}, \begin{matrix} n_{ij} \text{ integer} \\ 0 \leq n_i \leq W-R_i \\ i = 1, 2, \dots, 7 \end{matrix} \quad (10.2.6)$$

is not, unfortunately, a valid expression for determining the exact number of schedules that can be enumerated from every cyclic graph. Equation (10.2.6) is useful, however, because it does produce exact answers for a large subset of all cyclic graphs, and as such, can be used to illustrate the enormous number of distinct schedules that can be enumerated from a single graph.

Equation (10.2.6) is valid when it is used for cyclic graphs which contain at least one pair of period disjoint rays. Adjacent rays (a,b)\* are defined to be period disjoint if the number of period starts on ray b equals the total number of nodes on ray b (i.e., there are no non-start

\* Rays a and b are defined to be adjacent if they can be included in the same two-day recreation period. The notation (a,b) indicates that ray b "follows" ray a (in a clockwise direction) in the graph.

recreation days on ray  $b$ , or equivalently if  $n_b = R_b$ ). Period disjoint rays are easily identified in a cyclic graph; they are pairs of adjacent rays that have no period lines between them. As an example, the graph in figure 10.2.1 has two pairs of period disjoint rays: (Sunday, Monday) and (Thursday, Friday). A cyclic graph may have several or no period disjoint rays (e.g., the large cyclic graph in figure 10.2.4 with 20 nodes and seven recreation periods has no period disjoint rays; there is at least one recreation period line joining every pair of adjacent rays).

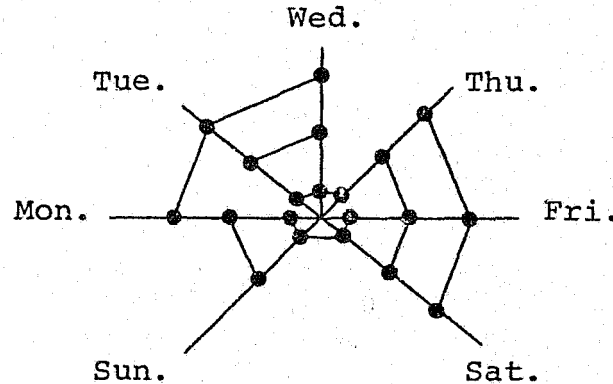


Figure 10.2.4

Cyclic Graph with Seven Recreation Periods

The validity of equation (10.2.6) for cyclic graphs with at least one pair of period disjoint rays can be shown with the following argument. It was noted above that the expression for each factor in (10.2.6) (i.e., the exact number of distinct arrangements of periods that begin on day  $i$ ), is valid with the assumption that all of the non-start recreation days on day  $i$  have been assigned to specific weeks within the schedule. Stated in another way, this condition implies that each recreation period that includes a non-start recreation day on day  $i$  has already had its initial recreation day, which must be on a different day of the week, assigned to a specific week in the schedule. Hence, a more accurate notation for equation (10.2.5) would be

$$N_{i/\{j\}} = \frac{(W-R_i+n_{i.})!}{(W-R_i)! \prod_j (n_{ij}!)} \quad \begin{array}{l} n_{ij} \text{ integer} \\ 0 \leq n_{i.} \leq W-R_i \end{array}$$

where  $N_{i/\{j\}}$  indicates that the number of distinct arrangements of the periods that begin on day  $i$  is conditional upon the assignment to specific weeks of all recreation periods which contain non-start recreation days on day  $i$  and begin on day  $j$ ,  $j = 1, 2, \dots, 7$ ,  $j \neq i$ .

Paralleling the logic used to calculate the probability of multiple events using conditional probabilities\*, the

---

\*  $p(ab \dots cd) = p(a) \cdot p(b/a) \dots p(d/ab \dots c)$ .

total number of arrangements for a cyclic graph can be accurately determined by equation (10.2.6) if the conditional numbers of arrangements for each day of the week correspond to the following generalized form of (10.2.6):

$$N = N_1 \cdot N_{2/\{1\}} \cdot N_{3/\{1,2\}} \cdot \dots \cdot N_{7/\{1,2,\dots,6\}} \quad (10.2.7)$$

The characteristics of equation (10.2.7) are:

- (1) the presence of one unconditional term ( $N_1$ ), and
- (2) the elements of the conditional set  $\{j\}$  for each term satisfy the rule that  $j < i$  for all days in  $\{j\}$  for day  $i$ .

