Chapter 4

Discrete System Simulation

- **Discrete Events:**

A simulation process will take system model through a series of state. We can describe the state of system at a given instant, since the model of system has a set of number to represent the state of a system. A number that is used to represent some aspect of some system state is called a state descriptor. As the simulation precedes the state descriptor changes their own values.

Discrete event can be defined as a set of circumstances that cause an instantaneous change in one or more state descriptor of the system. It is necessary that system changes must be sufficiently represent in the model. Simultaneous occurring changes in the state descriptor need not necessary to represent the single unit. In simulation, we concern with the occurrence of changes in the system but do not concern to when the changes are made to the model. However, it is necessary that changes must be made sequentially. So a system simulation must contain a number of representing time.

- **Representation of time**

It refers to number that record the passage of time. It is usually set to zero at the beginning of a simulation and subsequently indicates how many units of simulated time have pass since the beginning of the simulation.

Simulation Time:

It refers to the indicated clock time. Note that it is not the time that a computer has taken to carry out the simulation. The ratio of simulated time to the real time taken can vary to a great extend depending on the following factors:

- Nature of the system being simulated.
- The detail to which it is modeled.

Example: The simulation of an atomic model.

Changes occur in a fraction of a microsecond in real system. However, the simulation of these atomic models in a computer may take 1000 time more time than in the actual system.

Updating Clock Time:

There are two basic methods used to update clock time:

i) Event Oriented Method

In these method, the clock is advanced to the time at which next event is due to occur. Discrete system simulation is usually carried out by using the event oriented method.

ii) Interval Oriented Method

In these methods, the clock time is advanced by small and usually uniform interval of time and it is determined at each interval whether an event is due to occur at that time. Continuous system simulation normally uses the interval oriented method.

But it should be noted that no firm (visit) rule can be made about the way the time is represented in simulation for discrete and continuous system because:

- An interval oriented program will detect discrete changes and can therefore simulate discrete system.
- An event oriented program can be made to follow continuous changes by artificially introducing events that occurs at regular time interval and can therefore simulate continuous system.

Significant Event Simulation:

It is another method to represent the passage of time and is applicable to continuous system in which there is quiescent (continuous) period. A quiescent period is the interval between events in the event oriented approach but it involves the models representation of the system activities which create a notice of the event that terminates the interval. The significant event approach assumes that simple analytic function can be used to project the pan of a quiescent period. The significant event is the one with the least span. Example, an automobile travelling at constant acceleration, it movement might result in a significant event for several reasons:

- It might reach at the end the road.
- It velocity most reach some limit.
- It might come to rest.

If the initial conditions are known, the elapsed time for each these possible events can be calculated from simple formula.

- **Generation of Arrival Patterns**

The generation of exogenous arrival is an important aspect of discrete system simulation. It is possible that an exact sequence of arrivals has been specified for the simulation. For example, in the simulation of electronic circuit a particular sequence of signal might be used as the simulation input to verify the circuit.

- **Trace Driven Simulation**

It refers to the process of gathering a sequence of inputs based on the observation of a system. When there is no interaction between exogenous arrivals and the endogenous events of a system, it is permissible to create a sequence of arrival in preparation for the simulation. Usually, however, the simulation proceeds by creating new arrivals as they are needed.

- **Bootstrapping:**

It is the process of making one entity creates its successor. These methods require keeping only the arrival time of the next entity, it is therefore the prefer method of generating arrivals for computer simulation programs.

- **Simulation of Telephone System**

The simulation of a discrete system can be explained by simulating a telephone system as below: The system has number of telephone (here only 8 are shown), connected to a switch board by line. The switch board has a number of lines provided the condition that only one connection at time can be made to each line. Any call that cannot be connected at the time it arrives is immediately abandoned, and then the system is called **a lost call system**. A call may be lost due to the following reasons:

  - If the called party is engaged (busy)
  - If no link is available (a blocked call)

