A SIMULATION MODEL OF POLICE PATROL OPERATIONS: PROGRAM DESCRIPTION

PREPARED FOR THE DEPARTMENT
OF HOUSING AND URBAN DEVELOPMENT
AND FOR THE CITY OF NEW YORK

PETER KOLESAR
WARREN E. WALKER

R-1625/2-HUD/NYC
FEBRUARY 1975

THE
NEW YORK CITY
RAND
INSTITUTE
545 MADISON AVENUE NEW YORK NEW YORK 10022 (212) 758-2244

Rand

Published by The Rand Corporation

A SIMULATION MODEL OF POLICE PATROL OPERATIONS: PROGRAM DESCRIPTION

PREPARED FOR THE OFFICE OF POLICY DEVELOPMENT AND RESEARCH, DEPARTMENT OF HOUSING AND URBAN DEVELOPMENT,

AND FOR THE CITY OF NEW YORK

PETER KOLESAR
WARREN E. WALKER

R-1625/2-HUD/NYC
FEBRUARY 1975

THE
NEW YORK CITY
RAND
INSTITUTE
545 MADISON AVENUE NEW YORK NEW YORK 10022 (212) 758-2244

Rand

PREFACE

This Report is designed to provide computer programmers and systems analysts with detailed documentation of a computer program developed at The New York City-Rand Institute for simulating the activities of police patrol cars in a city or a region of a city. The development of the simulation program was supported under a contract with the New York City Police Department. Its generalization and documentation was supported by a contract with the Office of Policy Development and Research of the Department of Housing and Urban Development.

The HUD contract has as its objective the development, testing, and documentation of methods to improve the allocation of resources in municipal emergency service agencies. Making these techniques widely available should ultimately result in significant improvements in the delivery of municipal emergency services.

It is presumed that the reader of this Report knows something about police patrol operations and terminology, and quite a bit about computer simulation. A companion report is being prepared that will describe the simulation and its uses for police department administrators and other city officials:

R-1625/1-HUD, *A Simulation Model of Police Patrol Operations: Executive Summary*, Peter Kolesar and Warren E. Walker, The New York City-Rand Institute, forthcoming.

These two reports are part of a series that documents several different deployment models for police, fire, and ambulance services, and describes the application of the models in several cities. Further information can be obtained from The New York City-Rand Institute.

SUMMARY

The computer simulation model described in this report was written to study the police patrol operations of the New York City Police Department. The structure of the program, however, is sufficiently general that, with only minor modifications, it could be used to study the operations of any large metropolitan police department.

The simulation is designed to examine the effect of changes in a police department's deployment of its patrol resources. Among the policy options that can be tested are: changing the number of patrol units on duty; changing the boundaries of the regions assigned to particular patrol units for patrol and for dispatch to calls; and changing dispatching procedures. Alternative policies may be compared based on a wide range of performance measures provided by the simulation. These include the delay between when a call for service is received and when it is dispatched, the delay between when a call is dispatched and when a patrol car arrives at the scene, and the workload of individual patrol units.

The program is written in SIMSCRIPT II.5. This report describes each of the routines in the program and the program's data requirements in sufficient detail that a user who is familiar with SIMSCRIPT should be able to run or modify the program. Flow charts, a complete program listing, sample data decks, and program outputs are all included.

ACKNOWLEDGMENTS

The New York City-Rand Institute's Fire Operations Simulation, designed by Grace Carter and Edward Ignall, served as a model whose success we hoped to imitate. Although the details, problems, and programming language of the Police Patrol Simulation are different, many of the design concepts of Carter and Ignall served us in good stead. Captain Daniel Cawley of the New York City Police Department checked the reasonableness of many of the assumptions that we built into the model. Jack Hagouel and Harry Elam made modifications to our original design, which were incorporated in the version reported here. James Tien's detailed knowledge of dispatching operations was also helpful. The final report was considerably improved by the constructive comments of Grace Carter and John Schank of the Rand Corporation, and Calvin Clawson and Samson Chang of the Seattle Police Department, who reviewed an earlier draft.

Finally, we acknowledge the graciousness, patience, and dedication of the officers and patrolmen of the New York City Police Department in whose patrol cars we rode while learning something about what occurs in the real world of the streets of New York City.

CONTENTS

FIGURES

TABLES

I. INTRODUCTION

The simulation program described in this Report was written to provide the New York City Police Department with a tool for evaluating the effects of various changes in the deployment of its patrol resources. Although designed to reflect patrol operations in New York City, the structure of the program is sufficiently general so that it can be used, with minor modifications, to study the operations of any municipal police department.

Simulation, in the sense that it is used here, is an imitation through time of the events that occur during police patrol operations. The program that carries out the simulation maintains, inside the computer, a map of the region being simulated, on which it keeps track of the locations of the incidents requiring police service and the locations of the simulated patrol units. A simulation run imitates patrol activities over a fixed time period, say a series of tours of duty on a series of days. The user provides the computer program with specifications of conditions at the start of the time period being simulated, including the number and locations of the patrol units and the operating and dispatching rules being followed. Then, using an internal timing mechanism and a list of "future" jobs, the program carries out the assignment of patrol cars to jobs, their travel to the jobs, the queueing of calls, if any, and stores statistical summaries of response times, patrol availabilities, workloads, etc., that provide analysts with important information with which to judge the effectiveness of the deployment policies being studied. When the simulation is over, the program prints reports containing the statistics summarizing this information. Different deployment strategies or policies can thus be tried out and compared using the simulation as a kind of "pilot plant." Based on the simulation results, analysts and managers can quickly and economically eliminate options that are clearly bad while more attractive policies can be selected for further analysis and eventual implementation.

The purpose of this Report is to provide programmers and analysts with detailed documentation of this computer program, known as The New York City-Rand Institute Police Patrol Simulation Model. It is presumed that the reader knows something about police patrol operations and quite a bit about simulation. A companion document [7]* will provide an overview of the simula-

*Figures in square brackets identify references located at the end of this document.

tion from the managerial point of view. It will describe the purpose of the model, the costs associated with using it, and provide other information that will be helpful in determining when the simulation should be used and when it should not be used. Readers interested in improving their knowledge of simulation itself can consult a general reference such as [2], [3], or [8].

The programming language used for the simulation is SIMSCRIPT II.5 [6]. With the program listing and other information provided in this Report, anyone who has access to a SIMSCRIPT II.5 compiler and is familiar with its programming conventions should be able to use the program. Machine-dependent details of the use of SIMSCRIPT II.5 on the IBM 360 or 370 series of computers can be found in [5]. The simulation program is written as a set of modular subroutines. Each of the subroutines simulates one aspect of the operation of the system (e.g., dispatching, arrival of a patrol car at the scene of an incident, etc.). The user can carry out many applications using the simulation by simply assigning values to some input parameters. But for some applications, one or more of the subroutines may have to be rewritten. For experiments done for the New York City Police Department we have created several versions of some subroutines, but, for simplicity in this Report, only one version of each subroutine is presented in detail.

The simulation has been tested for both internal and external validity. Internal validation is the process of confirming that the computer program is a logically correct translation of the hypothesized model. We performed the internal validity check by comparing results from the simulation with the output from a queueing model that was a good approximation of the system being simulated. The results of this comparison are described in [4]. External validation is the process of confirming that an inference derived from the simulated system is correct for the actual system. For example, if the simulation has external validity and it shows that a certain deployment policy is better than another policy, we can be sure that it will turn out to be better if it is actually implemented. The external validity check of this simulation model was made by, first, gathering detailed data on actual activities of patrol cars in one region of New York City, and, then, running the simulation model using the observed jobs as input. Comparisons were made between performance measures produced by the simulation and the corresponding actual performance measures. Details of this validation are given in [1].

In Section II of this Report we discuss some of the general considerations that led us to the approach we took in designing the simulation. Section III explains the structural elements of the model. In Section IV the logical flow of the program is shown and each event and subroutine is explained in detail. Section V defines each piece of data required as input for the simulation, and Section VI describes the output statistics.

The program listing in Appendix A represents one complete version of the simulation. Appendixes B and C contain a sample initialization deck and sample job stream that can be used to test the program and to serve as a guide for preparing new input data. Appendix D defines each of the global variables used in the simulation.

II. DESIGN CONSIDERATIONS

POLICY OPTIONS

The simulation has been designed to provide a police department with the capability for evaluating a wide range of alternative patrol and dispatch policies. Among the policy options that can be tested are:

(a) changing the number of patrol units on duty;

(b) changing sector boundaries and sector assignments of patrol units;

(c) changing dispatching procedures, e.g., using new nomination lists, new priority rules, unit and job location information in unit assignment.

To test a new policy of type (a) or (b) only the initialization data deck has to be changed. Many dispatching changes can be implemented simply by changing input parameters. Some changes, however, may require writing a new subroutine. We decided that this approach was preferable to trying to write one subroutine that would accommodate all possible dispatching rules, since the number of such rules is extremely large. The simulation has been designed to minimize the effort required to code the new routines.

Separate subroutines are used not only for the decision rules, but for every identifiable submodel of the simulation. Although this structure results in many short subroutines, it increases the flexibility of the program and the ease with which it can be modified.

It is wise to make the first simulation run with the model representing the current practices in the department as closely as possible. This will enable the model to be calibrated (e.g., the travel distance function could be modified or the average response velocities adjusted). It will also provide a "base case" for evaluating new deployment policies.

It should be pointed out that, because of the assumptions about travel velocities, travel distances, incident durations, etc., in the simulation, the absolute numbers representing the results of the simulation are not to be regarded or used as facts, although they will probably be good estimates of actual values. The most important use of the simulation is as a tool for comparing alternative policies by noting the differences in their performance. For example, if the simulated average response times under one policy are significantly better than under another policy, then the same comparative advantage will probably carry over to the real world.

MEASURES OF PERFORMANCE

The performance characteristics produced as output by the simulation fall into the following three categories:

- Response Time Measures: The mean, variance, and distribution of response times (actually, travel times) are gathered by job priority and by sector.
- Queueing Delays: The mean, variance, and distribution of the dispatching queue sizes and waiting times are displayed by job priority.
- Car Activity: The proportion of time each patrol unit spends on patrol, working on jobs, etc., as well as the mean, variance, and distribution of the number of precinct cars available to respond to calls, are displayed.

These performance measures are only proxies for measures of patrol effectiveness associated with the goals of a police department; smaller response times and queueing delays and higher patrol availabilities should yield better police service. But, since the nature of the relationships between these performance measures and crime suppression, arrests, and other primary goals of the patrol force are not known, we have not attempted to incorporate them in the simulation.

JOB STREAM

The simulation program and the system parameters that specify the geography, number of cars, sectors, etc., can be viewed as a "black box." Into this box is "played" a previously created sequence of incidents or jobs in chronological order. This sequence, called the job stream, resides on disk or magnetic tape and contains, for each job, its entry time, location, duration, and priority. The playing of this job stream into the simulation model is analogous to the playing of a magnetic tape "containing" music into a tape recorder. As the tape recorder physically transforms the magnetic signals into sound, the simulation model mathematically and logically transforms the input jobs into a set of output performance statistics.

The generation of the job stream has been purposely kept separate from the structure of the simulation model itself. There are several good reasons for doing so:

(1) It enables the user to generate job streams from a wide variety

of sources. For example, actual job histories, projections of future call patterns, or results of probabilistic models of call generation can all be used to provide job streams to the simulation.

(2) It conserves computer time, since, when rerunning the same stream under several different deployment options, the jobs do not have to be regenerated for each simulation run.

(3) It enables some of the statistical analysis of the job stream to be done outside the simulation itself--and therefore to be done more economically.

III. THE STRUCTURE OF THE SIMULATION MODEL AND THE JOB STREAM

We now discuss in turn how the geography, patrol resources, and operating and deployment rules are represented in the simulation. We conclude the section with a description of the job stream and the way in which patrol units are dispatched.

GEOGRAPHY

The simulation models a region as a collection of discrete points, called *blocks*, at which calls for service occur. The blocks may correspond to the centroids of city blocks or census blocks, but they need not. The blocks are assigned grid coordinates according to a rectangular coordinate system whose axes should be chosen parallel to the predominant street directions. All incidents (jobs) occur at these points and all patrol units are located with reference to this coordinate system. Each block in the region is assigned an internal reference number between 1 and N.BLOCK (the total number of blocks in the region), and belongs to the class of permanent entities called BLOCK.* Table 1 describes the attributes associated with the permanent entity BLOCK.

The blocks are aggregated into N.NBD nonoverlapping *neighborhoods*, which must be assigned sequence numbers from 1 to N.NBD (each block belongs to exactly one neighborhood). Each neighborhood belongs to the class of permanent entities called NBD. Table 2 describes the attributes associated with the permanent entity NBD.

PATROL RESOURCES

The program simulates the activities of patrol cars, including both "sector cars" and supervisory or special cars. Each car is represented as a permanent entity (in the class called CAR), and is described by several attributes, including its name, sector responsibilities, current location, and the type of job on which it is working. The patrol area assigned to a particular car is called its *sector* (also called a *beat* in some cities). A car's sector is composed of the set of all neighborhoods for which that car has patrol responsibility. Each sector will be assigned one or more

---

*See [6] for an explanation of entities and attributes in SIMSCRIPT II.5.

Table 1

ATTRIBUTES ASSOCIATED WITH THE ENTITY "BLOCK"

Attribute	Description
TAXNO	The external identification number associated with the block
NBDID	The internal reference number of the neighborhood to which the block belongs
XCORD	The x coordinate of the center of the block
YCORD	The y coordinate of the center of the block

Table 2

ATTRIBUTES ASSOCIATED WITH THE ENTITY "NBD"

Attribute	Description
NBD.NAME	The 4-character alphanumeric name associated with the neighborhood
N.SECTOR.CARS	The number of sector cars assigned to the neighborhood
N.ADJACENT	The number of cars designated as adjacent resources for the neighborhood

sector cars, but some neighborhoods may have no sector cars assigned to them.

Figure 1 illustrates the partition of a sample rectangular region into five neighborhoods assigned to three sector cars. (Note: The individual blocks constituting each neighborhood are not shown.) Neighborhoods 1 and 2 jointly constitute Sector A and Neighborhoods 2 and 3 constitute Sector B. Thus, Sectors A and B overlap. Sector CD is composed of Neighborhoods 4 and 5. One sector car is assigned to Sector A, one to Sector B, and the third is responsible for patrolling Sector CD.

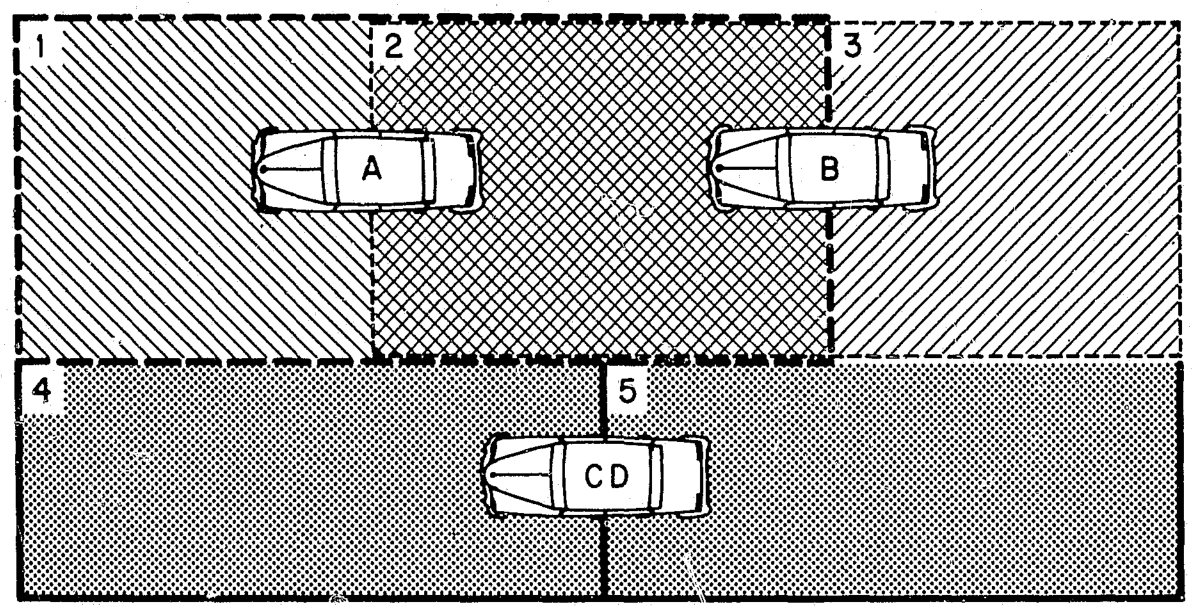
Patrol units are numbered from 1 to N.CAR (where N.CAR is the number of cars to be simulated) for internal reference. Table 3 describes the most important attributes associated with a patrol unit.

Sector cars and supervisory or special cars are differentiated only by their sector assignments and in the way they are nominated for dispatch. A sector car is assigned as the primary car for one or more neighborhoods and may be nominated as an alternate car for some other neighborhoods. This means that the simulation will attempt to assign it to jobs in its sector(s) if it is free. In addition, some types of jobs are only assigned to a sector car. Supervisory or special cars have no sector assignment and are dispatched only to high-priority jobs or to certain jobs when no sector cars are available.

MOVEMENT OF PATROL UNITS

To imitate the continuous movement of patrol units in the simulation would be very costly in terms of computing time and program complexity, and would change the results very little. Hence, the movement of patrol units is simulated as follows:

- Patrol units move between points as if on a dense rectangular street grid. Distances and times are calculated as shown below.
- If a unit completes a job in its sector, it is placed "on patrol" at the location of the job just completed, where it remains until its next assignment.
- If a unit completes a job outside its sector and is not dispatched to another job, it returns to the centroid of its sector (traveling on the street grid) and goes "on patrol" there. The centroid of each car's sector is specified by the user.



Legend

Sector	Shading	Neighborhoods Comprising Sector	Car Assigned
A		1 and 2	A
B		2 and 3	B
CD		4 and 5	CD

Fig. 1—An example of how neighborhoods and sectors are specified

Table 3

ATTRIBUTES ASSOCIATED WITH THE ENTITY "CAR"

Attribute	Description
NUMB.SECTORS	The number of neighborhoods to which the unit is assigned as a sector car
CAR.NAME	A 4-character alphanumeric name identifying the patrol unit for output purposes
XLOC	The x coordinate of the current location of the car
YLOC	The y coordinate of the current location of the car
ASSIGNMENT	The job to which the unit is currently assigned (if any)
CENTROID	The block to which the unit goes when it returns to patrol in its sector after responding to a job outside its sector (usually chosen to be at or near the centroid of the car's sector).
SPT	The priority of the job currently being serviced by the unit
SOT	An indicator variable that = 1 if the unit is working as primary car = 2 if the unit is working as backup car = 3 if the unit is working as tertiary car
SWT	An indicator variable that = 0 if the unit is on patrol in its sector = 1 if the unit is returning to sector from an outside call = 2 if the unit is responding to a call in sector = 3 if the unit is responding to a call out of sector = 4 if the unit is working on a call in sector = 5 if the unit is working on a call out of sector = 6 if the unit is out of service

Simulating the movement of patrol cars in this way makes the probability that a car is patrolling a given block equal to the conditional probability of a call for service at that block given that there is a call for service in the sector. This way busy blocks receive more patrol than less busy ones. In some cases it may be desirable to specify different patrol frequencies, e.g., if a block has few calls for service but those that do occur are very serious. This can be done by modifying the routine PLACE.CAR.ON. PATROL (see page 40). One could provide the simulation with a set of block patrol frequencies for each car or for the region being simulated. When a car is to return to patrol, the block to which it returns would be sampled at random according to the specified probability distribution.

The model assumes that patrol units travel from point to point as follows: If a patrol unit is currently at Block 1 with coordinates (x_1,y_1) and is dispatched to a job at Block 2 with coordinates (x_2,y_2), the simulation assumes that the unit travels parallel to the coordinate axes and that a pair of streets at right angles to each other pass through the points in question. Thus the program calculates the distance traveled as

$$d_{12} = | x_1 - x_2 | + | y_1 - y_2 |.$$

The time required to travel to Block 2 depends on the priority of the job, so that for a job of priority p we must specify a response velocity, v_p. The resulting response time is calculated as

$$t_{12} = d_{12}/v_p.$$

The simulation assumes that grid coordinates are assigned so that one grid coordinate unit represents one mile in each coordinate direction. If this is not the case, the user should supply velocities in terms of grid units per hour (instead of miles per hour) so that response times can be expressed in terms of minutes.

THE JOB STREAM

Jobs are created or collected outside the simulation and placed on a file called the job stream, which is "read and played out" by the simulation much in the same way that a tape recorder reads and plays a tape of music. As we have already pointed out, this procedure has several advantages

over generation of jobs within the simulation. There are two types of jobs included in the job stream: (1) calls for service, and (2) out-of-service jobs. The calls for service represent jobs carried out by patrolmen as part of their police functions. The out-of-service jobs represent functions other than preventive patrol carried out by patrolmen during a normal tour of duty. These include taking meal breaks, getting gas, having minor repairs made to the patrol car, etc. The out-of-service jobs are always associated with a specific patrol car.

Each call for service has the following information associated with it in the job stream:

- Job entry time: The "simulated clock" time at which the job enters the system.
- Priority: A number from 1 to 5 (1 is the highest priority).
- Job location: The internal identification number (from 1 to N.BLOCK) of the block at which the job occurs.
- Job duration: The length of time that the longest working patrol car spends at the job. (It should be noted that in the real world it is possible for the seriousness and duration of an incident to depend on the speed with which a patrol car gets to the scene. Since the nature of this dependency is not known, the simulation assumes that job duration and seriousness are unaffected by the speed of response. For comparing many types of deployment changes--particularly where the changes in response times are small--this should not be a serious limitation, but the user should be aware of this assumption in the model.)

Table 4 describes the attributes associated with every temporary entity JOB, which represents a call for service.

Each out-of-service job requires only two pieces of information:

- Patrol car: the internal identification number (from 1 to N.CAR) of the car to be placed out of service.
- Duration: the length of the out-of-service time.

The out-of-service jobs can be used to vary the number of patrol cars on duty over the course of the simulation. For example, if the calls for service represent calls over a whole day and if different numbers of cars

Table 4

ATTRIBUTES ASSOCIATED WITH THE ENTITY "JOB"

Attribute	Description
PRIORITY	A number from 1 to 5 indicating the priority (seriousness) of the call
LOCATION	The internal reference number of the block at which the job occurs
DURATION	The length of time that the longest working unit spends on the job
STATUS	An indicator variable that = 0 if no cars have arrived = 1 if the primary car has arrived = 2 if some cars, but not the primary car, have arrived
ASSIGNED.CARS	A list of the patrol cars assigned to the job

are on duty during each tour of the day, some cars could be put out of service for an entire tour at each tour-change time.

THE DISPATCHING ACTIVITY

The simulation imitates dispatching decisions, but does not simulate the dispatching activity per se. That is, it does not account for any time consumed during the dispatching function. For many applications, dispatching requires relatively little time and the dispatcher is not overloaded, so this model will be adequate. If, however, the dispatching-communication function is itself a potential source of delay, it would have to be modeled.

In the simulation, dispatching decisions are based on the location and priority of the job, and on the availability of patrol cars at the time of dispatch. A full description of how the simulation uses each of these characteristics in assigning cars is given in Section IV of this Report. The structure of the dispatching system we have coded is based on the computer-assisted dispatching system in use in New York City. We use the same type of information available to the New York City dispatcher and have rationalized the flexible rules and guidelines he follows. The simulation structure is general enough so that a variety of dispatching and

patrol policies quite different from New York City practice can be simulated by modifying only the program's initialization data. For example, different numbers of cars can be sent to different kinds of incidents; sector boundaries can be shifted or sectors can be eliminated entirely; certain calls can be held for specific cars.

Some dispatching policies, however, can be simulated only by changing the dispatching event routine (JOB.ENTRY). For example, the dispatching routine presented here does not make decisions based on <u>actual</u> car locations. It references all incoming jobs to a neighborhood, and dispatch decisions are made by referencing an ordered list of patrol units for that neighborhood that has been read in at the beginning of the simulation. Another version of the dispatching routine that we have used references an incoming call to its exact location and uses the exact locations of all patrol cars to determine the closest available cars to an incident. This version has been used to evaluate the desirability of car location devices.

IV. EVENTS AND SUBROUTINES

The simulation program is composed of six event routines and the subroutines that support them. The events are:

- JOB.ENTRY A call for police service is received at the dispatching center.
- ARRIVAL.AT.SCENE A patrol unit arrives at the scene of an incident.
- CALL.END A patrol unit finishes work at an incident or other activity.
- RETURN.TO.SECTOR A unit that has responded to an incident out of its sector has returned to patrol in its own sector.
- OUT.OF.SERVICE It is time for some unit to go out of service (for a meal or another nonservice-related purpose). If the unit is free, it goes out of service; if it is busy, the out-of-service period begins later.
- END.OF.SIMULATION The simulation is over and statistics are printed out.

The progress of an incident can be traced through the system as follows (names in capital letters that are not event names are names of subroutines): At a certain time (specified as part of the input job stream), the call is received by the dispatcher, causing the JOB.ENTRY event to be executed. Associated with the call are its location, priority, and duration. The program uses the dispatch policy for the job's priority class (DISPR1, DISPR2, DISPR3, DISPR4, or DISPR5) to decide which car (or cars) to send. Each car to be dispatched is ASSIGNed to the job. In the ASSIGN routine its distance from the job is determined by executing CALCULATE.DISTANCE, and the car's ARRIVAL.AT.SCENE is scheduled by SCHEDULE.CAR.ARRIVAL. If a car to be sent to the job is traveling back to its sector, the simulation first carries out a CANCEL.SECTOR.RETURN, then determines the car's current position with INTERPOLATE.LOCATION, and, finally, ASSIGNs it to the job. If a car to be sent to the job is currently responding to a lower priority job, the simulation will PREEMPT it, INTERPOLATE.LOCATION, and ASSIGN it to the higher priority job.