Application of equation (10.2.6) to any cyclic graph with a pair of period disjoint rays always yields a set of daily factors  $N_i$  which correspond to the right side of equation (10.2.7).

To illustrate, consider the cyclic graph in figure 10.2.1, and use the "b day" of one pair of period disjoint rays in the graph as day 1 for equation (10.2.7). In figure 10.2.1, either Monday or Friday can be used as day 1; assume Monday is selected. Applying equation (10.2.5), the  $N_1$  factor is unconditional because Monday contains no non-start recreation days, a property of the b ray of every pair of period disjoint rays. Applying (10.2.5) again, the  $N_2$  factor based on Tuesday is conditional only on day 1 (i.e.,  $N_{2/\{1\}}$ ) because the non-start recreation days on

Tuesday can only belong to periods that begin on Monday (day 1). Similarly, the  $N_3$  factor is conditional only on Monday and Tuesday (i.e.,  $N_3/\{1,2\}$ ). Continuing this reasoning for all seven days produces the following expression for the number of arrangements in the entire graph:

$$N = N_1 \cdot N_{2/\{1\}} \cdot N_{3/\{1,2\}} \cdot N_{4/\{3\}} \cdot N_5 \cdot N_{6/\{5\}} \cdot N_{7/\{5,6\}}$$

which satisfies both requirements cited above for equation (10.2.7). The results of this example can be generalized for all cyclic graphs which possess at least one pair of period disjoint rays by noting that:

- (1) the b ray of each pair of period disjoint rays always produces an unconditional term for equation (10.2.5) (e.g., the Monday ( $N_1$ ) and Friday ( $N_5$ ) factors above), and
- (2) if the b ray of a period disjoint ray is selected as day 1, the elements of the conditional set  $\{j\}$  for each subsequent day  $i$  satisfy the requirement that  $j < i$  for all days in set  $\{j\}$  for day  $i$ .

#### 10.2.4 Summary

For cyclic graphs with at least one pair of period disjoint rays, the exact number of distinct, non-cyclic schedules is given by

$$N = \prod_{i=1}^7 \frac{(W-R_i+n_i)!}{(W-R_i)! \prod_j (n_{ij}!)} \quad \begin{matrix} n_{ij} \text{ integer} \\ 0 \leq n_i \leq W-\kappa_i \end{matrix} \quad (10.2.8)$$

For one-shift cyclic schedules, equation (10.2.8) is reduced by a factor of  $W$ ; i.e.,  $N_c = N/W$ .

11. BIBLIOGRAPHY

1. Whisenand, P. M., B. L. Gates, and G. Medak. "The 4-Day Workweek in Law Enforcement," Management Information Service, 3:LS-9, Washington: International City Management Association, September 1971.
2. U. S. Department of Justice, Law Enforcement Assistance Administration. 4-10 Plan: Police Explore Potential of 4-Day Workweek, by J. T. McEwen. Selected Topic Digest No. 1. Washington: National Criminal Justice Reference Service, April 1972.
3. "4-Day Work Week Bringing 'Leisure Ethic' to Society," St. Louis Post-Dispatch, 28 July 1974.
4. "Workers in St. Louis Area Pleased With 4-Day Work Week," St. Louis Post-Dispatch, 29 July 1974.
5. "4-Day Work Week Spreading; Plan for 3-Day Week Offered," St. Louis Post-Dispatch, 30 July 1974.
6. Underwood, A. B. "What a 12-Hour Shift Offers," American Journal of Nursing, 75 (July 1975), 1176-1178.
7. Dobelis, M. C. "Boost Staff Productivity With a Three-Day Work Week," Computer Decision, (October 1971), 26-29.
8. "Firm's 3-Day Work Week Is Beneficial To Everyone," St. Louis Post-Dispatch, 10 November 1973.
9. Luce, W. J. "A Shift Scheduling Algorithm," Paper presented at the 44th national meeting of the Operations Research Society of America, November 1973, San Diego, California.
10. Byrne, J. L., and R. B. Potts. "Scheduling of Toll Collectors," Transportation Science, 7 (August 1973), 224-245.
11. Sharp, Gunder P., and Jean Chu. "A Combinatoric Approach to Scheduling Public Transit Operators," Paper presented at the 47th national meeting of the Operations Research Society of America, May 1975, Chicago, Illinois.
12. Baker, K. R. "Scheduling Full-Time and Part-Time Staff to Meet Cyclical Requirements," Operations Research Quarterly, 25 (March 1975), 65.
13. U. S. Civil Service Commission, "Flexitime - A Guide," by Barbara Fess, Pay and Labor Administration, Bureau of Policies and Standards. Washington: Government Printing Office, May 1974.

