Unit I: Introduction to Modeling and Simulation - Simulation and Modeling

System Concept:

The term system is derived from the Greek word **Systema**, which means an organized relationship between functioning units or components. It is aggregation or association of objects joined in some regular manner/interactions or independence.

The interaction/independence between the objects causes a change in the system. The system exists because it is designed to achieve one or more objectives.

There are more than a hundred definitions of the word system, but most seem to have a common thread that suggests that a system is an orderly grouping of interdependent components linked together according to a plan to achieve a specific objective.

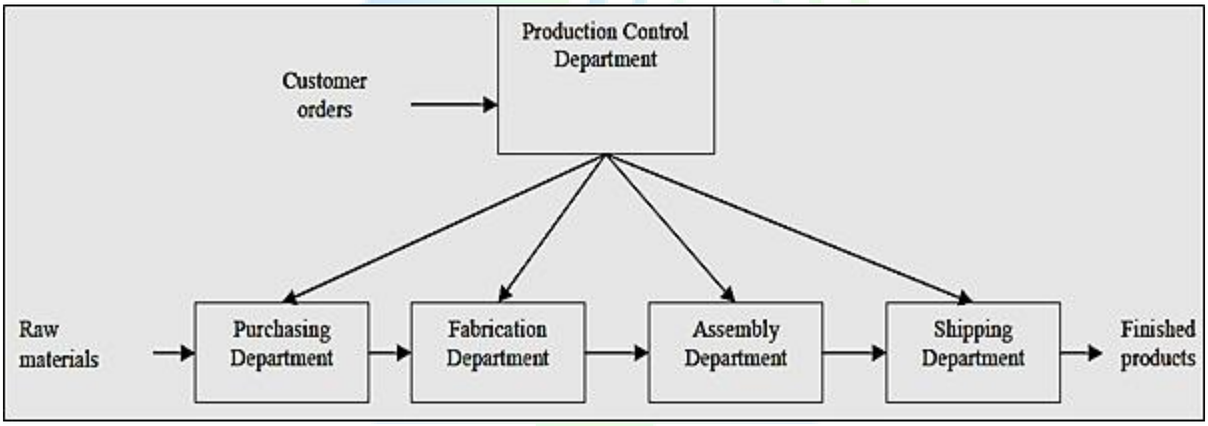


Fig: A Factory System

As shown in the above figure, two major components of the systems are the fabrication department making the parts and the assembly department producing the products.

A purchasing department maintains a supply of raw materials and a shipping department dispatches the finished products. A production control department receives orders and assigns work to the other departments.

Components of System:

1. Entity:

An entity is an object of interest in a system. *Example:* In the factory system, departments, orders, parts, and products are the entities.

2. Attribute:

An attribute denotes the property of an entity. *Example:* Quantities for each order, type of part, or several machines in a Department are attributes of the factory system.

3. Activity:

Any process causing changes in a system is called an activity. *Example:* Manufacturing process of the department.

4. State of the System:

The state of a system is defined as the collection of variables necessary to describe a system at any time, relative to the objective of study. In other words, the state of the system means a description of all the entities, attributes and activities as they exist at one point in time.

5. Event:

An event is defined as an instantaneous occurrence that may change the state of the system.

System	Entities	Attributes	Activities	Events	State Variables
Bank	Customers	Balance, Credit Status	Depositing, Withdrawal	Arrival, Departure	No. of busy tellers, No. of customers waiting
Production	Machines	Speed, Capacity	Welding, Stamping	Breakdown	Status of machine (busy, idle or down)
Communication	Messages	Length, Destination	Transmitting	Arrival at destination	Number waiting to be transmitted

Other Examples of System:

a. Traffic System
b. Telephone System
c. Supermarket System
d. Transportation Operation System
e. Hospital Facilities System and so on.

System Environment:

Everything that remains outside of the system under Consideration is called system environment. The system is often affected by change occurring outside the environment.

Some system activity may also produce changes that don't react on the system (but react outside the system). Such changes occurring outside the system is said to occur in the system environment.

When we are going to model the system, we must decide the boundary between the system and the environment. This decision depends on the purpose of the system study.

In the case of the factory system, factors controlling the arrival of the order may be considered to be the influence of the factory (i.e. part of the environment).

However, if the effect of the supply and demand is to be considered, there will be a relationship between the factory output and arrival of the order and this relationship must be considered as the activity of the system.

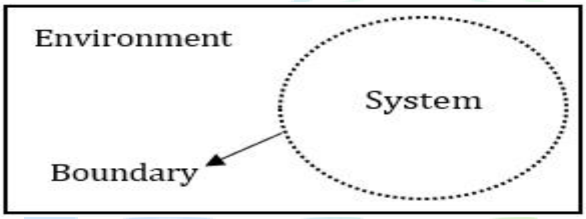


Fig: System Environment

System Types:

1. Closed System:

If any system shows endogenous activity then the system is said to be a closed system. A closed system is one where there is no interaction between the environment and the system components.

2. Open System:

If any system shows exogenous activity then the system is said to be an open system. The term exogenous is used to describe the activity in the environment that affects the system.

3. Isolated System:

The system, which is isolated from the external world (environment) is called an isolated system.

System Activities:

1. Endogenous Vs. Exogenous Activity:

The term ***endogenous*** is used to describe the activity that occurs within the system (activity occurs within an entity of system). And if there is only endogenous activity then the system is said to be a closed system.

The term ***exogenous*** is used to describe the activity in an environment that affects the system. Hence system, where there is an exogenous activity, is called an open system.

2. Deterministic Vs. Stochastic Activity:

If the outcome of the activity/system can be described in terms of the input (in terms of some mathematical function/formulae) then the activity is ***deterministic activity.***

When the effect of (outcome) of the activity vary over possible outcomes (can't be predicted using some mathematical function) then the activity is called **stochastic activity.**

The randomness of the stochastic activity is the part of the system environment because the exact outcome at any time is unknown. However, the random output can be measured and described in a probability distribution.

Discrete and Continuous System:

1. Discrete System:

Those systems whose state variable changes instantly at separate points in time is a discrete system. **Example:** Bank System. **State variable:** Number of customers.

Conclusion:

Due to these reasons in a bank, the number of customers may arrive only when a new customer arrives or leaves the bank. Here, the changes in the number of customers are in discrete time. Therefore, a bank can be taken as an example of a discrete system.

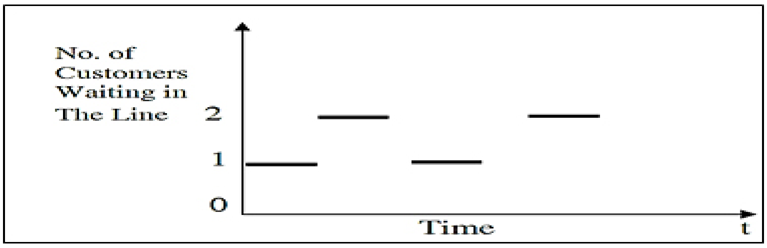


Fig: Discrete System

2. Continuous System:

Those systems whose state variables change continuously concerning time is a continuous system. **Example:** Aeroplane moving through the air. **State variable:** position and velocity.

Conclusion:

Here, the position of the Aeroplane is changing continuously with time. Therefore, an Aeroplane moving through the air is an example of a continuous system.

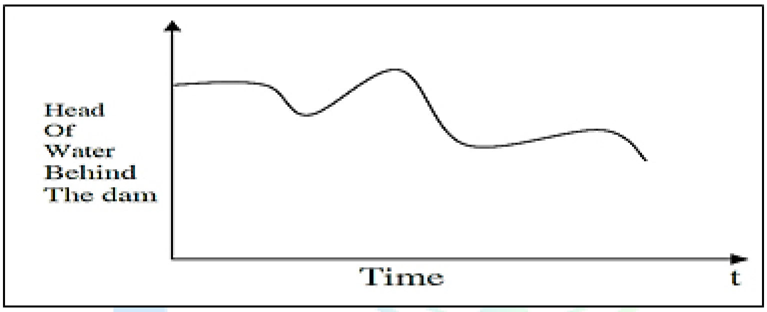


Fig: Continuous System

System Modeling:

A **Model** is defined as a representation of a system to study the system. In simple words, it is also defined as a *simplification of reality.* The model constructs a conceptual framework that describes a system.

It is necessary to consider those aspects of systems that affect the problem under investigation (unnecessary details to be removed).

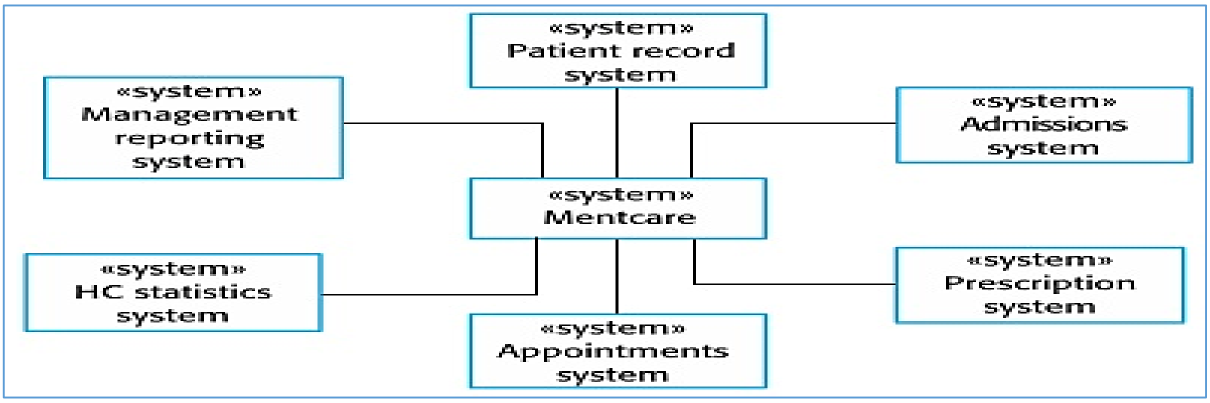


Fig: System Modeling of Mentcare System

A model of a system is a set of assumptions and approximations about how the system works and the task of deriving a model of a system is called system model.

Since the purpose of the study will determine the nature of the information that is gathered, there is no unique model of a system.

Different model of the same system will be produced by different analysts interested in different aspect of the system or by the same analyst as his understanding of the system changes. Studying a model instead of a real system is usually much easier, faster, cheaper and safer.

Deriving a Model:

A model of a system can be derived in the following two ways:

1. Establishing A Model Structure:

It determines the system boundary, entities, attributes, activities and state of a system.

2. Supplying Data:

In this phase, the values are provided to the system attributes that can have and relationships involve in the activities are defined.

These 2 jobs of creating a structure and providing the data are the parts of one task rather than as two separate tasks because they are so intimately related neither can be done without others.

The assumption about the system directs the gathering of data, and analysis of the data confirms the assumption. To illustrate this process, we consider the description of a supermarket.

Shoppers needing *several items* of shopping *arrive* at a supermarket. They *get* a basket, if one *available*, carry out their *shopping*, and then *queue* to *check-out* at one of several *counters*. After checking out, they *return* the *basket* and *leave*.

Here, in the assumption of the supermarket system, some keywords are made italic to point out the features of a system that must be reflected in the model. In this system, the entities are shopper, basket and counters. The attributes are the number of items, availability, and number of occupancy. The activities are arriving at the supermarket, getting a basket, returning a basket and leaving the supermarket.

Types of Models:

1. Physical Model:

In a physical model of a system, the system attributes are represented by measurement such as voltages or position of shafts. Here, the system activities are deflected in physical logic that derives a model. For example, the rate at which the shaft of the dc motor turn depends upon voltage applied to the motor.

If the applied voltage is used to represent the velocity of the vehicle, then the number of revolution of the shaft is a measure of the distance the vehicle has travelled.

2. Mathematical Model:

This model uses symbolic notation and mathematical equation to represent a system. In this model, attributes are represented by variables and activities are represented by a mathematic function that inter-relates variables.

3. Static Model:

This model can only show the values that system attributes takes when the system is in balance.

4. Dynamic Model:

This model can show the values of the system attributes that changes over time due to the effect of the system attributes.

5. Analytical Methods:

If the model is simple enough, it may be possible to work with its relationship and quantities to get an exact analytical method. This method uses the deductive region of mathematical theory to solve a model. It directly produces a general solution.

Let's consider that, we know the distance to be travelled and velocity, then we can work with the model to get as the time that will be required.

6. Numerical Methods (Simulation Method):

If an analytical solution to the mathematical is available and computationally efficient, it is usually desirable to study the model using an analytical method. However, many systems are highly complex, so those valid mathematical models of them are complex.

In this case, the model must be studied using simulation i.e. numerically exercising the model for the inputs in questions to see how they affect the outputs measures of performance.

Numerical methods use the computational produces to solve the equation. It produces a solution in steps, each step give a solution for once at a condition and calculation are repeated until a final solution is obtained.