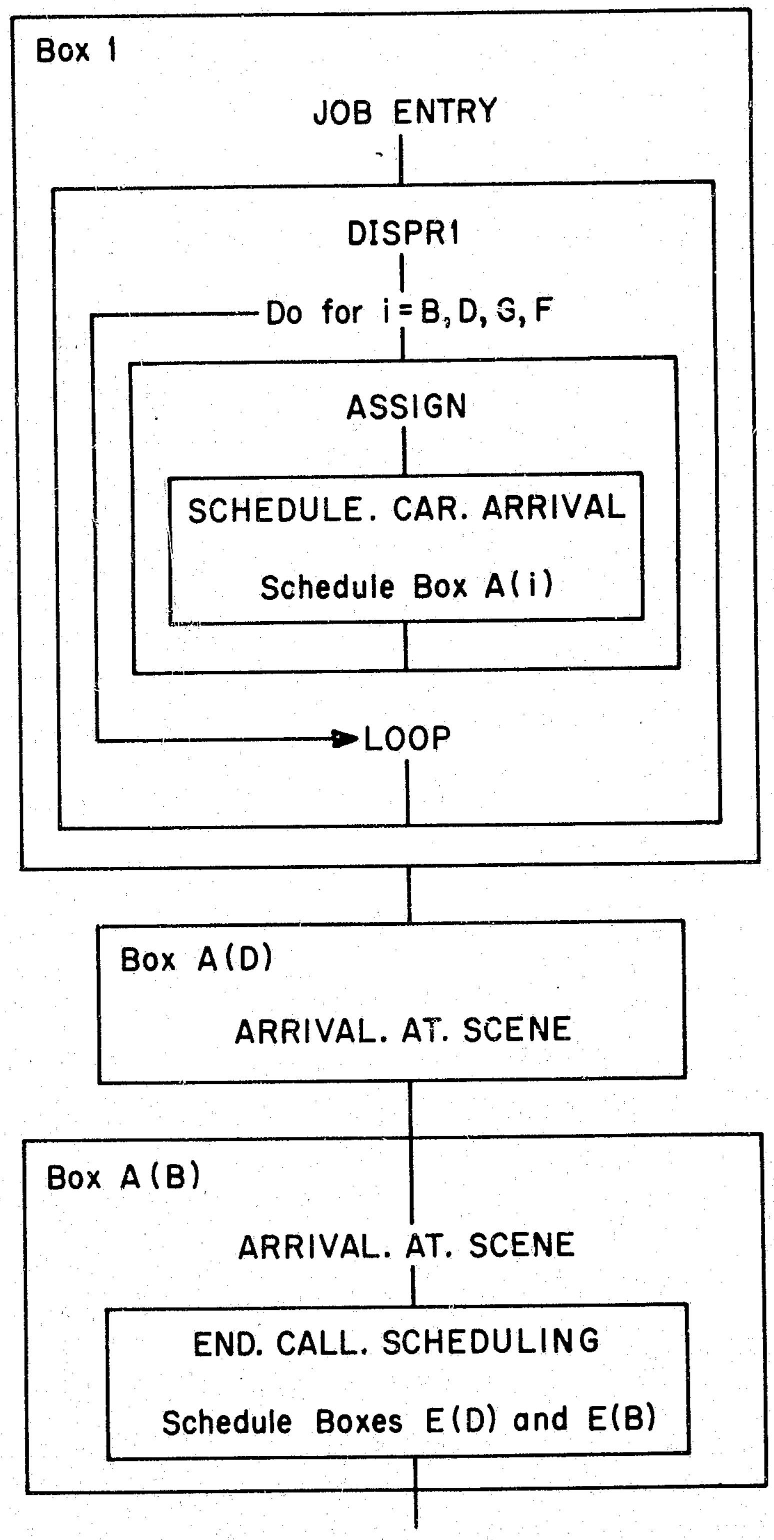
Each ARRIVAL.AT.SCENE causes a CALL.END to be scheduled for the arriving car at a time that depends on whether the car is the primary car at the job or a backup or tertiary car. When a car finishes a job, the simulated dispatcher checks the queues for another job. If a job is waiting for this car, the simulation dispatches the car to it with REASSIGN.CAR.TO.JOB. If no job is waiting that can be assigned to it, the simulation will PLACE.CAR.ON.PATROL. If currently within its sector, the car remains where it is; otherwise, it proceeds back to the centroid of its sector. In the latter case, a RETURN.TO.SECTOR event is scheduled.

Meals and other occurrences that place a car out of service are treated in much the same way as jobs. A car may be scheduled for its meal at any time during a tour (see input data section). The car will then be automatically scheduled for a meal at this (relative) time during each tour. When a car is scheduled to go out of service, an OUT.OF.SERVICE event occurs. If the car is available (on patrol), it begins its out-of-service time immediately; otherwise, the event is queued as jobs are. As soon as the car next becomes available, it will be placed out of service. The completion of an out-of-service period is treated exactly as if the car were completing a job.

In Fig. 2 we illustrate the sequence of events and the order of subroutine calls in the simulation by presenting a flow chart for an example of a single Priority 1 call. Each outer box in the flow chart denotes a simulated event, with time increasing as one proceeds down one page and onto the next. Small boxes inside the outer boxes denote subroutines called by the event routines. The subroutines may call other subroutines, which are shown in still smaller included boxes. Each outer box is labeled with an identifier (e.g., Box 1, Box A(D)) that may be used to refer to that specific event elsewhere in the flow chart.

In the example being simulated, the arrival of the call for service is simulated by the exogenous event JOB.ENTRY, which reads the description of the call from the input job stream. This information includes the fact that it is a Priority 1 incident. It also includes the location of the incident and the length of time it will take for the primary car to service it.

The JOB.ENTRY event calls subroutine DISPR1 to choose the patrol cars that will be dispatched to this incident. It determines that four cars, named B, D, G, and F will be sent, with B acting as primary car, D as backup car, and G and F as tertiary cars. Each car is sent to the incident through a separate call to the subroutine ASSIGN, which performs some bookkeeping functions and calls SCHEDULE.CAR.ARRIVAL to schedule the ARRIVAL.



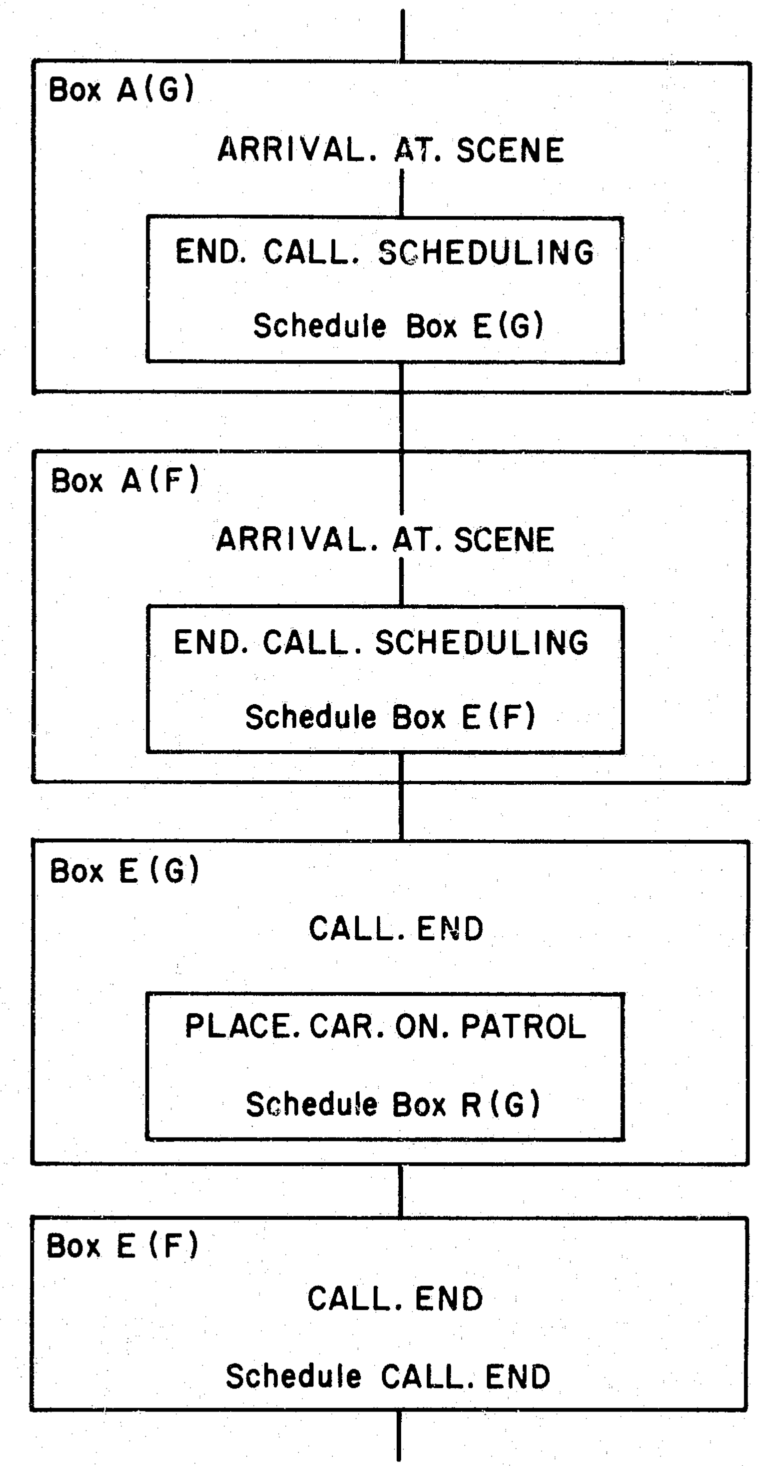
Exogenous event.
Read description
of call for service

DISPR1 selects cars B, D, G
and F to send.
Car B is primary car, D is
backup car, and G and F
are tertiary cars.

Car D (backup car)
arrives first.

Car B, the primary car, arrives.
Schedule end of call for all
cars on the scene.

Fig. 2—Flow chart for Priority 1 incident



Car G arrives.

Car F arrives.

Car G is sent back
to its own sector.

Car F is placed
out of service.

Fig. 2—(continued)

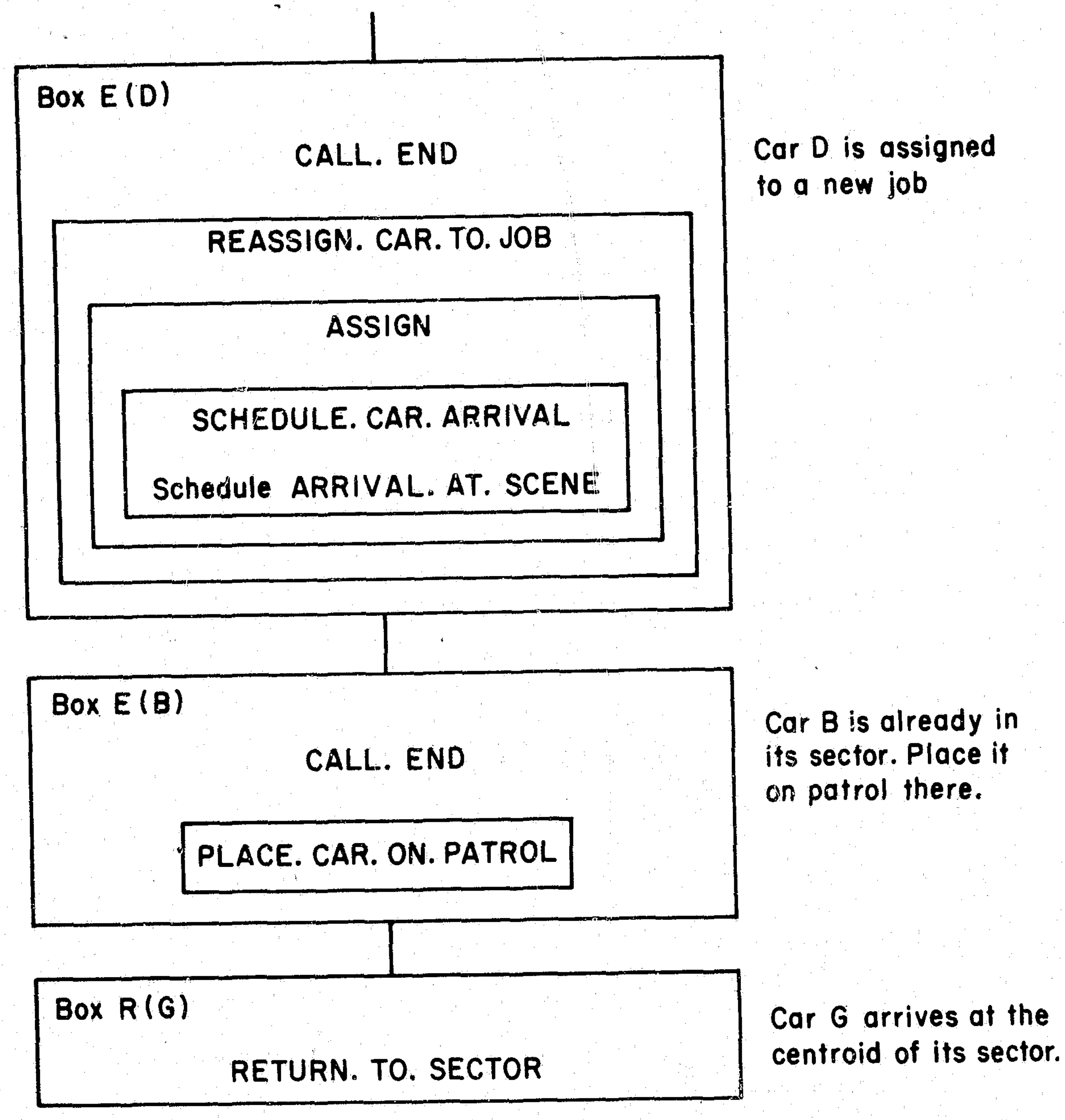


Fig. 2—(continued)

AT.SCENE event. The time required for each car to travel to the incident depends on the distance to be traveled and on the travel velocity.

Car D is the first car to arrive. Car B, the primary car, arrives next. Its arrival allows the simulation to schedule CALL.ENDs for all cars already on the scene (in this case only car D). Cars G and F arrive subsequently and their CALL.ENDs are scheduled as soon as they arrive.

Cars F and G, the tertiary cars, are the first to leave the scene. Car G, which has responded out of its sector, is placed on patrol and sent back to its sector (a RETURN.TO.SECTOR event is scheduled for it in Box E(G)). There is an out-of-service job waiting for car F, so it is placed out of service. When Car D, the backup car, is finished at the incident there is another job waiting for it. It is, therefore, dispatched to the new job. Car B, which is already in its sector, is placed on patrol at the scene of the incident after it completes its work there.

The flow chart ends with the arrival of car G at the centroid of its sector where it remains on preventive patrol. Of course, in an actual simulation run, additional events associated with other incidents would be interspersed among the events shown.

A complete description of each of the event routines and their supporting subroutines is given below.

MAIN

Every SIMSCRIPT II program must contain a routine called "MAIN." On each simulation run, the program execution begins at the first instruction (line 32000 in our program) in this MAIN routine and proceeds from there. In MAIN, the computer reads in the initialization deck, sets the dimensions for all arrays, and sets initial values for variables and arrays. It also prints a report displaying most of the initialization data. Before passing control to the internal SIMSCRIPT timing routine that controls the execution of the events, the program schedules the first meal time for each of the patrol cars, and then schedules the end of the simulation. The timing routine is part of the SIMSCRIPT compiler, and so is not included in our program documentation. The MAIN routine is listed below.

```
  MAIN
    DEFINE SIM.LENGTH AND MEAL.TIME AS REAL VARIABLES
 RESERVE REGION.NAME(*) AND TITLE(*) AS 20
  START NEW PAGE
 FOR I = 1 TO 20, READ TITLE(I) AS (20) A 4
 FOR I = 1 TO 20, READ REGION.NAME(I) AS (20) A 4
 READ SIM.LENGTH, MAX.SENT, MEAL.DURATION, AND TOUR.LENGTH
 WRITE AS " INPUT DATA FOR "
 FOR I= 1 TO 20, WRITE REGION.NAME(I) AS     A 4
 WRITE AS /
 SKIP 3 LINES
 PRINT 1 LINE WITH SIM.LENGTH AS FOLLOWS
 SIMULATION LENGTH =******.** HOURS
 SKIP 1 LINE
  READ N.BLOCK,N.NBD,N.CAR
    LET N.AVAILABLE=N.CAR
 PRINT 1 LINE WITH N.BLOCK,N.NBD AND N.CAR AS FOLLOWS
NO. OF BLOCKS = ****   NO. OF NBDS. = ****    NO. OF CARS = ****
  RESERVE ADJACENT.CARS(*,*) AS N.NBD BY N.CAR,AND SECTORS(*,*)
      AS N.CAR BY *
 CREATE EVERY BLOCK, NBD, CAR, AND P.CLASS(5)
 SKIP 1 OUTPUT LINE
 PRINT 1 LINE AS FOLLOWS
BLOCK ID   TAX NO.   NBD ID          X          Y
 FOR I=1 TO N.BLOCK, DO
           READ B,TAXNO(B),NBDID(B),XCORD(B),YCORD(B) USING UNIT 7
 PRINT 1 LINE WITH B,TAXNO(B),NBDID(B),XCORD(B),YCORD(B) THUS
 ****     *******     ****        **.*****  **.*****
  LOOP
    START NEW PAGE
 FOR I= 1 TO 20 , WRITE TITLE(I) AS    A 4
 WRITE AS /
 SKIP 2 LINES
 PRINT 2 LINES AS FOLLOWS
NBD ID      SECTORS         NUMBER OF         ORDER IN WHICH
                      SECT.CARS ADJ.CARS  CARS ARE NOMINATED
  FOR I=1 TO N.NBD, DO
      READ N,NBD.NAME(N),N.SECTOR.CARS(N),N.ADJACENT(N)
      FOR J=1 TO N.CAR,READ ADJACENT.CARS(N,J)
 PRINT 1 LINE WITH N,NBD.NAME(N),N.SECTOR.CARS(N),
  N.ADJACENT(N) AS FOLLOWS
 ****       ****          **       ***
       FOR K=1 TO N.CAR,DO PRINT 1 LINE WITH
 ADJACENT.CARS(I,K) AS FOLLOWS
                                                 ***
       LOOP
     LOOP
 FOR I=1 TO N.CAR    DO
 READ C,CAR.NAME(C),NUMB.SECTORS(C),CENTROID(C),MEAL.TIME
 RESERVE SECTORS(C,*) AS NUMB.SECTORS(C)
 FOR J=1 TO NUMB.SECTORS(C),  READ SECTORS(C,J)
      LET XLOC(C)=XCORD(CENTROID(C))
      LET YLOC(C)=YCORD(CENTROID(C))
  SCHEDULE AN OUT.OF.SERVICE GIVEN C IN MEAL.TIME HOURS
   LOOP
     FOR I=1 TO N.P.CLASS READ VELOCITY(I)
     RESERVE P.DURATION(*) AS 3
     FOR I = 1 TO 3 READ P.DURATION(I)
   CREATE AN END.OF.SIMULATION
  SCHEDULE THIS END.OF.SIMULATION IN SIM.LENGTH HOURS
   START SIMULATION
   STOP
   END
```


EVENT FOR JOB.ENTRY

This external event represents the arrival of a new incident into the system. For most types of jobs (all priorities except 4), the job arrives at the dispatching center. It is presumed that at this point the dispatcher has already obtained all information necessary for dispatch and can immediately assign a patrol car if one is available. If there are delays due to the dispatching operation itself, it will be necessary to have an event or events that can account for the dispatching delays. This might involve explicit modeling of the dispatcher's operation, including the modeling of delays that may occur when the citizen reporting the incident attempts to make telephone contact with the police, or while the police are obtaining the relevant information from the caller. We describe the simulated dispatching operation below in general terms. Note that Priority 4 jobs are intended to model those incidents that are discovered by the patrolling units themselves and that are not first reported to the police by telephone. Patrol cars are not, of course, dispatched to these jobs. Rather, the job is "picked up" by the appropriate unit. Yet in the following discussion we use the term "dispatch" somewhat loosely and incorporate under it the assignment of cars to Priority 4 jobs. And as a modeling convenience, we also place pickup jobs in a fictional queue where they "wait" for the appropriate patrol car to pick them up.

Data describing the new incident--its location, priority, and duration--are included as part of the input job stream. In order to determine whether there are patrol resources available to dispatch to the call and, if so, which resources to dispatch, the JOB.ENTRY event routine calls the appropriate dispatch subroutine based on the priority of the call for service. The dispatch subroutines in the version of the program documented here are called DISPR1, DISPR2, DISPR3, DISPR4, and DISPR5, corresponding to the five priority classes. The dispatch policy for each priority class is given in the description of each dispatch subroutine.

A general flow chart for the JOB.ENTRY event, including all the subroutines called by the various dispatch subroutines, is given in Fig. 3.

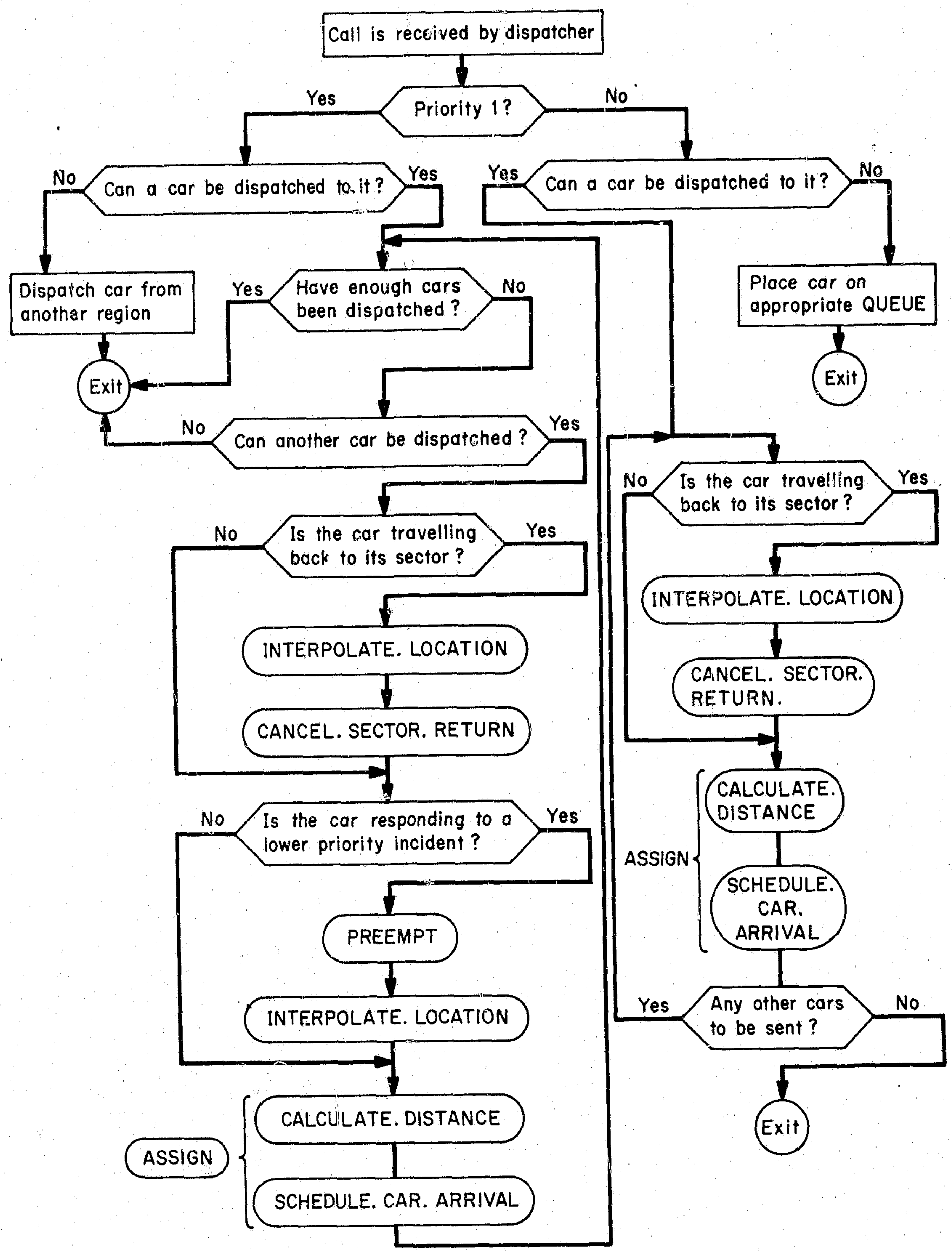


Fig. 3—JOB.ENTRY


```
EVENT FOR JOB.ENTRY                                          00059800
    DEFINE HOUR AND MOUR AS REAL VARIABLES                   00059900
    READ LOC,IPR,HOUR AND MOUR                               00060000
       CREATE A JOB CALLED J                                 00060100
       LET ENTRY.TIME(J)=TIME.V                              00060200
       LET LOCATION(J)=LOC                                   00060300
       LET PRIORITY(J)=IPR                                   00060400
    LET DURATION(J)=HOUR*60.0 + MOUR                         00060500
       LET STATUS(J)=0                                       00060600
       GO TO PR(IPR)                                         00060700
 'PR(1)'   CALL DISPR1(J)      RETURN                        00060800
 'PR(2)'   CALL DISPR2(J)      RETURN                        00060900
 'PR(3)'   CALL DISPR3(J)      RETURN                        00061000
 'PR(4)'   CALL DISPR4(J)      RETURN                        00061100
 'PR(5)'   CALL DISPR5(J)      RETURN                        00061200
       END                                                   00061300
```

Dispatch Subroutines: DISPR1, DISPR2, DISPR3, DISPR4, DISPR5

These subroutines are the heart of the simulation. With its five dispatching procedures, the simulation is capable of testing a wide range of dispatching alternatives by changing only the priorities assigned to jobs or the sector assignments of the patrol cars. If the user would like to test a policy that cannot be handled in this way, he can write his own dispatching routine(s) and substitute it for one or more of the five discussed here.

In all five policies the incoming jobs are referenced to a neighborhood, and dispatch decisions are made using an ordered list of patrol units associated with that neighborhood. The list consists of all patrol units that are to be considered possible candidates to be assigned to a job located in the neighborhood in question. Whether the unit is assigned, and in what capacity it will be assigned, depends upon the current status of the unit and of other units, upon the priority of the job, and upon the values given to various dispatch parameters. (Since neighborhoods can be as small as a single block or as large as the entire region being simulated, great flexibility is possible in the dispatching strategies that can be simulated.)

The following discussion of each of the dispatching procedures in the simulation depends on an understanding of the ordered list of patrol units. There is one such list for every neighborhood. For a particular neighborhood, say I, it is called ADJACENT.CARS(I). At the head of the list is the sector car(s) for the neighborhood, next come the sector cars for adjoining neighborhoods, then other cars in the region, and finally supervisory or special cars, if there are any. As will be seen in the following discussion, sector cars, neighboring sector cars, etc., are treated differently in dispatching. Associated with the list ADJACENT.CARS(I) are two pointers that are used in the dispatching routines:

- N.SECTOR.CARS(I) tells how many sector cars are assigned to Neighborhood I.
- N.ADJACENT(I) tells how many "adjacent sector cars" are associated with Neighborhood I.