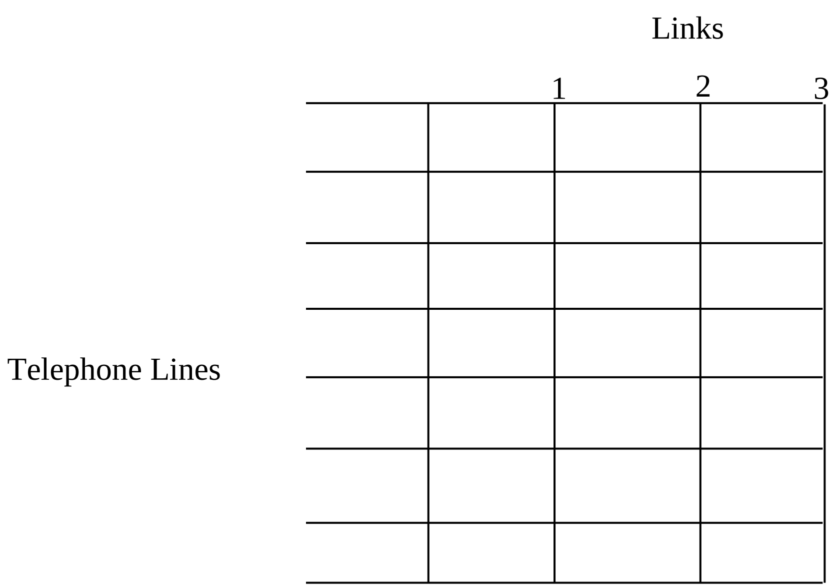
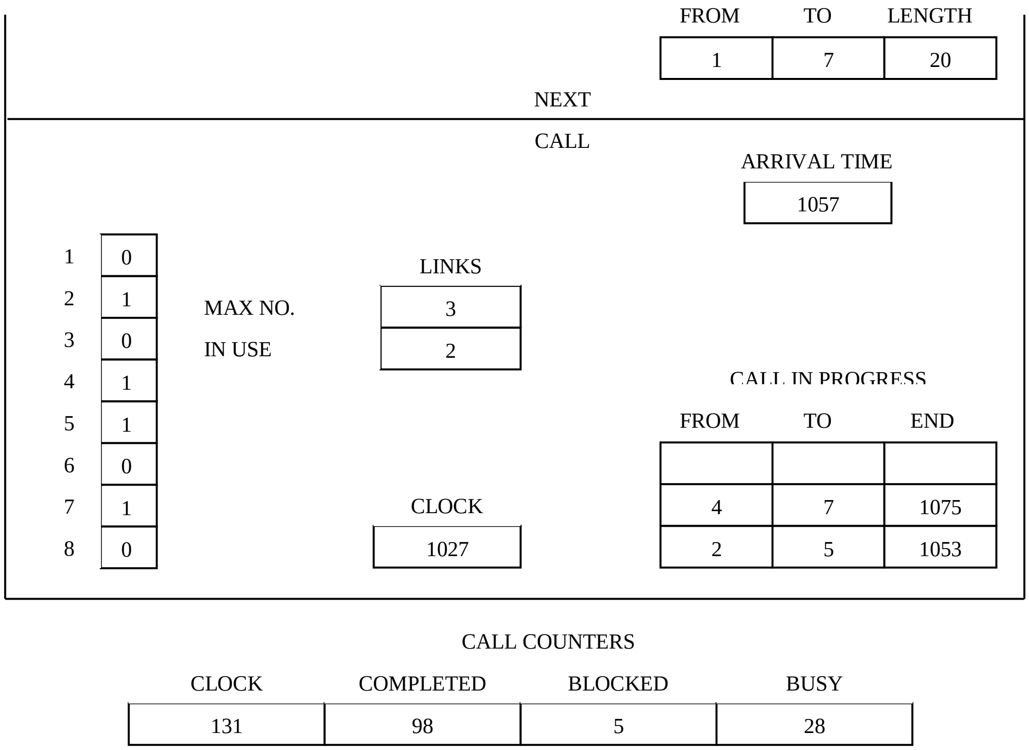


Figure 1: Simple Telephone System.

Object of Simulation:

To process a given number of calls and determine what portion are successfully completed, blocked or found to be busy calls. Lets us consider the current state of the system as below:



Here, line 2 is connected to line 5 and line 4 is connected to line 7. A 0 in the line tables means is free and 1 means, it is busy. Maximum number of links, in this case, is only 3 and 2 of them are in used. A number representing clock time is included to keep track of events. In this state, the clock time is shown to be 1027. The clock will be updated to the next occurrence of event as the simulation proceeds. Each column has its own attributes; its origin, destination, length and the time at which it call finishes. The “call in progress” tables shows the lines that are currently connected and the finish time. Arrival time and detail of next call are also shown to generate the arrival of calls; here bootstrap method can be used. It is assumed that the call is equally likely to come from any line, i.e. not busy at the time of arrival and that is can be directed to any line other than itself. To make easy to explain, the action of the simulation the attributes of next events are shown in figure 2, although in real practice, they would not have been generated until the clock reaches 1057, the clock of next arrival.

Two possible attributes can make to change to occur in the system state; a new call can arrive or an existing call can be finished. From the figure 2, it can be seen that three events can occur in the figure:

i) Call between 2 & 5 will finish at 1053
ii) A new call will arrive at 1057

iii) Call between 4 & 7 will finish at 1075

The simulation proceeds by executing a cycle below state to simulate each event:

a) Scans the events to determine which the next potential event is. Here, it is at 1053. The clock is updated then.