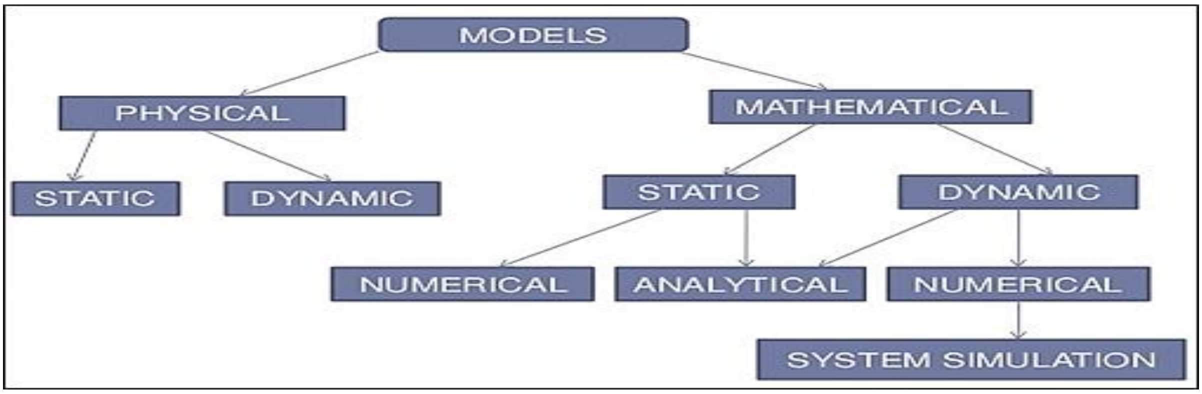


Fig: Types of Models

Principle of Modeling:

It is not possible to provide rules by which mathematical models are built, but several guiding principles can be stated. They do not describe distinct steps carried out in building a model. They describe different viewpoints from which to judge the information to be included in the model.

1.Block Building:

The description of the system should be organized in a series of blocks. The aim of constructing the blocks is to simplify the specification of the interactions within the system.

Each block describes a part of the system that depends upon a few, preferably one, input variables and results in a few output variables. The system as a whole can then be described in terms of the interconnections between the blocks. Correspondingly, the system can be represented graphically as a simple block diagram.

2.Relevance:

The model should only include those aspects of the system that are relevant to the study objectives.

As an example, if the factory system the study aims to compare the effects of different operating rules on efficiency, it is not relevant to consider the hiring of employees as an activity.

While irrelevant information in the model may not do any harm, it should be excluded because it increases the complexity of the model and causes more work in solving the model.

3. Accuracy:

The accuracy of the information gathered for the model should be considered. In an aircraft system, the accuracy with which the movement of aircraft is described depends upon the representation of the airframe. If these are not accurate it gives a false result while testing a system for output.

4. Aggregation:

A further factor to be considered is the extent to which several individual entities can be grouped together into larger entities.

Areas of Application:

System simulation is a technique, which finds application in almost each and every field. Some of the areas in which it can be successfully employed are listed below:

1. Manufacturing:

Design analysis and optimization of the productions system, materials management, capacity planning, layout planning and performance evaluation, evaluation of process quality.

2. Business:

Market analysis, prediction of consumer behavior, optimization of marketing strategy and logistics, comparative evaluation of marketing campaigns.

3. Military:

Testing of alternative combat strategies, air operations, sea operations, simulated war exercises, practicing ordinance effectiveness, and inventory management.

4. Healthcare Application:

Such as planning health services, expected patient density, facilities requirement, hospital staffing, estimating the effectiveness of a health care program.

5. Communication Application:

Such as network design and optimization, evaluating network reliability, manpower planning, sizing of message buffers.

6. Computer Application:

Such as designing hardware configuration and operating systems protocols, sharing, and networking.

7. Economic Application:

Such as portfolio management, forecasting the impact of Govt. Policies and internal market fluctuations on the economy, budgeting and forecasting market fluctuation.

8. Transport Application:

Design and testing of alternative transportation policy, transportation network- road, railways, airways, etc., evaluation of timetables, traffic and planning.

9. Environmental Application:

Solid waste management, performance evaluation of environmental program, evaluation of population control systems.

10. Biological Application:

Such as population genetics and the spread of epidemics.

Verification, Validation, and Calibration of Model:

After the development of a model is functionally complete, we should ask "does it work correctly". There are two paths to this question:

a. First does it operate the way the analyst intended?
b. Does it behave the way the real system works?

Verification:

Verification focuses on the internal consistency of a model. Verification checks that the implementation of the simulation model (program) corresponds to the model. It concerns with the building the model right. It is utilized in the comparison of the conceptual model to the computer representation that implements the conception.

It asks the question, is the model implemented correctly in the computer? Are the input parameters and logical structure of the model correctly represented? It is the process of comparing the computer code with the model to ensure that the code is a correct implementation of the model.

Validation:

Validation is concerned with the correspondence between the model and the reality. Its concern with building the right model. It is utilized to determine that a model is an accurate representation of the real system.

Validation is usually achieved through the calibration of the model, an iterative process of comparing the model to actual system behavior and using discrepancies between two and the insight gained to improve the model.

This process is repeated until model accuracy is a judge to be acceptable. It is the process of comparing the model's output with the behavior of the phenomenon. In other words: comparing model execution to reality (physical or otherwise).

Three Steps Approach in the Validation Process:

a. Build a model that has a high face validity
b. Validate model assumption
c. Compare the model input and output transformation to correspond input-output transformation of the real system.

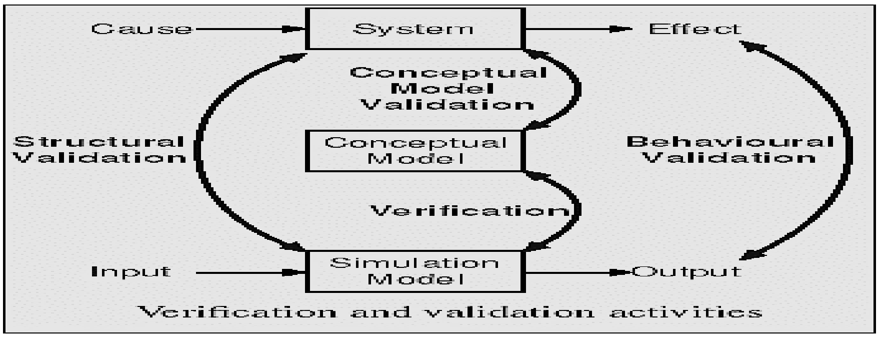


Fig: Verification and Validation Activities

Calibration:

It is an iterative process of comparing the model to the real system, making adjustments or major changes to the model, comparing the revised model to reality, making additional adjustments, comparing again and so on. It Checks that the data generated by the simulation matches real (observed) data.

The process of parameter estimation for a model. Calibration is a tweaking/tuning of existing parameters and usually does not involve the introduction of new ones, changing the model structure.

In the context of optimization, calibration is an optimization procedure involved in system identification or during experimental design.

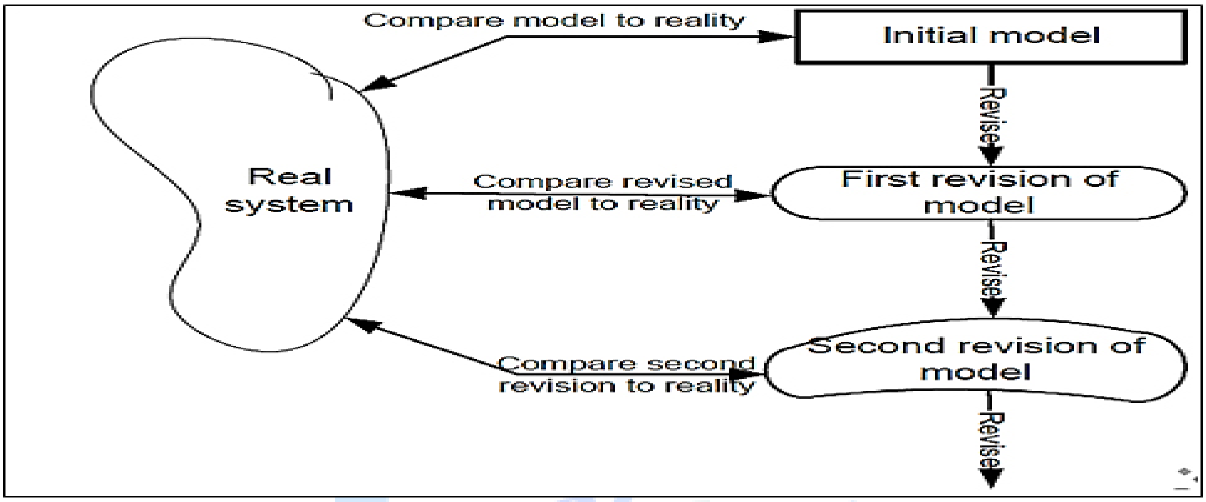


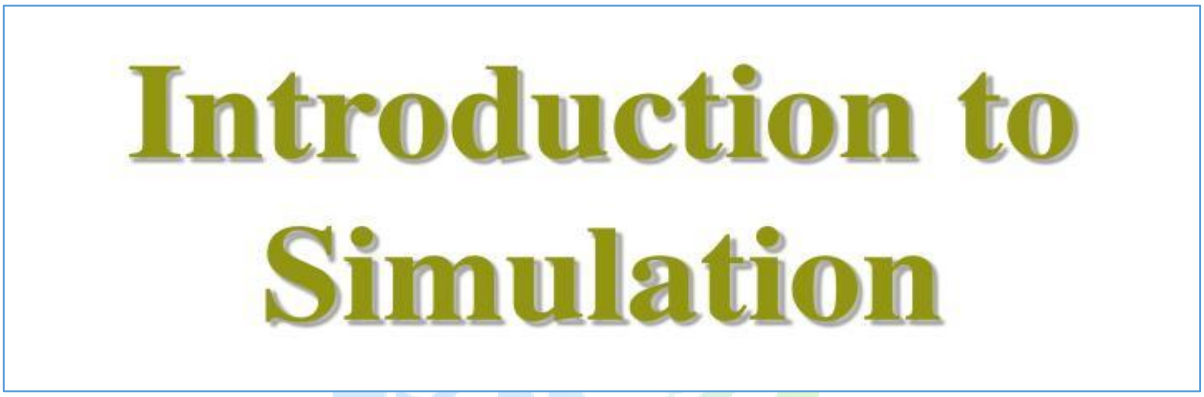
Fig: Calibration, Verification and Validation Relationship

Unit II: System Simulation - Simulation and Modeling

Introduction to Simulation:

Simulation is one of the most powerful tools available to decision-makers responsible for the design and operation of complex processes and systems. It makes possible to study, analysis and evaluation of a situation that would not be otherwise possible.

In an increasingly competitive world, simulation has become an essential problem-solving methodology for engineers, designers, and managers.



Simulation can be defined as the process of designing a model of a real system and conducting an experiment with these models of system and, or evaluating various strategies for the operation of the system. Thus, it is important that the model we design in such a way, model behavior mimics the behavior of a real system.

Simulation allows us to study the situation even though we are unable to experiment directly with the real system, either the system doesn't exist or it is too difficult or expensive to directly manipulate it.

We consider the simulation to include both constructions of the model and experimental use of the model for studying problems. Thus, we can think of simulation as an experiment and applied methodology which seeks to:

a. Describes the behavior of the system.
b. Use of model to predict future behavior i.e. the effect that will be produced by changes in the system or in its method of operation.

When To Use Simulation?

Following Are Some Of The Purposes For Which Simulation May Use:

a. Simulation is very useful for experiments with the internal interactions of a complex system, or of a subsystem within a complex system.

b. Simulation can be employed to experiment with new designs and policies, before implementing them.

c. Simulation can be used to verify the results obtained by analytical methods and to reinforce the analytical techniques.

d. Simulation is very useful in determining the influence of changes in input variable on the output of the system.

e. Simulation helps in suggestion modification in the system under investigation for its optimal performance.

Advantages:

Simulation has several advantages. First of all basic concept of simulation is easy to understand and hence often easier to justify to management or customer than some of the analytical methods.

Also, a simulation model may be more. Because its behavior has been compared to that of a real system or because it requires fewer simplifying assumptions and hence captures more of true characteristics of the system understudied.

Other Advantages Include:

a. We can taste new designs, layouts, etc. without assigning resources to their implementation.

b. It can be used to explore new staffing policy, operating procedure, design rules, organizational structure, information flows, etc. without disturbing the ongoing operation.

c. It allows us to identify a bottleneck (jams) in information, matters, products, flows and test option for increasing the flow rates.

d. It allows us to test a hypothesis about how or why a certain phenomenon occurs in the system.

e. It allows us to control time. Thus we can operate the system several month or years of experience in a matter of seconds allowing us to quickly look at a long time or we can slow down the phenomenon for study.

f. It allows us to gain insights into how a model the system actually works and understanding of which variables are most important up to a performance.

g. Its great strength is its ability to let us experiment with new and unfamiliar situations.