14. Bodin, L. "Towards a General Model for Manpower Scheduling - Part I," Journal of Urban Analysis, 1:2, 1972.
15. Bodin, L. "Towards a General Model for Manpower Scheduling - Part II," Journal of Urban Analysis, 1:2, 1972.
16. Heller, N. B. "Proportional Rotating Schedules." Ph.D. dissertation, University of Pennsylvania, 1969.
17. Heller, N. B., J. Thomas McEwen, and W. W. Stenzel. Computerized Scheduling of Police Manpower. 2 vols. Prepared for the National Institute of Law Enforcement and Criminal Justice (Grant No. NI72-018G). St. Louis: Board of Police Commissioners, St. Louis Metropolitan Police Department, March 1973.
18. Heller, N. B., and W. W. Stenzel. "Design of Police Work Schedules," Journal of Urban Analysis, 2 (1974), 21-49.
19. Ignall, E., P. Kolesar, and W. Walker. Linear Programming Models of Crew Assignments for Refuse Collection, P-4717. New York: The New York City-Rand Institute, November 1971.
20. Howell, J. P. "Cyclical Scheduling of Nursing Personnel," Hospitals, 40 (16 January 1966), 77-85.
21. Morrish, A. R., and A. R. O'Connor, "Cyclic Scheduling," Hospitals, 44 (16 February 1970), 67-71.
22. Frances, Sister M. A. "Implementing a Problem of Cyclical Scheduling of Nursing Personnel," Hospitals, 40 (16 July 1966), 108-125.
23. Smith, L. P. "The Application of an Inter-Active Algorithm to Develop Cyclical Rotational Schedules for Nursing Personnel," Paper presented at the 49th national meeting of the Operations Research Society of America, May 1975, Chicago, Illinois.
24. Maier-Rothe, C., and H. B. Wolfe. "Cycle Scheduling and Allocation of Nursing Staff," Cambridge: Arthur D. Little, Inc., October 1971.
25. Baker, K. R. "Workforce Allocation in Cyclical Scheduling Problems: Models and Applications." Graduate School of Business Administration, Paper No. 122. Durham: Duke University, n.d.