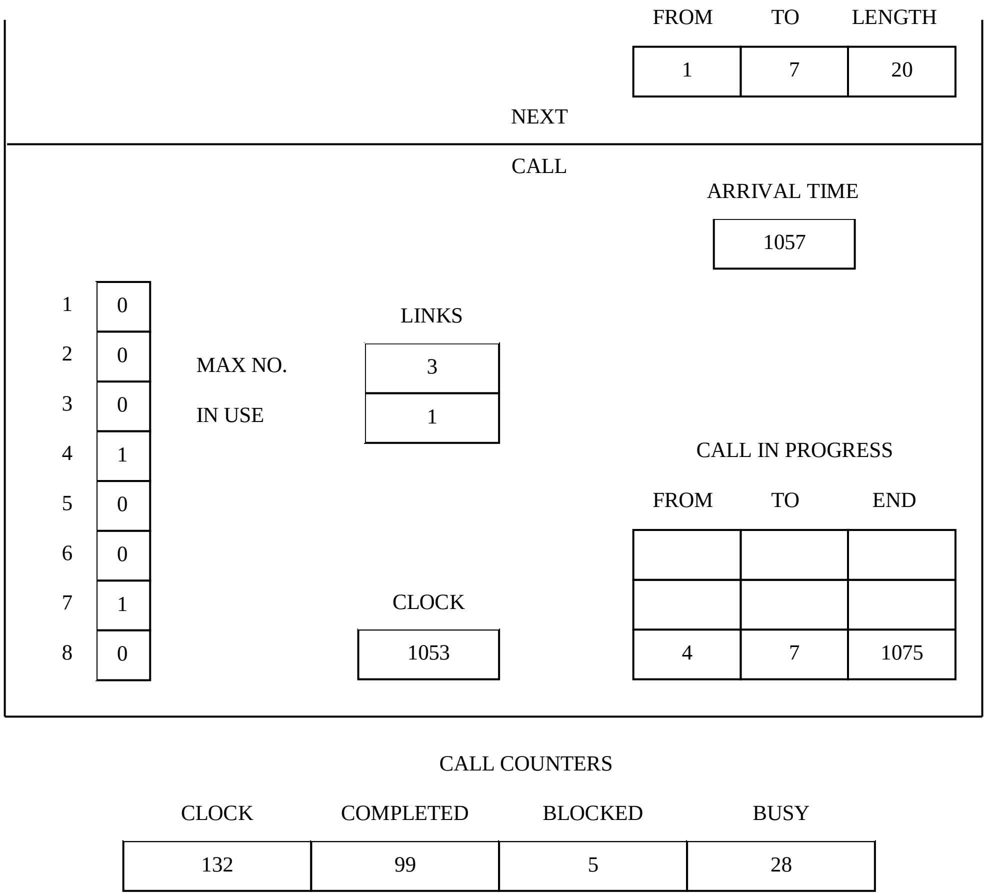
b) Select the activity that is to cause the event. Here, the activity is to disconnect a call between lines 2 & 5.

c) Test whether the potential event can be executed. Here, however, there are no conditions to disconnect a call.

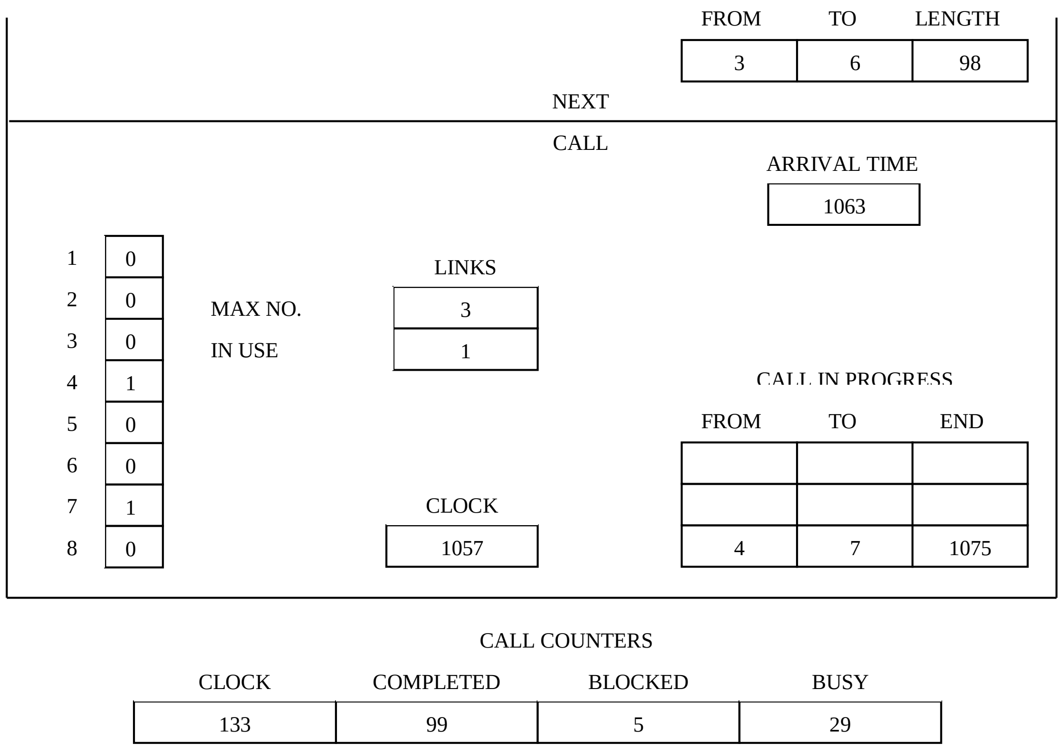
d) Change the record to reflect the effect of the event. Here the call is shown to be disconnected by setting to zero in the line table for line 2 & 5, reducing the number of links in used by 1 and removing the finish call from the “call in progress” table.

e) Gather some statistics for the simulation output. Here call counter are changed to record the number of calls that have been processed, completed, blocked or busy. The state of the system appears as shown in figure 3.