Disadvantages:

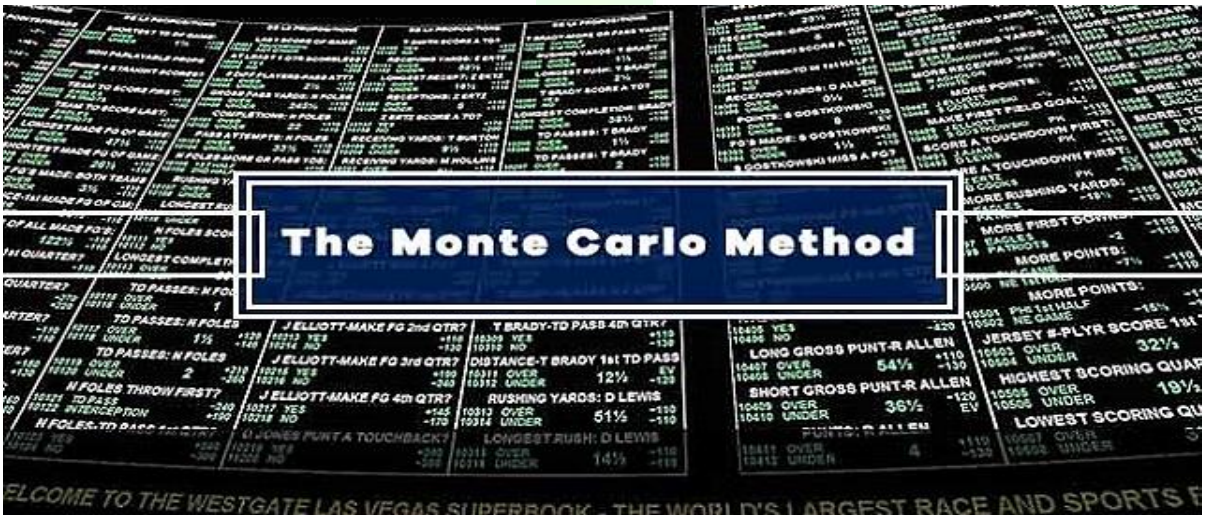
Even though simulation has many strengths and advantages, it is not without drawbacks. And some are:

a. Simulation is an art that requires specialized trainers and therefore, skill labels of practice vary widely. The utility of the study depends upon the quality of the model and the skill of the models.

b. Gathering highly reliable input data can be time consuming and the resulting data is sometimes highly comprised of insufficient data or poor management decisions.

c. Simulation models are input and output models i.e. they yield the portable output of the system for a given input. These are, therefore, run rather than solved. They do not yield optional rather they serve as a tool for analysis of the behavior of a system under conditions specified by the experiments.

Technique of Simulation – Monte Carlo Method:



Monte Carlo simulation is a computerized mathematical technique to generate random sample data, based on some known distribution for numerical experiments. This method is applied to risk quantitative analysis and decision-making problems.

This method is used by the professionals of various profiles such as finance, project management, energy, manufacturing, engineering, research & development, insurance, oil & gas, transportation, etc.

This method was first used by scientists working on the atom bomb in 1940. This method can be used in those situations where we need to make an estimate and uncertain decisions such as weather forecast predictions.

Important Characteristics:

Following Are The Three Important Characteristics Of Monte-Carlo Method:

a. Its output must generate random samples.
b. Its input distribution must be known.
c. Its result must be known while performing an experiment.

Advantages:

a. Easy to implement.
b. Provides statistical sampling for numerical experiments using the computer.
c. Provides an approximate solution to mathematical problems.
d. Can be used for both stochastic and deterministic problems.

Disadvantages:

a. Time consuming as there is a need to generate a large number of sampling to get the desired output.
b. The results of this method are only the approximation of true values, not the exact.

Flow Diagram:

The Monte Carlo simulation that is used to measure the EENS must reflect accurately the way a power system is operated and all the information that is available in the operational time frame.

The Following Illustration Shows A Generalized Flowchart Of Monte Carlo Simulation:

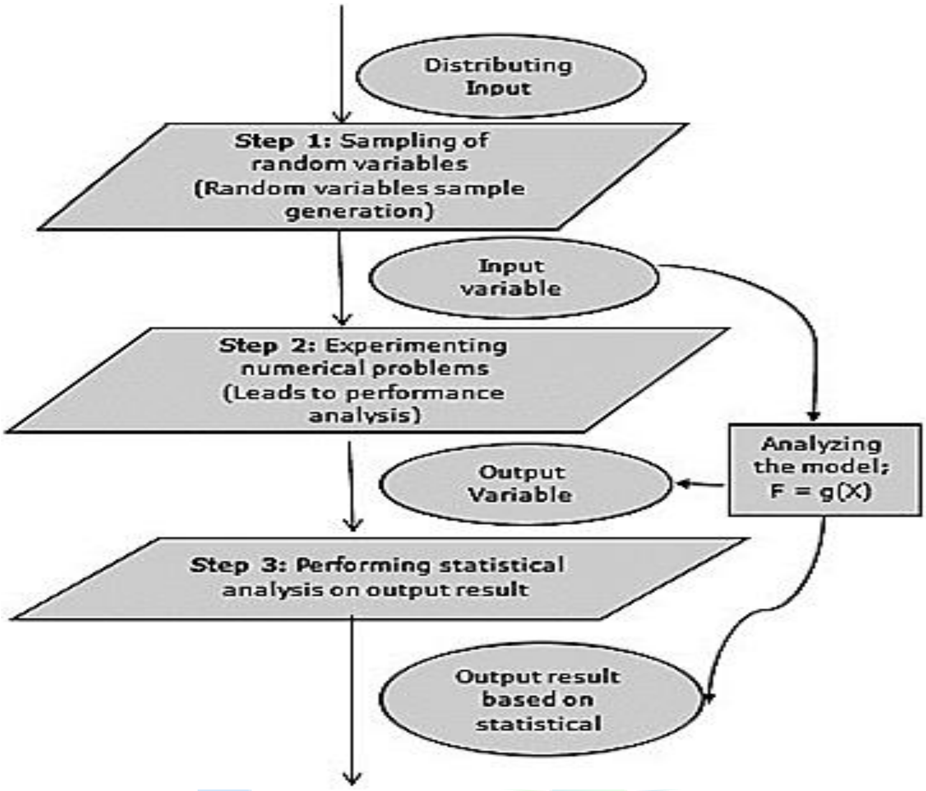
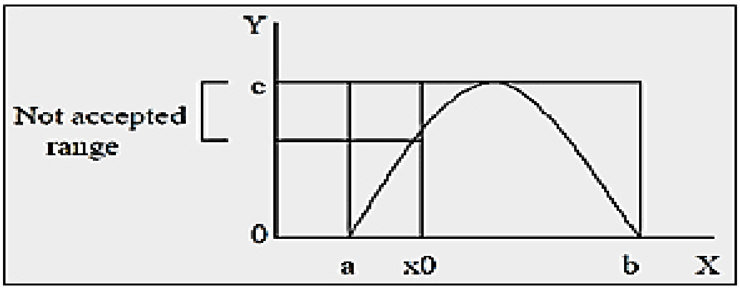


Fig: Flowchart of Monte Carlo Simulation

Problem Depicting Monte Carlo Method:

This method is applied to solve both deterministic as well as stochastic problems. There are many deterministic problems also which are solved by using random numbers and interactive procedure of calculations. In such a case, we convert the deterministic model into a stochastic model, and the results obtained are not exact values, but only estimates.



For example, we shall consider the problem taking integral of a single variable over a range which corresponds to finding the area under graph representing the function f(x). Let us suppose that f(x) is positive and has **a** and **b** as bounds above by **c**.

Then as shown in the figure, the function f(x) will be contained within the rectangle of sides c and (b-a). Now, we can pick up points at random within the rectangle and determine whether they lie beneath the curve or not. The points selected are assumed to be obtained from a uniformly distributed random number generator.

Two successive samplings are made to get X and Y coordinates so that X is in the range **a** to **b** and Y is in the range **0** to **c**. The fraction of points that fall or below the curve will be approximately the ratio of the area under the curve to the area of the rectangle. If **N** points are drawn and **n** of them fall under curve then;

$$\frac{\textit{number of points inside the curve}}{\textit{number of points falling inside rectangle}} = \frac{\textit{area of curve}}{\textit{area of rectangle}}$$

Here,

Curve = f(x),
n=randomly selected points laying inside the curve,
N=total numbers of points selected,
Area of rectangle=c*(b-a)

Now we have;

$$\frac{n}{N} = \frac{\int_a^b f(x)dx}{c * (b - a)}$$

Which is the mathematical statement of the Monte Carlo method?

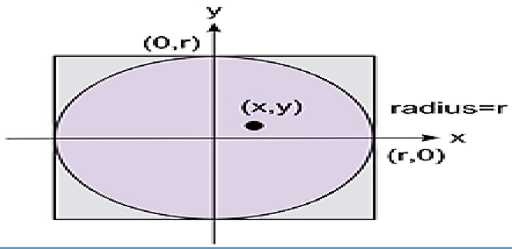
The accuracy increase as **N** increases. After enough points have taken, the value of the integral (i.e. the area under the curve represented by the function f(x)) is obtained by **n/N * c * (b - a).**

The computational technique is shown in the figure. At each trial, the value of x is selected at random between 'a' and 'b', say X0. Similarly, the second random number is selected between 0 and C to give y. If y <= f(x0) then point is inside the curve and count 'n' otherwise point will not lie in the curve and the next point will be picked.

The application of the Monte Carlo Method for evaluation of Pi (π) is converting a deterministic model into a stochastic model. Some examples that use random sampling in problem-solving are as follows:

a. To find the area of irregular surface figure
b. Numerical Integration of single-variable function
c. A Gambling Game.
d. Random Walk Problem

Determine The Value Of Pi (Π) Using Monte Carlo Method:



$$\frac{Area\ of\ quadrant\ of\ circle}{Area\ of\ Rectangle} = \frac{Number\ of\ points\ inside\ the\ curve}{Number\ of\ points\ inside\ the\ rectangle}$$

$$or, \frac{{}^{1}/_{4}\,\pi r^2}{r^2} = \frac{n}{N}$$

$$\therefore\ \pi = \frac{4n}{N}$$

We use random number generation method to determine the sample points that lie inside or outside the curve. Let (x_0, y_0) be an initial guess for the sample point than from a linear congruential method of random number generation:

$X_{i+1} = (a_{xi} + c) \bmod m$

$Y_{i+1} = (a_{yi} + c) \bmod m$

Where a & c are constants, m is the upper limit of generated random number. If $y \leq y_i$ then increment n.

Example:

We have circle equation = $x^2 + y^2 = 1$ or $y = \sqrt{(1 - x^2)}$ Now, generate the random numbers x and y within the interval 0 and 1.

For x: $x_0 = 27$, a = 17, c = 0, m = 100

For y: $y_0 = 47$, a = 17, c = 0, m = 100

X'	Y'	$\sqrt{(1 - x^2)}$	In/Out
0.59	0.99	0.962	In
0.03	0.83	0.99	In
0.51	0.11	0.86	In
0.67	0.87	0.74	Out

Now,

Points inside the curve (n) = 3

Points inside the rectangle (N) = 4

Value of pi = (n/N)*4 = 3

Numerical Integration using Monte Carlo Method:

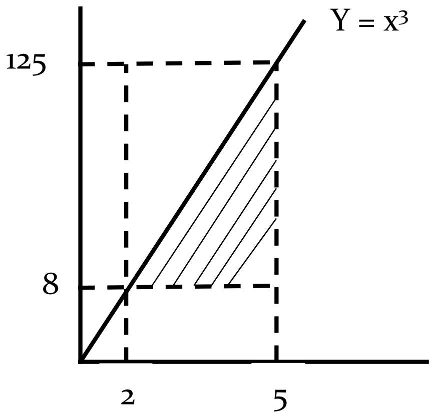
$$I = \int_a^b f(x)dx$$

From Monte Carlo Method

$$I = \frac{n}{N} * (b - a) * c$$

Example:

$$I = \int_{2}^{5} x^3 dx \ \ using\ Monte\ Carlo\ Method$$



We have to generate a random number for x and y. For x (random number be in range 2 to 5) & for y (random number be in range 8 to 125).

Here, the area of the rectangle under the given condition = (5 - 2) * (125 - 8) = 351, also we know, ***I = n/N * area of rectangle***

Now, we can select the random points inside the curve (using the random number generation method).

For x: x0 = 23, a = 17, c = 0, m = 50

Fir y: y0 = 61, a = 59, c = 0, m = 125

X	X' = X*0.1	Y	X'3	In/Out
23	2.3	59	12.167	Out
41	4.1	99	68.921	Out
47	4.7	91	103.823	In
49	4.9	119	117.649	Out
33	3.3	21	35.937	In
11	1.1	114	1.331	Out
37	3.7	101	50.653	Out

We get,

Points inside the curve (n) = 2

Points inside the rectangle (N) = 6

I = n/N * area of rectangle

I = 2/6 * 351

I = 117

Comparison of Simulation and Analytical Method:

Once we have built a mathematical model, it must then be examined to see how it can be used to answer the questions of interest in the system is supposed to represent. If the model is simple enough, it may be possible to work with its relationships and quantities to get an exact, analytical solution.

In the d = v*t example, if we know the distance to be travelled and the velocity, then we can work with the model to get t = d/v as the time that will be required.

This is a very simple, closed-form solution obtainable with just paper and pencil, but some analytical solutions can become extraordinarily complex, requiring vast computing resources; inverting a large non-sparse matrix is a well-known example of a situation in which there is an analytical formula known in principle, but obtaining it numerically in a given instance is far from trivial.

If an analytical solution to a mathematical model is available and is computationally efficient, it is usually desirable to study the model in this way rather than via a simulation.

However, many systems are highly complex, so that valid mathematical models of them are themselves complex, precluding any possibility of an analytical solution. In this case, the model must be studied using simulation, i.e., numerically exercising the model for the inputs in question to see how they affect the output measures of performance.

While there may be an element of truth to pejorative old saws such as “method of last resort” sometimes used to describe simulation, the fact is that we are very quickly led to simulation in many situations, due to the sheer complexity of the systems of interest and of the models necessary to validly represent them.

Given, then, that we have a mathematical model to be studied using simulation (henceforth referred to as a simulation model), we must then look for particular tools to execute this model (i.e. actual simulation).