26. Guha, Dilip, and James Browne. "Optimal Scheduling of Tours and Days Off," Paper presented at the ORSA/TSS Workshop on Automated Techniques for Scheduling of Vehicle Operators for Urban Public Transportation Services, April 1975, Chicago, Illinois.
27. Monroe, G. "Scheduling Manpower for Service Operations," Industrial Engineering, 2 (August 1970), 10-17.
28. Butterworth, R. W. and G. T. Howard. "A Method of Determining Highway Patrol Manning Schedules," Paper presented at the 44th national meeting of the Operations Research Society of America, November 1973, San Diego, California.
29. Kolesar, P. J., K. L. Rider, T. B. Crabill, and W. E. Walker. A Queueing - Linear Programming Approach to Scheduling Police Patrol Cars, P-5260. New York: The New York City - Rand Institute, June 1974.
30. Church, J. G. "Sure Staf: A Computerized Staff Scheduling System for Telephone Business Offices," Management Science, 20 (December 1973), 708-720.
31. Segal, M. "The Operator - Scheduling Problem: A Network-Flow Approach," Operations Research, 22 (July - August 1974), 808-823.
32. Tibrewala, R., D. Philippe, and J. Browne. "Optimal Scheduling of Two Consecutive Idle Periods," Management Science, 19 (September 1972), 71-75.
33. Baker, K. R. "Scheduling a Full-Time Workforce to Meet Cyclic Staffing Requirements," Management Science, 20 (August 1974), 1561-1568.
34. Guha, D. K. "An Optimal Procedure for Allocating Manpower with Cyclic Requirements: General Case," The Port Authority of New York and New Jersey, August 1974.
35. Abernathy, W. J., N. Baloff, J. C. Hershey, and S. Wandell. "A Three-Stage Manpower Planning and Scheduling Model - A Service-Sector Example," Operations Research, 21 (May - June 1973), 693 - 711.
36. Jelinek, R. C., T. K. Zenn, G. L. Delon, and R. A. Aleman. "A Nurse Scheduling and Allocation System in Operation," Technical Papers, 23rd Annual American Institute of Industrial Engineers Conference. (1972), 273-281.

37. Warner, D. M., and J. Prawda "A Mathematical Programming Model for Scheduling Nursing Personnel in a Hospital." School of Public Health. Ann Arbor: University of Michigan, August 1971.
38. Warner, D. M. "A Two-Phase Model for Scheduling Nursing Personnel in a Hospital." Ph.D. dissertation, University of Tulane, 1969.
39. Henderson, W. B., and W. L. Berry. "Determining Optimal Shift Schedules for Telephone Exchange," Institute for Research in the Behavioral, Economic, and Management Sciences, Paper No. 507. West Lafayette: Purdue University, April 1975.
40. Henderson, W. B., and W. L. Berry. "Heuristic Methods for Telephone Operator Shift Scheduling: An Experimental Analysis," Center for Business and Economic Research, Paper No. 20. Knoxville: University of Tennessee, February 1975.
41. Baker, K. R., T. Crabill, and M. Magazine. "An Optimal Procedure for Allocating Manpower with Cyclic Requirements," AIIE Transactions, 5 (June 1973), 119-126.
42. Edie, L. C. "Traffic Delays at Toll Booths," Operations Research, 2 (May 1954), 107-138.
43. Rubin, J. "Scheduling of Airline Crews for Aircraft Schedules with Frequency Exceptions," Technical Report 20. Cambridge: IBM Cambridge Scientific Center, 1974.
44. Rubin, J. "A Technique for the Solution of Massive Set Covering Problems, With Application to Airline Crew Scheduling," Technical Report No. 320-3004. Philadelphia: IBM Philadelphia Scientific Center, September 1971.
45. Nicoletti, Bernardo. "Automatic Crew Rostering," Transportation Science, 9 (February 1975), 33-42.
46. Kansas City (Missouri) Police Department. Survey of Municipal Police Departments, 1968. Kansas City: Kansas City Police Department, 1968.
47. Elmaghraby, S. E. (ed.). Symposium on the Theory of Scheduling and It's Applications. Berlin: Spring-Verlag, 1973.
48. Hadley, G., and T. M. Whiten. Analysis of Inventory Systems. Englewood Cliffs: Prentice-Hall, Inc., 1963.
49. Hillier, F. S., and G. J. Lieberman. Introduction to Operations Research. San Francisco: Holden-Day, Inc., 1967.