Figure 4 is a representation of the ordered list for Neighborhood I and its members. We assume for illustration that there are six patrol units in the field, numbered 1 through 6, and one supervisor's car, numbered 7. There is one sector car for Neighborhood I, namely car 4; and there are two adjacent cars, numbered 3 and 1. Thus, N.SECTOR.CARS(I) = 1 and N.ADJACENT(I) = 2, and as shown schematically in Fig. 3 these point to appropriate cars on the list.

ADJACENT.CARS(I)	
car 4	N.SECTOR.CARS(I) = 1
car 3	
car 1	N.ADJACENT(I) = 2
car 2	
car 5	
car 6	
car 7	

Fig. 4—The structure of the dispatcher's list ADJACENT.CARS(I)

When a job in Neighborhood I comes into the system, the car(s) dispatched depends on the priority of the job and on the ordered list ADJACENT.CARS(I). A description of each of the five dispatch subroutines follows. As mentioned above, each of the subroutines calls the ASSIGN routine to assign a car to a job. If a car to be sent to a job is returning to its sector, the routines CANCEL.SECTOR.RETURN and INTERPOLATE.LOCATION are called before the ASSIGN routine is executed.

ROUTINE TO DISPR1 GIVEN J. Priority 1 jobs are the most serious incidents. They generally include robberies in progress, police officers in

danger, etc. In DISPR1, the simulation dispatches the first MAX.SENT of the available cars on ADJACENT.CARS(I).* If fewer than MAX.SENT are available, all of the available cars are dispatched. Those cars dispatched to the job are all associated with the internal identification number assigned to the job (variable J). Cars are available if they are on patrol or responding to lower priority incidents. The preempting of cars that are responding to lower priority calls in order to respond to Priority 1 jobs is the only preempting included in the simulation. Starting from the top of ADJACENT.CARS(I), the available cars are assigned as follows:

- A primary car (the first available car on the list)--the job is assumed to start upon its arrival at the scene and it works for the entire duration.
- A backup car (the second available car on the list)--works for a proportion, P.DURATION(2), of the duration of the job, starting work upon the arrival of the primary car.
- Tertiary cars (all other available cars on the list up to a total of MAX.SENT-2)--each of these work for a proportion, P.DURATION(3), of the duration of the job, starting work upon the arrival of the primary car.

If no cars are available it is assumed that the job is handled by units from another region. Where this occurs, it is recorded by adding "1" to the variable OUTSIDE.DISPATCH and the job is not considered further by the simulation. The number of times this happens is shown in the output report "Number of Precinct Cars Sent to Calls" (Fig. 16). Thus, the simulation never queues Priority 1 jobs.

```
ROUTINE TO DISPR1 GIVEN J
LET SECT=NBDID(LOCATION(J))                    LET MARKER=0
    FOR I=1 TO N.CAR,DO ''NOMINATE CARS ''
             LET K=ADJACENT.CARS(SECT,I)
          IF SWT(K) LE 3 AND SPT(K) NE 1  ''CAR IS ELIGIBLE ''
          IF MARKER=0 LET SOT(K)=1
                      LET MARKER=1
                      GO TO CALC1
          OTHFRWISE   IF MARKER = 1 LET SOT(K)=2
                                    LET MARKER=2
                                    GO TO CALC1
```

*MAX.SENT is a variable whose value is specified by the user. It must be less than or equal to N.CAR.

```
              OTHERWISE IF MARKER = MAX.SENT GO OUT
              OTHERWISE ADD 1 TO MARKER LET SOT(K)=3
'CALC1' IF SWT(K)=1 CALL INTERPOLATE.LOCATION GIVEN K AND ASSIGNMENT(K)
        REGARDLESS
            IF SPT(K) GT 1 CALL PREEMPT(K)
            REGARDLESS CALL ASSIGN(K,J)  LET SWT(K)=2
                IF I GT N.SECTOR.CARS(SECT) LET SWT(K)=3
                REGARDLESS
            REGARDLESS
        LOOP
'OUT'  IF MARKER = 0 ADD 1 TO OUTSIDE.DISPATCH
                       DESTROY JOB CALLED J
          REGARDLESS LET COUNT(1)=MARKER
        RETURN
    END
```

ROUTINE TO DISPR2 GIVEN J. Priority 2 jobs are less serious than Priority 1 jobs. Either one or two patrol cars are assigned to Priority 2 jobs, as follows:

- If no cars are available in the region being simulated, the job is queued for the first available car. Jobs in the Priority 2 queue get assigned units on a first-in-first-out basis before jobs in other queues. A job that is queued gets assigned only one car.
- One car is sent unless a sector car is available to act as primary car and another sector car or one of the N.ADJACENT cars from the neighboring sectors is available. In this case, both a primary and backup car are sent. The primary car works for the duration of the job and the backup car works for a proportion, P.DURATION(2), of the duration. Work starts upon the arrival of the primary car.

```
ROUTINE TO DISPR2 GIVEN J
LET SECT=NBOID(LOCATION(J))          LET MARKER=0
FOR I=1 TO N.CAR, DO        ''NOMINATE CARS ''
    LET K=ADJACENT.CARS(SECT,I)
      IF SPT(K)=0        '' CAR IS ELIGIBLE ''
          IF MARKER=0 LET SOT(K)=1  LET MARKER=1 GO TO CALC2
          OTHERWISE
         IF MARKER=1 AND I LE N.ADJACENT(SECT)+N.SECTOR.CARS(SECT)
          LET SOT(K)=2 LET MARKER=2
'CALC2'        IF SWT(K)=1 CALL INTERPOLATE.LOCATION GIVEN K AND
               ASSIGNMENT(K)
               REGARDLESS CALL ASSIGN(K,J) LET SWT(K)=2
               IF I GT N.SECTOR.CARS(SECT) LET SWT(K)=3
               REGARDLESS
          REGARDLESS
    REGARDLESS
     LOOP
   IF MARKER=0 FILE J IN QUEUE(2)    RETURN
  OTHERWISE LET COUNT(2)=MARKER RETURN
END
```


ROUTINE TO DISPR3 GIVEN J. Exactly one car is assigned to Priority 3 jobs. This car can be any available car in the region being simulated. The car dispatched is the first available car on the ADJACENT.CARS list associated with the neighborhood from which the call emanated. If no cars in the region are available the job is queued for the first available car. Priority 3 jobs are dispatched only if the Priority 2 queue (QUEUE(2)) is empty.

```
ROUTINE TO DISPR3 GIVEN J
'' SEND ANY AVAILABLE CAR ''
LET SECT=NBOID(LOCATION(J))
FOR I=1 TO N.CAR ,  DO
     LET K=ADJACENT.CARS(SECT,I)
     IF SPT(K)=0      ''CAR IS ELIGIBLE''
          LET SOT(K)=1
          IF SWT(K)=1 CALL INTERPOLATE.LOCATION GIVEN K AND
               ASSIGNMENT(K)
               REGARDLESS CALL ASSIGN(K,J)
               LET SWT(K)=2
               IF I GT N.SECTOR.CARS(SECT) LET SWT(K)=3
               REGARDLESS LET COUNT(3)=1  RETURN
          OTHERWISE LOOP
     FILE J IN QUEUE(3)
     RETURN
     END
```

ROUTINE TO DISPR4 GIVEN J. These jobs represent "pickup" jobs--incidents spotted by patrolmen during the course of their preventive patrol activities. As is the case for Priority 3 jobs, exactly one car is assigned. But the car assigned will always be a sector car whose area of responsibility includes the block where this incident occurs. Since the car is already at the location of the incident when it occurs, it does not have to travel to reach the scene. To imitate this, the simulation instantaneously changes the car's location to the grid coordinates associated with the pickup job.

If the appropriate sector car is available when the Priority 4 job is received, it will be immediately assigned to the job. If no sector car is available, there are two possible courses of action: (1) the call is queued until a sector car is available or (2) the call is ignored (since no car was available at that instant to spot the incident). The former approach is taken in this simulation program. This approach is appropriate when the calls for service in the job stream represent an actual historical sequence of calls. In this case, the fact that a specific car was unavailable when a pickup job occurred indicates a difference between the real and simulated worlds. But elimination of the job would result in changing the workload of the cars on duty and would lead to difficulties in comparing the simulation results produced for the same job stream by different deployment policies.

However, there is an alternative way that pickup jobs can be inserted into the job stream. They may be generated randomly according to the (known) frequency with which a car on preventive patrol in a region would encounter pickup jobs. In this case, if a Priority 4 call arrives when the sector car is busy, the call should be ignored. As a result, the more time a car has for preventive patrol, the more often it will service pickup jobs. Similarly, a busy car will have correspondingly fewer pickup jobs. In order to modify the simulation to implement this approach to Priority 4 jobs, the following statement should be substituted for statement number 00069100 in ROUTINE TO DISPR4:

DESTROY JOB CALLED J.

(Even in this case there will be difficulties in comparing the simulation results produced for the same job stream by different deployment policies.)

```
00067700 ROUTINE TO DISPR4 GIVEN J
00067800      '' PICK-UP JOB ''
00067900 LET SECT = NBDID(LOCATION(J))
00068000 FOR I = 1 TO N.SECTOR.CARS(SECT), DO
00068100      LET K = ADJACENT.CARS(SECT,I)
00068200      IF SPT(K) = 0      ''CAR IS AVAILABLE''
00068300           LET XLOC(K) = XCORD(LOCATION(J))
00068400           LET YLOC(K) = YCORD(LOCATION(J))
00068500           CALL ASSIGN(K,J)
00068600           LET SWT(K) = 2      LET SOT(K) = 1
00068700           LET COUNT(4) = 1
00068800           RETURN
00068900      OTHERWISE
00069000 LOOP
00069100 FILE J IN QUEUE(4)        ''NO SECTOR CAR FREE''
00069200 RETURN
00069300 END
00069400
```

ROUTINE TO DISPR5 GIVEN J. Priority 5 jobs are similar to Priority 4 jobs, but no instantaneous travel is assumed. These jobs are assigned only to a sector car for the neighborhood associated with the job. This class is meant to imitate low priority jobs that may be assigned by a supervisor, such as taking reports on past burglaries, etc. If no sector car is available the job is placed on QUEUE(5). A Priority 5 job is assigned only when the Priority 2 and Priority 3 queues are empty and QUEUE(4) does not contain a job waiting for the same sector car(s).


```
ROUTINE TO DISPR5 GIVEN J
'' ASSIGN JOB ONLY TO SECTOR CAR ''
LET SECT=NBDID(LOCATION(J))
FOR I= 1 TO N.SECTOR.CARS(SECT),  DO    ''LOOK FOR A SECTOR CAR''
      LET K=ADJACENT.CARS(SECT,I)
      IF SPT(K)=0      ''CAR IS ELIGIBLE''
           IF SWT(K)=1 CALL INTERPOLATE.LOCATION GIVEN K AND
                     ASSIGNMENT(K)
                REGARDLESS CALL ASSIGN(K,J) LET SWT(K)=2
         LET SOT(K)=1     LET COUNT(5)=1    RETURN
           OTHERWISE
      LOOP
   FILE J IN QUEUE(5)         ''NO SECTOR CAR FREE''
   RETURN
END
```

ROUTINE TO INTERPOLATE.LOCATION GIVEN K AND J

This routine is called by each of the dispatch subroutines whenever the car to be dispatched is en route to a different location (either returning to its sector from a previous job or responding to a lower priority job). It calculates the approximate current position of CAR K, which is traveling to JOB J (the "job" might be a return to patrol duty), and assigns the x and y coordinates of CAR K's current simulated position to the variables XLOC(K) and YLOC(K). The routine makes the assumption that a car travels only in right-angle directions during the course of a trip--first traveling on a straight path in the x direction, then on a straight path in the y direction. The total distance already traveled by the car is assumed to be equal to the fraction of the total duration of the trip that has already elapsed.

If CAR K had been on its way to its sector, the event RETURN.TO.SECTOR, which had already been scheduled for this car, is now canceled by calling the CANCEL.SECTOR.RETURN routine.

```
ROUTINE TO INTERPOLATE.LOCATION GIVEN K AND J
   '' CALLED FROM PREEMPT AND FROM JOB.ENTRY ''
NORMALLY MODE IS REAL
DEFINE LJ,K,J,R,PATROL.RETURN AS INTEGER VARIABLES
LET LJ=LOCATION(J)
LET V=VELOCITY(PRIORITY(J))
LET XD=XCORD(LJ)-XLOC(K)
LET YD=YCORD(LJ) - YLOC(K)
LET T=(TIME.V - T.START.RESPONSE(K))*24
LET TX=ABS.F(XD)/V
LET DELT=T - TX
IF DELT GT 0 LET XLOC(K)=XCORD(LJ)
             LET YLOC(K)=V*DELT*SIGN.F(YD)+YLOC(K)
   GO TO CHECK.IF.PATROL
OTHERWISE LET XLOC(K)=V*(-DELT)*SIGN.F(XD)+XLOC(K)
```

```
'CHECK.IF.PATROL'  IF SWT(K) NE 1 RETURN
                   OTHERWISE CALL CANCEL.SECTOR.RETURN GIVEN K
                   RETURN
    END
```

ROUTINE TO CANCEL.SECTOR.RETURN GIVEN K

This subroutine is called when CAR K, which is returning to its sector, is dispatched to a new job before it reaches its sector. It cancels the event RETURN.TO.SECTOR, which had already been scheduled for this car.

```
 ROUTINE TO CANCEL.SECTOR.RETURN GIVEN K
 LET PATROL.RETURN = ASSIGNMENT(K)
  DESTROY JOB CALLED PATROL.RETURN
 LET R=NEXT.EVENT(K)
 CANCEL RETURN.TO.SECTOR CALLED R
DESTROY RETURN.TO.SECTOR CALLED R
 RETURN
 END
```

ROUTINE TO PREEMPT(K)

If a car is responding to a lower priority incident when the dispatcher receives a Priority 1 call, this routine redirects it to the Priority 1 incident. It cancels CAR K's ARRIVAL.AT.SCENE for the lower priority incident and removes it from the list of ASSIGNED.CARS for that incident.

If no car has yet arrived at the lower priority incident, it places the incident on the appropriate QUEUE for redispatch. Otherwise, it readjusts the assignments of the cars at the lower priority incident and, if the primary car is the one that has been preempted, it reschedules the end of the incident.

```
ROUTINE TO PREEMPT(K)
LET J = ASSIGNMENT(K)
IF J IS NOT IN A QUEUE
    IF STATUS(J)=0 FILE J FIRST IN QUEUE(PRIORITY(J))
    REGARDLESS
REGARDLESS LET E= NEXT.EVENT(K) CANCEL THE ARRIVAL.AT.SCENE CALLED E
DESTROY THE ARRIVAL.AT.SCENE CALLED E
REMOVE K FROM ASSIGNED.CARS(J)  LET SPT(K)=0
NOW INTERPOLATE.LOCATION GIVEN K AND J
IF N.ASSIGNED.CARS(J) =1 LET F=F.ASSIGNED.CARS(J)
    IF SOT(F)=2 AND SWT(F) GE 4
    LET E = NEXT.EVENT(F)
    CANCEL THE CALL.END CALLED E
    RESCHEDULE THE CALL.END CALLED E IN DURATION(J) MINUTES
 REGARDLESS
    LET SOT(F)=1
REGARDLESS
LET T.START.RESPONSE(K)= TIME.V
RETURN
END
```


ROUTINE TO ASSIGN(K,J)

This routine assigns CAR K to work on JOB J. It assigns the car the priority associated with the job and places the car on the list of cars assigned to this job (ASSIGNED.CARS(J)). It also schedules the time at which the car will arrive at the scene by calling the routine SCHEDULE.CAR. ARRIVAL(K,J).

```
ROUTINE TO ASSIGN(K,J)
LET SPT(K)=PRIORITY(J)
LET ASSIGNMENT(K)=J
NOW SCHEDULE.CAR.ARRIVAL(K,J)
FILE     K  IN ASSIGNED.CARS(J)
   IF SWT(K) LE 1 SUBTRACT 1 FROM N.AVAILABLE REGARDLESS
RETURN
END
```

ROUTINE TO SCHEDULE CAR.ARRIVAL(K,J)

This routine, called from the ASSIGN subroutine, schedules the arrival of CAR K at the location of JOB J. The function CALCULATE.DISTANCE is used to calculate the distance to be traveled. It is assumed that the car travels from its current location at a constant velocity that is determined by the priority of the job. The event that ends the trip is ARRIVAL.AT.SCENE. The identity of this event is stored in the attribute NEXT.EVENT(K), and the current time is stored in T.START.RESPONSE(K).

```
ROUTINE TO SCHEDULE.CAR.ARRIVAL(K,J)
DEFINE DIST AS A REAL VARIABLE
LET DIST=CALCULATE.DISTANCE(XCORD(LOCATION(J)),YCORD(LOCATION(J)),
XLOC(K),YLOC(K))
LET T.START.RESPONSE(K)=TIME.V
CREATE AN ARRIVAL.AT.SCENE
SCHEDULE THIS ARRIVAL.AT.SCENE GIVEN K IN
DIST/VELOCITY(PRIORITY(J)) HOURS
LET NEXT.EVENT(K)=ARRIVAL.AT.SCENE
RETURN
END
```

ROUTINE TO CALCULATE.DISTANCE GIVEN X1, Y1, X2 AND Y2

This routine is used to calculate the distance, D, between a point with grid coordinates (X1, Y1) and a point with grid coordinates (X2, Y2). The function may be easily changed to suit the city being simulated. The version of the routine that is reproduced here calculates the right angle distance, as follows:

$$D = | X2 - X1 | + | Y2 - Y1 |.$$

This equation assumes that a dense rectangular street network exists, connecting all pairs of points. For some of our experiments we found it necessary--and easy--to modify the function to model the effects of natural or man-made barriers such as large bodies of water or railroad tracks.

```
ROUTINE TO CALCULATE.DISTANCE GIVEN X1,Y1,X2 AND Y2
DEFINE D,X1,X2,Y1, AND Y2 AS REAL VARIABLES
LET D = ABS.F(X2-X1) + ABS.F(Y2-Y1)
RETURN WITH D
END
```

EVENT FOR ARRIVAL.AT.SCENE(K)

This internal event, which is flow-charted in Fig. 5, represents the arrival of CAR K at the scene of an incident. The first statements in the routine change the state of the system to reflect this event. What follows depends on whether the car has been assigned to be the primary car for the incident or one of the backup cars. If it is the primary car and no other car has arrived, its CALL.END event is scheduled. If other cars have arrived, CALL.END events are scheduled for each of these cars (since the duration of a job is measured from the time of arrival of the primary car). If the car is a backup car and the primary car is already there, then a CALL.END is scheduled for the car. Otherwise, it must wait for the arrival of the primary car before its CALL.END can be scheduled. Finally, if this is the first car to arrive at the scene, its travel time is calculated and added to both the overall travel time statistics and the travel time statistics corresponding to the priority of this job.

```
    EVENT FOR ARRIVAL.AT.SCENE(K)
    DEFINE D AS A REAL VARIABLE
    LET LKAR=LOCATION(ASSIGNMENT(K))
    LET XLOC(K)=XCORD(LKAR)
    LET YLOC(K)=YCORD(LKAR)
    IF SWT(K)=2   LET SWT(K)=4 ADD 1 TO IN.SECT.JOBS(K) GO TO BOTH
    OTHERWISE     LET SWT(K)=5 ADD 1 TO OUT.SECT.JOBS(K)
'BOTH'
    LET J=ASSIGNMENT(K)
    LET JPR=PRIORITY(J)
    LET D = DURATION(J)
    IF SOT(K)=1 ''THIS IS THE PRIMARY CAR AT JOB J'' GO  TO PRIME.CAR
    OTHERWISE
    IF STATUS(J)=0  ''NO CAR HAS ARRIVED AND THIS CAR IS NOT PRIMARY''
    GO TO FIRST.CAR
    OTHERWISE
    IF STATUS(J)=1  '' PRIMARY CAR HAS ALREADY ARRIVED  ''
    PERFORM END.CALL.SCHEDULING GIVEN K AND D
    REGARDLESS RETURN '' IF NO PRIMARY CAR DELAY SCHEDULING A CALL.END''
'FIRST.CAR'
```

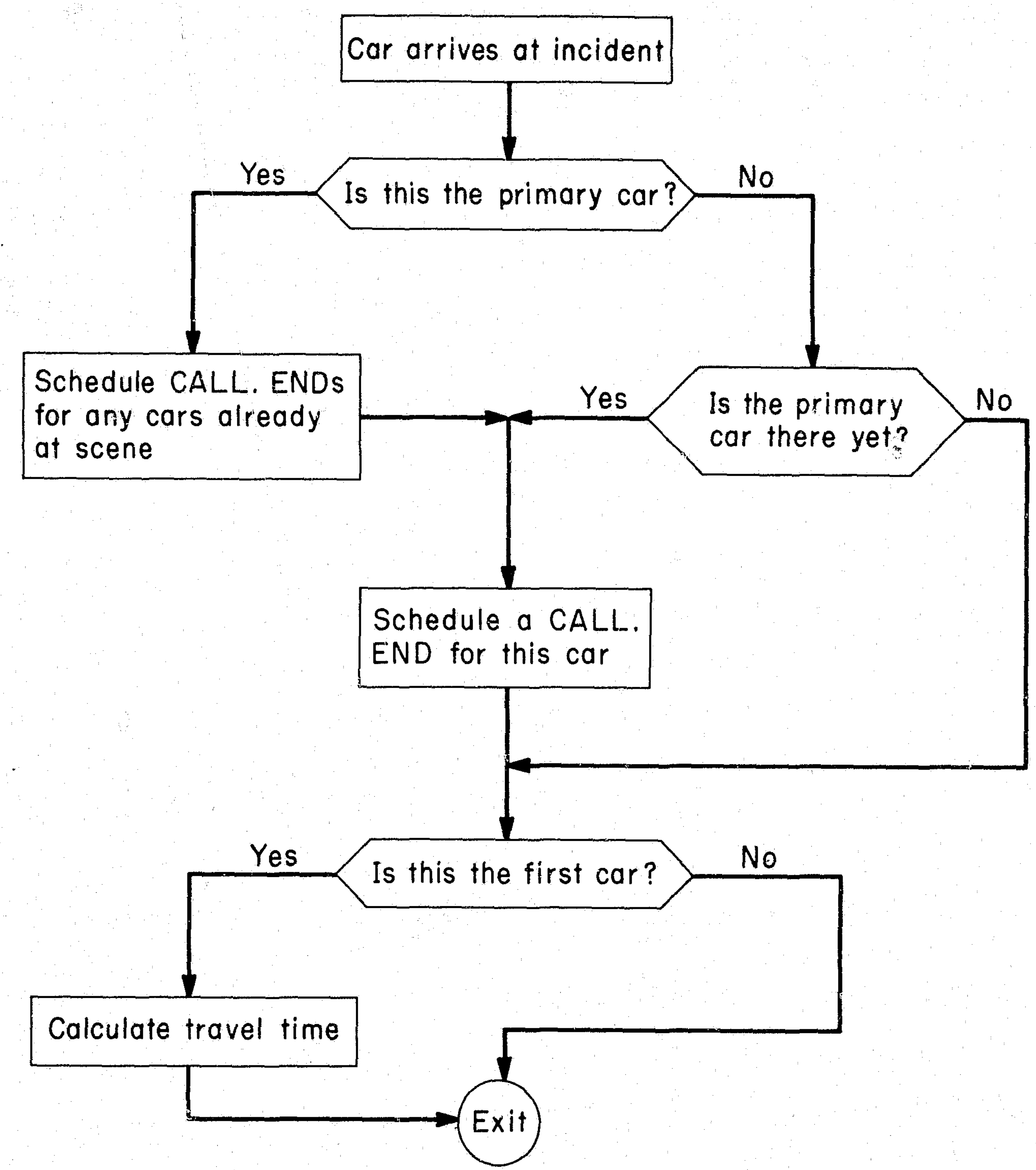



Fig. 5—ARRIVAL.AT.SCENE

```
   LET STATUS(J)=2
   LET JOB.RESPONSE.TIME(JPR)=(TIME.V -
    T.START.RESPONSE(K))*1440.0
   GO TO RT(JPR)
'RT(1)' LET P1.NBD.RSPONSE(NBDID(LKAR))=JOB.RESPONSE.TIME(JPR) RETURN
'RT(2)' LET P2.NBD.RSPONSE(NBDID(LKAR))=JOB.RESPONSE.TIME(JPR) RETURN
'RT(3)' LET P3.NBD.RSPONSE(NBDID(LKAR))=JOB.RESPONSE.TIME(JPR) RETURN
'RT(4)' LET P4.NBD.RSPONSE(NBDID(LKAR))=JOB.RESPONSE.TIME(JPR) RETURN
'RT(5)' LET P5.NBD.RSPONSE(NBDID(LKAR))=JOB.RESPONSE.TIME(JPR) RETURN
'PRIME.CAR'
    IF STATUS(J) NE 0 '' SOME CARS ALREADY THERE''
          FOR EACH CAR IN ASSIGNED.CARS(J) WITH SWT(CAR) GE 4  DO
          PERFORM END.CALL.SCHEDULING GIVEN CAR AND D
          LOOP  LET STATUS(J)=1     RETURN
    OTHERWISE
    LET JOB.RESPONSE.TIME(JPR)=(TIME.V -
     T.START.RESPONSE(K))*1440.0
    CALL END.CALL.SCHEDULING GIVEN K AND D
    LET STATUS(J)=1
    GO TO RT(JPR)
    END
```

ROUTINE FOR END.CALL.SCHEDULING GIVEN K AND DUR

This routine, called from the ARRIVAL.AT.SCENE event, schedules the CALL.END event for CAR K. DUR is the length of time that the primary car will spend at the incident (i.e., the length of the incident). T, the length of time CAR K spends at the incident, depends on whether it is the parimary car (SOT(K)=1), a backup car (SOT(K)=2), or a tertiary car (SOT(K) =3), as follows:

$$T = P.DURATION(SOT(K))* DUR,$$

where P.DURATION(I) is the proportion of DUR that a car of type I spends at an incident. The CALL.END for CAR K is then scheduled to occur in T minutes.

```
 ROUTINE FOR END.CALL.SCHEDULING GIVEN K AND DUR
DEFINE T AND DUR AS REAL VARIABLES
  LET T=P.DURATION(SOT(K))*DUR
 CREATE A CALL.END
  SCHEDULE THIS CALL.END GIVEN K IN T MINUTES
 RETURN
 END
```