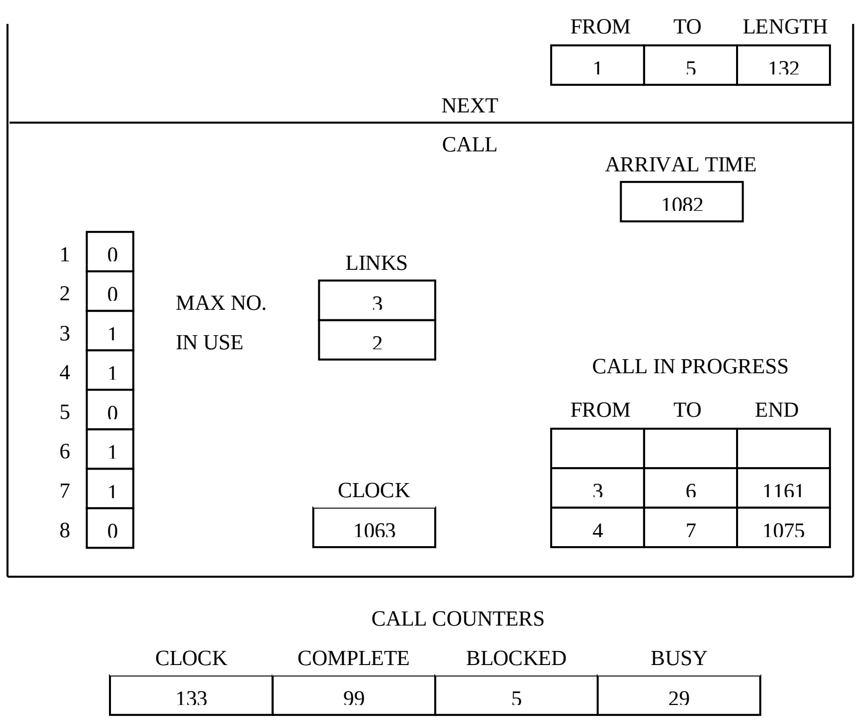
f) The above steps are repeated to continue the simulation.



In figure, it can be seen that the next potential event is the arrival of a call at time 1057 sec. the clock is updated to 1057 and the attributes of new arrival is generated. Since the selected activity is to connect a call, it is necessary to test; to find whether a link is available and to find whether the party is busy or not. In this case, the called party, the line 7 is busy so the call is lost. Both the processed call and the busy call counter are increased by one. A new arrival is generated. These are shown in figure 4.



Suppose the next arrival time is 1063, and the call will be from line 3 to 6 and will spend 20 sec and at this time, the arriving call can be connected. The new state of the system is shown in figure 5. The procedural is repeated to a certain limit until a good statistics can be gathered.