50. Sasieni, M., A. Yaspan, and L. Friedman. Operations Research: Methods and Problems. New York: John Wiley and Sons, Inc., 1959.
51. Beltrami, E. J., and L. Bodin. "Networks and Vehicle Routing for Municipal Waste Collection," Report No. UPS 72-18. Stony Brook: State University of New York, August 1972.
52. Bodin, L. "A Taxonomic Structure for Vehicle Routing and Scheduling Problems," Program for Urban and Policy Sciences. Stony Brook: State University of New York, n.d.
53. Bodin, L., and S. J. Kursh. "A Computerized System for the Routing and Scheduling of Street Sweepers," Program for Urban and Policy Sciences. Stony Brook: State University of New York, August 1974.
54. Etschmaier, M., and M. Rothstein. "Operations Research in the Management of the Airlines," Omega, 2 (April 1974), 157-179.
55. Angel, R. D., W. L. Caudle, R. Noonan, A. Whinston. "Computer-Assisted School Bus Scheduling," Management Science, 18 (February 1972), B279-B288.
56. Heller, N. B., and J. Thomas McEwen. The Use of an Incident Seriousness Index in the Deployment of Police Patrol Manpower. 2 vols. Prepared for the National Institute of Law Enforcement and Criminal Justice (Grant No. NI71-036G). St. Louis: Board of Police Commissioners, St. Louis Metropolitan Police Department, 1972.
57. Dantzig, G. B. "A Comment on Edie's 'Traffic Delays at Toll Booths,'" Operations Research, 2 (August 1954), 339-341.
58. Rothstein, M. "Scheduling Manpower by Mathematical Programming," Industrial Engineering, 4 (April 1972), 29-33.
59. Altman, S., E. Beltrami, S. Rappaport, and G. Schoenflo. "A Nonlinear Programming Model of Crew Assignments for Household Refuse Collection," IEEE Transactions on Systems, Man, and Cybernetics, SMC-1 (July 1973), 289-291.
60. Bennet, B. and R. Potts. "Rotating Roster for a Transit System," Transportation Science, 2 (1968), 14-33.
61. St. Louis Metropolitan Police Department. "1973 District Patrol Plan," Intra-Department Report, 22 August 1973.

62. Cochran, James L., and Milan Zeleny (eds.). Multiple Criteria Decision Making. Columbia: University of South Carolina Press, 1973.
63. Dawes, R. M. "A Case Study of Graduate Admissions," American Psychologist, (February 1971). 80-188.
64. Slovic, P., and S. Lichtenstein. "Comparison of Bayesian and Regression Approaches to the Study of Information Processing in Judgement," Organizational Behavior and Human Performance, 6 (1971), 651-730.
65. Slovic, P. "Analyzing the Expert Judge: A Descriptive Study of a Stockbroker's Decision Process," Journal of Applied Psychology, 53 (1969), 255-263.
66. Hoffman, P. J., P. Slovic, and L. G. Rorer. "An Analysis of Variance Model for the Assessment of Configural Cue Utilization in Clinical Judgment," Psychology Bulletin, 69 (1968), 338-349.
67. Raiffa, H. Preferences for Multi-Attributed Alternatives. RM-5868. Santa Monica: RAND Corporation, 1969.
68. Churchman, C. W., and R. L. Ackoff. "An Approximate Measure of Value," Operations Research, 2 (1954), 172-187.
69. Fishburn, P. C. "Additive Utilities with Finite Sets: Applications in the Management Sciences," Naval Research Logistics Quarterly, 14 (1967), 1-10.
70. Miller, J. R. III. Professional Decision Making. New York: Praeger, 1970.
71. Vesper, K. H., and Y. Sayeki. "A Quantitative Approach for Decision Analysis," Working Paper 71-1-14, University of Washington, 1971.
72. de Neufville, R. and R. L. Keeney. "Use of Decision Analysis in Airport Development in Mexico City," Analysis for Public Systems, A. Drake et al (eds.). Cambridge: MIT Press, 1972.
73. Keeney, R. L. "Multi-Dimension Utility Functions: Theory, Assessment, and Applications," Technical Report 43, Operations Research Center, MIT, 1969.
74. Fishburn, P. C. "Independence in Utility Theory with Whole Product Sets," Operations Research, 13(1965), 28-45.