Difference between Simulation and Analytic:

Basis	Simulation	Analytic
Input Parameterization	Measured or Invented	Measured or invented (with certain limitations)
Model Components	Virtually anything	Composed of limited basic building blocks
Model Outputs	Anything that can be measured	Equilibrium measures
Effort To Construct Model	Arbitrary	Modest
Computational Cost	Typically large	Typically small

Underlying Concepts	Probability or Statistics	Algebra to stochastic processes
Special Properties	Credible	Insight, optimization

Experimental Nature of Simulation:

The simulation technique makes no specific attempt to isolate (separate) the relationship between any particular variables; instead, it observes how all variables of the model change with time.

The relationship between the variables must be derived from these observations. Simulation is, therefore, essentially an experimental problem-solving technique. Many simulation runs have to be made to understand the relationships involved in the system, so the use of simulation in a study must be planned as a series of experiments.

Types of Simulation:

A simulator is a device, computer program, or system that performs a simulation. A simulation is a method for implementing a model over time. There are three types of commonly uses simulations:

1. Live:

Simulation involving real people operating real systems.

a. Involve individuals or groups
b. May use actual equipment
c. Should provide a similar area of operations
d. Should be close to replicating the actual activity

2. Virtual:

Simulation involving real people operating simulated systems. Virtual simulations inject Human-In-The-Loop in a central role by exercising:

a. Motor control skills (e.g. flying an aeroplane)
b. Decision skills (e.g. committing fire control resources to action)
c. Communication skills (e.g. members of a C4I team)

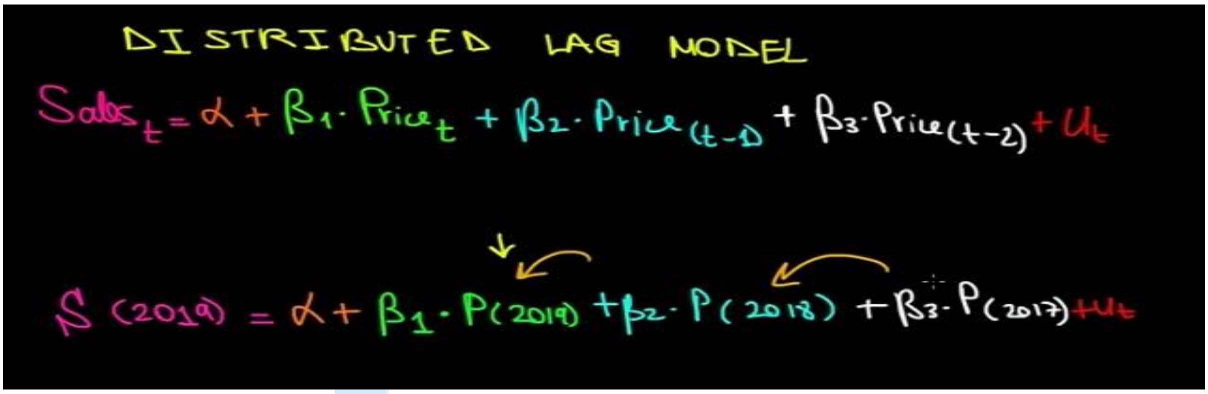
3. Constructive:

Simulation involving simulated people operating simulated systems. Real people can stimulate (make inputs) but are not involved in determining outcomes. Constructive simulations offer the ability to:

a. Analyze concepts
b. Predict possible outcomes
c. Stress large organizations
d. Make measurements
e. Generate statistics
f. Perform analysis

Distributed Lag Model:

If the regression model includes not only the current but also the lagged (past) values of the explanatory variables (the x's) it is called a distributed lag model. If the model includes one or more lagged values of the dependent variable among its explanatory variables, it is called an autoregressive model. This is known as a dynamic model.



In other word, Distributed Lag Model is defined as a type of model that have the property of changing only at fixed interval of time and based on current values of variables on other current values of variables and values that occurred in previous intervals.

In economic studies, some economic data are collected over uniform time intervals such as a month or year. This model consists of linear algebraic equations that represent continuous system but data are available at fixed points in time.

For Example: Mathematical Model of National Economy

Let,

C = Consumption
I = Investment
T = Taxes
G = Government Expenditures
Y = National Income

Then

C = 20 + 0.7 (Y – T)
I = 2 + 0.1 Y
T = 0.2 Y
Y = C + I + G

All the equation are expressed in billions of rupees. This is static model and can be made dynamic by lagging all the variables as follows:

C = $20 + 0.7(Y_{-1} - T_{-1})$
I = $2 + 0.1Y_{-1}$
T = $0.2Y_{-1}$
Y = $C_{-1} + I_{-1} + G_{-1}$

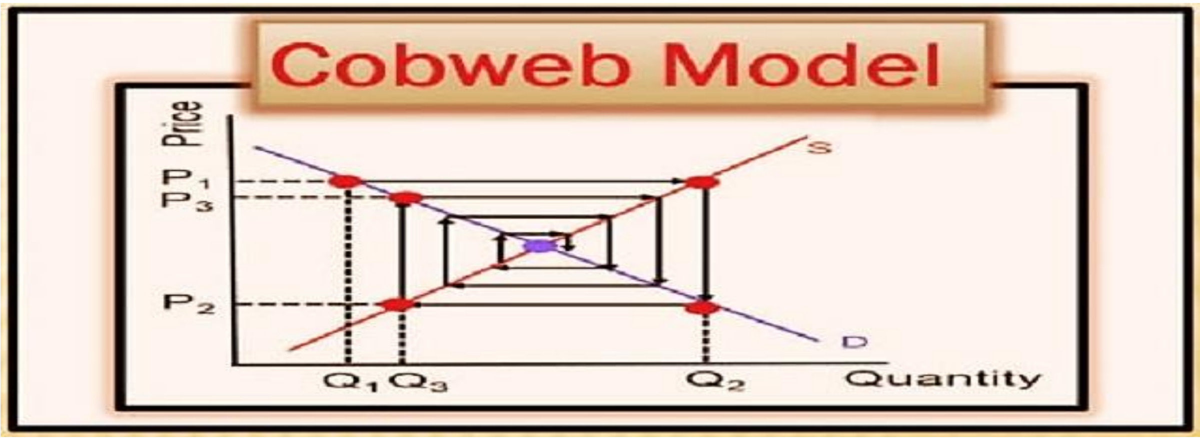
Any variable that can be expressed in the form of its current value and one or more previous value is called a lagging variable. And hence this model is given the name distributed lag model. The variable in a previous interval is denoted by attaching –n suffix to the variable. Where -n indicates the n^{th} interval.

Advantages of Distributed Lag Model:

a. Simple to understand and can be computed by hand, computers are extensively used to run them.
b. There is no need for special programming language to organize the simulation task.

Cobweb Model:

Cobweb theory is the idea that price fluctuations can lead to fluctuations in supply which cause a cycle of rising and falling prices.



In a simple cobweb model, we assume there is an agricultural market where supply can vary due to variable factors, such as the weather.

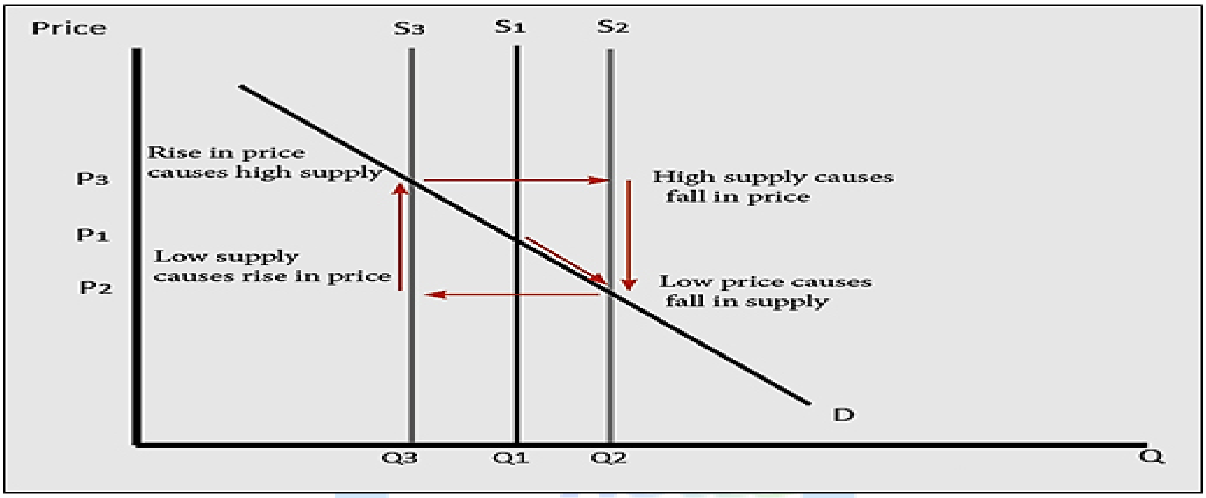
Assumptions of Cobweb theory:

a. In an agricultural market, farmers have to decide how much to produce a year in advance, before they know what the market price will be. (supply is price inelastic in short-term)

b. A key determinant of supply will be the price from the previous year.

c. A low price will mean some farmers go out of business. Also, a low price will discourage farmers from growing that crop in the next year.

d. Demand for agricultural goods is usually price inelastic (a fall in price only causes a smaller % increase in demand).



Explanation:

1. If there is a very good harvest, then supply will be greater than expected and this will cause a fall in price.

2. However, this fall in price may cause some farmers to go out of business. Next year farmers may be put off by the low price and produce something else. The consequence is that if we have one year of low prices, next year farmers reduce the supply.

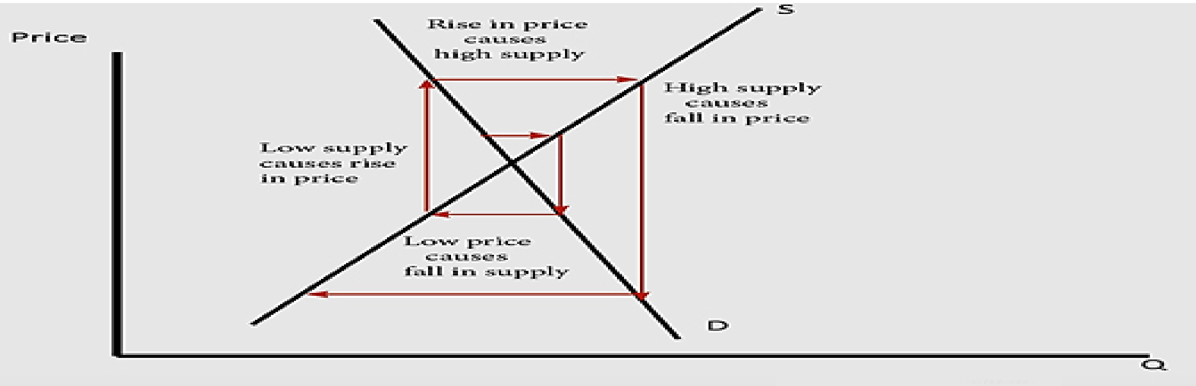
3. If supply is reduced, then this will cause the price to rise.

4. If farmers see high prices (and high profits), then next year they are inclined to increase supply because that product is more profitable.

In theory, the market could fluctuate between high price and low price as suppliers responds to past prices.

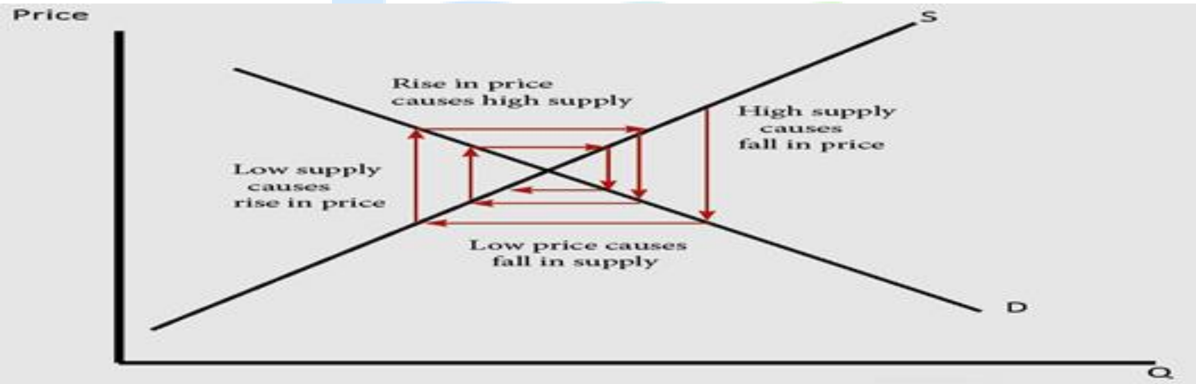
Cobweb Theory and Price Divergence:

The price will diverge from the equilibrium when the supply curve is more elastic than the demand curve, (at the equilibrium point).



If the slope of the supply curve is less than the demand curve, then the price changes could become magnified and the market more unstable.

Cobweb Theory and Price Convergence:



At the equilibrium point, if the demand curve is more elastic than the supply curve, we get the price volatility falling, and the price will converge on the equilibrium

Limitations of Cobweb theory:

1. Rational Expectations:

The model assumes farmers base next year's supply purely on the previous price and assume that next year's price will be the same as last year (adaptive expectations). However, that rarely applies in the real world. Farmers are more likely to see it as a 'good' year or 'bad year and learn from price volatility.

2. Price Divergence Is Unrealistic And Not Empirically Seen:

The idea that farmers only base supply on last year's price means, in theory, prices could increasingly diverge, but farmers would learn from this and pre-empt changes in price.

3. It May Not Be Easy Or Desirable To Switch:

A potato grower may concentrate on potatoes because that is his specialty. It is not easy to give up potatoes and take to aborigines.