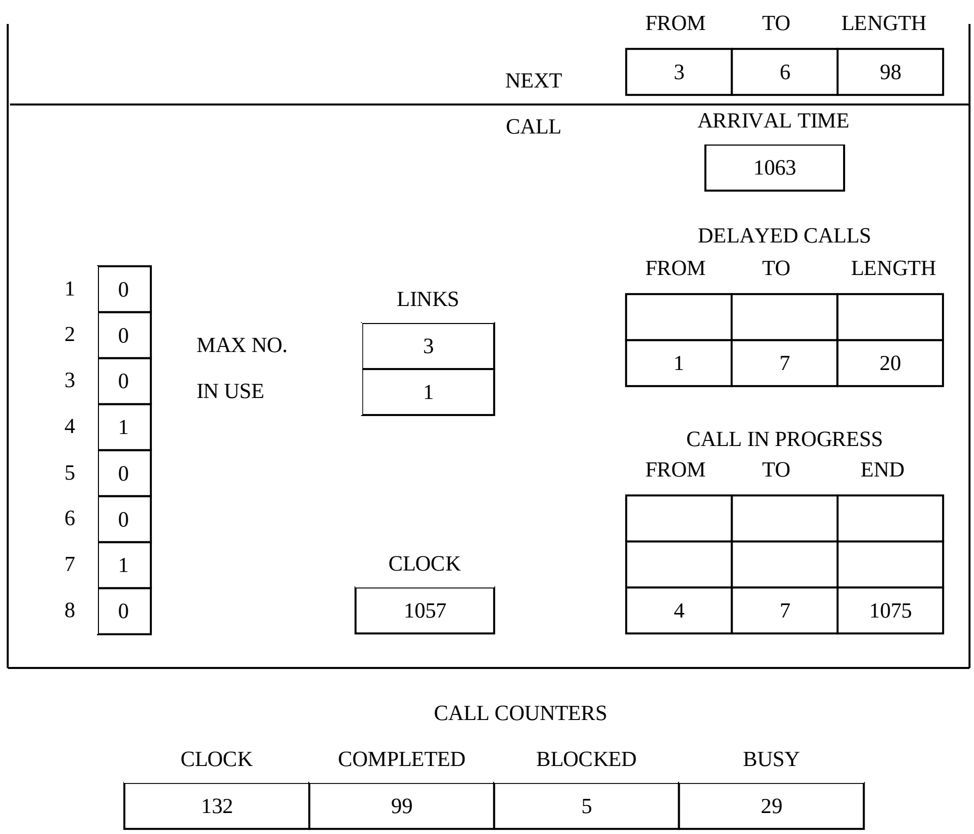


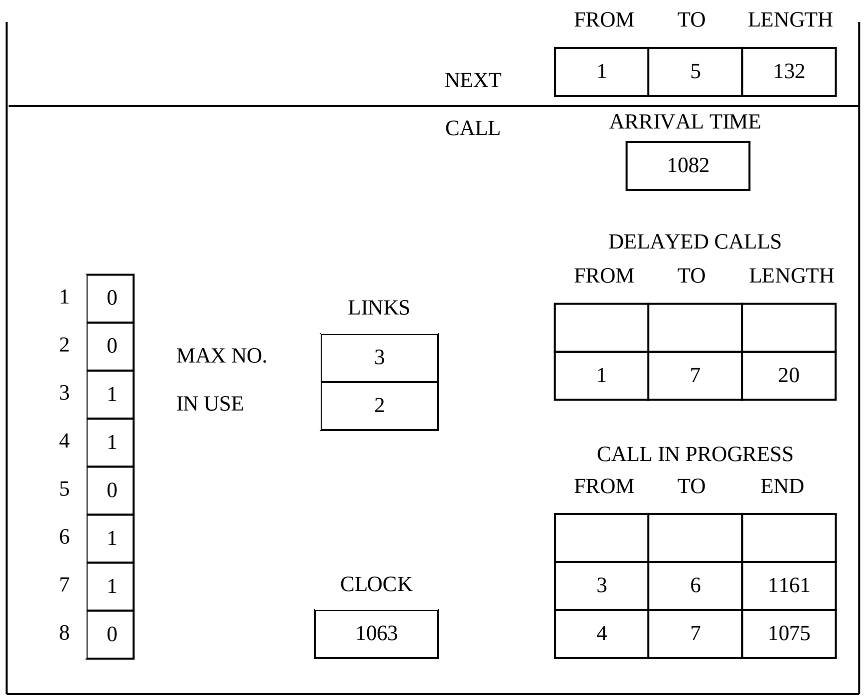
Delayed Call

When the telephone system is modified so that calls that cannot be connected are not lost, and then they will wait until they can be connected later. Such calls are refer to as delayed call. We know that it is not possible in the case a real time telephone system but possible to message in a switching system that has store and forward capability.

To keep record of delayed call, it is necessary to build another list like the call in progress list. After arriving a call, if it cannot be connected, then it is placed in the delayed call list, waiting to next time. When a previous call is completed, it is necessary to check the delayed call list to find if a waiting call can be connected. This is illustrated in the figures 6, 7, 8, and 9.

From the above figure, it is clear that the next potential event is the arrival of next call from line 1 to 7. Since the line 7 is not free the next call cannot be executed. It is then placed in the delay call list waiting for the next time and the new stated of the system is shown in below figure.

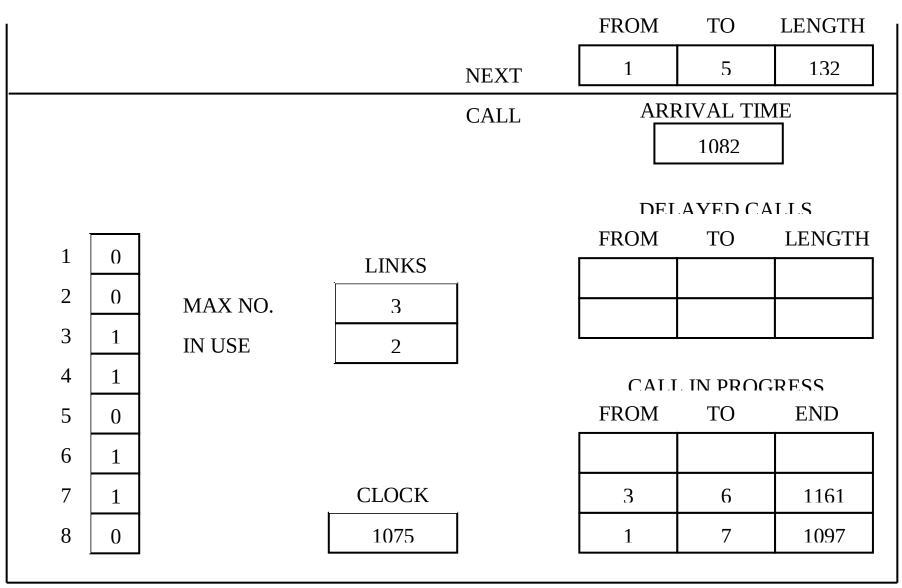




CALL COUNTERS

CLOCK	COMPLETE	BLOCKED	BUSY
132	99	5	29

When the call from line 4 and 7 completed at time 1075, then for next event, the delayed call list is checked first. At these time, the lines 1 and 7 can be connected which is shown in below figure.



CALL COUNTERS

CLOCK	COMPLETED	BLOCKED	BUSY
133	100	5	29

Simulation Programming Task

There are mainly three tasks to be performed in simulation programming. They are:

i) Generating a Model

The first task in simulation programming is to generate a model and initialize it. In this state, a set of number of most created to represent the state of the system. Thus, these set of number is called the system image. The activities that occur in the system are in routines.

ii) Simulation

After generating a model, the next is to program the procedure that execute the cycle of actions involved in carrying out the simulation. This procedure is refers to as simulation algorithm.

iii) Generating Report

After programming the simulation algorithm, the next and final task is to run the simulation to generate on output report. The statistic gather (data collected) during the simulation will be organized by a report generator.