EVENT CALL.END GIVEN K

This internal event is called whenever a car is scheduled to finish work at an incident or to return to service after being out of service. The subroutine first adjusts the state of the system and checks whether any other cars are still assigned to the incident. If there are none, the incident is erased from the system. Next, a check is made to see if this car has

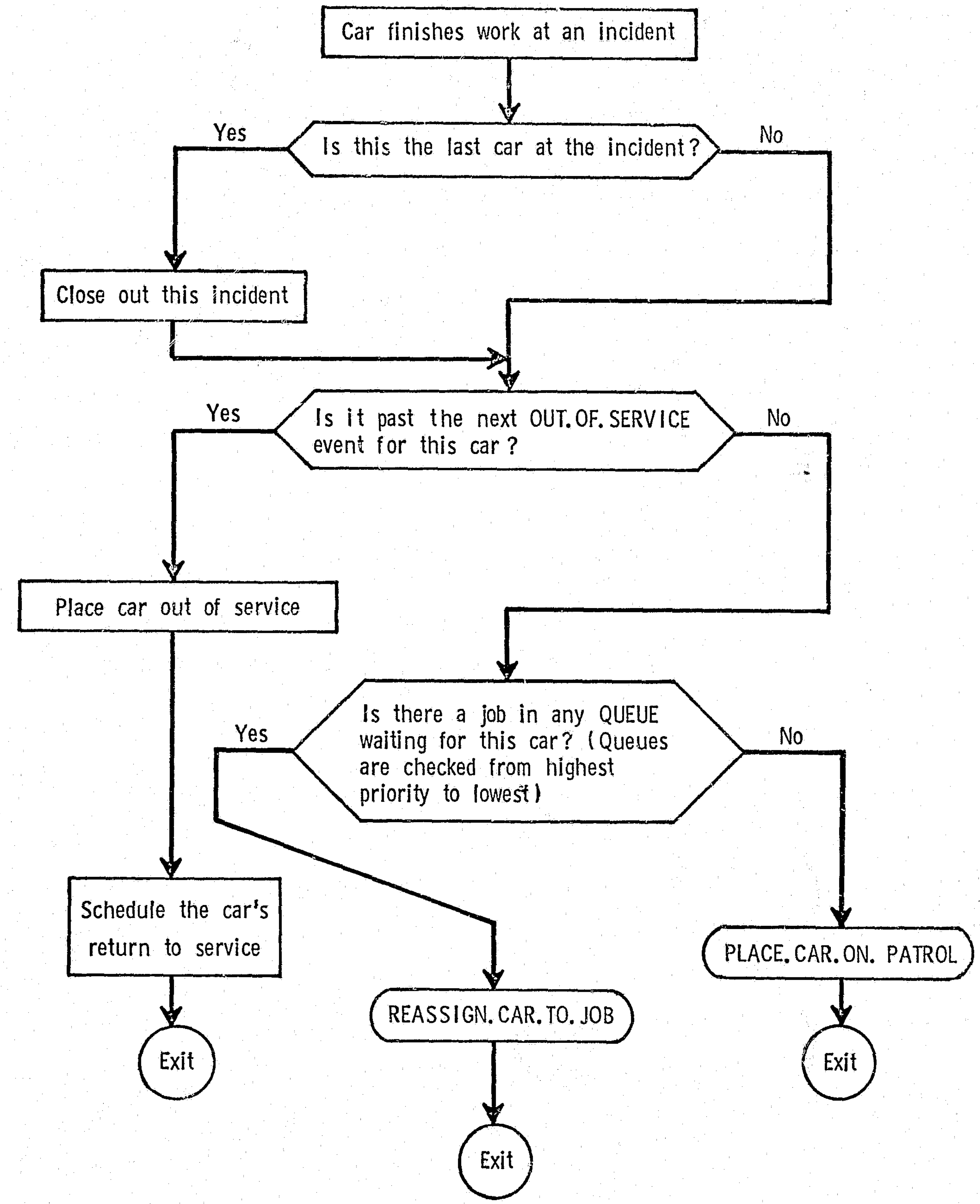


Fig. 6—CALL.END

been scheduled to go out of service. If so, the car is placed out of service for the required amount of time. Otherwise, the QUEUE is checked (from highest priority to lowest) to see if this car is needed at, and eligible for, some other incident. If so, REASSIGN.CAR.TO.JOB is called to dispatch the car to that incident. If there is no job waiting for this car, PLACE.CAR.ON.PATROL is called to return the car to preventive patrol in its sector. A flow chart for this event is given in Fig. 6.

```
00083700   EVENT CALL.FND GIVEN K
00083800   IF SWT(K)=6 GO TO CHECK.Q     ''END OF OUT-OF-SERVICE''
00083900    OTHERWISE LET J=ASSIGNMENT(K)
00084000   REMOVE K FROM ASSIGNED.CARS(J)
00084100   IF ASSIGNED.CARS(J) IS EMPTY DESTROY JOB CALLFD J
00084200   REGARDLESS
00084300 'CHECK.Q'
00084400         FOR EACH JOB ON QUEUE(1) ''THIS IS THE OUT-OF-SERVICE QUEUE''
00084500         WITH LOCATION(JOB)=K, DO LET ASSIGNMENT(K)=JOB  LET SWT(K)=6
00084600   SCHEDULE A CALL.END GIVEN K IN DURATION(JOB) MINUTES
00084700   LET SPT(K)=6  LET SOT(K)=6
00084800   LET XLOC(K)=XCORD(CENTROID(K))
00084900   LET YLOC(K)=YCORD(CENTROID(K))
00085000        REMOVE JOB FROM QUEUE(1)
00085100        RETURN
00085200        LOOP
00085300   FOR I=2 TO N.P.CLASS  DO
00085400     IF QUEUE(I) IS EMPTY   GO TO NEXT.Q
00085500     OTHERWISE IF I LE 3 NOW REASSIGN.CAR.TO.JOB(K,F.QUEUE(I))  RETURN
00085600                 OTHERWISE    ''MUST FIND A SECTOR CAR''
00085700                     FOR EACH JOB ON QUEUE(I) DO
00085800                                 LET NHOOD=NBDID(LOCATION(JOB))
00085900                                 FOR J=1 TO NUMB.SECTORS(K)    DO
00086000                                       IF NHOOD=SECTORS(K,J)
00086100                                             NOW REASSIGN.CAR.TO.JOB(K,JOB)
00086200                                             RETURN
00086300                                       OTHERWISE
00086400                                 LOOP
00086500                     LOOP
00086600  'NEXT.Q'  LOOP
00086700 'PREVENT.PATROL'
00086800      ADD 1 TO N.AVAILABLE
00086900    NOW PLACE.CAR.ON.PATROL(K)
00087000    RETURN
00087100    END
```

ROUTINE TO REASSIGN.CAR.TO.JOB GIVEN K AND J

This routine is called from the CALL.END event if CAR K is to be dispatched to JOB J, which has been waiting in a QUEUE. It changes the state of the system to reflect this dispatch, calculates the time that the job has spent waiting in queue, and calls the routine ASSIGN to assign the car to the job.


```
00087200
00087300     ROUTINE TO REASSIGN.CAR.TO.JOB GIVFN K AND J
00087400     LET SECT=NBDID(LOCATION(J))
00087500     IF PRIORITY(J)=4 LET XLOC(K)=XCORD(LOCATION(J))
00087600                      LET YLOC(K)=YCORD(LOCATION(J)) REGARDLESS
00087700     NOW ASSIGN(K,J)
00087800     FOR I = 1 TO N.CAR DO
00087900     LET KK=ADJACENT.CARS(SECT,I)
00088000          IF KK NE K GO TO LOOP1
00088100          OTHFRWISE LET SWT(K) =2
00088200               IF I GT N.SECTOR.CARS(SECT) LET SWT(K)=3
00088300               REGARDLESS GO TO OUT
00088400 'LOOP1'
00088500     LOOP
00088600 'OUT'
00088700     LET PRI=PRIORITY(J)
00088800     REMOVE J FROM QUEUE(PRI)
00088900     LET WAIT.TIME(PRI) = (TIME.V - ENTRY.TIME(J))*1440.0
00089000     LET COUNT(PRI)=1
00089100     LET SOT(K)=1
00089200     RETURN
00089300     END
```

ROUTINE TO PLACE.CAR.ON.PATROL GIVEN K

This routine is called from the CALL.END event if there are no jobs in any QUEUE for CAR K to service. First, the car's status is changed to reflect its availability. If the car is already inside its sector, it remains at its current location. (In reality, cars on patrol move around in their sectors more or less at random. Keeping the car where it is--the random location of the last job it serviced--is meant to imitate this with a minimum of computing effort, since, if the car moves at random, its expected location is its last location.) Otherwise, the car begins to travel back to the centroid of its sector, and a RETURN.TO.SECTOR event is scheduled for the car at a time determined from the distance it must travel and the average travel velocity for Priority 3 jobs. This routine will select blocks for patrol within a car's sector according to the relative frequency of calls for service at each of the blocks in the sector. The patrol frequencies could be modified by inserting into this routine (at line 00089700) a procedure to select the car's location according to a probability distribution associated with the neighborhood or a probability distribution specific to that car. The PREAMBLE would have to be modified to include the appropriate variables, and the desired patrol frequencies would have to be supplied in the initialization deck.

```
ROUTINE TO PLACE.CAR.ON.PATROL GIVEN K
DEFINE D AS A REAL VARIABLE
LET SPT(K)=0 LET SOT(K)=0
IF SWT(K)=4 OR SWT(K)=6  LET SWT(K)=0  RETURN   ''CAR WAS IN SECTOR'
'' ALREADY.... LEAVE HIM THERE ''
'OTHERWISE
CREATE A JOB CALLED PATROL.RETURN
LET SWT(K)=1 LET LJ=CENTROID(K) LET LOCATION(PATROL.RETURN)=LJ
LET ASSIGNMENT(K)=PATROL.RETURN
 LET PRIORITY(PATROL.RETURN)=3
LET D = CALCULATE.DISTANCE(XCORD(LJ),YCORD(LJ),XLOC(K),YLOC(K))
LET T.START.RESPONSE(K)=TIME.V
CREATE A RETURN.TO.SECTOR CALLED R
LET CARNUMBER(R)=K    ''THE CAR NUMBER OF THIS PATROL.RETURN''
SCHEDULE THE RETURN.TO.SECTOR CALLED R IN D/VELOCITY(3) HOURS
LET NEXT.EVENT(K)=R
RETURN
END
```

EVENT RETURN.TO.SECTOR GIVEN K

This event, scheduled internally, occurs whenever an available car that is returning from an out-of-sector job arrives at the centroid of its sector. The event changes the status of the system to reflect the car's arrival.

```
EVENT RETURN.TO.SECTOR GIVEN K
 LET J=ASSIGNMENT(K)
LET XLOC(K)=XCORD(CENTROID(K))
LET YLOC(K)=YCORD(CENTROID(K))
LET SWT(K)=0
DESTROY JOB CALLED J
RETURN
END
```

EVENT OUT.OF.SERVICE GIVEN K SAVING THE EVENT NOTICE

This event can either be scheduled internally (meal times will usually be scheduled in this way) or externally (other out-of-service conditions, such as the breakdown of a car, will usually be scheduled as part of the input job stream).

If the event has been scheduled internally, CAR K will be placed out of service for the number of minutes specified by the input variable MEAL. DURATION, and another OUT.OF.SERVICE event will be scheduled for this car in TOUR.LENGTH hours (TOUR.LENGTH is another input variable). If the event has been scheduled externally, the amount of time the car will remain out of service (in minutes) is read from the card in the input job stream that scheduled this event.

If the car scheduled to go out of service is currently available, it is immediately placed out of service and the status of the system is changed to reflect this. If returning to its sector, the car's sector return is canceled and it is immediately placed at the centroid of its sector.

If the car scheduled to go out of service is currently busy, it will be placed out of service as soon as it finishes the job. This is accomplished by placing notice of the out-of-service "job" on QUEUE(1). (Priority 1 jobs are never queued, so only out-of-service jobs will appear on QUEUE(1).) If an out-of-service job for CAR K already exists on QUEUE(1), the time of this new out-of-service job is added to the duration of the previously scheduled job. A flow chart for the OUT.OF.SERVICE event is given in Fig. 7.

```
EVENT OUT.OF.SERVICE GIVEN K SAVING THE EVENT NOTICE
  DEFINE T.OUT.OF.SERVICE AS A REAL VARIABLE
  LET X.MEAL = OUT.OF.SERVICE
  IF FUNIT.A(OUT.OF.SERVICE) EQ 0
  SCHEDULE AN OUT.OF.SERVICE GIVEN K IN TOUR.LENGTH HOURS
  LET T.OUT.OF.SERVICE=MEAL.DURATION  GO AROUND  OTHERWISE READ K,
  T.OUT.OF.SERVICE
'AROUND'
  DESTROY THE OUT.OF.SERVICE CALLED X.MEAL
  IF SWT(K) GE 2 GO TO DELAY
  OTHERWISE
     IF SWT(K)=1
     CALL CANCEL.SECTOR.RETURN  GIVEN K
     LET XLOC(K)=XCORD(CENTROID(K))
     LET YLOC(K)=YCORD(CENTROID(K))
     REGARDLESS
     SUBTRACT 1 FROM N.AVAILABLE
     LET SWT(K)=6  LET SPT(K)=6  LET SOT(K)=6
     CREATE CALL.END
      SCHEDULE THIS CALL.END GIVEN K IN T.OUT.OF.SERVICE MINUTES
  RETURN
'DELAY'
  FOR EVERY JOB ON QUEUE(1) DO
  IF LOCATION(JOB)=K GO TO ALREADY.SCHEDULED
  OTHERWISE LOOP

  CREATE JOB
  LET LOCATION(JOB)=K
  LET ENTRY.TIME(JOB)=TIME.V
  LET DURATION(JOB)=T.OUT.OF.SERVICE
  LET PRIORITY(JOB)=6
  LET STATUS(JOB)=6
  FILE JOB IN QUEUE(1)
  RETURN
'ALREADY.SCHEDULED'
  ADD T.OUT.OF.SERVICE TO DURATION(JOB)    RETURN
  END
```

EVENT FOR END.OF.SIMULATION

This internal event signals the end of a simulation run. It prints several reports that summarize the simulated patrol activity during the run.

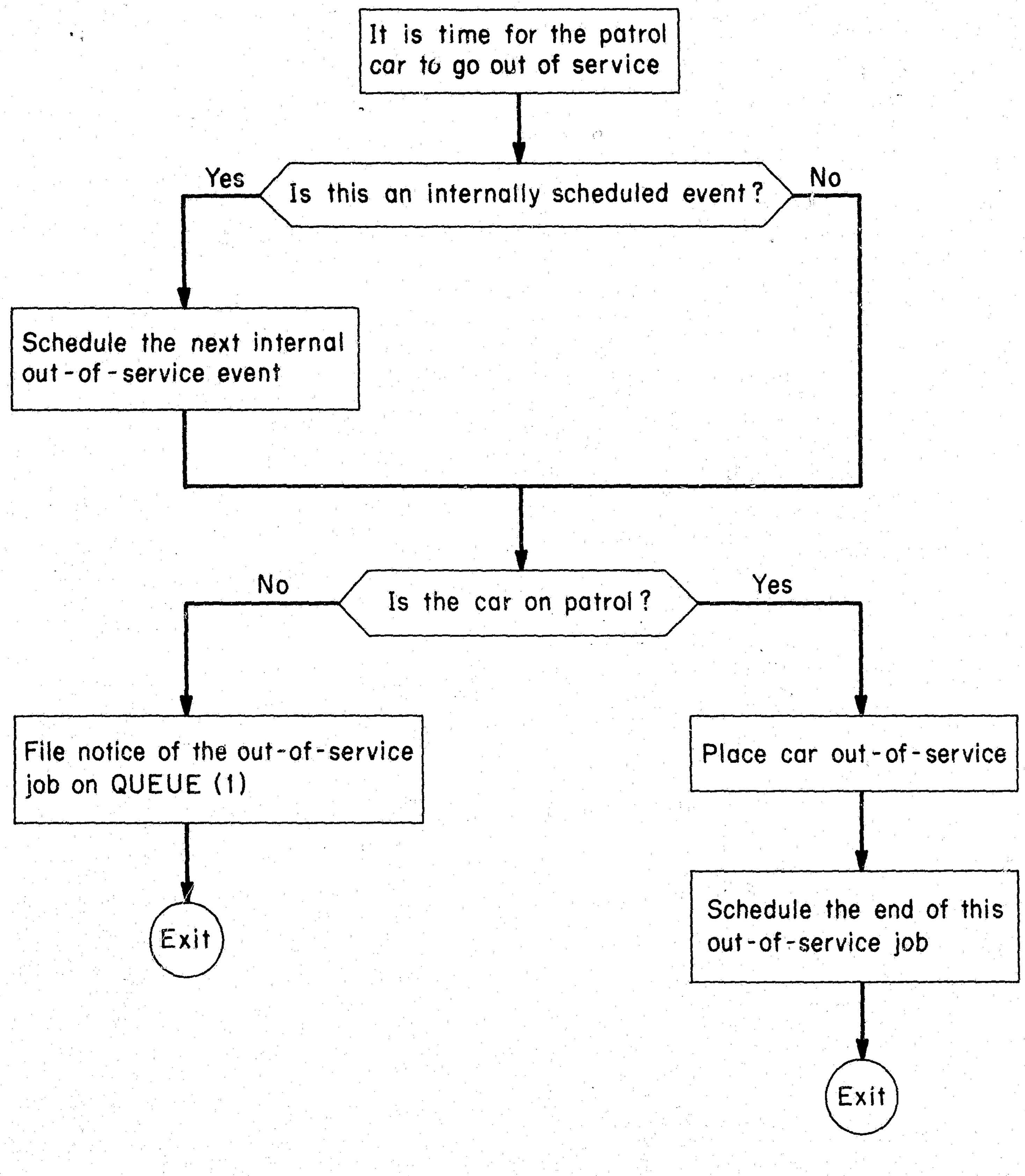


Fig. 7—OUT.OF.SERVICE

Reports are produced presenting the following information:

- The workload statistics for each CAR and the averages of these statistics for all CARs.
- Queueing statistics for each of the priority classes.
- Histograms, averages, and variances of travel times for each of the priority classes.
- A summary of responses by priority class, broken down by neighborhood.
- The average, variance, and distribution of patrol car availability over the course of the simulation.
- Statistics on the number of cars actually dispatched to incidents, broken down by priority class.

Details and examples of each of the reports are provided in Section VI.

```
 EVENT FOR END.OF.SIMULATION
DEFINE M.V,T,TIN,TOUT,TOTIN,TOTOUT AS REAL VARIABLES
DEFINE X AS A REAL 1-DIMENSIONAL ARRAY
RESERVE X(*) AS 10

 START NEW PAGE
   PRINT 2 DOUBLE LINES AS FOLLOWS

                     THE NEW YORK CITY-RAND INSTITUTE

                     POLICE PATROL SIMULATION MODEL
WRITE AS "SIMULATION RESULTS FOR "
FOR I= 1 TO 20, WRITE REGION.NAME(I) AS      A 4
WRITE AS /
SKIP 1 LINE
FOR I= 1 TO 20 , WRITE TITLE(I) AS     A 4
WRITE AS /
SKIP 5 OUTPUT LINES
PRINT 1 LINE AS FOLLOWS

LET IJ=0      LET OJ=0                   CAR ACTIVITY SUMMARY
SKIP 1 OUTPUT LINE
 PRINT 2 DOUBLE LINES AS FOLLOWS
CAR      NO. OF JOBS        ON PATROL        RESPONDING        WORKING
TOT. IN SERVICE
          IN        OUT       IN      OUT       IN       OUT       IN       OUT
 IN       OUT      OUT OF SERVICE
 FOR K= 1 TO N.CAR,DO
LET TIN=(UTILIZ(K,1) +UTILIZ(K,3) +UTILIZ(K,5))/TIME.V
LET TOUT=(UTILIZ(K,2)+UTILIZ(K,4)+UTILIZ(K,6))/TIME.V
PRINT 1 DOUBLE LINE WITH CAR.NAME(K),IN.SECT.JOBS(K),OUT.SECT.JOBS(K),
UTILIZ(K,1)/TIME.V,UTILIZ(K,2)/TIME.V,UTILIZ(K,3)/TIME.V,
UTILIZ(K,4)/TIME.V,UTILIZ(K,5)/TIME.V,UTILIZ(K,6)/TIME.V,
TIN,TOUT,UTILIZ(K,7)/TIME.V AS FOLLOWS
```

```
   ***    ****     ****    *.***   *.***    *.***    *.***    *.***    *.***
*.***   *.***          *.***
   FOR J=1 TO 7, DO   ADD UTILIZ(K,J)/TIME.V TO X(J)     LOOP
   ADD IN.SECT.JOBS(K) TO IJ
   ADD OUT.SECT.JOBS(K) TO OJ
      ADD TIN TO TOTIN
      ADD TOUT TO TOTOUT
    LOOP
  FOR J=1 TO 7 DO        LET X(J)=X(J)/N.CAR    LOOP
  LET TOTIN=TOTIN/N.CAR          LET TOTOUT=TOTOUT/N.CAR
  SKIP 1 OUTPUT LINE
   PRINT 1 DOUBLE LINE WITH IJ,OJ,X(1),X(2),X(3),X(4),X(5),X(6),
       TOTIN,TOTOUT,X(7) THUS
TOTALS  ****     ****    *.***   *.***    *.***    *.***    *.***    *.***
*.***    *.***          *.***

   FOR I=2 TO N.P.CLASS,    DO    START NEW PAGE
   FOR J= 1 TO 20, WRITE TITLE(J) AS A 4
   WRITE AS /
   SKIP 2 LINES
    PRINT 1 LINE WITH I AS FOLLOWS
        QUEUEING STATISTICS FOR PRIORITY CLASS **
    SKIP 2 OUTPUT LINES
    PRINT 2 LINES WITH MQ(I) AND VQ(I) AS FOLLOWS
     HISTOGRAM OF QUEUE-SIZE            AVERAGE QUEUE-SIZE = ***.***
 NO. IN QUEUE    FREQUENCY              VARIANCE                ****.***
    SKIP 1 OUTPUT LINE
        FOR J= 1 TO 30 , WITH PROBQ(I,J) GT 0, DO
        PRINT 1 LINE WITH J-1 AND PROBQ(I,J)/TIME.V AS FOLLOWS
   **              *.***
    LOOP

   SKIP 2 OUTPUT LINES
   LET T=NWT(I)*MWT(I)
   LET NTOT=NRT(I)
    IF NTOT=0  LET T=0   GO TO PRT
   OTHERWISE LET T=T/NTOT
'PRT'  PRINT 3 DOUBLE LINES WITH NWT(I),MWT(I),VWT(I),NTOT,T THUS
    HISTOGRAM OF WAITING TIMES        AVERAGE WAITING TIME OF THE ***
JOBS DELAYED= ***.***
   WAITING TIME          FREQUENCY
VARIANCE     = ***.***
    (MINUTES)                         AVERAGE WAITING TIME OF ALL ****
JOBS DISP'D = ***.***
        FOR J=1 TO 24, WITH HWT(I,J) GT 0, DO
           PRINT 1 LINE WITH 5*J AND HWT(I,J) THUS
      < ***          *****
   LOOP
  SKIP 3 OUTPUT LINES
  LET J=1
  PRINT 3 LINES AS FOLLOWS
   CURRENT QUEUE CONTENTS
 POSITION         TIME SINCE RECEIPT
                      (MINUTES)
  SKIP 1 OUTPUT LINE
  FOR EVERY JOB IN QUEUE(I), DO
  PRINT 1 LINE WITH J AND (TIME.V-ENTRY.TIME(JOB))*1440.0 THUS
  ***              ***.**
  ADD 1 TO J
  LOOP
    LOOP
  CALL RESULTS
  END    ''SIMULATION''
```



```
    ROUTINE FOR RESULTS
    DEFINE M,V,T  AS REAL VARIABLES
     BEGIN REPORT PRINTING FOR I=1 TO N.P.CLASS IN GROUPS OF 6 PER PAGE
    BEGIN HEADING
    FOR J= 1 TO 20, WRITE TITLE(J) AS A 4
    WRITE AS /
    SKIP 2 LINES
    PRINT 1 LINE AS FOLLOWS
      RESPONSE TIME HISTOGRAMS
     SKIP 2 OUTPUT LINES
    END ''HEADING''
    PRINT 1 DOUBLE LINE WITH A GROUP OF I FIELDS THUS
    PRIORITY CLASS        **          **          **          **          **
  **
    SKIP 1 OUTPUT LINE
    PRINT 2 LINES AS FOLLOWS
     RESPONSE TIME
      (MINUTES)
         FOR J= 1 TO 16, DO
   PRINT 1 DOUBLE LINE WITH J AND A GROUP OF FREQ(I,J) FIELDS THUS
    <**.**          ****        ****        ****        ****        ****
 ****
    LOOP
    SKIP 1 OUTPUT LINE
   PRINT 1 DOUBLE LINE WITH A GROUP OF NRT(I) FIELDS THUS
NUMBER OF RESPONSES    *****       *****       *****       *****       *****
 *****
   PRINT 1 DOUBLE LINE WITH A GROUP OF MRT(I) FIELDS THUS
 AVERAGE R.T.          **.**       **.**       **.**       **.**       **.**
 **.**
   PRINT 1 DOUBLE LINE WITH A GROUP OF VRT(I) FIELDS THUS
 VARIANCE OF R.T.     ***.**      ***.**      ***.**      ***.**      ***.**
***.**
   END  ''REPORT''
   START NEW PAGE
   FOR I= 1 TO 20 , WRITE TITLE(I) AS    A 4
   WRITE AS /
   SKIP 2 LINES
   PRINT 1 DOUBLE LINE AS FOLLOWS
                                                 SUMMARY OF RESPONSES BY
 NEIGHBORHOOD
   SKIP 2 OUTPUT LINES
   PRINT 2 DOUBLE LINES AS FOLLOWS
     PRIORITY-->    1                        2                         3
                  5                TOTALS
  NBD          NO. AVG RT   VAR          NO. AVG RT   VAR         NO. AVG RT
 VAR        NO. AVG RT   VAR        NO.  AVG RT   VAR
   SKIP 1 OUTPUT LINE
   FOR J=1 TO N.NBD DO
   LET N1=NSRT1(J) LET N2=NSRT2(J) LET N3=NSRT3(J) LET N4=NSRT5(J)
    LET T=N1+N2+N3+N4
   LET M=N1*MSRT1(J)+N2*MSRT2(J)+N3*MSRT3(J)+N4*MSRT5(J)
   LET V=N1*N1*VSRT1(J)+N2*N2*VSRT2(J)+N3*N3*VSRT3(J)+N4*N4*VSRT5(J)
    IF T=0  LET M=0  LET V=0  GO TO PRI          OTHERWISE
    LET M=M/T
    LET V=V/(T*T)
 'PRI' PRINT 1 DOUBLE LINE WITH NBD.NAME(J),N1,MSRT1(J), VSRT1(J),N2,
   MSRT2(J),VSRT2(J),N3,MSRT3(J),VSRT3(J),N4,MSRT5(J),VSRT5(J),T,M,V
    THUS
  ****        **** **.*** **.***       **** **.*** **.***       **** **.*** *
*.***      **** **.*** **.***       ***** **.*** **.***
```

```
    LOOP
    SKIP 1 OUTPUT LINE
    LET T=0  LET M=0   LET V=0
    FOR I=1 TO 4  DO
         IF I EQ 4, LET I = 5 REGARDLESS
         LET N=NRT(I)          ADD N TO T
         ADD N*MRT(I) TO M    ADD N*N*VRT(I) TO V
    LOOP
    LET M=M/T        LET V=V/(T*T)
    PRINT 1 DOUBLE LINE WITH NRT(1),MRT(1),VRT(1),NRT(2),MRT(2),
    VRT(2),NRT(3),MRT(3),VRT(3),NRT(5),MRT(5),VRT(5),T,M,V THUS
 TOTALS       **** ** *** **.***      **** **.*** **.***      **** **.*** *
*.***      **** **.*** **.***      ***** **.*** **.***
    SKIP 9 OUTPUT LINES
    PRINT 2 LINES AS FOLLOWS
            PICK-UP (PRIORITY 4)
  NBD           NO. AVG RT    VAR
    SKIP 1 OUTPUT LINE
    FOR J=1 TO N.NBD DO
    LET N5 = NSRT4(J)
    PRINT 1 LINE WITH NBD.NAME(J),N5,MSRT4(J),VSRT4(J) AS FOLLOWS
  ****        **** **.*** **.***
    LOOP
    SKIP 1 OUTPUT LINE
    PRINT 1 LINE WITH NRT(4), MRT(4), VRT(4) THUS
 TOTALS       **** **.*** **.***
    START NEW PAGE
    FOR I= 1 TO 20 , WRITE TITLE(I) AS     A 4
    WRITE AS /
    SKIP 2 LINES
    PRINT 1 LINE AS FOLLOWS
      DISTRIBUTION OF CAR AVAILABILITY
    SKIP 2 OUTPUT LINES
    PRINT 2 LINES AS FOLLOWS
          NO. OF CARS        PER CENT
          ON PATROL          OF TIME
    FOR I=0 TO N.CAR DO
      PRINT 1 LINE WITH I AND 100*HISTAV(I+1)/TIMF.V THUS
             **                *.***
    LOOP
      SKIP 1 OUTPUT LINE
    PRINT 1 LINE WITH MAV THUS
        AVERAGE NUMBER AVAILABLE = **.***
    PRINT 1 LINE WITH VAV THUS
                     VARIANCE  =***.***
     START NEW PAGE
    FOR I= 1 TO 20 , WRITE TITLE(I) AS     A 4
    WRITE AS /
    SKIP 2 LINES
      PRINT 1 LINE AS FOLLOWS
  NUMBER OF PRECINCT CARS SENT TO CALLS
    SKIP 2 OUTPUT LINES
      PRINT 1 DOUBLE LINE THUS
  PRIORITY     0     1     2     3     4     5     6     7     8     9
 10     11     12     13     14     15    AVERAGE    VARIANCE
    WRITE AS /,"       1  "
      FOR I=1 TO 16,WRITE HISTC(1,I) AS I 6
    WRITE MC(1) AND VC(1) AS S 6,D(6,2),S 5,D(7,2),/
    FOR I=2 TO N.P.CLASS, DO
    PRINT 1 DOUBLE LINE WITH I,HISTC(I,1),HISTC(I,2),HISTC(I,3),
         HISTC(I,4),MC(I), AND VC(I) THUS
     **    ****  ****  ****  ****
                                           ***.**      ****.**
    LOOP
    STOP
     END
```