4. Other Factors Affecting Price:

There are many other factors affecting price than a farmers decision to supply. In global markets, supply fluctuations will be minimized by the role of importing from abroad. Also, demand may vary. Also, supply can vary due to weather factors.

5. Buffer Stock Schemes:

Governments or producers could band together to limit price volatility by buying surplus.

Steps of Simulation Study:

1. Problem Formulation:

The study begins with defining the problem statement. It can be developed either by the analyst or client. If the statement is provided by the client, then the analyst must take extreme care to ensure that the problem is clearly understood.

If a problem statement is prepared by the simulation analyst, the client must understand and agree with the formulation. Even with all of these precautions, the problem may need to be reformulated as the simulation study progresses.

2. Setting of Objectives and Overall Project Plan:

Another way to state this step is to "prepare a proposal." The objectives indicate the questions to be answered by the simulation study. Whether the simulation is appropriate or not is to be decided at this stage.

The overall project plan should include a statement of the alternative systems and a method for evaluating the effectiveness of these alternatives.

The plan includes several personnel, number of days to complete the task, stages in the investigation, output at each stage, cost of the study and billing procedures if any.

3. Model Conceptualization:

Model is a simplification of reality. The real-world system under investigation is abstracted by a conceptual model. It is recommended that modelling begins with a simple model and grows until a model of appropriate complexity has been achieved.

For example, consider the model of a manufacturing and material handling system. The basic model with the arrivals, queues, and servers is constructed. Then, add the failures and shift schedules.

Next, add the material-handling capabilities. Finally, add special features. Constructing an excessive complex model will add to the cost of the study and the time for its completion, without increasing the quality of the output.

Maintaining client involvement will enhance the quality of the resulting model and increase the client's confidence in its use.

4. Data Collection:

This step involves gathering the desired input data. The data changes over the complexity of the model. Data collection takes a huge amount of total time required to perform a simulation. It should be started at early stages together with model building. The collection of data should be relevant to the objectives of the study.

5. Model Translation:

The conceptual model constructed in Step 3 is coded into a computer recognizable form, an operational model. The suitable simulation language is used.

6. Verified:

Verification is concerning the operational model. Is it performing properly? If the input parameters and logical structure of the model are correctly represented in the computer, then verification is completed.

7. Validated:

Validation is the determination, that the model is an accurate representation of the real system. This is done by calibration of the model; an iterative process of comparing the model to the actual system behavior. This process is repeated until model accuracy is acceptable.

8. Experimental Design:

The alternatives to be simulated must be determined. For each the scenario that is to be simulated, decisions need to be made concerning the length of the simulation run, the number of runs (also called replications), and the length of the initialization period.

9. Production Runs And Analysis:

The production runs, and their subsequent analysis is used to estimate measures of performance for the system design that is being simulated.

10. More Runs:

After the completion of the analysis of runs, the simulation the analyst determines if additional runs are needed and any additional experiments should follow.

11. Documentation And Reporting:

There are two types of Documentation: Program and Progress. Program documentation is necessary for numerous reasons. If the program is going to be used again by the same or different analysts, it may be necessary to understand how the program operates. This will enable confidence in the program so that the client can make decisions based on the analysis.

Also, if the model is to be modified, this can be greatly facilitated by adequate documentation. Progress reports provide a chronology of work done and decisions made. It is the written history of a simulation project. The result of all the analyses should be reported clearly and concisely. This will enable the client to review the final formulation.

12. Implementation:

If the client has been involved throughout the study period, and the simulation analyst has followed all of the steps rigorously, then the likelihood of a successful implementation is increased.

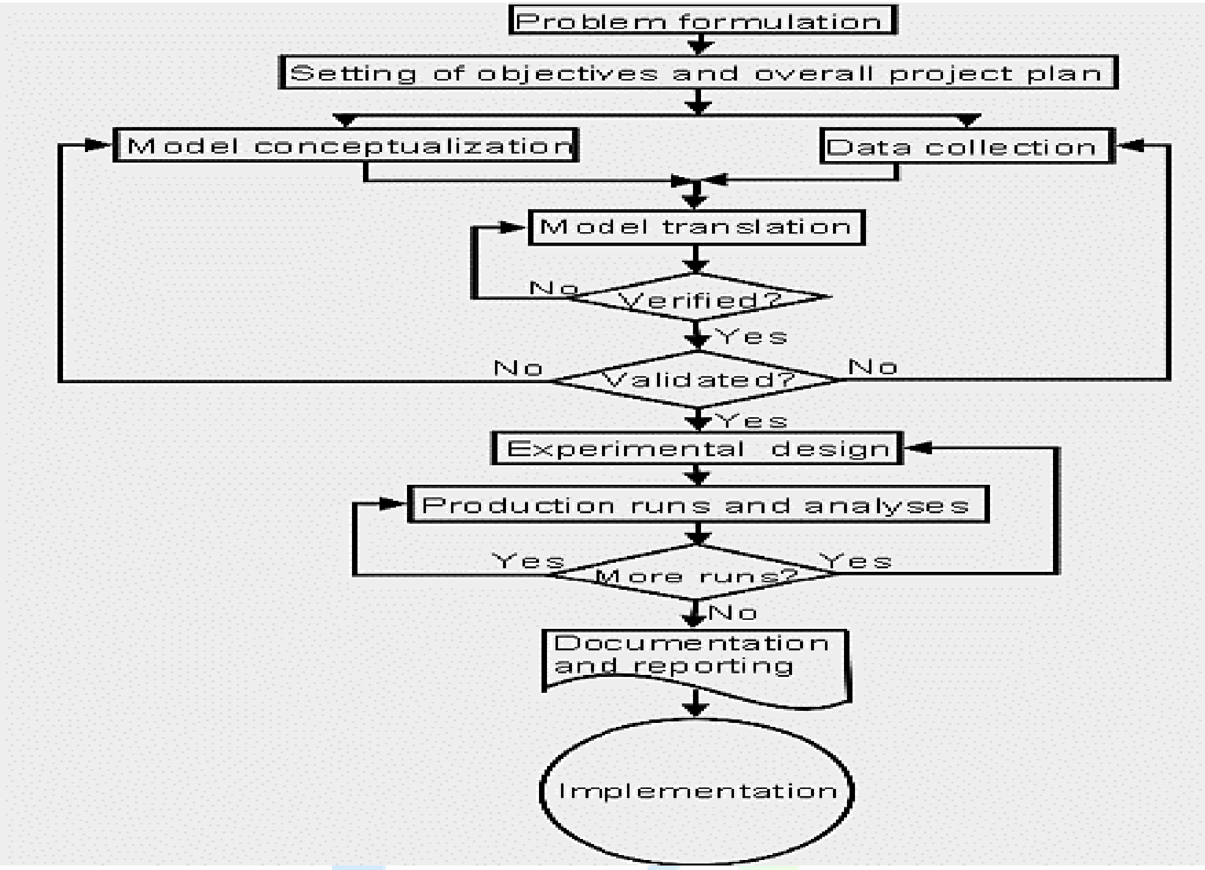


Fig: Steps of Simulation Study

Time Advancement Mechanism:

The simulation models we consider will be discrete, dynamic, and stochastic. Discrete event simulation concerns the modelling of a system as it evolves over time by a representation in which the state variables change instantaneously at separate points in time.

These points in time are the one which an event occurs, where an event is defined as an instantaneous occurrence that may change the state of a system.

Because of the dynamic nature of discrete-event simulation model, we must keep track of the current value of simulation time as the simulation proceeds and we also need a mechanism to advance simulation time from one variable to another. We call the variable in a simulation model that gives the current value of simulation time the simulation clock or simulation time.

Simulation time means the integral clock time and not the time a computer was taken to carry out the simulation. Two principal approaches for advancing the simulation clock are:

a. Next Event Time Advance
b. Fixed Increment Time Advance

Queuing Models and its Characteristics:

The Queuing theory provides predictions about waiting times, the average number of waiting customers, the length of a busy period and so forth. These predictions help us to anticipate situations and to take appropriate measures to shorten the queues.

A further attractive feature of the theory is quite an astonishing range of its applications. Some of the more prominent of these are telephone conversation, machine repair, toll booths, taxi stands, inventory control, the loading, and unloading of ships scheduling patients in the hospital clinics, production flow and applications in the computer field concerning program scheduling, etc.

A queuing system may be described as one having a service facility, at which units of some kind (called customers) arrive for service and whenever there are more units in the system than the service facility can handle simultaneously, a queue or waiting line develops.

The waiting units take their turn for service according to a pre-assigned rule and after service, they leave the system. Thus, the input to the system consists of the customers demanding service and the output is the serviced customers.

Characteristics of the Queuing System:

1. The Arrival Pattern:

The arrival pattern describes how a customer may become a part of the queuing system. The arrival time for any customer is unpredictable. Therefore, the arrival time and the number of customers arriving at any specified time intervals are usually random variables.

A Poisson distribution of arrivals corresponds to arrivals at random. In Poisson distribution, successive customers arrive after intervals which independently are and exponentially distributed. The Poisson distribution is important, as it is a suitable mathematical model of many practical queuing systems as described by the parameter "the average arrival rate".

2. The Service Mechanism:

The service mechanism is a description of the resources required for service. If there are an infinite number of servers, then there will be no queue. If the number of servers is finite, then the customers are served according to a specific order.

The time taken to serve a particular customer is called the service time. The service time is a statistical variable and can be studied either as the number of services completed in a given time or the completion period of service.

3. The Queue Discipline:

The most common queue discipline is the "First Come First Served" (FCFS) or "First-in, First-out" (FIFO). Situations like waiting for a haircut, ticket-booking counters follow FCFS discipline.

Other disciplines include "Last In First Out" (LIFO) where the last customer is serviced first, "Service In Random Order" (SIRO) in which the customers are serviced randomly irrespective of their arrivals.

"Priority service" is when the customers are grouped in priority classes based on urgency. "Preemptive Priority" is the highest priority given to the customer who enters into the service, immediately, even if a customer with lower priority is in service. "Non-preemptive priority" is where the customer goes ahead in the queue but will be served only after the completion of the current service.

The Number of Customers Allowed In the System:

In certain cases, a service system is unable to accommodate more than the required number of customers at a time. No further customers are allowed to enter until space becomes available to accommodate new customers. Such types of situations are referred to as finite (or limited) source queue.

Examples of finite source queues are cinema halls, restaurants, etc. On the other hand, if a service system can accommodate any number of customers at a time, then it is referred to as an infinite (or unlimited) source queue.

For example, in a sales department, where the customer orders are received; there is no restriction on the number of orders that can come in so that a queue of any size can form.

The Number of Service Channels:

The more the number of service channels in the service facility, the greater the overall service rate of the facility. The combination of arrival rate and service rate is critical for determining the number of service channels.

When there are several service channels available for service, then the arrangement of service depends upon the design of the system's service mechanism. Parallel channels mean, several channels providing identical service facilities so that several customers may be served simultaneously.

Series channel means a customer goes through successive ordered channels before service is completed. A queuing system is called a **one-server model,** i.e., when the system has only one server, and a **multi-server model** i.e., when the system has several parallel channels, each with one server.

1. Arrangement Of Service Facilities In Series:

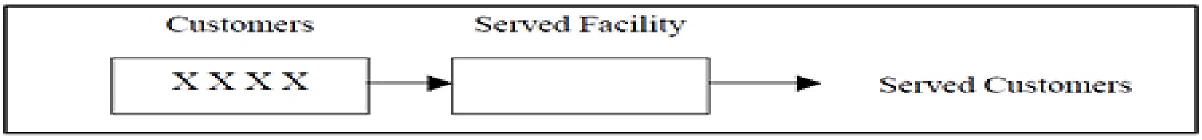


Fig: Single Queue Single Server

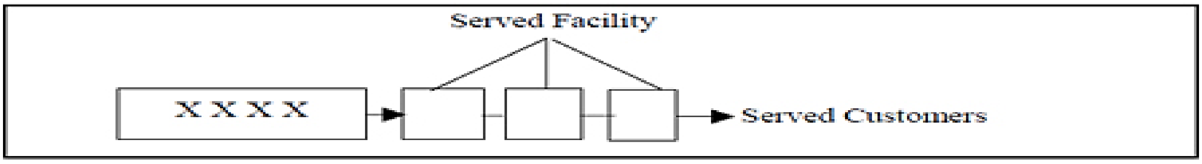
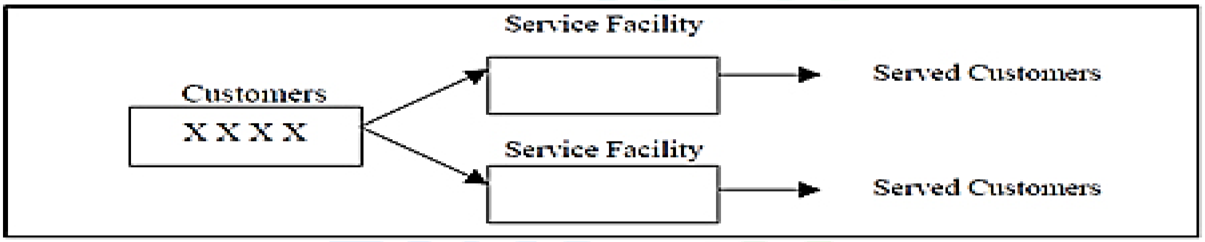
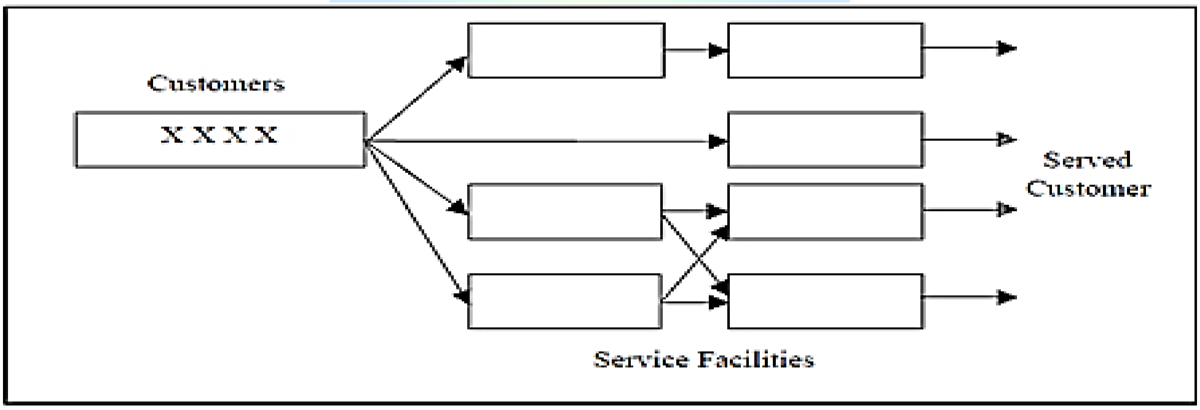


Fig: Single Queue, Multiple Server

2. Arrangement of Service Facilities In Parallel:



3. Arrangement of Mixed Service facilities:



Measuring the Performance of the System:

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

1. 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.