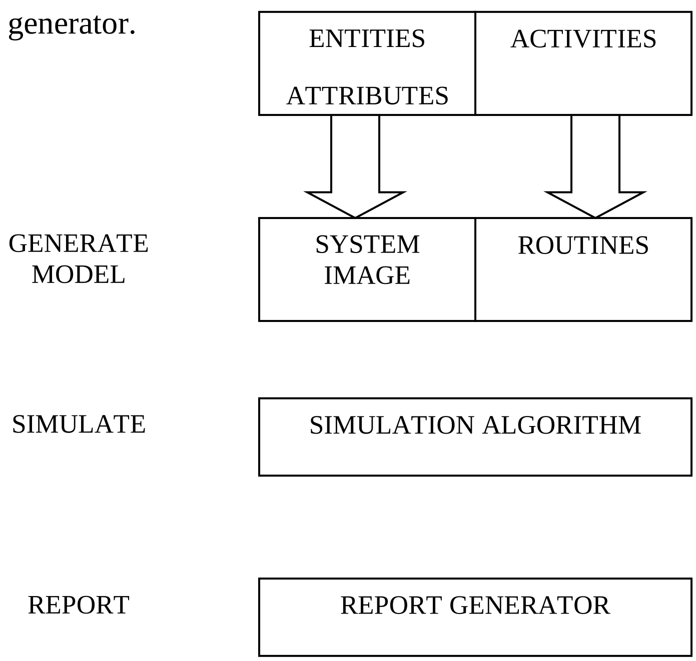


Figure: Simulation Programming Task.

Gathering Statistics:

To print out the statistics gathered during the run, simulation programming system include a report generator. The exact statistics required from a model depend upon the study being performed. Some of the commonly required statistic which is usually included in the output during the simulation is as follows:

a) Counts

This gives the number of entities at a particular type or the number of times some events occurs.

b) Summary measures

This includes measuring some quantities such as extreme values, mean values and standard deviation.

c) Utilization

This measure the time in fraction or percentage, that some entity is engaged.

d) Occupancy

These give the percentage of a group of entities in use on the average.

e) Distribution

These record the distribution of important variables such as queue length or waiting time.

f) Transit time

These records the time taken from an entity to move from one part of the system to some otherpart.

When there are stochastic effects operating in the system, all these measures will fluctuate as a simulation proceeds, and the particular values reached at the simulation are taken as estimates of the true values they are designed to measure.

Counter and Summary Statistics:

Counter which are the basic for most statistics are used to accumulate totals and to record current values of some level in the system. For example, in the simulation of telephone system, counter are used to record the total number of lost call, busy call, and to keep track of how many link were in use at any time. Whenever a new value of a count is established, it is compared with the record of the current maximum or minimum, and the record is changed when necessary. The accumulator sum of observation must be kept to derive the mean value and standard deviation as below:

The mean of set of N observation $X_i(i = 1,2,3, \dots \dots \dots . n)$ is given by

Mean, M=$\frac{1}{N}\sum_{i=1}^{N} X_i$

A mean is derived by accumulating the total value of the observation, and also accumulating a count of the number of observations.

And, Standard Deviation,$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(M - X_i)^2}$

The accumulated sum of the observations must be kept to derive the mean value, and so the additional record needed to derive a standard deviation is the sum of the squares.

Measuring Utilization & Occupancy

To measure the load on some entity such as an item of equipment, the simplest way is to determine what fraction/percentage of time, the item as engaged during the simulation. Measuring those statistic is refer to as the utilization of that equipment.

The time history of an equipment usage might appear as shown in figure below:

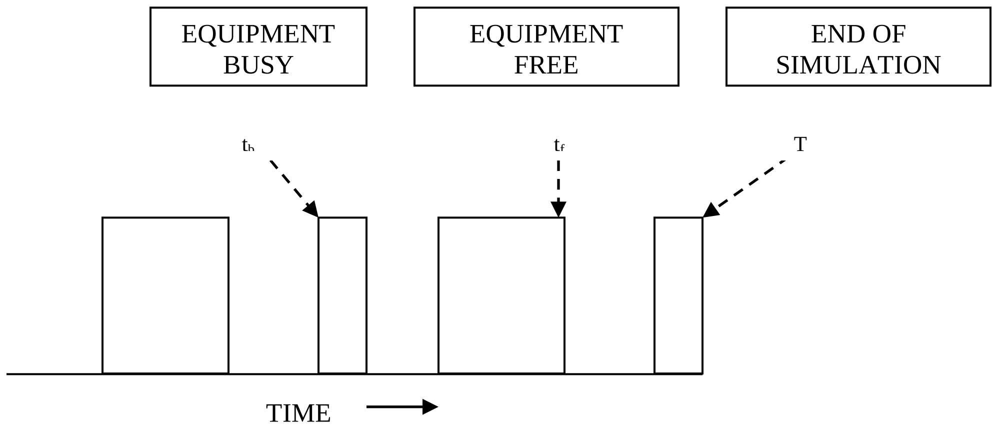


Figure: Utilization of equipment.

Let, t_b= the time at which item last become busy.

t_f= the time at which the item last become free.