V. THE INPUT DATA

There are two basic sets of data needed to run the simulation. The first input data set, called the "initialization deck," follows the job control cards and specifies the geographical configuration of the region being simulated, the patrol unit assignments, and initial parameter values such as the number of hours in a tour, response velocities, meal times, the duration of the simulation, etc. (The initialization deck is the GO.SYSIN data set.) The second data set is the series of calls for service to be responded to during the course of the simulation, and is referred to as the "job stream." (It is the SIMU04 data set.) In what follows, we describe, card by card, all the data needed for the simulation. For easy reference to the program, the variable name used to identify each piece of data in the simulation is given in upper case in parenthesis. So, for example, the simulation variable N.CAR is used to refer to the number of patrol units assigned to the region being simulated.

The Initialization Deck (GO.SYSIN)

(5.1) Card 1--supplies the title for the simulation run. This title will be printed at the top of every output report. The title can be up to 80 characters long (the contents of one complete card).

(5.2) Card 2--a description of the region being simulated (e.g., 26th Precinct, 7th Division, etc.). This is printed on the first page of the output report. The description can be up to 80 characters long (the contents of one complete card).

(5.3) Card 3--supplies the initial values for four system parameters:

- (a) The length of the simulation (SIM.LENGTH) in hours.
- (b) The maximum number of cars that will be dispatched to a Priority 1 incident (MAX.SENT).
- (c) The duration of an internally scheduled OUT.OF.SERVICE job (MEAL.DURATION) in minutes.
- (d) The length of a single tour of duty (TOUR.LENGTH) in hours. This variable is used to schedule an OUT.OF.SERVICE event for each car one time during every tour of duty (e.g., for a meal). After the first internally scheduled OUT.OF. SERVICE event for a car (scheduled at a time specified as

described in paragraph (5.7) below), an OUT.OF.SERVICE event for the car will occur every TOUR.LENGTH hours.

(5.4) Card 4--the number of BLOCKs (N.BLOCK) in the region being simulated (A BLOCK is the smallest geographical unit in the simulation), the number of neighborhoods (N.NBD), and the number of patrol units (N.CAR). These three integers are punched in three fields, separated by at least one blank space. Note that, for internal reference, the blocks will be numbered 1 to N.BLOCK, the neighborhoods from 1 to N.NBD, and the cars from 1 to N.CAR.

(5.5) N.BLOCK Cards--one for each block, containing:

(a) The block number used as an internal reference number for the block (B). These must run in sequence from 1 through N.BLOCK.

(b) The external identification number associated with the block (TAXNO(B)). This number can be its tax block number, census block, or any other identifying number.

(c) The internal reference number of the neighborhood to which the block belongs (NBDID(B)).

(d) The x and y coordinates of the center of the block (XCORD(B), YCORD(B)).

This information is punched in five fields, each separated by at least one blank space. The variables must be integers, except for the coordinates, which can be punched with decimal points. We emphasize that NBDID(B) is a number and not an alphanumeric sector name associated with the neighborhood. Further, NBDID(B) must correspond to the appropriate internal neighborhood sequence number (N) used when neighborhood information is being input. (See paragraph (5.6) below.) These N.BLOCKs may comprise a separate data set. If so, they must be kept separate from the GO.SYSIN data, and a data definition (DD) card for the additional data set must be included in the job control cards. In addition, the statement in the MAIN routine that reads this data (statement number 34200) should indicate the logical unit from which this data should be read (e.g., the version given here reads this data from logical unit 7).

(5.6) N.NBD Sets of Cards--one set for each neighborhood, consisting of:

(a) One card containing the following four fields, separated by blanks:

(1) The internal neighborhood reference number (N). These must run in sequence from 1 to N.NBD.

(2) An alphanumeric sector name associated with the neighborhood (NBD.NAME(N)).

(3) The number of sector cars assigned to the neighborhood (N.SECTOR.CARS(N)).

(4) The number of patrol units designated as adjacent resources for the neighborhood (N.ADJACENT(N)).

(b) One or more cards containing a list of all the cars in the simulation in the order in which they will be nominated for dispatching in this neighborhood (ADJACENT.CARS(N.J). The members of the list are car numbers, punched in N.CAR fields, each separated by at least one blank space. The car numbers used must correspond to the internal car numbers (C), discussed in paragraph (5.7) below.

(5.7) N.CAR Sets of Cards--one set for each patrol unit, consisting of:

(a) One card containing the following five fields, separated by blanks:

(1) The internal reference number for the unit (C). These must run in sequence from 1 to N.CAR.

(2) The 4-character alphanumeric name associated with the unit (CAR.NAME(C)).

(3) The number of neighborhoods to which the unit is assigned as a "sector" car (NUMB.SECTORS(C)). If none (e.g., a supervisor's car), put a 1 in this field and a zero in the following card.

(4) The internal block number of the block at which the unit is "stationed" while on patrol (CENTROID(C)). This number must correspond to one of the block numbers (B) discussed in paragraph (5.5) above.

(5) The scheduled time into a tour (in decimal hours, e.g., 1.5) at which the car will be placed out of service (MEAL.TIME).

(b) One or more cards containing the list of neighborhoods to which the unit is assigned as a "sector car." This list should contain NUMB.SECTORS(C) entries, which are the internal neighborhood reference numbers (N) described in paragraph (5.6) above. Overlapping sectors are achieved by assigning some or all of the same neighborhoods to two or more cars.

(5.8) 1 Card--giving the response velocity in miles per hour for each of the five priority classes (VELOCITY(P)), punched as real numbers in five fields, each separated by at least one blank space, in order of priority class, starting with Priority class 1.

(5.9) 1 Card--giving the proportion of the job duration (P.DURATION (I)), assigned to each of the three types of responding units (primary, backup, and tertiary), punched as real numbers in three fields, each separated by at least one blank space.

The Job Stream (SIMU04)

(5.10) In chronological order, as many cards as there are calls for service and externally scheduled OUT.OF.SERVICE events. For each call for service, the card must contain the following 9 fields describing the job:

(1) The event name JOB.ENTRY punched in columns 1-9.
(2-4) The time the job enters the system (ENTRY.TIME(J)), measured in days, hours, and minutes from the start of the simulation, punched in three fields.
(5) The location of the job (LOCATION(J)), given by the internal block number. This number must correspond to one of the values of B given in paragraph (5.5) above.
(6) The priority of the job (PRIORITY(J)).
(7-8) The duration of the job in hours and minutes, punched in two fields (HDUR and MDUR).
(9) An asterisk (*).

All fields must be separated by at least one blank space. There should always be at least one job whose entry time is later than SIM.LENGTH, the end of the simulation.

For each OUT.OF.SERVICE event, the card must contain the following three fields of information:

(1) The event name OUT.OF.SERVICE punched in columns 1-14.
(2) The internal indentification number of the car to be placed out of service. This number must correspond to one of the values of C given in paragraph (5.7) above.
(3) The length of the out-of-service time in minutes.

AN EXAMPLE

We illustrate the data input with sample data sets for a simulation of the 71st Precinct in the New York City Police Department. The 71st Precinct has been broken down into 305 blocks, which, in this simulation run, are grouped into 12 neighborhoods. Each neighborhood corresponds to one of 12 nonoverlapping sectors. Figure 8 is a street map of the 71st Precinct and Fig. 9 is a computer-generated map that uses the coordinate system of the simulation and shows each individual tax block and the sector to which it belongs. In this example, there are seven patrol units: six are sector cars, each of which is assigned to a pair of neighborhoods, and one is a sergeant's car. Primary cars work for the entire duration of the incident; backup cars work for half the duration of the incident, and tertiary cars are released as soon as they arrive at the incident. Responses to all incidents, no matter what their priority, are made at a velocity of 20 mph. The data set describing each of the blocks is read from logical unit 7. A listing of the control cards and initialization deck for this example is given in Appendix B. Some of the key data elements are explained in Tables 5 and 6. Appendix C lists the first few cards in a sample job stream for a 16-hour simulation of this region.

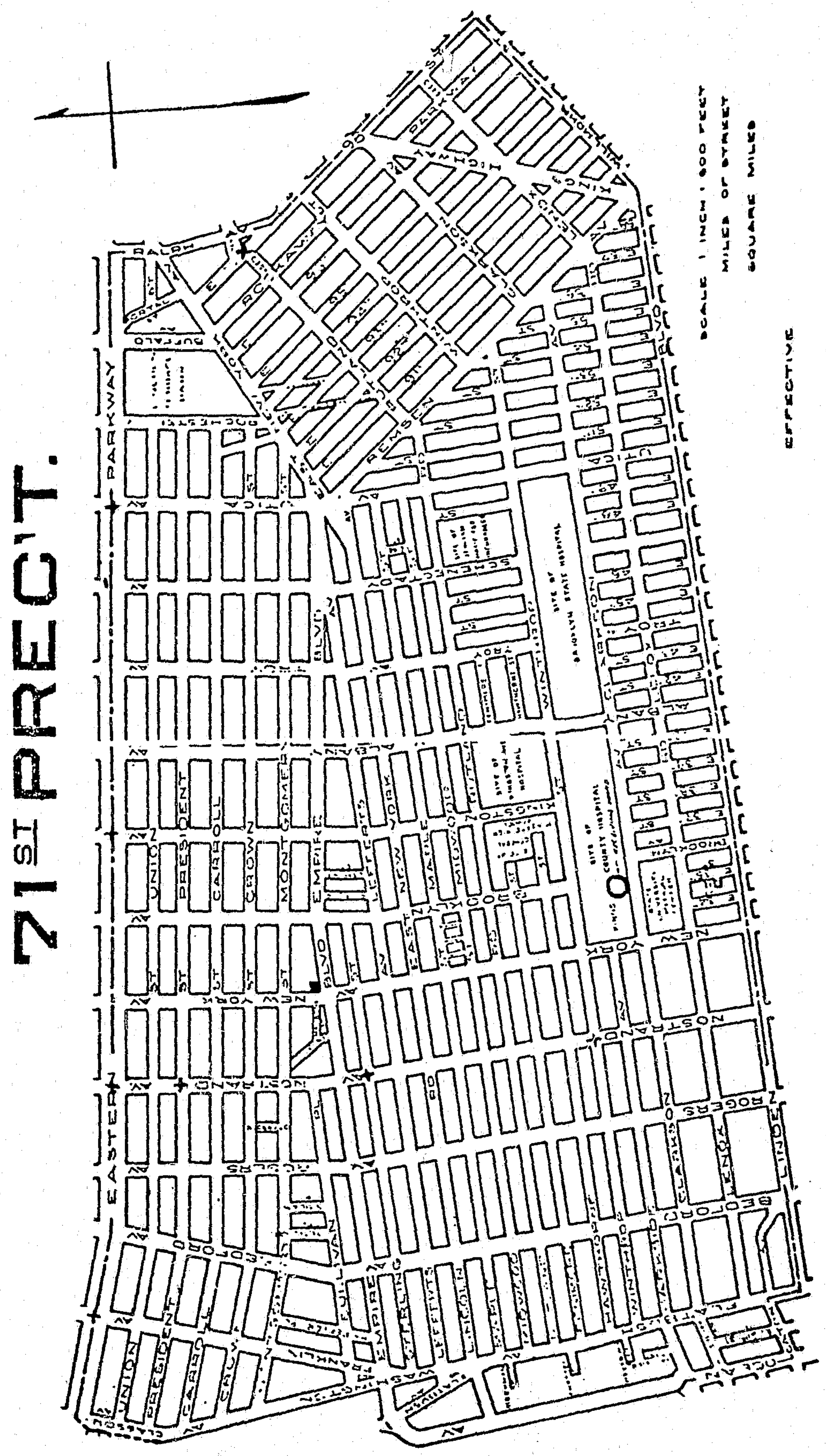


Fig. 8—Map of 71st Precinct—Brooklyn

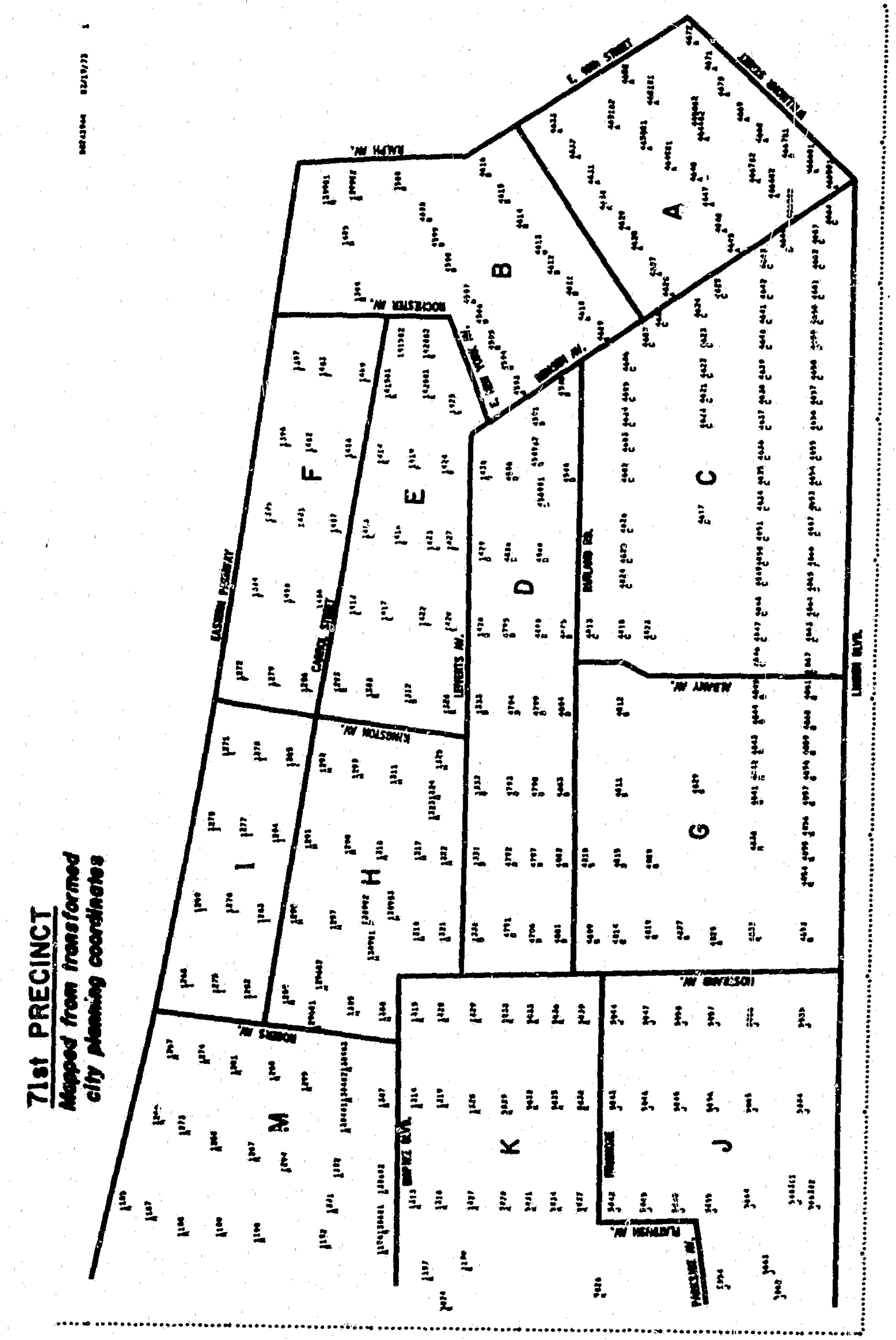


Fig. 9—Computer-generated map of tax blocks and sectors in the 71st Precinct

Table 5

SAMPLE SIMULATION: CAR PARAMETERS

Car Number (C)	Car Name (CAR.NAME(C))	Number of Sectors (NUMB.SECTORS(C))	Meal Time (MEAL.TIME)	Patrol Block CENTROID(C))	Assigned SECTORS(C,J))
1	AB	2	3	144	1,2
2	CD	2	4	222	3,4
3	EF	2	5	100	5,6
4	HI	2	3	43	8,9
5	JG	2	4	296	7,10
6	KM	2	5	63	11,12
7	SGT	0	4	59	0

Table 6

SAMPLE SIMULATION: NEIGHBORHOOD DATA

Neighborhood Number (N)	Neighborhood Name (NBD.NAME(N))	Number of Sector Cars N.SECTOR.CARS(N))	Number of Adjacent Cars (N.ADJACENT(N))	Adjacent Car List (ADJACENT.CARS(N,J))
1	A	1	1	(1, 2, 3, 4, 5, 6, 7)
2	B	1	1	(1, 2, 3, 4, 5, 6, 7)
3	C	1	1	(2, 3, 4, 5, 6, 1, 7)
4	D	1	1	(2, 3, 4, 5, 6, 1, 7)
5	E	1	1	(3, 1, 4, 2, 6, 5, 7)
6	F	1	1	(3, 1, 4, 2, 6, 5, 7)
7	G	1	1	(5, 2, 6, 4, 3, 1, 7)
8	H	1	1	(4, 6, 3, 2, 5, 1, 7)
9	I	1	1	(4, 6, 3, 2, 5, 1, 7)
10	J	1	1	(5, 2, 6, 4, 3, 1, 7)
11	K	1	1	(6, 5, 4, 2, 3, 1, 7)
12	M	1	1	(6, 5, 4, 2, 3, 1, 7)

VI. OUTPUT REPORTS

The simulation program produces several types of output. We illustrate the various output reports by showing those produced by a 16-hour simulation of New York City's 71st Precinct, using the sample input data presented in Appendix B. First, after reading the initalization deck, the program prints out the title and the following input data;

- The length of the simulation (in hours).
- The number of blocks, neighborhoods, and patrol cars in the region being simulated.
- A listing of the data associated with each block, including its internal reference number, external identification number, x and y coordinates; and the data associated with each neighborhood, including its internal reference number, the number of sector cars and adjacent cars assigned to it, and the ordered list of patrol cars constituting its set of ADJACENT.CARS.

A sample of this initialization output is shown in Figs. 10 and 11.

After simulating the patrol operations in the region for the required number of hours, the program prints a series of six reports that summarize the simulated activity. Each of these reports is described below.

(1) CAR ACTIVITY SUMMARY (Fig. 12)

A summary is given for each car, showing how it spent its time during the course of the simulation. The following information is printed for each car:

- The number of jobs to which it responded within its sector and the number to which it responded outside its sector.
- The proportion of time the car spent on preventive patrol in its sector and the proportion of time it was available to respond from outside its sector (while returning to its sector from an outside job).
- The proportions of its time spent responding to calls that occurred inside its sector and responding to calls outside its sector.

- The proportions of its time spent working on jobs in its sector and working outside its sector.
- The proportion of its total in-service time spent within its sector and the proportion of time spent out of its sector.
- The proportion of the total simulated time during which the car was out of service.

The last line of the report prints the values of each of the above quantities, averaged over the N.CAR patrol units in the simulation.

(2) QUEUEING STATISTICS (Fig. 13)

A one-page report is printed for each priority class having a queue (i.e., all classes except Priority 1). This report contains three types of information:

- Queue sizes--a frequency histogram, plus the average and variance of the queue size. (The histogram shows frequencies for queues of up to 50 jobs.)
- Waiting times--a frequency histogram of the waiting time of jobs that were queued before being dispatched, the average and variance of the waiting time for these jobs, and the average waiting time for all jobs dispatched, including those that were not delayed. (The histogram shows waiting time frequencies in five-minute intervals up to four hours.)
- Current (final) queue contents--a list of those calls in queue and waiting to be dispatched at the instant the simulation ends, together with the time elapsed since each call was received (the time each call has already spent in queue).

(3) RESPONSE TIME HISTOGRAMS (Fig. 14)

A report listing, for each priority class, the number of calls that had response times falling into each one-minute interval (i.e., 0-1 minute, 1-2 minutes, etc.) up to sixteen minutes, where response time is defined as the time from dispatch of call until the arrival of the first patrol car. In addition, the total number of responses is printed for each priority class.

(4) NEIGHBORHOOD ACTIVITY SUMMARY (Fig. 15)

This report shows the workload (number of jobs) and response time information for each neighborhood, broken down by priority class. Thus, the number of jobs responded to, together with the average and variance of the response times to these jobs, is printed for each priority class, neighborhood by neighborhood. Totals are printed for each neighborhood and each priority class.

(5) DISTRIBUTION OF CAR AVAILABILITY (Fig. 16)

This report shows the percentage of time that a given number of cars were on preventive patrol in the region, from 0 patrol units to N.CAR patrol units. In addition, the average number of available cars and the variance are printed.

(6) NUMBER OF PRECINCT CARS SENT TO CALLS (Fig. 17)

Since under different dispatching rules different numbers of cars may be sent to calls, this report shows how often any given number of cars (from 0 to 15) are sent to calls in each priority class. In some priority classes exactly one car is always sent (e.g., Classes 3, 4, and 5), but Priority 1 calls receive up to MAX.SENT of the available cars. The number of Priority 1 calls that received zero cars indicates the number of times a Priority 1 call was received when all N.CAR patrol units in the region being simulated were busy.