2. 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

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

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 *Ti* be the total time during the simulation that the queue is of length *i,* then $T(n) = T_1+T_2+T_3+$…… and $\hat{P}i = Ti/T(n)$ and so that we can rewrite the above equation as:

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

3. 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.

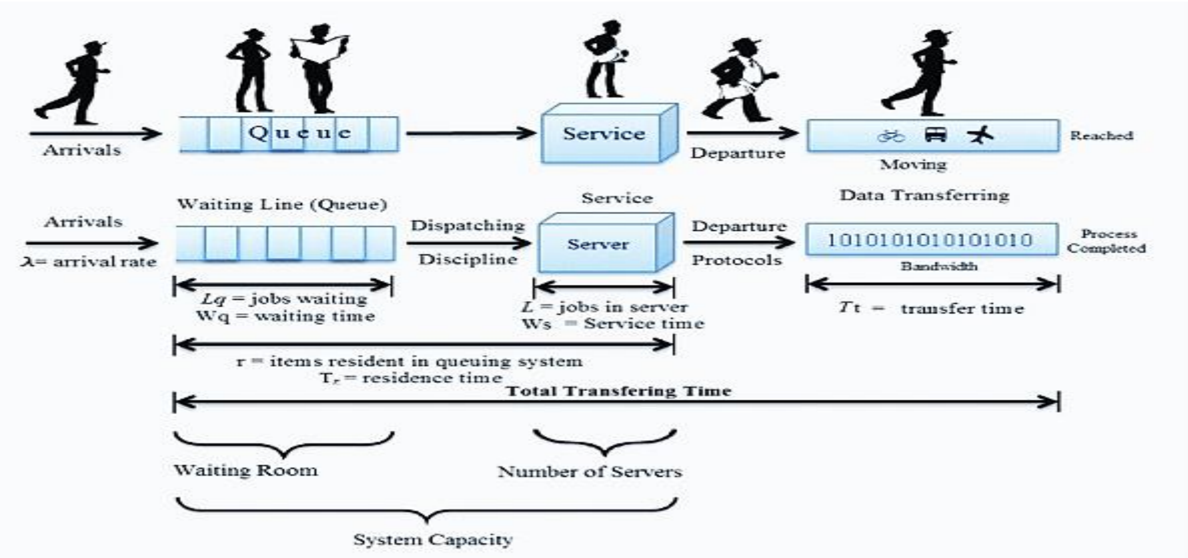
Let us define the busy function as:

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

$\quad\quad\;\; 0$ 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: $\hat{u}(n) = \frac{\int_0^{T(n)} B(t)dt}{T(n)}$

Single Server Queuing System:



Consider a single server queuing system, where the inter arrival time A1, A2… are independent and identically distributed (IID) random variables. A customer 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. A customer who arrives and finds the server busy, joins the end of the single queue.

b. After completing service for a customer1, the server chooses the next customer from the queue (if any) in a FIFO manner.

c. The simulation will begin in the empty and idle state i.e. no customers are present and the service is idle.

d. At time 0, the system will begin waiting for the arrival of the 1st customer will occur, after the 1st inter-arrival time A1 rather than at time 0.

e. The simulation will continue until N numbers of customers have completed their delays in a queue.

Unit III: Continuous System - Simulation and Modeling

Continuous System Simulation and System Dynamics:

Continuous System Simulation:

Continuous System Simulation describes systematically and methodically how mathematical models of dynamic systems, usually described by sets of either ordinary or partial differential equations possibly coupled with algebraic equations, can be simulated on a digital computer.

Modern modelling and simulation environments relieve the occasional user from having to understand how simulation really works. Once a mathematical model of a process has been formulated, the modelling and simulation environment compiles and simulates the model and curves of result trajectories appear magically on the user's screen.

Yet, magic has a tendency to fail, and it is then that the user must understand what went wrong, and why the model could not be simulated as expected.

Classification of Continuous Simulation Languages

1. Block Oriented Simulation Languages:

Block oriented simulation languages are based on the methodology of analogue computers. The system must be expressed as a block diagram that defines the interconnection of functional units and their quantitative parameters.

"Programming" means entering the interconnection of the blocks and their description. Then the user adds statements and/or directives that control the simulation. If the system is described as a set of equations, they must be converted to a block diagram.

This conversion is a simple straightforward process. The typical blocks available in most continuous block oriented languages are integrators, limiters, delays, multipliers, hysteresis, constant values, adders, holders, gain (coefficient) and other.

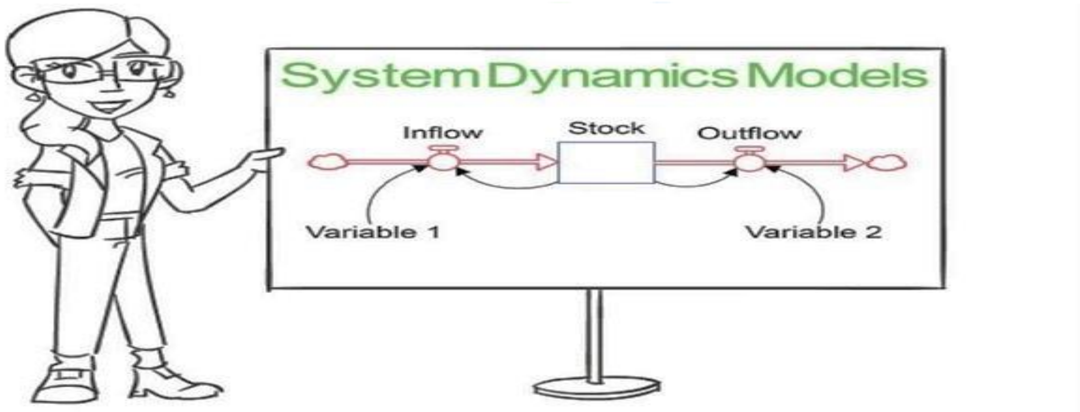
2. Expression Oriented Continuous Languages:

Expression oriented continuous languages are based on writing expressions (equations) that represent the mathematical model. So the system simulated must be expressed by a set of equations. Then the user adds statements and/or directives that control the simulation. Some languages enable both block and expression based ways of system definition.

Simulation control means selection of: the integration method (because some languages offer more), the integration step, the variables (outputs of blocks) that should be observed, the intervals for collecting data for printing and/or plotting, scaling of outputs (that may be also done automatically), duration of the simulation runs, number of repetitions and the way certain values are changed in them, etc.

System Dynamics:

System Dynamics is a computer-aided approach to policy analysis and design. It applies to dynamic problems arising in complex social, managerial, economic, or ecological systems, literally any dynamic systems characterized by interdependence, mutual interaction, information feedback, and circular causality.



System Dynamics Approach Involves:

a. Defining problems dynamically, in terms of graphs over time.
b. Striving for an endogenous, behavioral view of the significant dynamics of a system, a focus inward on the characteristics of a system that themselves generate or exacerbate the perceived problem.

c. Thinking of all concepts in the real system as continuous quantities interconnected in loops of information feedback and circular causality.

d. Identifying independent stocks or accumulations (levels) in the system and their inflows and outflows (rates).

e. Formulating a behavioral model capable of reproducing, by itself, the dynamic problem of concern. The model is usually a computer simulation model expressed in nonlinear equations, but is occasionally left unquantified as a diagram capturing the stock-and-flow/causal feedback structure of the system.

f. Deriving understandings and applicable policy insights from the resulting model.

g. Implementing changes resulting from model-based understandings and insights.

Continuous System Models:

Continuous system simulation is one, in which predominant activities of the system cause smooth changes in the attributes of the system entities. When such a system is modelled mathematically, the variable of the model representing the attributes are controlled by continuous functions.

In general, in continuous, the relationship describes the rate at which attributes changes, so that the model consists of differential equations.

If a system can be represented using simple differential equation model, then it is often possible to solve the model without the use of simulation, otherwise we use simulation to solve those models which are complex to solve analytically.

Differential Equations:

A differential equation is a mathematical equation that relates some function with its derivatives where the functions usually represent physical quantities, the derivatives represent their rates of change, and the equation defines a relationship between the two.

Because of such relations are extremely common, differential equations play a prominent role in many disciplines including engineering, physics, economics, and biology. We can use the differential equation to represent the behavior of a continuous system.

An example of a linear differential equation with constant coefficient is one that describes the wheel suspension of an automobile.

The equation is: $\mathbf{M\ddot{x}+D\dot{x}+Kx=KF(t)}$

Where,

$\ddot{x}$ = acceleration
$\dot{x}$ = velocity
x = displacement
K = stiffness of spring
D = measure of viscosity (thickness) of shock absorber
F(t) = input of system depends on independent variable t

When more than one independent variable occurs in a differential equation, the equation is said to be a partial differential equation. It can involve the derivatives of the same dependent variable with respect to each of the independent variables.

An example is an equation describing the flow of heat in a three-dimensional body. There are four independent variables, representing the three dimensions and time, and one dependent variable, representing temperature.

Differential equation occurs repeatedly in scientific and engineering studies. The reason for this prominence is that most physical and chemical process involves rates of change, which require differential equations for their mathematical description.

Since a differential coefficient can also represent a growth rate, continuous models can also be applied to a problem of a social or economic nature where there is a need to understand the general effect of growth trends.

Ordinary Differential Equations:

An ordinary differential equation (*ODE*) is an equation containing an unknown function of one real or complex variable *x*, its derivatives, and some given functions of *x*. The unknown function is generally represented by a variable (often denoted *y*), which, therefore, *depends* on *x*.

Thus *x* is often called the independent variable of the equation. The term "*ordinary*" is used in contrast with the term partial differential equation which may be with respect to *more than* one independent variable.

Linear differential equations are the differential equations that are linear in the unknown function and its derivatives. Their theory is well developed, and, in many cases, one may express their solutions in terms of integrals.

Most ODE that is encountered in physics are linear, and, therefore, most special functions may be defined as solutions of linear differential equations.

Partial Differential Equation (PDE):

Partial Differential Equation is a differential equation that contains unknown multivariable functions and their partial derivatives. (This is in contrast to ordinary differential equations, which deal with functions of a single variable and their derivatives.)

PDEs are used to formulate problems involving functions of several variables, and are either solved in closed form or used to create a relevant computer model. PDEs can be used to describe a wide variety of phenomena in nature such as sound, heat, electrostatics, electrodynamics, fluid flow, elasticity, or quantum mechanics.

These seemingly distinct physical phenomena can be formalized similarly in terms of PDEs. Just as ordinary differential equations often model one-dimensional dynamical systems, partial differential equations often model multidimensional systems. PDEs find their generalization in stochastic partial differential equations.

Non-Linear Differential Equations:

Non-linear differential equations are formed by the *products of the unknown function and its derivatives* are allowed and its degree is > 1. Nonlinear differential equations can exhibit very complicated behavior over extended time intervals.

Linear Differential Equations:

A linear differential equation with constant coefficients is always of this form, although derivatives of any order may other forms, such as being raised to a power, or are combined in any way- for example, by being multiplied together, the differential equation is said to be non-linear.

Analog Computers:

Analog computers are generally used to solve continuous model but sometimes are also used to solve static models. Some device whose behavior is equivalent to a mathematical operation such as addition or integration is combined together in a manner specified by a mathematical model of a system to allow the system to be simulated.

That combination is used in the simulation of a continuous system is referred to as an analogue computer or when they are used to solve differential equation they are referred to as differential analyzer.

Simulation with an analog computer is more properly described as being based on a mathematical model than as being a physical model. The most widely used form of analog computers is the electronics analog computers based on the operational amplifiers.

Voltages in the computers are equated to mathematical variables and the operational amplifiers can add and integrate the voltage. With appropriate circuits, an amplifier can be made to add several input voltages, each representing a variable of model, to produce a voltage representing the sum of the input variables.

Different scale factors can be used on the input to represent the coefficient of the model equations. Such amplifiers are called summer. Another circuit arrangement produces an integrator for which the output is the integral with respect to time of single input voltage or the sum of several input voltages.