75. Keeney, R. L. "Utility Independence and Preferences for Multiattributed Consequences," Operations' Research, 19 (1971), 875-893.
76. Lee, S. and E. R. Clayton. "A Goal Programming Model for Academic Resource Allocation," Management Science, 18 (1972), B395-B417.
77. Courtney, J., T. D. Klastoren, and T. W. Ruefli. "A Goal Programming Approach to Urban - Suburban Location Preferences," Management Science, 18 (1972), B258-B267.
78. Charnes, A. and W. W. Cooper. Management Models and Industrial Applications of Linear Programming. New York: Wiley, 1961.
79. Geoffreon, A., J. S. Dyer, and A. Feinberg. "An Interactive Approach for Multi-Criterion Optimization, with an Application to the Operation of an Academic Department," Western Management Science Institute, Working Paper 176, University of California, Los Angeles, 1971.
80. Larsen, R. C. "Approximating the Performance of Urban Emergency Service Systems," Operations Research, 23 (September - October 1975), 845-868.
81. Kleinmuntz, B. "The Processing of Clinical Information by Man and Machine," Formal Representation of Human Judgment, B. Kleinmuntz (ed.). New York: Wiley, 1968.
82. Clarkson, G. Portfolio Section. Englewood Cliffs: Prentice-Hall, 1962.
83. Smith, R., and P. S. Greenlaw. "Simulation of a Psychological Decision Process in Personnel Selection," Management Science, 13 (1967), B409-B419.
84. Beltman, J. "A Graph Theory Approach to Comparing Consumer Information Processing Models," Management Science, 18 (1971), P114-P129.
85. Coombs, C. H. A Theory of Data. New York: Wiley, 1964.
86. Dawes, R. M. "Social Selection Based on Multidimensional Criteria," Journal of Abnormal and Social Psychology, 68 (1964), 104-109.
87. Luce, R. D. "Semiororders and a Theory of Utility Discrimination," Econometrica, 24 (1956), 178-191.

88. Tversky, A. "Intransitivity of Preferences," Psychological Review, 76 (1969), 31-48.
89. Shepard, R. N. "On Subjectively Optimum Selection Among Multi-Attribute Alternatives," Human Judgments and Optimality, M. W. Shelby and G. L. Bryan (eds.). New York: Wiley, 1964.
90. Fishburn, P. C. "Lexicographic Orders, Utilities and Decision Rules: A Survey," Working paper, Pennsylvania State University, 1972.
91. Feller, William. An Introduction to Probability Theory and Its Applications. Vol. I. 2nd ed. New York: John Wiley and Sons, Inc., 1957.
92. Hall, Marshall. Combinatorial Theory. Waltham, Massachusetts: Blaisdell Publishing Company, 1967.
94. Agin, N. "Optimum Seeking With Branch and Bound," Management Science, 13 (Dec. 1966), B176-B185.
93. Mitten, L. G. "Branch-and-Bound Methods: General Formulation and Properties," Operations Research, 18 (Jan. - Feb. 1970), 24-34.
95. Bellmore, M., and John C. Malone. "Pathology of Travelling-Salesman Subtour-Elimination Algorithms", Operations Research, 19 (March-April 1971), 278-307.

12. VITA

Biographical items on the author of the thesis, Mr. William W. Stenzel

- 1) Born February 7, 1942
- 2) Attended the University of Wisconsin from September 1960 to June 1964. Received the degree of Bachelor of Science in Applied Mathematics and Engineering Physics in June 1964.
- 3) Attended the University of Wisconsin from September 1964 to June 1966. Received the degree of Master of Science in Statistics in June 1966.
- 4) Operations Analyst Engineer, Advanced Design Spacecraft, McDonnell Douglas Astronautics Company from June 1966 to September 1971.
- 5) Attended Washington University from September 1971 to June 1973 as a full-time student and from September 1973 to the present date as a special student. Awarded a Washington University Graduate Engineering Traineeship in the academic year 1971-1972, and a Law Enforcement Assistance Administration Doctoral Dissertation Grant for the academic year 1972-1973. Received the degree of Master of Science in Applied Mathematics and Computer Sciences in June 1972.
- 6) Research Analyst, Missouri Law Enforcement Assistance Council, Region 5, from June 1973 to October 1974.
- 7) Associate Director for The Institute for Public Program Analysis from October 1974 to the present date.
- 8) Membership in Honor Societies: Phi Eta Sigma, Phi Kappa Phi.

December 1976

Short Title: Proportional Work Schedules Stenzel, D.Sc. 1976



**END**