T =total time in simulation run.

Then, the utilization u is given by:

$$u = \frac{1}{T}\sum_{i=1}^{N}(t_f - t_b)_i$$

In dealing with group of entities, rather than individual items, the calculation is similar, requiring that information about the number of entities involved also kept. The below figure represents as a function of time and the number of links in telephone system that are busy. To find average number of link in use, a record must be kept of the number of links currently in use and the time at which the last change occurred. If the number changes at time t_i to the value n_i, then, at the time of the next change t_{i+1}, the quantity $n_i(t_{i+1}-t_i)$ must be calculated and added to an accumulated total. The average number in use during the simulation run, A, is then calculated at the end of the run by dividing the total by the total simulation time T, so that:

$$A = \frac{1}{T} n_i(t_{i+1} - t_i)$$

The below figure might also represent number of entities waiting on a queues.

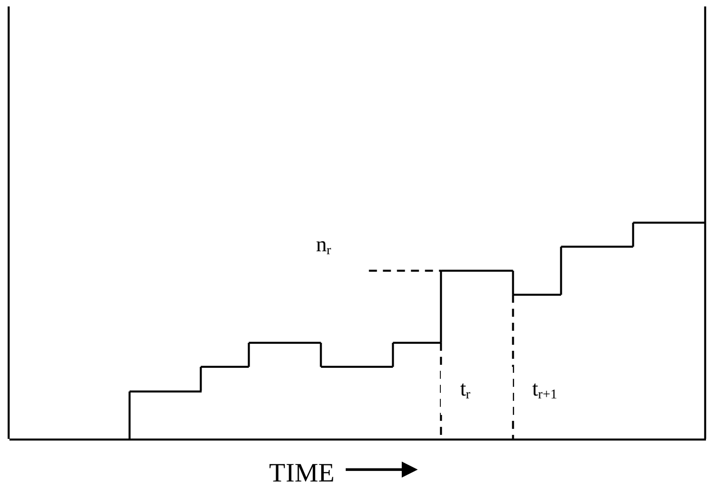


Figure: Time history of busy telephone links.

If there is an upper limit on the number of entities, as there was a limit on a number of links in the telephone system then the occupancy is defined as the ratio of the average number in use to the maximum number.

Occupancy = $\frac{\text{Average no is use}}{\text{Maximum number}}$

If M= links in a telephone

n_i = number of busy in the interval t_i and t_{i+1} then the average occupancy, assuming the number n_i changes n times is given by:

Occupancy (B) =$\frac{1}{NM}\sum_{i=1}^{N} n_i(t_{i+1} - t_i)$

Recording Distribution & Transit Time

To determine the distribution of a variable it requires counting the number of times the value of the variable fall within specific interval. For this purpose, a table with location to define the internal and to accumulate the count has to be maintained. When an observation is made, the value is compared with the defined intervals and the appropriate must be incremented by one.

The definition of a destination table required specification for lower limit of the tabulation, the interval size and the number of intervals. Normally, the tabulation interval sizes are uniform.

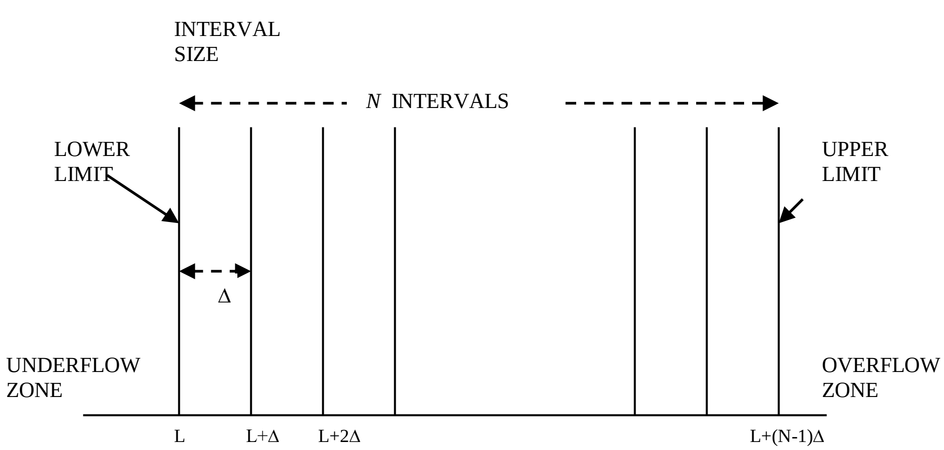


Figure: Distribution Table.

Generally, it is required to keep count the number of time and observation falls below the lower limit and beyond the upper limit rather than to obtain accurately the potential range of valuation. To determine the mean and standard deviation, it will be required to accumulate number of observation (X_i) and the sum of squares$\sum(X_i)^2$. For each observation X_i,

- One is added to appropriate counter
- $X_i$is added to the sum, $\sum(X_i)$
- $(X_i)^2$ is added to the sum, $\sum(X_i)^2$.