All voltages can be positive or negative to correspond to the sign of the variable represented. To satisfy the equation of the model, it is sometimes necessary to use a sing inverter.

Advantages:

a. Parallel operation: many signal values can be computed simultaneously.

b. Computation can be done for some applications without the requirement for transducers to convert the inputs/outputs to/from digital electronic form.

c. Setup requires the programmer to scale the problem for the dynamic range of the computer. This can give insight into the problem and the effects of various errors.

Disadvantages:

a. Computation elements have a limited useful dynamic range, usually not much more than 120 dB, about 6 significant digits of accuracy.

b. Useful solution to problems of any size can take an inordinate amount of setup time (though modern analog computers have interfaces that make setup substantially easier than it used to be).

c. For a given size (mass) and power consumption, digital computers can solve larger problems.

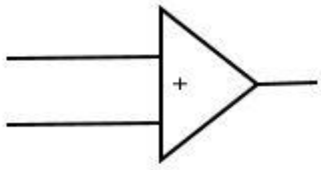
d. Solutions appear in real (or scaled) time, and maybe difficult to record for later use or analysis.

e. The range of useful time constants is limited. Problems that have components operating on vastly different time scales are difficult to deal with accurately.

Components of Analog Computer:

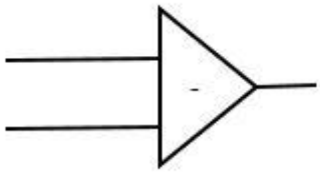
1. Adder:

With an appropriate circuit, an amplifier made to add several input voltage each representing the variable of the model to produce a voltage each representing a sum of the input voltage.



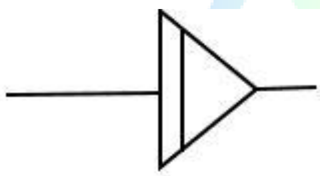
2. Subtractor:

With an appropriate circuit, an amplifier made to subtract several input voltage each representing the variable of the model to produce the voltage each representing a difference of input voltage.



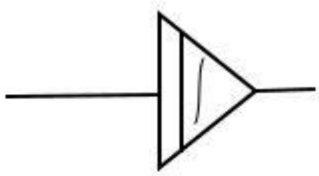
3. Differentiator:

An op-amp differentiator or a differentiating the amplifier is a circuit configuration which produces output voltage amplitude that is proportional to the rate of change of the applied input voltage.



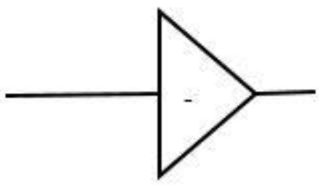
4. Integrator:

The circuit arrangement for which the output is integral with respect to time of single input voltage or the sum of several input voltage.



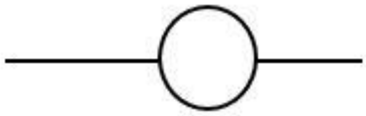
5. Invertor:

It is an amplifier designed to cause the output to reverse the sign of the input.



6. Scale Factor:

This circuit multiplies each input by a factor (the factor is determined by circuit design) and then adds these values together. The factor that is used to multiply each input is determined by the ratio of the feedback resistor to the input resistor.



Analog Method:

The general methods by which analog computer are applied can be demonstrated using the second-order differential equation given as: $M\ddot{x}+D\dot{x}+Kx=KF(t)$

Solving the equation for the highest order derivate gives: $M\ddot{x}=KF(t)-D\dot{x}-Kx$

Suppose a variable representing the input F(t) is supplied, and assume for the time being that there exist variables representing **-x** and $-\dot{x}$. These three variables can be scaled and added with a summer to produce a voltage representing $M\ddot{x}$.

Integrating this variable with a scale factor of **1/M** produce $\dot{(x)}$. Changing the sign produce $\dot{x}$, which supplies one of the variables initially assumed; and further integration produces **-x**, which was other assumed variables. For convenience, a further sign inverter is included to produce **+x** as an output.

A block diagram to solve the problem in this manner is shown below. The symbols used in the figure are standard symbols for drawing block diagrams representing analog computer arrangements.

The circle indicates scale factors applied to the variable. The triangular symbol at the left of the figure represents the operating of the adding variables. The triangular symbol with a vertical bar represents integration, and the containing a minus sign is a sign changer.

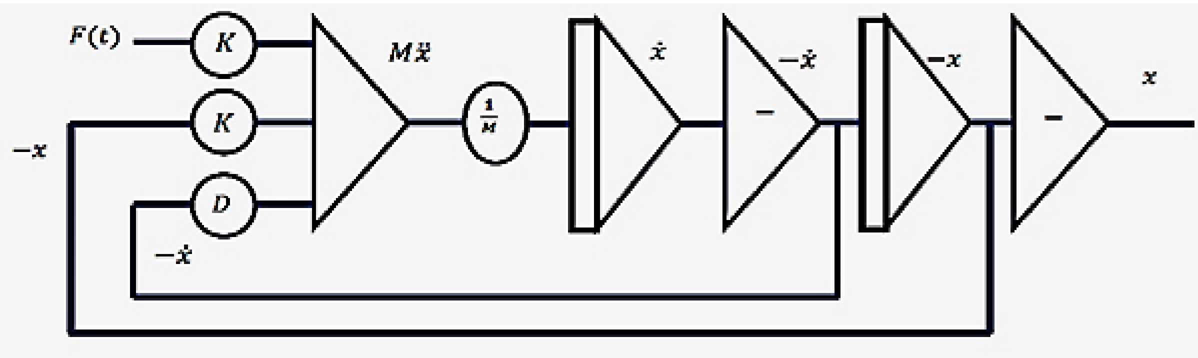


Fig: analog model of automobile suspension problem

Draw The Analog Model Of The Liver With Following Set Of Equations:

(dx_1)/dt=-k12x_1+k21x_2
(dx_2)/dt=k12x_1-(k21+k23) x_2
(dx_3)/dt=k23x_2

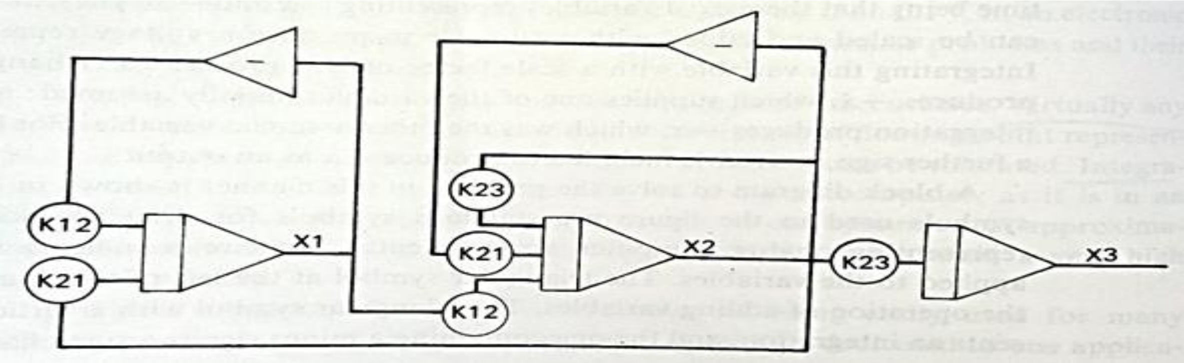


Fig: Analog Model of the Liver

Hybrid Computers:

The term hybrid computers have emerged to describe the combination of traditional analog computer elements (that gives the smooth continuous output and carry out non-linear operations) as well as the circuit components that have the capacity of storing values, switching operations and performing a logical operation.

The scope of analog computers has been considerably extended by developments of a solid-logic electronic device. Hybrid computer may be used to simulate a system that is mainly continuous and also have some digital elements.

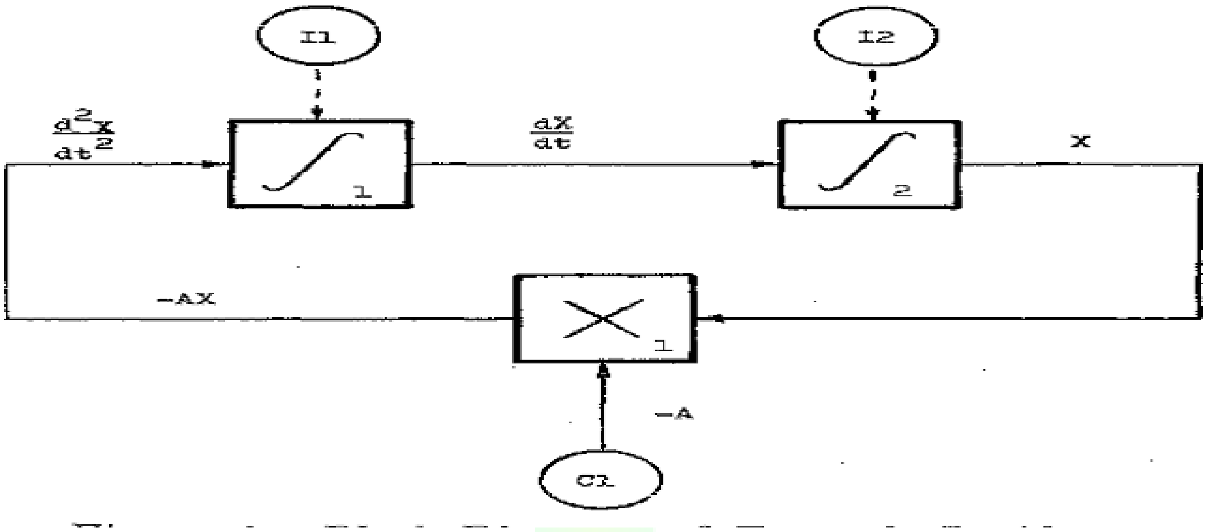
For e.g. an artificial satellite for which both the continuous equation of motion and the digital controls signal must be simulated. A hybrid computer is useful when a system that can be adequately represented by an analog computer model is subject of a repetitive study.

Digital Analog Simulators:

To avoid the disadvantage of analog computers, many digital computer programming languages have written to produce digital-analogue simulator. These allow a continuous model to be programmed on a digital computer in the same way as it is solved on an analog computer.

These languages contain macro instructions that carry out the action of address, integrators and sign chargers. A program uses these macro-instructions to link them together in essentially the same way as operational amplifiers are connected in analog computers.

Later more powerful techniques of applying digital computers to the simulation of the continuous system have been developed. Due to these digital-analogue simulators are not now in common uses.



Continuous System Simulation Language:

It is a restriction to keep the digital computer within the limit to a routine that represents as it is done with a digital-analog simulator. To remove the restriction a number of continuous system simulation language have been developed.

They use familiar statement type of input for a digital computer, allowing a problem to be programmed directly from the equation of mathematical the model rather than requiring the equation to be broken down in functional elements.

A CSSL include macros or subroutines that forms the function of specific analog elements so that it is possible to incorporate the convenience of an analog simulator. To allow the users to define special-purpose elements that correspond to an operation that are particularly important in a specific type of application.

It includes a variety of algebraic and logical expression to describe the relation between variable. Therefore, they remove the orientation towards linear differential equation which characterizes analog computer. One particular CSSL that illustrates the nature of these languages is the Continuous System Modeling Program.

CSMP III (Continuous System Modeling Programming III):

A CSMP III program is constructed from three general types of statements:

Structural Statement:

It defines the model. They consist of FORTRAN like statement and functional block designed for an operation that frequently occurs in a model definition. It can make use of the operation of addition, subtraction, multiplication, division, and exponential using the same notation and rules used in FORTRAN.

Data Statement:

It assigns numeric values to the parameter constant and initial condition.

Control Statement:

It specifies the option in the assembly and execution of program and the choice of output.

For example, the model includes the equation: **X = 6Y/W+(Z-2)^2**

The following statement would be used: **X = 6.0*Y/W+(Z-2.0)**2.0**

Note that real constants are specified in decimal notation. Exponent notation may also be used; for example, 1.2E-4 represents 0.00012. Fixed value constants may also be declared. Variable names may have up to six characters.

Write A CSMP Program Of Following Differential Equation.

$M\ddot{x}+D\dot{x}+Kx=KF(t)$

Here,

Structural Statement

$M\ddot{x}=KF(t)-D\dot{x}-Kx$
$\ddot{x}=(KF(t)-D\dot{x}-Kx)/M$
x_2 dot= 1.0/M [KF(t)-D$\dot{x}$-Kx]
x_2 dot= 1.0/M [K*F(t)-D*$\dot{x}$-K*x]

x_2 dot=(1.0/M)*[K*F(t)-D*ẋ-K*x]
x_2 dot=INTGRL (0.0,x_2 dot)
x=INTGRL(0.0,x_1 dot)

Data Statement

M = 3.0
F(t) = 1.0
K = 4.0

Control Statement

DELT (Integral Interval) = 0.05
FINTIME (Finish Time) = 1.5
PRDEL (Integral at which to print result) = 2

Hybrid Statement:

The system to be studied is either continuous or discrete and we have to select the analog or digital computer for the study of the system. There are many advantages and disadvantages of analog and digital computer.

To achieve the advantages of both (analog and digital computer) we can combine both analog/digital computer system into a single form and simulate the system through it.

In this case, one computer is simulating the system being studied while others providing the simulation of the environment in which the system is to operate. The hybrid simulation requires some extra technical improvement, high-speed converters are used to convert the signal from one form to another.

Feedback System:

Feedback systems have a closed-loop structure that brings results from past action of the system back to control future action, so feedback systems are influenced by their own past behavior.