Input data for the 71st Precinct, Brooklyn, New York

SIMULATION LENGTH = 16.00 HOURS

NO. OF BLOCKS = 305 NO. OF NBDS. = 12 NO. OF CARS = 7

BLOCK ID	TAX NO.	NBD ID	X	Y
1	1185	12	.81170	2.18140
2	1187	12	.79450	2.12950
3	1188	12	.76100	2.07780
4	1189	12	.76830	2.00850
5	1190	12	.75940	1.94630
6	1192	12	.74620	1.83550
7	1196	12	.73350	1.73110
8	1197	11	.69360	1.65230
9	1198	11	.71180	1.58130
10	1266	12	.96310	2.12540
11	1267	12	1.08660	2.09430
12	1268	9	1.21150	2.07400
13	1269	9	1.35220	2.05450
14	1270	9	1.49190	2.03140
15	1271	9	1.63190	2.01260
16	1272	6	1.77100	1.98720
17	1273	12	.94520	2.07420
18	1274	12	1.07370	2.04180
19	1275	9	1.20350	2.02170
20	1276	9	1.34540	2.00060
21	1277	9	1.48500	1.97950
22	1278	9	1.62440	1.95770
23	1279	6	1.76420	[illegible]
24	1280	12	.92480	2.01610
25	1281	12	1.05880	1.98270
26	1282	9	1.19540	1.96240
27	1283	9	1.33520	1.94160
28	1284	9	1.47580	1.92010
29	1285	9	1.61570	1.89770
30	1286	6	1.75480	1.87730
31	1287	12	.90450	1.95950
32	1288	12	1.04540	1.92460
33	1289	8	1.18560	1.90230
34	1290	8	1.32660	1.88150
35	1291	8	1.46690	1.86140
36	1292	8	1.60610	1.83970
37	1293	5	1.74740	1.81760
38	1294	12	.88370	1.90050
39	1295	12	1.02960	1.86610
40	129601	8	1.14210	1.84800
41	129602	8	1.21210	1.83870
42	1297	8	1.31790	1.82290
43	1298	8	1.45730	1.79970
44	1299	8	1.59730	1.78100
45	1300	5	1.73730	1.75870
46	1301	12	.80640	1.82220

Fig. 10—Listing of the input data: block data

Input data for the 71st Precinct, Brooklyn, New York

6 PATROL CARS AND 1 SERGEANT'S CAR

NBD ID	SECTORS	NUMBER OF SECT.CARS	NUMBER OF ADJ.CARS	ORDER IN WHICH CARS ARE NOMINATED
1	A	1	1	1 2 3 4 5 6 7
2	B	1	1	1 2 3 4 5 6 7
3	C	1	1	2 3 4 5 6 1 7
4	D	1	1	2 3 4 5 6 1 7
5	E	1	1	3 1 4 2 6 5 7
6	F	1	1	3 1 4 2 6 5 7
7	G	1	1	5

Fig. 11—Listing of the input data: neighborhood data

THE NEW YORK CITY-RAND INSTITUTE
POLICE PATROL SIMULATION MODEL

SIMULATION RESULTS FOR THE 71ST PRECINCT, BROOKLYN, NEW YORK

6 PATROL CARS AND 1 SERGEANT'S CAR

CAR ACTIVITY SUMMARY

CAR	NO. OF JOBS		ON PATROL		RESPONDING		WORKING		TOT. IN SERVICE		OUT OF SERVICE
	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	
AB	8	6	.311	.009	.014	.031	.207	.196	.533	.236	.231
CD	5	16	.297	.017	.009	.054	.120	.339	.426	.410	.164
EF	8	11	.272	.016	.013	.032	.215	.255	.499	.303	.198
HI	5	16	.232	.023	.009	.055	.093	.428	.333	.506	.160
JG	8	12	.362	.026	.015	.039	.210	.183	.588	.248	.165
KM	8	10	.296	.017	.012	.038	.257	.232	.565	.287	.148
SGT	0	12	.465	.016	0.	.032	0.	.321	.465	.369	.167
TOTALS	42	83	.319	.018	.010	.040	.157	.279	.487	.337	.176

Fig. 12—Car activity summary

6 PATROL CARS AND 1 SERGEANT'S CAR

QUEUEING STATISTICS FOR PRIORITY CLASS 3

HISTOGRAM OF QUEUE-SIZE

NO. IN QUEUE	FREQUENCY
0	.758
1	.092
2	.031
3	.029
4	.030
5	.036
6	.022
7	.000
8	.002

AVERAGE QUEUE-SIZE = .691
VARIANCE 2.314

HISTOGRAM OF WAITING TIMES

WAITING TIME (MINUTES)	FREQUENCY
< 5	8
< 10	9
< 15	9
< 20	5
< 25	3
< 30	5
< 35	3
< 40	2
< 45	1

AVERAGE WAITING TIME OF THE 45 JOBS DELAYED= 15.877
VARIANCE = 123.109
AVERAGE WAITING TIME OF ALL 104 JOBS DISP'D = 6.870

CURRENT QUEUE CONTENTS

POSITION	TIME SINCE RECEIPT (MINUTES)

Fig. 13—Queuing statistics for Priority Class 3

6 PATROL CARS AND 1 SERGEANT'S CAR

RESPONSE TIME HISTOGRAMS

PRIORITY CLASS	1	2	3	4	5
RESPONSE TIME (MINUTES)					
< 1.00	1	0	4	0	0
< 2.00	2	0	29	0	0
< 3.00	2	0	26	0	0
< 4.00	3	0	17	0	0
< 5.00	1	0	11	0	0
< 6.00	0	0	9	0	0
< 7.00	0	0	5	0	0
< 8.00	0	0	2	0	0
< 9.00	0	0	1	0	0
<10.00	0	0	0	0	0
<11.00	0	0	0	0	0
<12.00	0	0	0	0	0
<13.00	0	0	0	0	0
<14.00	0	0	0	0	0
<15.00	0	0	0	0	0
<16.00	0	0	0	0	0
NUMBER OF RESPONSES	9	0	104	0	0
AVERAGE R.T.	2.11	0.	2.63	0.	0.
VARIANCE OF R.T.	1.43	0.	3.00	0.	0.

Fig. 14—Response time histograms

6 PATROL CARS AND 1 SERGEANT'S CAR

SUMMARY OF RESPONSES BY NEIGHBORHOOD

PRIORITY-->	1			2			3			5			TOTALS		
NBD	NO.	AVG RT	VAR	NO.	AVG RT	VAR	NO.	AVG RT	VAR	NO.	AVG RT	VAR	NO.	AVG RT	VAR
A	0	0.	0.	0	0.	0.	10	4.000	3.000	0	0.	0.	10	4.000	3.000
B	2	2.000	0.	0	0.	0.	16	2.938	4.559	0	0.	0.	18	2.833	3.602
C	0	0.	0.	0	0.	0.	7	2.286	1.347	0	0.	0.	7	2.286	1.347
D	0	0.	0.	0	0.	0.	6	2.167	2.139	0	0.	0.	6	2.167	2.139
E	1	0.	0.	0	0.	0.	7	2.000	2.857	0	0.	0.	8	1.750	2.187
F	0	0.	0.	0	0.	0.	6	2.333	2.889	0	0.	0.	6	2.333	2.889
G	1	3.000	0.	0	0.	0.	8	2.500	2.000	0	0.	0.	9	2.556	1.580
H	2	3.500	.250	0	0.	0.	3	1.667	2.889	0	0.	0.	5	2.400	1.080
I	0	0.	0.	0	0.	0.	8	2.125	.859	0	0.	0.	8	2.125	.859
J	1	1.000	0.	0	0.	0.	15	3.400	3.973	0	0.	0.	16	3.250	3.492
K	1	1.000	0.	0	0.	0.	12	2.083	.743	0	0.	0.	13	2.000	.633
M	1	3.000	0.	0	0.	0.	6	1.833	.472	0	0.	0.	7	2.000	.347
TOTALS	9	2.111	1.432	0	0.	0.	104	2.625	3.004	0	0.	0.	113	2.584	2.553

	PICK-UP (PRIORITY 4)		
NBD	NO.	AVG RT	VAR
A	0	0.	0.
B	0	0.	0.
C	0	0.	0.
D	0	0.	0.
E	0	0.	0.
F	0	0.	0.
G	0	0.	0.
H	0	0.	0.
I	0	0.	0.
J	0	0.	0.
K	0	0.	0.
M	0	0.	0.
TOTALS	0	0.	0.

Fig. 15—Neighborhood activity summary

6 PATROL CARS AND 1 SERGEANT'S CAR

DISTRIBUTION OF CAR AVAILABILITY

NO. OF CARS ON PATROL	PER CENT OF TIME
0	29.886
1	8.427
2	14.250
3	13.111
4	18.421
5	10.210
6	4.974
7	.721

AVERAGE NUMBER AVAILABLE = 2.359
VARIANCE = 3.914

Fig. 16—Distribution of car availability

6 PATROL CARS AND 1 SERGEANT'S CAR

NUMBER OF PRECINCT CARS SENT TO CALLS

PRIORITY	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	AVERAGE	VARIANCE
1	1	3	0	6	0	0	0	0	0	0	0	0	0	0	0	0	2.10	1.29
2	0	0	0	0													0.	0.
3	0	109	0	0													1.00	0.
4	0	0	0	0													0.	0.
5	0	0	0	0													0.	0.