The space required for the tabulation shown in figure below:

i=1	L, LOWER LIMIT	UNDERFLOW COUNT
	L+Δ	1^{st} INTERVAL COUNT
	L+iΔ	i^{th} INTERVAL COUNT
i=N-1	L + (N-1)Δ	$(N-1)^{th}$INTERVAL COUNT
i=N		OVERFLOW COUNT
	NUMBER OF ENTRIES	
	TOTAL OF ENTRIES	
	SUM OF SQUARES	

Since values of observation are matched to an interval the derived destination is an approximation. However, note that the mean and standard deviation will be accurate within the accuracy limit of the computer even if some observations fall outside the table limits.

The instances when the observations are made are determined by the native of random variable being measured. To understand this, consider the following two examples:

To measure the mean waiting time for a service, an observation must be taken as each entity starts to receive a service so that the times at which the observations are tabulated are randomly spaced.

To measure the distribution of the number of entities waiting, observation should be taken at uniform interval time. The final output weight also expresses the data in other commercial form as required. For example, cumulative distribution may be given or the distribution may be resolved to express the counts as percentage of total observation. These all results are calculated at the end of simulation.

A clock is used in the manner of time stamp to measure transit time. When an entity reaches a point from which a measured of transit time is to start. A note of the time of arrival is made. Later when the entity reaches the point, at which, the measured ends, a note of the clock time upon arrival is made and these two clock time noted are used to compute the elapsed time.

Discrete Simulation Language

To simplify a task of writing a discrete simulation program, a number of programming languages are sued. The two most commonly used are:

a) GPSS
b) SIMSCRIPT

These languages are used describe a system and establish a system image and execute a simulation algorithm. Most programming also provides report generators. These language also offer many convenient facilities such as,

- Automatic generators of streams of pseudo random numbers for any desired statistical destination.
- Automatic data collection
- Their statistical analysis and report generations
- Automatic handling of queues, etc.

In addition a good simulation also provides a nodded builder with the view of the world that next nodded building easier. Every discrete system simulation language must provide the concept and statement for:

a) Representing the state of a system at a single point in time (static modeling)
b) Moving a system from state to state (Dynamic modeling), &

c) Performing relevant task such as random number of generation, data analysis and report generation.

Discrete system simulation languages can be classified into three main categories:

a) Event oriented languages
b) Activity oriented languages
c) Process oriented languages.

Simulation of Single Server Queuing System

Consider a single server queuing system as shown in figure below for which the inter arrival time A1, A2 …….. are independent and identically distributed (IID) random variables. A customers who arrives and finds the server idle, enters the service immediately and the service time S1, S2,…. of the successive customers are IID random variables that are independent of inter arrival times.

- A customer who arrives and finds the server busy, joins the end of the single queue.
- After completing service for a customer1, the server chooses the next customer from the queue (if any) in FIFO manner.
- The simulation will begin in the empty and idle state i.e. no customers are present and the service is idle.
- At time 0, the system will begin waiting for arrival of the 1^{st} customer will occur, after the 1^{st} inter arrival time A1 rather than at time 0.
- The simulation will continue until N numbers of customers have completed their delays in queue.

Measuring the performance of the system

To measure the performance of the system, we estimate the following three qualities:

a) Estimate the expected average delay d(n) in queue of (n) customers. The actual average delay for n customers depends on the inter arrival and service time. From a single sum of simulation with customers delays D1, D2,…….., Dn, the estimate of d(n) is given by:

$\hat{d}(n) = \frac{\sum_{i=1}^{n} Di}{n}$……………… (1)

Note that a customer could have a delay of zero in case of an arrival finding the system empty or idle. If many delays were 0 then this could represent the system providing very good service.

This estimate of d(n) gives information about the system performance form the customer point of view.

b) Estimate the expected average number of customers in the queue (but not being served), denoted by q(n). It is different from the average delay in queue because it takes over continuous time rather than over customer (being discrete).

Let Q(t) = number of customers in queue at time (t), (t≥0).

T(n) = time required for 'n' delays in queue.

Pj = expected proportion of time that Q(t) = 1 (value of Pi will be between 0-1)

Then, average number of customer queue

q(n)= $\sum_{i=0}^{\infty} iP_i$

Expected average number of customer in queue:

$\hat{q}(n) = \sum_{i=0}^{\infty} i\hat{P}_i$.......................... (2)

where$\hat{P}_i$ is observed (rather than expected) portion of the time during the simulation that there were i customers in queue. Also, let T_i be the total time during the simulation that the queue is of length *i*, then $T(n) = T_1 + T_2 + T_3 + \cdots$ .... and $\hat{P}_i = T_i/T(n)$ and so that we can rewrite the above equation as:

$$\hat{q}(n) = \frac{\sum_{i=0}^{\infty} iT_i}{T(n)}$$

c) Measure how busy the server is:

The expected utilization of the server is the expected proportion of time during the simulation that the server is busy and is thus, the number between 0 & 1; denote by $u(n)$. From a single simulation, then, our estimate of $u(n)$ is $\hat{u}(n)$ = the observed proportion of time during the simulation that the server is busy. Let us define the busy function as:

$B(t)$ =1 if the server is busy at time t

0 if the server is idle at time t

Then, $\hat{u}(n)$ could be expressed the proportion of time when $B(t)$ equals to 1 and is given by:

$$\hat{u}(n) = \frac{\int_0^{\mathrm{T(n)}} \mathrm{B(t)dt}}{\mathrm{T(n)}}$$