Fig. 17—Number of precinct cars sent to calls

Appendix A

LISTING OF THE SIMULATION PROGRAM

Note that the program begins with a section called the PREAMBLE. This section plays a special role in any SIMSCRIPT II program. In it, one defines variables, entities, attributes, lists, and events, specifies the number of dimensions for arrays, and makes provision for gathering statistics. Lines 25000 through 31700 constitute the PREAMBLE of this simulation. The PREAMBLE is followed by the MAIN routine (lines 31900-37800), which is followed by the events and subroutines that comprise the simulation program.

```
PREAMBLE ''
 LAST COLUMN IS 72  ''
 EXTERNAL EVENT UNIT IS 4
 NORMALLY MODE IS INTEGER AND DIMENSION IS 0
DEFINE $ TO MEAN IN ARRAY

 PERMANENT ENTITIES
EVERY BLOCK HAS A TAXNO $ 1,AN XCORD $ 2,A YCORD $ 3 AND A NBDID $ 4
EVERY NBD HAS A NBD.NAME $ 5,AN N.ADJACENT $ 6,
 AN N.SECTOR.CARS $ 7,A P1.NBD.RSPONSE $ 24,A P2.NBD.RSPONSE $ 25,
 A P3.NBD.RSPONSE $ 26, A P4.NBD.RSPONSE $ 27,
 AND A P5.NBD.RSPONSE $ 28
EVERY CAR HAS A CAR.NAME $ 8,AN XLOC $ 9,A YLOC $ 10,A PRI.CAR $ 11,
A NUMB.SECTORS $ 12,AN ASSIGNMENT $ 13,A T.START.RESPONSE $ 14,
A NEXT.EVENT $ 15, A (SPT(1/4),SOT(2/4),SWT(2/2)) $ 16,
A CENTROID $ 17,AN IN.SECT.JOBS $ 18,AN OUT.SECT.JOBS $ 19,
AND MAY BELONG TO AN ASSIGNED.CARS
EVERY P.CLASS OWNS A QUEUE,AND HAS A JOB.RESPONSE.TIME $ 20,
A VELOCITY $ 21,A COUNT $ 22, AND A WAIT.TIME $ 23

   DEFINE XCORD,YCORD,XLOC,YLOC,T.START.RESPONSE,VELOCITY,WAIT.TIME
  AS REAL VARIABLES
 DEFINE    NBD.NAME AND CAR.NAME AS ALPHA VARIABLES
 DEFINE ADJACENT.CARS AND SECTORS AS 2-DIMENSIONAL ARRAYS
   DEFINE P.DURATION AS A 1-DIMENSIONAL REAL ARRAY
DEFINE OUTSIDE.DISPATCH AS AN INTEGER VARIABLE
   DEFINE N.AVAILABLE AS AN INTEGER VARIABLE
DEFINE MAX.SENT, MEAL.DURATION, AND TOUR.LENGTH AS INTEGER VARIABLES
DEFINE REGION.NAME AND TITLE AS 1-DIMENSIONAL ALPHA ARRAYS

 TEMPORARY ENTITIES
 EVERY JOB HAS A LOCATION,AN ENTRY.TIME,A PRIORITY,A STATUS,
 A DURATION,MAY BELONG TO A QUEUE,AND OWNS AN ASSIGNED.CARS
 DEFINE ASSIGNED.CARS AS A FIFO SET WITHOUT FB,FA,RF AND RL ROUTINES
 DEFINE QUEUE AS A FIFO SET WITHOUT FB AND FA ROUTINES
DEFINE ENTRY.TIME AND DURATION AS REAL VARIABLES

 EVENT NOTICES INCLUDE END.OF.SIMULATION
EVERY ARRIVAL.AT.SCENE HAS A CARNUMBER IN WORD 6
EVERY CALL.END HAS A CARNUMBER IN WORD 6
EVERY OUT.OF.SERVICE HAS A CARNUMBER IN WORD 6
EVERY RETURN.TO.SECTOR HAS A CARNUMBER IN WORD 6
EXTERNAL EVENTS ARE JOB.ENTRY AND OUT.OF.SERVICE
PRIORITY ORDER IS END.OF.SIMULATION, CALL.END,
           JOB.ENTRY, OUT.OF.SERVICE

 ACCUMULATE MQ AS THE MEAN,VQ AS THE VARIANCE,AND
 PROBQ(0 TO 50 BY 1) AS THE HISTOGRAM OF N.QUEUE
ACCUMULATE UTILIZ(0 TO 6 BY 1) AS THE HISTOGRAM OF SWT
TALLY MC  AS THE MEAN,VC  AS THE VARIANCE, AND HISTC(0 TO 15 BY 1)
AS THE HISTOGRAM OF COUNT
TALLY MRT AS THE MEAN,VRT AS THE VARIANCE,NRT AS THE NUMBER AND
FREQ(0 TO 15 BY  1) AS THE HISTOGRAM OF JOB.RESPONSE.TIME
TALLY MWT AS THE MEAN, VWT AS THE VARIANCE, NWT AS THE NUMBER,
AND HWT(0 TO 240 BY 5) AS THE HISTOGRAM OF WAIT.TIME
TALLY MSRT1 AS THE MEAN,VSRT1 AS THE VARIANCE AND NSRT1 AS THE
NUMBER OF P1.NBD.RSPONSE
TALLY MSRT2 AS THE MEAN,VSRT2 AS THE VARIANCE AND NSRT2 AS THE
NUMBER OF P2.NBD.RSPONSE
TALLY MSRT3 AS THE MEAN,VSRT3 AS THE VARIANCE AND NSRT3 AS THE
```



```
 NUMBER OF P3.NBD.RSPONSE
 TALLY MSRT4 AS THE MEAN,VSRT4 AS THE VARIANCE AND NSRT4 AS THE
 NUMBER OF P4.NBD.RSPONSE
 TALLY MSRT5 AS THE MEAN,VSRT5 AS THE VARIANCE AND NSRT5 AS THE
 NUMBER OF P5.NBD.RSPONSE
 ACCUMULATE MAV AS THE MEAN,VAV AS THE VARIANCE
 AND HISTAV(0 TO 12 BY 1) AS THE HISTOGRAM OF N.AVAILABLE
 DEFINE CALCULATE.DISTANCE AS A REAL FUNCTION WITH 4 ARGUMENTS
 END

  MAIN
    DEFINE SIM.LENGTH AND MEAL.TIME AS REAL VARIABLES
 RESERVE REGION.NAME(*) AND TITLE(*) AS 20
  START NEW PAGE
 FOR I = 1 TO 20, READ TITLE(I) AS (20) A 4
 FOR I = 1 TO 20, READ REGION.NAME(I) AS (20) A 4
 READ SIM.LENGTH, MAX.SENT, MEAL.DURATION, AND TOUR.LENGTH
 WRITE AS " INPUT DATA FOR "
 FOR I= 1 TO 20, WRITE REGION.NAME(I) AS    A 4
 WRITE AS /
 SKIP 3 LINES
 PRINT 1 LINE WITH SIM.LENGTH AS FOLLOWS
 SIMULATION LENGTH =******.** HOURS
 SKIP 1 LINE
  READ N.BLOCK,N.NBD,N.CAR
    LET N.AVAILABLE=N.CAR
 PRINT 1 LINE WITH N.BLOCK,N.NBD AND N.CAR AS FOLLOWS
NO. OF BLOCKS = ****    NO. OF NBDS. = ****    NO. OF CARS = ****
  RESERVE ADJACENT.CARS(*,*) AS N.NBD BY N.CAR,AND SECTORS(*,*)
      AS N.CAR BY *
 CREATE EVERY BLOCK, NBD, CAR, AND P.CLASS(5)
 SKIP 1 OUTPUT LINE
 PRINT 1 LINE AS FOLLOWS
BLOCK ID    TAX NO.    NBD ID           X          Y
 FOR I=1 TO N.BLOCK, DO
          READ B,TAXNO(B),NBDID(B),XCORD(B),YCORD(B) USING UNIT 7
 PRINT 1 LINE WITH B,TAXNO(B),NBDID(B),XCORD(B),YCORD(B) THUS
  ****     *******    ****        **.*****   **.*****
  LOOP
    START NEW PAGE
 FOR I= 1 TO 20 , WRITE TITLE(I) AS     A 4
 WRITE AS /
 SKIP 2 LINES
 PRINT 2 LINES AS FOLLOWS
NBD ID      SECTORS        NUMBER OF          ORDER IN WHICH
                       SECT.CARS ADJ.CARS  CARS ARE NOMINATED
  FOR I=1 TO N.NBD, DO
      READ N,NBD.NAME(N),N.SECTOR.CARS(N),N.ADJACENT(N)
      FOR J=1 TO N.CAR,READ ADJACENT.CARS(N,J)
 PRINT 1 LINE WITH N,NBD.NAME(N),N.SECTOR.CARS(N),
  N.ADJACENT(N) AS FOLLOWS
 ****       ****         **        ***
        FOR K=1 TO N.CAR,DO PRINT 1 LINE WITH
 ADJACENT.CARS(I,K) AS FOLLOWS
                                                            ***
        LOOP
      LOOP
 FOR I=1 TO N.CAR     DO
 READ C,CAR.NAME(C),NUMB.SECTORS(C),CENTROID(C),MEAL.TIME
 RESERVE SECTORS(C,*) AS NUMB.SECTORS(C)
 FOR J=1 TO NUMB.SECTORS(C),   READ SECTORS(C,J)
```

```
        LET XLOC(C)=XCORD(CENTROID(C))
        LET YLOC(C)=YCORD(CENTROID(C))
   SCHEDULE AN OUT.OF.SERVICE GIVEN C IN MEAL.TIME HOURS
    LOOP
      FOR I=1 TO N.P.CLASS READ VELOCITY(I)
      RESERVE P.DURATION(*) AS 3
      FOR I = 1 TO 3 READ P.DURATION(I)
    CREATE AN END.OF.SIMULATION
   SCHEDULE THIS END.OF.SIMULATION IN SIM.LENGTH HOURS
    START SIMULATION
    STOP
    END

    EVENT FOR END.OF.SIMULATION
   DEFINE M,V,T,TIN,TOUT,TOTIN,TOTOUT AS REAL VARIABLES
   DEFINE X AS A REAL 1-DIMENSIONAL ARRAY
   RESERVE X(*) AS 10

    START NEW PAGE
      PRINT 2 DOUBLE LINES AS FOLLOWS

                  THE NEW YORK CITY-RAND INSTITUTE

                  POLICE PATROL SIMULATION MODEL
   WRITE AS "SIMULATION RESULTS FOR "
   FOR I= 1 TO 20, WRITE REGION.NAME(I) AS     A 4
   WRITE AS /
   SKIP 1 LINE
   FOR I= 1 TO 20 , WRITE TITLE(I) AS     A 4
   WRITE AS /
   SKIP 5 OUTPUT LINES
   PRINT 1 LINE AS FOLLOWS
                              CAR ACTIVITY SUMMARY
   LET IJ=0      LET OJ=0
   SKIP 1 OUTPUT LINE
    PRINT 2 DOUBLE LINES AS FOLLOWS
   CAR     NO. OF JOBS      ON PATROL       RESPONDING       WORKING
TOT. IN SERVICE
         IN       OUT     IN      OUT      IN       OUT      IN       OUT
  IN       OUT     OUT OF SERVICE
    FOR K= 1 TO N.CAR,DO
   LET TIN=(UTILIZ(K,1) +UTILIZ(K,3) +UTILIZ(K,5))/TIME.V
   LET TOUT=(UTILIZ(K,2)+UTILIZ(K,4)+UTILIZ(K,6))/TIME.V
  PRINT 1 DOUBLE LINE WITH CAR.NAME(K),IN.SECT.JOBS(K),OUT.SECT.JOBS(K),
   UTILIZ(K,1)/TIME.V,UTILIZ(K,2)/TIME.V,UTILIZ(K,3)/TIME.V,
   UTILIZ(K,4)/TIME.V,UTILIZ(K,5)/TIME.V,UTILIZ(K,6)/TIME.V,
   TIN,TOUT,UTILIZ(K,7)/TIME.V AS FOLLOWS
   ***    ****     ****    *.***  *.***    *.***    *.***    *.***    *.***
*.***    *.***         *.***
   FOR J=1 TO 7, DO    ADD UTILIZ(K,J)/TIME.V TO X(J)     LOOP
   ADD IN.SECT.JOBS(K) TO IJ
   ADD OUT.SECT.JOBS(K) TO OJ
      ADD TIN TO TOTIN
      ADD TOUT TO TOTOUT
    LOOP
   FOR J=1 TO 7 DO       LET X(J)=X(J)/N.CAR    LOOP
   LET TOTIN=TOTIN/N.CAR          LET TOTOUT=TOTOUT/N.CAR
   SKIP 1 OUTPUT LINE
    PRINT 1 DOUBLE LINE WITH IJ,OJ,X(1),X(2),X(3),X(4),X(5),X(6),
        TOTIN,TOTOUT,X(7) THUS
  TOTALS  ****     ****    *.***  *.***    *.***    *.***    *.***    *.***
```



```
*.***    *.***        *.***

    FOR I=2 TO N.P.CLASS,    DO    START NEW PAGE
    FOR J= 1 TO 20, WRITE TITLE(J) AS A 4
    WRITE AS /
    SKIP 2 LINES
     PRINT 1 LINE WITH I AS FOLLOWS
         QUEUEING STATISTICS FOR PRIORITY CLASS **
     SKIP 2 OUTPUT LINES
     PRINT 2 LINES WITH MQ(I) AND VQ(I) AS FOLLOWS
      HISTOGRAM OF QUEUE-SIZE           AVERAGE QUEUE-SIZE = ***.***
   NO. IN QUEUE    FREQUENCY            VARIANCE             ****.***
     SKIP 1 OUTPUT LINE
         FOR J= 1 TO 30 , WITH PROBQ(I,J) GT 0, DO
         PRINT 1 LINE WITH J-1 AND PROBQ(I,J)/TIME.V AS FOLLOWS
  **             *.***
     LOOP

    SKIP 2 OUTPUT LINES
    LET T=NWT(I)*MWT(I)
    LET NTOT=NRT(I)
     IF NTOT=0  LET T=0    GO TO PRT
    OTHERWISE LET T=T/NTOT
 'PRT'  PRINT 3 DOUBLE LINES WITH NWT(I),MWT(I),VWT(I),NTOT,T THUS
     HISTOGRAM OF WAITING TIMES          AVERAGE WAITING TIME OF THE ***
JOBS DELAYED= ***.***
    WAITING TIME         FREQUENCY
VARIANCE     = ***.***
     (MINUTES)                           AVERAGE WAITING TIME OF ALL ****
JOBS DISP'D = ***.***
         FOR J=1 TO 24, WITH HWT(I,J) GT 0, DO
               PRINT 1 LINE WITH 5*J AND HWT(I,J) THUS
      < ***           *****
     LOOP
    SKIP 3 OUTPUT LINES
    LET J=1
    PRINT 3 LINES AS FOLLOWS
     CURRENT QUEUE CONTENTS
  POSITION            TIME SINCE RECEIPT
                          (MINUTES)
    SKIP 1 OUTPUT LINE
    FOR EVERY JOB IN QUEUE(I), DO
    PRINT 1 LINE WITH J AND (TIME.V-ENTRY.TIME(JOB))*1440.0 THUS
    ***             ***.**
    ADD 1 TO J
    LOOP
      LOOP
    CALL RESULTS
    END     ''SIMULATION''
    ROUTINE FOR RESULTS
    DEFINE M,V,T  AS REAL VARIABLES
     BEGIN REPORT PRINTING FOR I=1 TO N.P.CLASS IN GROUPS OF 6 PER PAGE
    BEGIN HEADING
    FOR J= 1 TO 20, WRITE TITLE(J) AS A 4
    WRITE AS /
    SKIP 2 LINES
    PRINT 1 LINE AS FOLLOWS
      RESPONSE TIME HISTOGRAMS
     SKIP 2 OUTPUT LINES
    END ''HEADING''
    PRINT 1 DOUBLE LINE WITH A GROUP OF I FIELDS THUS
```

```
    PRIORITY CLASS        **          **          **          **          **
   **
    SKIP 1 OUTPUT LINE
    PRINT 2 LINES AS FOLLOWS
     RESPONSE TIME
     (MINUTES)
        FOR J= 1 TO 16, DO
   PRINT 1 DOUBLE LINE WITH J AND A GROUP OF FREQ(I,J) FIELDS THUS
    <**.**            ****        ****        ****        ****        ****
  ****
    LOOP
    SKIP 1 OUTPUT LINE
   PRINT 1 DOUBLE LINE WITH A GROUP OF NRT(I) FIELDS THUS
NUMBER OF RESPONSES    *****       *****       *****       *****       *****
 *****
   PRINT 1 DOUBLE LINE WITH A GROUP OF MRT(I) FIELDS THUS
 AVERAGE R.T.             **.**       **.**       **.**       **.**       **.**
 **.**
   PRINT 1 DOUBLE LINE WITH A GROUP OF VRT(I) FIELDS THUS
 VARIANCE OF R.T.      ***.**      ***.**      ***.**      ***.**      ***.**
***.**
   END  ''REPORT''
   START NEW PAGE
   FOR I= 1 TO 20 , WRITE TITLE(I) AS    A 4
   WRITE AS /
   SKIP 2 LINES
   PRINT 1 DOUBLE LINE AS FOLLOWS
                                                   SUMMARY OF RESPONSES BY
 NEIGHBORHOOD
   SKIP 2 OUTPUT LINES
   PRINT 2 DOUBLE LINES AS FOLLOWS
     PRIORITY-->    1                          2                          3
                 5                        TOTALS
  NBD           NO. AVG RT    VAR          NO. AVG RT    VAR          NO. AVG RT
 VAR          NO. AVG RT    VAR          NO.  AVG RT    VAR
   SKIP 1 OUTPUT LINE
   FOR J=1 TO N.NBD DO
   LET N1=NSRT1(J) LET N2=NSRT2(J) LET N3=NSRT3(J) LET N4=NSRT5(J)
    LET T=N1+N2+N3+N4
   LET M=N1*MSRT1(J)+N2*MSRT2(J)+N3*MSRT3(J)+N4*MSRT5(J)
   LET V=N1*N1*VSRT1(J)+N2*N2*VSRT2(J)+N3*N3*VSRT3(J)+N4*N4*VSRT5(J)
    IF T=0  LET M=0  LET V=0  GO TO PRI          OTHERWISE
    LET M=M/T
    LET V=V/(T*T)
'PRI' PRINT 1 DOUBLE LINE WITH NBD.NAME(J),N1,MSRT1(J), VSRT1(J),N2,
   MSRT2(J),VSRT2(J),N3,MSRT3(J),VSRT3(J),N4,MSRT5(J),VSRT5(J),T,M,V
    THUS
  ****        **** **.*** **.***      **** **.*** **.***      **** **.*** *
*.***     **** **.*** **.***      ***** **.*** **.***
   LOOP
   SKIP 1 OUTPUT LINE
   LET T=0  LET M=0   LET V=0
   FOR I=1 TO 4  DO
        IF I EQ 4, LET I = 5 REGARDLESS
        LET N=NRT(I)          ADD N TO T
        ADD N*MRT(I) TO M   ADD N*N*VRT(I) TO V
   LOOP
   LET M=M/T       LET V=V/(T*T)
   PRINT 1 DOUBLE LINE WITH NRT(1),MRT(1),VRT(1),NRT(2),MRT(2),
   VRT(2),NRT(3),MRT(3),VRT(3),NRT(5),MRT(5),VRT(5),T,M,V THUS
 TOTALS       **** **.*** **.***      **** **.*** **.***      **** **.*** *
```



```
*.***     **** **.*** **.***      ***** **.*** **.***
    SKIP 9 OUTPUT LINES
    PRINT 2 LINES AS FOLLOWS
            PICK-UP (PRIORITY 4)
  NBD          NO. AVG RT   VAR
    SKIP 1 OUTPUT LINE
    FOR J=1 TO N.NBD DO
    LET N5 = NSRT4(J)
    PRINT 1 LINE WITH NBD.NAME(J),N5,MSRT4(J),VSRT4(J) AS FOLLOWS
  ****         **** **.*** **.***
    LOOP
    SKIP 1 OUTPUT LINE
    PRINT 1 LINE WITH NRT(4), MRT(4), VRT(4) THUS
 TOTALS        **** **.*** **.***
    START NEW PAGE
    FOR I= I TO 20 , WRITE TITLE(I) AS    A 4
    WRITE AS /
    SKIP 2 LINES
    PRINT 1 LINE AS FOLLOWS
      DISTRIBUTION OF CAR AVAILABILITY
    SKIP 2 OUTPUT LINES
    PRINT, 2 LINES AS FOLLOWS
          NO. OF CARS       PER CENT
          ON PATROL         OF TIME
    FOR I=0 TO N.CAR DO
      PRINT 1 LINE WITH I AND 100*HISTAV(I+1)/TIME.V THUS
            **                 *.***
    LOOP
      SKIP 1 OUTPUT LINE
    PRINT 1 LINE WITH MAV THUS
        AVERAGE NUMBER AVAILABLE = **.***
    PRINT 1 LINE WITH VAV THUS
                       VARIANCE  =***.***
     START NEW PAGE
    FOR I= 1 TO 20 , WRITE TITLE(I) AS    A 4
    WRITE AS /
    SKIP 2 LINES
      PRINT 1 LINE AS FOLLOWS
  NUMBER OF PRECINCT CARS SENT TO CALLS
    SKIP 2 OUTPUT LINES
      PRINT 1 DOUBLE LINE THUS
  PRIORITY     0      1      2      3      4      5      6      7      8      9
 10     11     12     13     14     15     AVERAGE     VARIANCE
    WRITE AS /,"       1  "
      FOR I=1 TO 16,WRITE HISTC(1,I) AS I 6
    WRITE MC(1) AND VC(1) AS S 6,D(6,2),S 5,D(7,2),/
    FOR I=2 TO N.P.CLASS, DO
    PRINT 1 DOUBLE LINE WITH I,HISTC(I,1),HISTC(I,2),HISTC(I,3),
        HISTC(I,4),MC(I), AND VC(I) THUS
     **     ****  ****  ****  ****
                                                ***.**      ****.**
    LOOP
    STOP
     END

EVENT FOR JOB.ENTRY
    DEFINE HDUR AND MDUR AS REAL VARIABLES
    READ LOC,IPR,HDUR AND MDUR
       CREATE A JOB CALLED J
       LET ENTRY.TIME(J)=TIME.V
       LET LOCATION(J)=LOC
```

```
      LET PRIORITY(J)=IPR
   LET DURATION(J)=HDUR*60.0 + MDUR
      LET STATUS(J)=0
      GO TO PR(IPR)
'PR(1)'    CALL DISPR1(J)       RETURN
'PR(2)'    CALL DISPR2(J)       RETURN
'PR(3)'    CALL DISPR3(J)       RETURN
'PR(4)'    CALL DISPR4(J)       RETURN
'PR(5)'    CALL DISPR5(J)       RETURN
      END
   ROUTINE TO DISPR1 GIVEN J
   LET SECT=NBDID(LOCATION(J))           LET MARKER=0
       FOR I=1 TO N.CAR,DO ''NOMINATE CARS ''
                   LET K=ADJACENT.CARS(SECT,I)
               IF SWT(K) LE 3 AND SPT(K) NE 1  ''CAR IS ELIGIBLE ''
               IF MARKER=0 LET SOT(K)=1
                       LET MARKER=1
                       GO TO CALC1
               OTHERWISE   IF MARKER = 1 LET SOT(K)=2
                                         LET MARKER=2
                                         GO TO CALC1
               OTHERWISE IF MARKER = MAX.SENT GO OUT
               OTHERWISE ADD 1 TO MARKER LET SOT(K)=3
'CALC1' IF SWT(K)=1 CALL INTERPOLATE.LOCATION GIVEN K AND ASSIGNMENT(K)
        REGARDLESS
            IF SPT(K) GT 1 CALL PREEMPT(K)
            REGARDLESS CALL ASSIGN(K,J)  LET SWT(K)=2
               IF I GT N.SECTOR.CARS(SECT) LET SWT(K)=3
               REGARDLESS
            REGARDLESS
         LOOP
'OUT'  IF MARKER = 0 ADD 1 TO OUTSIDE.DISPATCH
                        DESTROY JOB CALLED J
         REGARDLESS LET COUNT(1)=MARKER
         RETURN
   END
   ROUTINE TO DISPR2 GIVEN J
   LET SECT=NBDID(LOCATION(J))           LET MARKER=0
   FOR I=1 TO N.CAR, DO       ''NOMINATE CARS ''
        LET K=ADJACENT.CARS(SECT,I)
            IF SPT(K)=0       '' CAR IS ELIGIBLE ''
               IF MARKER=0 LET SOT(K)=1  LET MARKER=1 GO TO CALC2
               OTHERWISE
              IF MARKER=1 AND I LE N.ADJACENT(SECT)+N.SECTOR.CARS(SECT)
               LET SOT(K)=2 LET MARKER=2
'CALC2'             IF SWT(K)=1 CALL INTERPOLATE.LOCATION GIVEN K AND
                    ASSIGNMENT(K)
                    REGARDLESS CALL ASSIGN(K,J) LET SWT(K)=2
                    IF I GT N.SECTOR.CARS(SECT) LET SWT(K)=3
                    REGARDLESS
               REGARDLESS
      REGARDLESS
         LOOP
      IF MARKER=0 FILE J IN QUEUE(2)    RETURN
     OTHERWISE LET COUNT(2)=MARKER RETURN
   END
   ROUTINE TO DISPR3 GIVEN J
   '' SEND ANY AVAILABLE CAR ''
   LET SECT=NBDID(LOCATION(J))
   FOR I=1 TO N.CAR ,  DO
        LET K=ADJACENT.CARS(SECT,I)
```



```
        IF SPT(K)=0      ''CAR IS ELIGIBLE''
            LET SOT(K)=1
            IF SWT(K)=1 CALL INTERPOLATE.LOCATION GIVEN K AND
                 ASSIGNMENT(K)
            REGARDLESS CALL ASSIGN(K,J)
            LET SWT(K)=2
            IF I GT N.SECTOR.CARS(SECT) LET SWT(K)=3
            REGARDLESS LET COUNT(3)=1  RETURN
        OTHERWISE LOOP
FILE J IN QUEUE(3)
RETURN
END
ROUTINE TO DISPR4 GIVEN J
     '' PICK-UP JOB ''
LET SECT = NBDID(LOCATION(J))
FOR I = 1 TO N.SECTOR.CARS(SECT), DO
     LET K = ADJACENT.CARS(SECT,I)
     IF SPT(K) = 0        ''CAR IS AVAILABLE''
          LET XLOC(K) = XCORD(LOCATION(J))
          LET YLOC(K) = YCORD(LOCATION(J))
          CALL ASSIGN(K,J)
          LET SWT(K) = 2      LET SOT(K) = 1
          LET COUNT(4) = 1
          RETURN
     OTHERWISE
LOOP
FILE J IN QUEUE(4)         ''NO SECTOR CAR FREE''
RETURN
END

ROUTINE TO DISPR5 GIVEN J
'' ASSIGN JOB ONLY TO SECTOR CAR ''
LET SECT=NBDID(LOCATION(J))
FOR I= 1 TO N.SECTOR.CARS(SECT),   DO    ''LOOK FOR A SECTOR CAR''
       LET K=ADJACENT.CARS(SECT,I)
       IF SPT(K)=0        ''CAR IS ELIGIBLE''
           IF SWT(K)=1 CALL INTERPOLATE.LOCATION GIVEN K AND
                     ASSIGNMENT(K)
                REGARDLESS CALL ASSIGN(K,J) LET SWT(K)=2
        LET SOT(K)=1     LET COUNT(5)=1    RETURN
           OTHERWISE
       LOOP
    FILE J IN QUEUE(5)          ''NO SECTOR CAR FREE''
    RETURN
END

ROUTINE TO INTERPOLATE.LOCATION GIVEN K AND J
   '' CALLED FROM PREEMPT AND FROM JOB.ENTRY ''
NORMALLY MODE IS REAL
DEFINE LJ,K,J,R,PATROL.RETURN AS INTEGER VARIABLES
LET LJ=LOCATION(J)
LET V=VELOCITY(PRIORITY(J))
LET XD=XCORD(LJ)-XLOC(K)
LET YD=YCORD(LJ) - YLOC(K)
LET T=(TIME.V - T.START.RESPONSE(K))*24
LET TX=ABS.F(XD)/V
LET DELT=T - TX
IF DELT GT 0 LET XLOC(K)=XCORD(LJ)
           LET YLOC(K)=V*DELT*SIGN.F(YD)+YLOC(K)
    GO TO CHECK.IF.PATROL
OTHERWISE LET XLOC(K)=V*(-DELT)*SIGN.F(XD)+XLOC(K)
```

```
'CHECK.IF.PATROL'  IF SWT(K) NE 1 RETURN
                   OTHERWISE CALL CANCEL.SECTOR.RETURN GIVEN K
                   RETURN
END

ROUTINE TO ASSIGN(K,J)
LET SPT(K)=PRIORITY(J)
LET ASSIGNMENT(K)=J
NOW SCHEDULE.CAR.ARRIVAL(K,J)
FILE    K  IN ASSIGNED.CARS(J)
  IF SWT(K) LE 1 SUBTRACT 1 FROM N.AVAILABLE REGARDLESS
RETURN
END

ROUTINE TO CANCEL.SECTOR.RETURN GIVEN K
LET PATROL.RETURN = ASSIGNMENT(K)
 DESTROY JOB CALLED PATROL.RETURN
LET R=NEXT.EVENT(K)
CANCEL RETURN.TO.SECTOR CALLED R
DESTROY RETURN.TO.SECTOR CALLED R
RETURN
END

ROUTINE TO PREEMPT(K)
LET J = ASSIGNMENT(K)
IF J IS NOT IN A QUEUE
    IF STATUS(J)=0 FILE J FIRST IN QUEUE(PRIORITY(J))
    REGARDLESS
REGARDLESS LET E= NEXT.EVENT(K) CANCEL THE ARRIVAL.AT.SCENE CALLED E
DESTROY THE ARRIVAL.AT.SCENE CALLED E
REMOVE K FROM ASSIGNED.CARS(J)  LET SPT(K)=0
NOW INTERPOLATE.LOCATION GIVEN K AND J
IF N.ASSIGNED.CARS(J) =1 LET F=F.ASSIGNED.CARS(J)
    IF SOT(F)=2 AND SWT(F) GE 4
    LET E = NEXT.EVENT(F)
    CANCEL THE CALL.END CALLED E
    RESCHEDULE THE CALL.END CALLED E IN DURATION(J) MINUTES
 REGARDLESS
    LET SOT(F)=1
REGARDLESS
LET T.START.RESPONSE(K)= TIME.V
RETURN
END

ROUTINE TO SCHEDULE.CAR.ARRIVAL(K,J)
DEFINE DIST AS A REAL VARIABLE
LET DIST=CALCULATE.DISTANCE(XCORD(LOCATION(J)),YCORD(LOCATION(J)),
XLOC(K),YLOC(K))
LET T.START.RESPONSE(K)=TIME.V
CREATE AN ARRIVAL.AT.SCENE
SCHEDULE THIS ARRIVAL.AT.SCENE GIVEN K IN
DIST/VELOCITY(PRIORITY(J)) HOURS
LET NEXT.EVENT(K)=ARRIVAL.AT.SCENE
RETURN
END

EVENT FOR ARRIVAL.AT.SCENE(K)
DEFINE D AS A REAL VARIABLE
LET LKAR=LOCATION(ASSIGNMENT(K))
LET XLOC(K)=XCORD(LKAR)
LET YLOC(K)=YCORD(LKAR)
```



```
    IF SWT(K)=2   LET SWT(K)=4 ADD 1 TO IN.SECT.JOBS(K) GO TO BOTH
    OTHERWISE     LET SWT(K)=5 ADD 1 TO OUT.SECT.JOBS(K)
'BOTH'
    LET J=ASSIGNMENT(K)
    LET JPR=PRIORITY(J)
    LET D = DURATION(J)
    IF SOT(K)=1 ''THIS IS THE PRIMARY CAR AT JOB J'' GO  TO PRIME.CAR
    OTHERWISE
    IF STATUS(J)=0  ''NO CAR HAS ARRIVED AND THIS CAR IS NOT PRIMARY''
    GO TO FIRST.CAR
    OTHERWISE
    IF STATUS(J)=1  '' PRIMARY CAR HAS ALREADY ARRIVED  ''
    PERFORM END.CALL.SCHEDULING GIVEN K AND D
    REGARDLESS RETURN '' IF NO PRIMARY CAR DELAY SCHEDULING A CALL.END''
'FIRST.CAR'
    LET STATUS(J)=2
    LET JOB.RESPONSE.TIME(JPR)=(TIME.V -
     T.START.RESPONSE(K))*1440.0
    GO TO RT(JPR)
'RT(1)' LET P1.NBD.RSPONSE(NBDID(LKAR))=JOB.RESPONSE.TIME(JPR) RETURN
'RT(2)' LET P2.NBD.RSPONSE(NBDID(LKAR))=JOB.RESPONSE.TIME(JPR) RETURN
'RT(3)' LET P3.NBD.RSPONSE(NBDID(LKAR))=JOB.RESPONSE.TIME(JPR) RETURN
'RT(4)' LET P4.NBD.RSPONSE(NBDID(LKAR))=JOB.RESPONSE.TIME(JPR) RETURN
'RT(5)' LET P5.NBD.RSPONSE(NBDID(LKAR))=JOB.RESPONSE.TIME(JPR) RETURN
'PRIME.CAR'
    IF STATUS(J) NE 0 '' SOME CARS ALREADY THERE''
        FOR EACH CAR IN ASSIGNED.CARS(J) WITH SWT(CAR) GE 4  DO
        PERFORM END.CALL.SCHEDULING GIVEN CAR AND D
        LOOP  LET STATUS(J)=1      RETURN
    OTHERWISE
    LET JOB.RESPONSE.TIME(JPR)=(TIME.V -
     T.START.RESPONSE(K))*1440.0
    CALL END.CALL.SCHEDULING GIVEN K AND D
    LET STATUS(J)=1
    GO TO RT(JPR)
    END

    ROUTINE TO CALCULATE.DISTANCE GIVEN X1,Y1,X2 AND Y2
    DEFINE D,X1,X2,Y1, AND Y2 AS REAL VARIABLES
    LET D = ABS.F(X2-X1) + ABS.F(Y2-Y1)
    RETURN WITH D
    END

    ROUTINE FOR END.CALL.SCHEDULING GIVEN K AND DUR
   DEFINE T AND DUR AS REAL VARIABLES
    LET T=P.DURATION(SOT(K))*DUR
    CREATE A CALL.END
    SCHEDULE THIS CALL.END GIVEN K IN T MINUTES
    RETURN
    END
    EVENT CALL.END GIVEN K
    IF SWT(K)=6 GO TO CHECK.Q    ''END OF OUT-OF-SERVICE''
     OTHERWISE LET J=ASSIGNMENT(K)
    REMOVE K FROM ASSIGNED.CARS(J)
    IF ASSIGNED.CARS(J) IS EMPTY DESTROY JOB CALLED J
    REGARDLESS
'CHECK.Q'
        FOR EACH JOB ON QUEUE(1) ''THIS IS THE OUT-OF-SERVICE QUEUE''
        WITH LOCATION(JOB)=K, DO LET ASSIGNMENT(K)=JOB  LET SWT(K)=6
   SCHEDULE A CALL.END GIVEN K IN DURATION(JOB) MINUTES
   LET SPT(K)=6  LET SOT(K)=6
```

```
LET XLOC(K)=XCORD(CENTROID(K))
LET YLOC(K)=YCORD(CENTROID(K))
    REMOVE JOB FROM QUEUE(1)
    RETURN
    LOOP
FOR I=2 TO N.P.CLASS  DO
  IF QUEUE(I) IS EMPTY   GO TO NEXT.Q
  OTHERWISE IF I LE 3 NOW REASSIGN.CAR.TO.JOB(K,F.QUEUE(I))   RETURN
            OTHERWISE    ''MUST FIND A SECTOR CAR''
               FOR EACH JOB ON QUEUE(I) DO
                         LET NHOOD=NBDID(LOCATION(JOB))
                         FOR J=1 TO NUMB.SECTORS(K)     DO
                              IF NHOOD=SECTORS(K,J)
                                   NOW REASSIGN.CAR.TO.JOB(K,JOB)
                                   RETURN
                              OTHERWISE
                         LOOP
               LOOP
'NEXT.Q'  LOOP
'PREVENT.PATROL'
  ADD 1 TO N.AVAILABLE
NOW PLACE.CAR.ON.PATROL(K)
RETURN
END

ROUTINE TO REASSIGN.CAR.TO.JOB GIVEN K AND J
LET SECT=NBDID(LOCATION(J))
IF PRIORITY(J)=4 LET XLOC(K)=XCORD(LOCATION(J))
                 LET YLOC(K)=YCORD(LOCATION(J))  REGARDLESS
NOW ASSIGN(K,J)
FOR I = 1 TO N.CAR DO
LET KK=ADJACENT.CARS(SECT,I)
    IF KK NE K GO TO LOOP1
    OTHERWISE LET SWT(K) =2
        IF I GT N.SECTOR.CARS(SECT) LET SWT(K)=3
        REGARDLESS GO TO OUT
'LOOP1'
    LOOP
'OUT'
LET PRI=PRIORITY(J)
REMOVE J FROM QUEUE(PRI)
LET WAIT.TIME(PRI) = (TIME.V - ENTRY.TIME(J))*1440.0
LET COUNT(PRI)=1
LET SOT(K)=1
RETURN
END
ROUTINE TO PLACE.CAR.ON.PATROL GIVEN K
DEFINE D AS A REAL VARIABLE
LET SPT(K)=0 LET SOT(K)=0
IF SWT(K)=4 OR SWT(K)=6  LET SWT(K)=0  RETURN   ''CAR WAS IN SECTOR'
'' ALREADY.... LEAVE HIM THERE ''
OTHERWISE
CREATE A JOB CALLED PATROL.RETURN
LET SWT(K)=1 LET LJ=CENTROID(K) LET LOCATION(PATROL.RETURN)=LJ
LET ASSIGNMENT(K)=PATROL.RETURN
 LET PRIORITY(PATROL.RETURN)=3
LET D = CALCULATE.DISTANCE(XCORD(LJ),YCORD(LJ),XLOC(K),YLOC(K))
LET T.START.RESPONSE(K)=TIME.V
CREATE A RETURN.TO.SECTOR CALLED R
LET CARNUMBER(R)=K   ''THE CAR NUMBER OF THIS PATROL.RETURN''
SCHEDULE THE RETURN.TO.SECTOR CALLED R IN D/VELOCITY(3) HOURS
```



```
LET NEXT.EVENT(K)=R
RETURN
END

EVENT RETURN.TO.SECTOR GIVEN K
 LET J=ASSIGNMENT(K)
LET XLOC(K)=XCORD(CENTROID(K))
LET YLOC(K)=YCORD(CENTROID(K))
LET SWT(K)=0
DESTROY JOB CALLED J
DESTROY THIS RETURN.TO.SECTOR
RETURN
END

EVENT OUT.OF.SERVICE GIVEN K SAVING THE EVENT NOTICE
 DEFINE T.OUT.OF.SERVICE AS A REAL VARIABLE
 LET X.MEAL = OUT.OF.SERVICE
 IF EUNIT.A(OUT.OF.SERVICE) EQ 0
 SCHEDULE AN OUT.OF.SERVICE GIVEN K IN TOUR.LENGTH HOURS
 LET T.OUT.OF.SERVICE=MEAL.DURATION  GO AROUND  OTHERWISE READ K,
 T.OUT.OF.SERVICE
'AROUND'
DESTROY THE OUT.OF.SERVICE CALLED X.MEAL
IF SWT(K) GE 2 GO TO DELAY
OTHERWISE
    IF SWT(K)=1
    CALL CANCEL.SECTOR.RETURN  GIVEN K
    LET XLOC(K)=XCORD(CENTROID(K))
    LET YLOC(K)=YCORD(CENTROID(K))
    REGARDLESS
    SUBTRACT 1 FROM N.AVAILABLE
    LET SWT(K)=6  LET SPT(K)=6  LET SOT(K)=6
    CREATE CALL.END
     SCHEDULE THIS CALL.END GIVEN K IN T.OUT.OF.SERVICE MINUTES
RETURN
'DELAY'
FOR EVERY JOB ON QUEUE(1) DO
IF LOCATION(JOB)=K GO TO ALREADY.SCHEDULED
OTHERWISE LOOP

CREATE JOB
LET LOCATION(JOB)=K
LET ENTRY.TIME(JOB)=TIME.V
LET DURATION(JOB)=T.OUT.OF.SERVICE
LET PRIORITY(JOB)=6
LET STATUS(JOB)=6
FILE JOB IN QUEUE(1)
RETURN
'ALREADY.SCHEDULED'
ADD T.OUT.OF.SERVICE TO DURATION(JOB)    RETURN
END
```

CONTINUED

1 OF 2

Appendix B

SAMPLE CONTROL CARDS AND INITIALIZATION DECK FOR THE 71ST PRECINCT IN NEW YORK CITY

The following control cards are sufficient for running the police patrol simulation, assuming that

(1) the compiled program has been named HPATROL;

(2) the output is to be listed on logical unit 3;

(3) the job stream is to be read from logical unit 4;

(4) the initialization data (except for the BLOCK data) is to be read from logical unit 5;

(5) the initialization data for the BLOCKs is to be read from logical unit 7.

```
//GO EXEC PGM=HPATROL,REGION=150K
//GO.SIMU03 DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=1729)
//GO.SIMU04 DD DISP=SHR,DSN=RA701.R9057.PD.SHIFT3,UNIT=ONLINE,
// VOL=SER=RES306
//GO.SIMU05 DD DDNAME=SYSIN
//GO.SIMU07 DD DISP=SHR,DSN=RA701.R9057.HBE.TXBL71,UNIT=ONLINE,
// VOL=SER=RES333
//GO.SIMU17 DD DISP=SHR,DSN=A001.SIM2.ERRLIB
```

The following cards constitute the complete initialization deck, except for the BLOCK data, for the sample simulation of the 71st Precinct in New York City.

```
//GO.SYSIN DD *
            6 PATROL CARS AND 1 SERGEANT'S CAR
            THE 71ST PRECINCT, BROOKLYN, NEW YORK
 16.0    3      60    8
  305  12    7
1 A    1     1
 1 2 3 4 5 6 7
2 B    1     1
1 2 3 4 5 6  7
3  C    1     1
2 3 4 5 6 1  7
4  D    1     1
2 3 4 5 6 1  7
5 E    1     1
3 1 4 2 6 5  7
6  F    1     1
3 1 4 2 6 5  7
7  G    1     1
5 2 6 4 3 1  7
8  H    1     1
4 6 3 2 5 1  7
9  I    1     1
4 6 3 2 5 1  7
10  J    1     1
5 2 6 4 3 1  7
11  K    1     1
6 5 4 2 3 1  7
12  M    1     1
6 5 4 2 3 1  7
1  AB  2    144  3.0
1   2
2  CD  2    222  4.0
3   4
3  EF  2    100  5.0
5   6
4  HI  2    43   3.0
8   9
5  JG  2    296  4.0
7   10
6  KM  2    63   5.0
11  12
7  SGT  1  59  3.5
0
  20.0  20.0  20.0  20.0 20.0
  1.0  0.5  0.0
```


The following represents the initialization data for the 305 BLOCKs in the simulation of Precinct 71. This is the SIMOU7 data set.

Block ID	Tax No.	NBD ID	X	Y
1	1185	12	.81170	2.18140
2	1187	12	.79450	2.12950
3	1188	12	.76100	2.07780
4	1189	12	.76830	2.00850
5	1190	12	.75940	1.94630
6	1192	12	.74620	1.83550
7	1196	12	.73350	1.73110
8	1197	11	.69360	1.65230
9	1198	11	.71180	1.58130
10	1266	12	.96310	2.12540
11	1267	12	1.08660	2.09430
12	1268	9	1.21150	2.07400
13	1269	9	1.35220	2.05450
14	1270	9	1.49190	2.03140
15	1271	9	1.63190	2.01260
16	1272	6	1.77100	1.98720
17	1273	12	.94520	2.07420
18	1274	12	1.07370	2.04180
19	1275	9	1.20350	2.02170
20	1276	9	1.34540	2.00060
21	1277	9	1.48500	1.97950
22	1278	9	1.62440	1.95770
23	1279	6	1.76420	1.93680
24	1280	12	.92480	2.01610
25	1281	12	1.05880	1.98270
26	1282	9	1.19540	1.96240
27	1283	9	1.33520	1.94160
28	1284	9	1.47580	1.92010
29	1285	9	1.61570	1.89770
30	1286	6	1.75480	1.87730
31	1287	12	.90450	1.95950
32	1288	12	1.04540	1.92460
33	1289	8	1.18560	1.90230
34	1290	8	1.32660	1.88150
35	1291	8	1.46690	1.86140
36	1292	8	1.60610	1.83970
37	1293	5	1.74740	1.81760
38	1294	12	.88370	1.90050
39	1295	12	1.02960	1.86610
40	129601	8	1.14210	1.84800
41	129602	8	1.21210	1.83870
42	1297	8	1.31790	1.82290
43	1298	8	1.45730	1.79970
44	1299	8	1.59730	1.78100
45	1300	5	1.73730	1.75870
46	1301	12	.80640	1.82220

Internal ID No.	Tax Block No.	Neigh-bor-hood	Coordinates X	Coordinates Y
47	1302	12	.87830	1.81010
48	130401	12	.95710	1.79960
49	130402	12	1.00830	1.79570
50	130403	12	1.06330	1.79180
51	1305	8	1.16870	1.78460
52	130601	12	.77060	1.73210
53	130602	12	.85110	1.73100
54	1307	12	1.00270	1.73130
55	1308	8	1.16010	1.73150
56	130901	8	1.26380	1.75460
57	130902	8	1.33160	1.75960
58	130903	8	1.34150	1.72010
59	1310	8	1.45020	1.74000
60	1311	8	1.58850	1.72220
61	1312	5	1.72900	1.70080
62	1313	11	.82030	1.67640
63	1314	11	.99810	1.67710
64	1315	11	1.15810	1.67760
65	1316	8	1.30580	1.67730
66	1317	8	1.44560	1.67720
67	1318	11	.81620	1.62650
68	1319	11	.99820	1.62640
69	1320	11	1.15740	1.62750
70	1321	8	1.30670	1.62610
71	1322	8	1.44200	1.62670
72	1323	8	1.52150	1.64870
73	1324	8	1.55640	1.64640
74	1325	8	1.61270	1.64270
75	1326	5	1.71830	1.63390
76	1327	11	.81830	1.57220
77	1328	11	.99750	1.57280
78	1329	11	1.15790	1.57220
79	1330	4	1.30760	1.57490
80	1331	4	1.43800	1.57330
81	1332	4	1.56730	1.57320
82	1333	4	1.71040	1.57480
83	1394	6	1.91240	1.96530
84	1395	6	2.04970	1.94250
85	1396	6	2.18520	1.92190
86	1397	6	2.32580	1.90050
87	1398	2	2.45530	1.79940
88	139901	2	2.62920	1.85060
89	139902	2	2.63780	1.80370
90	1400	6	1.90440	1.91300
91	1401	6	2.04200	1.89240
92	1402	6	2.17770	1.87080
93	1403	6	2.31880	1.84980
94	1405	2	2.55820	1.81830
95	1406	6	1.89670	1.85610
96	1407	6	2.03370	1.83450
97	1408	6	2.16860	1.81340
98	1409	6	2.31010	1.78990
99	1412	5	1.88650	1.79710
100	1413	5	2.02310	1.77590
101	1414	5	2.15970	1.75460

Internal ID No.	Tax Block No.	Neigh-bor-hood	Coordinates X	Coordinates Y
102	141501	5	2.28210	1.73600
103	141502	5	2.35200	1.72440
104	1417	5	1.87830	1.73790
105	1418	5	2.01610	1.71720
106	1419	5	2.15180	1.69550
107	142001	5	2.28330	1.67640
108	142002	5	2.35070	1.67000
109	1422	5	1.86820	1.67910
110	1423	5	2.00810	1.65790
111	1424	5	2.14620	1.63810
112	1425	5	2.25270	1.62550
113	1426	5	1.85940	1.62390
114	1427	5	2.00810	1.62390
115	1428	4	1.84840	1.57600
116	1429	4	1.99180	1.57620
117	1430	4	2.13790	1.57670
118	3508	2	2.65480	1.72790
119	4588	4	2.13750	1.52400
120	458901	4	2.08990	1.47750
121	458902	4	2.16220	1.47830
122	4590	4	2.13740	1.42910
123	4591	4	2.23600	1.48140
124	4592	4	2.28500	1.43570
125	4593	2	2.29240	1.51300
126	4594	2	2.33320	1.53740
127	4595	2	2.37400	1.56140
128	4596	2	2.41740	1.58400
129	4597	2	2.45710	1.60800
130	4598	2	2.50370	1.63550
131	4599	2	2.55220	1.66030
132	4600	2	2.60180	1.69050
133	4602	3	2.13480	1.32890
134	4603	3	2.18570	1.32870
135	4604	3	2.23660	1.32850
136	4605	3	2.28370	1.32860
137	4606	3	2.33190	1.32860
138	4607	3	2.38110	1.29030
139	4608	3	2.41630	1.26550
140	4609	2	2.38960	1.37600
141	4610	2	2.42850	1.40290
142	4611	2	2.46780	1.43170
143	4612	2	2.50640	1.45760
144	4613	2	2.54640	1.48570
145	4614	2	2.58980	1.51540
146	4615	2	2.63300	1.54640
147	4616	2	2.68120	1.57960
148	4617	3	2.06040	1.19530
149	4620	3	2.23470	1.19600
150	4621	3	2.28240	1.19570
151	4622	3	2.32950	1.19580
152	4623	3	2.37950	1.19490
153	4624	3	2.43130	1.20020
154	4625	3	2.47560	1.16690
155	4626	1	2.47160	1.25940
156	4627	1	2.51090	1.28660

Internal ID No.	Tax Block No.	Neigh-bor-hood	Coordinates X	Coordinates Y
157	4628	1	2.54930	1.31470
158	4629	1	2.58870	1.34240
159	4630	1	2.62640	1.36880
160	4631	1	2.67160	1.39930
161	4632	1	2.71640	1.42930
162	4633	1	2.76630	1.45810
163	4634	3	2.08580	1.09510
164	4635	3	2.13350	1.09600
165	4636	3	2.18060	1.09610
166	4637	3	2.23170	1.09560
167	4638	3	2.27970	1.09520
168	4639	3	2.32760	1.09480
169	4640	3	2.37960	1.09520
170	4641	3	2.42650	1.09540
171	4642	3	2.47390	1.09420
172	4643	3	2.52340	1.09250
173	4644	3	2.55930	1.06370
174	4645	1	2.55360	1.14280
175	4646	1	2.59320	1.17030
176	4647	1	2.63210	1.19680
177	4648	1	2.67720	1.21480
178	464901	1	2.70030	1.26510
179	464902	1	2.75010	1.20820
180	465001	1	2.73410	1.30840
181	465002	1	2.78450	1.21220
182	465101	1	2.81440	1.29180
183	465102	1	2.77090	1.35630
184	4652	1	2.85240	1.34150
185	4653	3	2.08960	1.00080
186	4654	3	2.13640	.99880
187	4655	3	2.18390	.99980
188	4656	3	2.23270	1.00240
189	4657	3	2.28060	1.00200
190	4658	3	2.32640	1.00180
191	4659	3	2.37720	1.00270
192	4660	3	2.42540	1.00090
193	4661	3	2.47260	1.00110
194	4662	3	2.52480	1.00010
195	4663	3	2.57370	1.00490
196	4664	3	2.60990	.97480
197	466501	1	2.67020	.98220
198	466502	1	2.62160	1.04310
199	466601	1	2.69840	1.01740
200	466602	1	2.65290	1.08440
201	466701	1	2.72890	1.05930
202	466702	1	2.67920	1.11850
203	4668	1	2.75880	1.10030
204	4669	1	2.79210	1.13420
205	4670	1	2.83560	1.16500
206	4671	1	2.87900	1.19580
207	4672	1	2.92990	1.22960
208	4791	4	1.30910	1.52100
209	4792	4	1.43910	1.52080
210	4793	4	1.56750	1.52140
211	4794	4	1.71160	1.52230

Internal ID No.	Tax Block No.	Neigh-bor-hood	Coordinates X	Coordinates Y
212	4795	4	1.84880	1.52400
213	4796	4	1.30610	1.47270
214	4797	4	1.43910	1.47270
215	4798	4	1.56750	1.47420
216	4799	4	1.71310	1.47420
217	4800	4	1.84910	1.47710
218	4801	4	1.30760	1.42470
219	4802	4	1.43940	1.42650
220	4803	4	1.56870	1.42680
221	4804	4	1.71270	1.42630
222	4805	4	1.84950	1.42860
223	4806	4	1.99050	1.52530
224	4808	4	1.98920	1.47710
225	4809	7	1.30800	1.37620
226	4810	7	1.43990	1.37820
227	4811	7	1.56890	1.32550
228	4812	7	1.71060	1.32680
229	4813	3	1.84890	1.38080
230	4814	7	1.30790	1.32340
231	4815	7	1.43960	1.32550
232	4818	3	1.84980	1.32960
233	4819	7	1.30950	1.26970
234	4820	7	1.44130	1.27380
235	4823	3	1.84980	1.27700
236	4824	3	1.94300	1.32820
237	4825	3	1.99040	1.32930
238	4826	3	2.03970	1.32910
239	4827	7	1.30850	1.21740
240	4828	7	1.30810	1.16120
241	4829	7	1.57740	1.19310
242	4837	7	1.30970	1.08990
243	4838	7	1.47760	1.08990
244	4841	7	1.55720	1.09110
245	4842	7	1.60540	1.09110
246	4843	7	1.65270	1.09200
247	4844	7	1.70060	1.09150
248	4845	7	1.75080	1.09090
249	4846	3	1.80110	1.09230
250	4847	3	1.84890	1.09200
251	4848	3	1.89570	1.09350
252	4849	3	1.94980	1.09340
253	4850	3	1.99180	1.09470
254	4851	3	2.03300	1.09520
255	4853	7	1.31070	.99860
256	4854	7	1.40970	.99920
257	4855	7	1.45760	.99880
258	4856	7	1.50540	.99820
259	4857	7	1.55720	1.00000
260	4858	7	1.60490	.99970
261	4859	7	1.65200	.99980
262	4860	7	1.70050	1.00020
263	4861	7	1.74840	.99980
264	4862	3	1.79960	.99930
265	4863	3	1.84670	.99940
266	4864	3	1.89380	.99960

Internal ID No.	Tax Block No.	Neigh-bor-hood	Coordinates X	Coordinates Y
267	4865	3	1.94580	.99990
268	4866	3	1.99220	.99940
269	4867	3	2.03990	1.00030
270	5024	11	.63870	1.61740
271	5026	11	.66850	1.35430
272	5028	11	.81760	1.52210
273	5029	11	.99770	1.52090
274	5030	11	1.15780	1.52170
275	5031	11	.81750	1.47280
276	5032	11	.99870	1.47350
277	5033	11	1.15870	1.47400
278	5034	11	.81720	1.42510
279	5035	11	.99890	1.42630
280	5036	11	1.15810	1.42740
281	5037	11	.81650	1.37850
282	5038	11	.99930	1.37790
283	5039	11	1.15890	1.37780
284	5042	10	.81660	1.32430
285	5043	10	.99990	1.32450
286	5044	10	1.15900	1.32360
287	5045	10	.82090	1.27320
288	5046	10	.99890	1.27260
289	5047	10	1.15910	1.26940
290	5048	10	.82330	1.21900
291	5049	10	.99980	1.21780
292	5050	10	1.15880	1.21700
293	5054	10	.68120	1.14170
294	5055	10	.82170	1.16240
295	5056	10	.99940	1.16290
296	5057	10	1.16060	1.16170
297	5062	10	.66040	1.03930
298	5063	10	.71380	1.06110
299	5064	10	.82830	1.09330
300	5065	10	1.00090	1.09160
301	5066	10	1.16120	1.09070
302	508301	10	.83560	1.01700
303	508302	10	.83140	.98100
304	5084	10	1.00040	1.00010
305	5085	10	1.16080	1.00070

Appendix C
SAMPLE JOB STREAM

The following series of jobs represents a possible job stream for a 16-hour sample simulation of the 71st Precinct in New York City. There are 114 calls for service and 10 OUT.OF.SERVICE events.

Event Name	Time of Event			Block†	Priority	Duration‡		
	Day	Hour	Min.			Hour	Min.	
JOB.ENTRY	0	0	04	110	1	01	02	*
JOB.ENTRY	0	0	04	99	3	00	05	*
JOB.ENTRY	0	0	07	282	3	01	25	*
JOB.ENTRY	0	0	12	118	3	00	03	*
JOB.ENTRY	0	0	20	62	3	00	16	*
JOB.ENTRY	0	0	27	51	3	00	18	*
OUT.OF.SERVICE	0	0	31	2			21	*
JOB.ENTRY	0	0	38	38	3	00	10	*
JOB.ENTRY	0	0	42	17	1	00	22	*
JOB.ENTRY	0	0	48	21	3	00	43	*
JOB.ENTRY	0	0	49	19	3	00	38	*
JOB.ENTRY	0	0	49	255	1	00	19	*
JOB.ENTRY	0	0	51	274	3	00	52	*
JOB.ENTRY	0	0	52	118	3	00	09	*
JOB.ENTRY	0	1	21	290	3	00	08	*
JOB.ENTRY	0	1	26	242	3	00	19	*
JOB.ENTRY	0	1	33	161	3	00	40	*
JOB.ENTRY	0	1	35	130	3	00	06	*
JOB.ENTRY	0	1	37	24	3	00	06	*
JOB.ENTRY	0	1	59	275	3	00	14	*
JOB.ENTRY	0	2	11	89	3	01	06	*
JOB.ENTRY	0	2	15	123	3	00	48	*
JOB.ENTRY	0	2	16	116	3	00	58	*
JOB.ENTRY	0	2	17	293	1	00	21	*
JOB.ENTRY	0	2	35	91	3	00	04	*
JOB.ENTRY	0	2	40	292	3	00	09	*
JOB.ENTRY	0	3	01	118	3	00	12	*
OUT.OF.SERVICE	0	3	18	4			15	*
JOB.ENTRY	0	3	22	61	3	00	10	*
JOB.ENTRY	0	3	26	213	3	00	33	*
JOB.ENTRY	0	3	50	31	3	00	40	*

Event Name	Time of Event			Block	Priority	Duration	
	Day	Hour	Min.			Hour	Min.
JOB.ENTRY	0	3	55	21	3	00	26 *
JOB.ENTRY	0	4	02	294	3	00	35 *
OUT.OF.SERVICE	0	4	02	3			70 *
JOB.ENTRY	0	4	12	155	3	00	54 *
JOB.ENTRY	0	4	15	155	3	00	15 *
JOB.ENTRY	0	4	19	294	3	00	09 *
JOB.ENTRY	0	4	22	77	3	00	17 *
JOB.ENTRY	0	4	23	34	1	00	42 *
JOB.ENTRY	0	4	27	84	3	00	05 *
JOB.ENTRY	0	4	39	67	3	00	38 *
JOB.ENTRY	0	4	41	300	3	00	16 *
JOB.ENTRY	0	4	47	301	3	00	36 *
JOB.ENTRY	0	4	58	285	3	00	15 *
JOB.ENTRY	0	4	58	260	3	00	13 *
JOB.ENTRY	0	5	06	17	3	00	06 *
JOB.ENTRY	0	5	10	301	3	00	06 *
JOB.ENTRY	0	5	10	218	3	00	08 *
JOB.ENTRY	0	5	11	218	3	02	06 *
JOB.ENTRY	0	5	22	34	1	00	20 *
JOB.ENTRY	0	5	30	16	1	00	08 *
JOB.ENTRY	0	5	34	290	3	00	55 *
JOB.ENTRY	0	5	36	145	3	00	03 *
JOB.ENTRY	0	5	45	126	3	00	12 *
JOB.ENTRY	0	5	51	161	3	00	07 *
JOB.ENTRY	0	6	01	241	3	00	16 *
JOB.ENTRY	0	6	32	69	3	00	19 *
JOB.ENTRY	0	6	37	241	3	01	08 *
JOB.ENTRY	0	6	51	103	3	00	01 *
JOB.ENTRY	0	7	11	99	3	00	09 *
OUT.OF.SERVICE	0	7	13	1			42 *
JOB.ENTRY	0	7	15	15	3	00	06 *
JOB.ENTRY	0	7	19	16	3	00	21 *
JOB.ENTRY	0	7	25	135	3	00	39 *
JOB.ENTRY	0	8	05	11	3	00	39 *
JOB.ENTRY	0	8	09	13	3	00	16 *
JOB.ENTRY	0	8	12	126	3	00	25 *
JOB.ENTRY	0	8	24	90	3	01	03 *
JOB.ENTRY	0	8	25	156	3	00	31 *
JOB.ENTRY	0	8	34	293	3	00	13 *
JOB.ENTRY	0	8	35	291	3	00	16 *
JOB.ENTRY	0	8	40	118	3	00	17 *
JOB.ENTRY	0	8	40	125	3	01	00 *
JOB.ENTRY	0	8	42	86	3	00	18 *
JOB.ENTRY	0	9	04	166	3	00	23 *
OUT.OF.SERVICE	0	9	28	4			19 *
JOB.ENTRY	0	9	30	15	3	01	22 *
JOB.ENTRY	0	9	42	62	3	00	20 *
OUT.OF.SERVICE	0	9	57	5			38 *
JOB.ENTRY	0	10	11	293	3	01	28 *
OUT.OF.SERVICE	0	10	20	2			16 *

Event Name	Time of Event			Block	Priority	Duration	
	Day	Hour	Min.			Hour	Min.
JOB.ENTRY	0	10	46	89	3	00	01 *
JOB.ENTRY	0	10	47	20	3	00	24 *
JOB.ENTRY	0	10	48	125	1	00	02 *
JOB.ENTRY	0	10	52	89	1	01	15 *
JOB.ENTRY	0	11	00	86	3	00	07 *
JOB.ENTRY	0	11	05	271	3	00	01 *
JOB.ENTRY	0	11	13	10	3	00	38 *
JOB.ENTRY	0	11	16	127	3	00	55 *
JOB.ENTRY	0	11	18	162	3	00	09 *
JOB.ENTRY	0	11	28	297	3	00	18 *
JOB.ENTRY	0	11	43	146	3	00	08 *
JOB.ENTRY	0	11	45	104	3	00	54 *
JOB.ENTRY	0	11	45	205	3	00	10 *
JOB.ENTRY	0	11	51	146	3	00	18 *
JOB.ENTRY	0	11	53	271	3	00	32 *
JOB.ENTRY	0	12	16	243	3	00	01 *
OUT.OF.SERVICE	0	12	18	1			60 *
JOB.ENTRY	0	12	31	284	3	00	18 *
JOB.ENTRY	0	12	47	253	3	00	54 *
JOB.ENTRY	0	12	58	221	3	00	11 *
JOB.ENTRY	0	12	59	134	3	00	30 *
JOB.ENTRY	0	13	01	225	3	00	08 *
JOB.ENTRY	0	13	11	242	3	01	02 *
JOB.ENTRY	0	13	17	76	3	00	08 *
JOB.ENTRY	0	13	23	129	3	00	08 *
JOB.ENTRY	0	13	33	162	3	00	32 *
JOB.ENTRY	0	13	37	57	3	00	09 *
JOB.ENTRY	0	13	39	104	3	00	06 *
JOB.ENTRY	0	13	40	129	3	00	02 *
OUT.OF.SERVICE	0	13	42	6			22 *
JOB.ENTRY	0	13	45	278	3	00	51 *
JOB.ENTRY	0	14	21	19	3	00	30 *
JOB.ENTRY	0	14	31	133	3	00	02 *
JOB.ENTRY	0	14	31	299	3	00	09 *
JOB.ENTRY	0	14	45	160	3	00	10 *
JOB.ENTRY	0	14	49	203	3	00	31 *
JOB.ENTRY	0	15	00	281	1	00	06 *
JOB.ENTRY	0	15	09	57	3	00	22 *
JOB.ENTRY	0	15	09	253	3	00	21 *
OUT.OF.SERVICE	0	15	14	7			40 *
JOB.ENTRY	0	15	18	242	3	00	12 *
JOB.ENTRY	0	15	23	191	3	00	09 *
JOB.ENTRY	0	15	53	107	3	00	03 *
JOB.ENTRY	0	16	10	176	3	00	32 *

†For OUT.OF.SERVICE events this field contains the number of the car to be placed out of service.

‡For OUT.OF.SERVICE events this field contains the length of time that the car is to remain out of service (in minutes).

Appendix D

A DEFINITION OF EACH GLOBAL SIMULATION VARIABLE

I. ENTITIES AND THEIR ATTRIBUTES

Entity	Type	Attribute	Mode*	Description
BLOCK	Permanent	TAXNO	I	The external identification number associated with the block.
		XCORD	R	The x coordinate of the center of the block.
		YCORD	R	The y coordinate of the center of the block.
		NBDID	I	The internal reference number of the neighborhood to which the block belongs.
NBD	Permanent	NBD.NAME	A	The 4-character name associated with the neighborhood.
		N.ADJACENT	I	The number of cars designated as adjacent resources for the neighborhood.
		N.SECTOR.CARS	I	The number of sector cars assigned to the neighborhood.
		P_n.NBD.RSPONSE	R	The time until the arrival of the first car at an incident of priority n(n=1,...,5) that occurred in the neighborhood (used as statistics-gathering variables).
CAR	Permanent	CAR.NAME	A	The 4-character name associated with the car.
		XLOC	R	The x coordinate of the car's current location.
		YLOC	R	The y coordinate of the car's current location.
		NUMB.SECTORS	I	The number of neighborhoods to which this car is assigned as a sector car.
		ASSIGNMENT	I	The internal identification number of the job on which the car is currently working (if any).
		T.START.RESPONSE	R	The time at which the car began responding to its current job.
		NEXT.EVENT	I	The internal identification number of the next event that is scheduled to occur for the car.

*See notes on p. 97.

Entity	Type	Attribute	Mode*	Description
		SPT	I	The priority of the job currently being serviced by the car (out-of-service jobs have Priority 6; if the car is available, its SPT is zero).
		SOT	I	= 1 if the unit is working as primary car. = 2 if the unit is working as backup car. = 3 if the unit is working as a tertiary car.
		SWT	I	= 0 if the unit is on patrol in its sector. = 1 if the unit is returning to sector from an outside call. = 2 if the unit is responding to a call in sector. = 3 if the unit is responding to a call outside of sector. = 4 if the unit is working on a call in sector. = 5 if the unit is working on a call out of sector. = 6 if the unit is out of service.
		CENTROID	I	The block to which the unit goes when it returns to patrol in its sector after responding to a job outside its sector.
		IN.SECT.JOBS	I	The total number of sector jobs the car has responded to so far.
		OUT.SECT.JOBS	I	The total number of out-of-sector jobs the car has responded to so far.
P.CLASS	Permanent	JOB.RESPONSE.TIME	R	The time until the arrival of the first car at a given incident of this priority class (used as a statistics-gathering variable).
		VELOCITY	R	The average speed at which cars respond to incidents of this priority class.
		COUNT	I	The number of cars actually dispatched to a specific incident of this priority class (a statistics-gathering variable).
		WAIT.TIME	R	The time that a specific incident of this priority class spends waiting in queue before a unit is dispatched to it (a statistics-gathering variable).

Entity	Type	Attribute	Mode*	Description
JOB	Temporary	LOCATION	I	The internal block number within which the job occurs.
		ENTRY.TIME	R	The time at which the call is received (in days since the start of the simulation).
		PRIORITY	I	The priority of the job (1-5).
		STATUS	I	= 0: No car has yet arrived at the scene. = 1: The primary car has already arrived at t-e scene. = 2: Some cars, but not the primary car, have already arrived at the scene.
		DURATION	R	The length of time the primary car is required to work at the job (in minutes).

II. LISTS

Name	Owner Entity	Member Entity	Type	Definition
QUEUE	P.CLASS	JOB	FIFO	Jobs waiting to be dispatched.
ASSIGNED.CARS	JOB	CAR	FIFO	The cars assigned to a job.

---

*The symbols used in this column have the following meaning:

I Integer
R Real
A. Alphanumeric

III. ARRAYS

Name	No. of Dimensions	1st Dimension	2nd Dimension	
ADJACENT.CARS	2	N.NBD	N.CAR	The dispatch nomination list for each neighborhood.
SECTORS	2	N.CAR	NUMB.SECTORS(CAR)	The sector responsibilities for each car.
P.DURATION	1	3		The fraction of a job's duration that the primary car, secondary car, and tertiary car spend working at the job.
REGION.NAME	1	20		The alphanumeric name of the region being simulated (maximum 80 characters).
TITLE	1	20		A title identifying the simulation run (maximum 80 characters). The title is printed at the top of each output report.

IV. SYSTEM VARIABLES - INPUT

Variable	Mode*	Description
N.BLOCK	I	The total number of BLOCKs in the region being simulated.
N.NBD	I	The total number of NBDs in the region being simulated.
N.CAR	I	The number of cars being simulated.
N.P.CLASS	I	The number of priority classes (set at 5 in the simulation docmented here).
MAX.SENT	I	The maximum number of cars that can be dispatched to a Priority 1 incident.
MEAL.DURATION	I	The length of an internally scheduled OUT.OF.SERVICE job.
TOUR.LENGTH	I	The length of each car's work day.

V. SYSTEM VARIABLES - INTERNAL AND OUTPUT

Variable	Description
N.AVAILABLE	The current number of cars available for dispatch.
OUTSIDE.DISPATCH	The number of times that no car in the region being simulated is available for dispatch to a Priority 1 call for service. It is assumed that a car from outside the region is sent.
MQ, VQ, PROBQ	The mean, variance, and histogram of the queue lengths for each of the QUEUEs.
UTILIZ	The histogram of the amount of simulated time a car spends at each value of SWT.
MC, VC, HISTC	The mean, variance, and histogram of COUNT, the number of cars sent to an incident.
MRT, VRT, NRT, FREQ	The mean, variance, number, and histogram of JOB.RESPONSE.TIME.
$MSRT_n$, $VSRT_n$, $NSRT_n$,	The mean, variance, and number of P_n.NBD.RESPONSE (n=1, 2,..., 5).
MAV, VAV, HISTAV	The mean, variance, and histogram of N.AVAILABLE.

REFERENCES

1. Crabill, T. B., W. E. Walker, and P. Kolesar, "Validation of a Police Patrol Simulation Model," Proceedings of the 8th Annual Simulation Symposium, Tampa, Florida, March 1975.

2. Fishman, George S., Concepts and Methods in Discrete Event Digital Simulation, John Wiley & Sons, Inc., New York, 1973.

3. Gordon, Geoffrey, System Simulation, Prentice-Hall, Inc., Englewood Cliffs, 1969.

4. Ignall, E., P. Kolesar, and W. Walker, "The Use of Simulation in the Development and Empirical Validation of Analytical Models for Emergency Services," Proceedings of the 1974 Winter Simulation Conference, Washington, D.C., January 1974.

5. Johnson, G. D., SIMSCRIPT II.5 User's Manual S/360-370 Version, Consolidated Analysis Centers, Inc., Santa Monica, California, 1972.

6. Kiviat, P. J., R. Villanueva, and H. M. Markowitz, SIMSCRIPT II.5 Programming Language, Consolidated Analysis Centers, Inc., Santa Monica, California, 1973.

7. Kolesar, P., and W. E. Walker, "A Simulation Model of Police Patrol Operations: Executive Summary," R-1665/1-HUD, The New York City-Rand Institute, forthcoming.

8. Naylor, T. H., J. C. Balintfy, D. S. Burdick, and K. Chu, Computer Simulation Techniques, John Wiley & Sons, Inc., New York, 1966.

END