Extending the blind control example, a feedback system would be a system that not only opens the blinds when the sun rises but also adjusts the blinds during the day to ensure the room is not subjected to direct sunlight.

Even though the open system can consist of many parts and thus become very complex (these systems have high detail complexity), experience shows that the behavior of even small feedback systems consisting of only a few parts (and thus low detail complexity) can be very difficult to predict in practice: despite low detail complexity, these systems have high dynamic complexity.

In business prototyping, we deal with both kinds of systems; system dynamics is particularly good at capturing the dynamics of feedback systems.

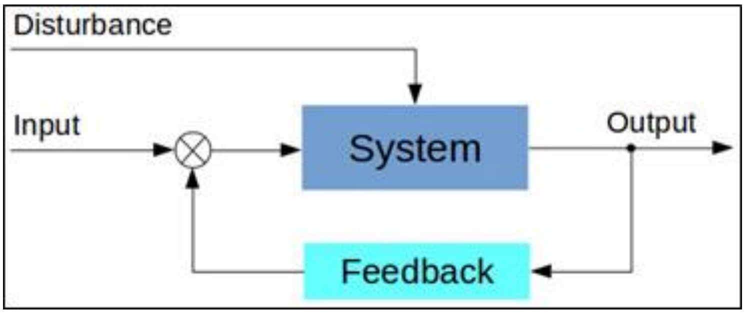


Fig: Feedback System

Interactive Systems:

Interactive systems are computer systems characterized by significant amounts of interaction between humans and the computer. Most users have grown up using Macintosh or Windows computer operating systems, which are prime examples of graphical interactive systems.

Editors, CAD-CAM (Computer Aided Design-Computer Aided Manufacture) systems and data entry systems are all computer systems involving a high degree of human-computer interaction. Games and simulations are interactive systems.

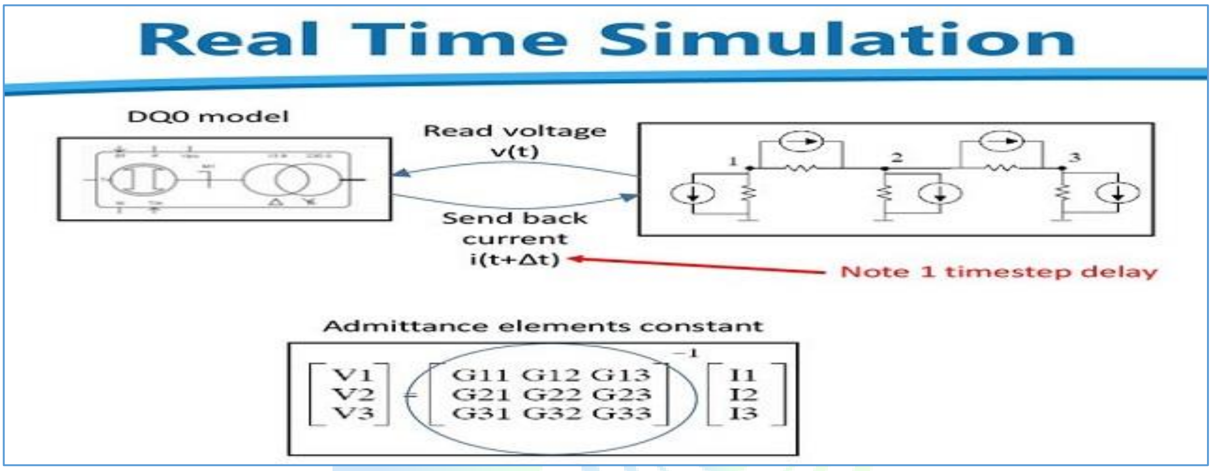
Web browsers and Integrated Development Environments (IDEs) are also examples of very complex interactive systems. Some estimates suggest that as much as 90 percent of computer technology development effort is now devoted to enhancements and innovations in interface and interaction.

To improve efficiency and effectiveness of computer software, programmers, and designers not only need a good knowledge of programming languages, but a better understanding of human information processing capabilities as well.

They need to know how people perceive screen colors, why and how to construct unambiguous icons, what common patterns or errors occur on the part of users, and how user effectiveness is related to the various mental models of systems people possess.

Real-Time Simulation:

Real-time simulation refers to a computer model of a physical system that can execute at the same rate as the actual "wall clock" time. In other words, the computer model runs at the same rate as the actual physical system. For example, if a tank takes 10 minutes to fill in the real-world, the simulation would take 10 minutes as well.



Real-time simulation occurs commonly in computer gaming, but also is important in the industrial market for operator training and off-line controller tuning. Computer languages like LabVIEW, VisSim, and Simulink allow quick creation of such real-time simulations and have connections to industrial displays and Programmable Logic Controllers via OLE for process control or digital and analog I/O cards.

Several real-time simulators are available on the market like xPC Target and RT-LAB for mechatronic systems and using Simulink, eFPGAsim and eDRIVEsim for power electronic simulation and eMEGAsim, HYPERSIM and RTDS for power grid real-time (RTS) simulation.

Predator-Prey Model:

It is also called the parasite-host model. An environment consists of two population i.e. predator and prey. It is also a mathematical model. The prey is passive but the predator depends on the prey for their source of food.

Let,

x(t) = number of prey population at time t.
y(t) = number of predator population at time t.
r.x(t) = rate of growth of prey for some +ve 'r', where r = natural birth and death rate.

Because of the interaction between predator and prey, it will be reasonable to assume that the death rate of prey is proportional to the product of two population size x(t).y(t) or the death rate of prey is a.x(t).y(t). Therefore the overall rate of change of prey population, dx/dt is given by, dx/dt=r.x(t)-a.x(t).y(t).

Where a is positive constant of proportion. Also, the predator population depends on the prey for their existence, the rate of the predator in the absence of prey is –s.y(t) for some positives.

The interaction between two population cause predator population to increase at a rate of proportion x(t).y(t). Thus, overall change predator population dy/dt=-s.y(t)-b.x(t).y(t). Where, b is positive constant.

As the predator population increases the prey population decreases. This cause a decrease in the rate of an increased predator, which eventually results in a decrease in the number of predators. These in turns cause the number of prey population to increase.

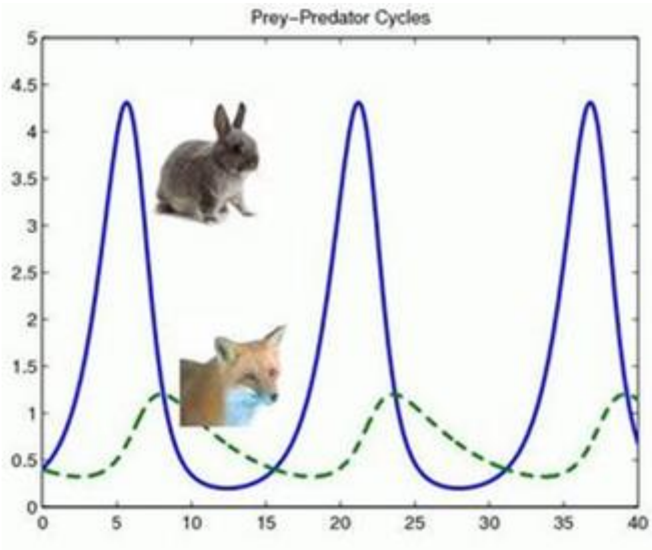


Fig: Prey-Predator Cycles

Unit IV: Discrete System Simulation - Simulation and Modeling

Introduction to Discrete System Simulation:

In discrete systems, the changes in the system state are discontinuous and each change in the state of the system is called an **event**. For example, the arrival or departure of a customer in a queue is an event. Similarly, sales of an item from stock is an event in the inventory system.

Therefore, the discrete system is often referred to as a discrete system simulation. The model used in a discrete system simulation has a set of numbers to represent the state of the system, called as a **state descriptor**.

Representation of Time:

It refers to a number that records 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, 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 extent depending on the following factors:

a. Nature of the system being simulated.
b. The detail to which it is modelled.

Example: The simulation of an atomic model. Changes occur in a fraction of a microsecond in a 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:

1. Event Oriented Method:

In these methods, 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.

2. Interval Oriented Method:

In these methods, the clock time is advanced by small and usually, a 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's (visit) rule can be made about the way the time is represented in simulation for discrete and continuous system because:

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

Significant Event Simulation:

It is another method to represent the passage of time and applies to a continuous system in which it is quiescent (continuous) period. A quiescent period is an interval between events in the event-oriented approach but it involves the model's representation of the system activities which create a notice of the event that terminates the interval.

The significant event approach assumes that a 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:

a. It might reach at the end of the road.
b. Its velocity most reach some limit.
c. It might come to rest.

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

General Arrival Pattern:

The generation of exogenous arrival is an important aspect of discrete system simulation. An exact sequence of arrivals may have been specified for the simulation. For example, in the simulation of the 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 preferred 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 several telephones (here only 8 are shown), connected to a switchboard by line. The switchboard has several lines provided the condition that only one connection at a 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:

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

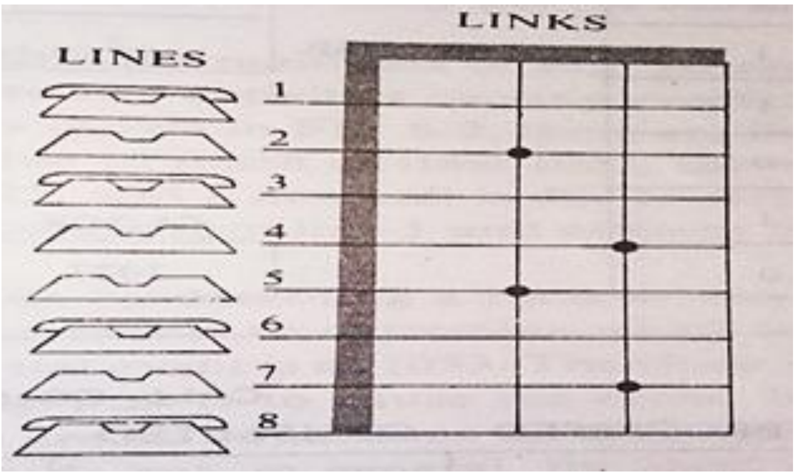


Figure 1: Simple Telephone System

Object of Simulation:

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

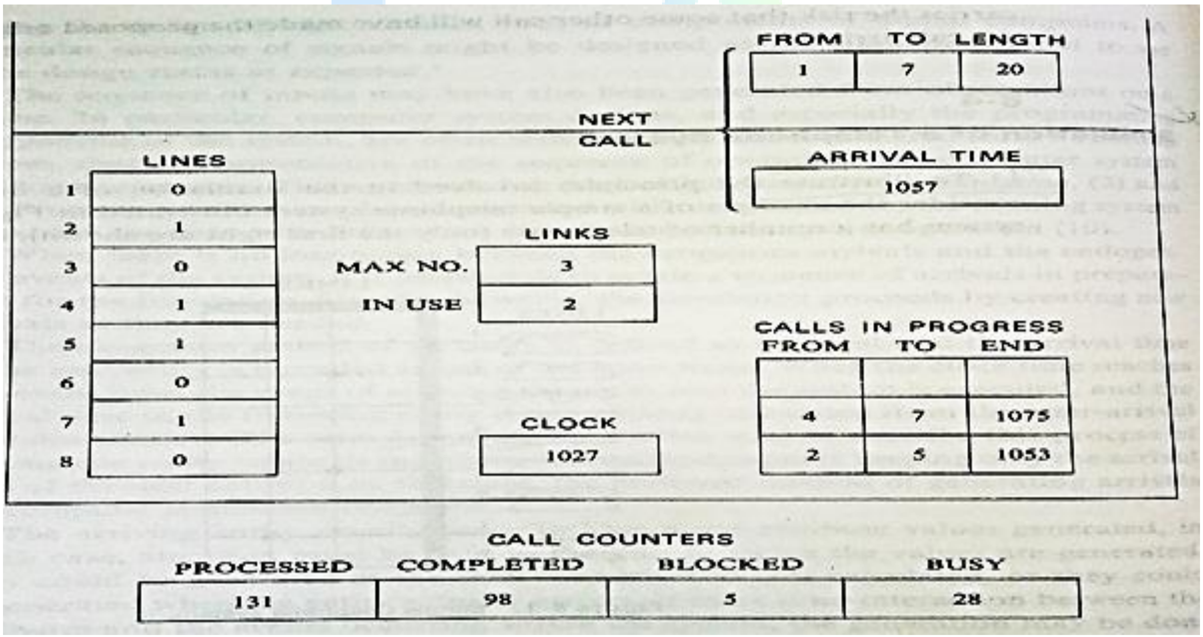


Figure 2

Here, line 2 is connected to line 5 and line 4 is connected to line 7. 0 in the line tables means is free and 1 means, it is busy. The maximum number of links, in this case, is only 3 and 2 of them are in use.

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 the event as the simulation proceeds.

Each column has its own attributes; its origin, destination, length and the time at which it calls finishes.

The "call-in progress" tables show the lines that are currently connected and 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 figure 2, it can be seen that three events can occur in the figure:

a. Call between 2 & 5 will finish at 1053
b. A new call will arrive at 1057
c. Call between 4 & 7 will finish at 1075

The Simulation Proceeds by Executing a Cycle below State to Simulate Each Event:

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

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

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

4. 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 is used by 1 and removing the finish call from the "call-in progress" table.

5. Gather some statistics for the simulation output. Here call counter is 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.

6. The above steps are repeated to continue the simulation.

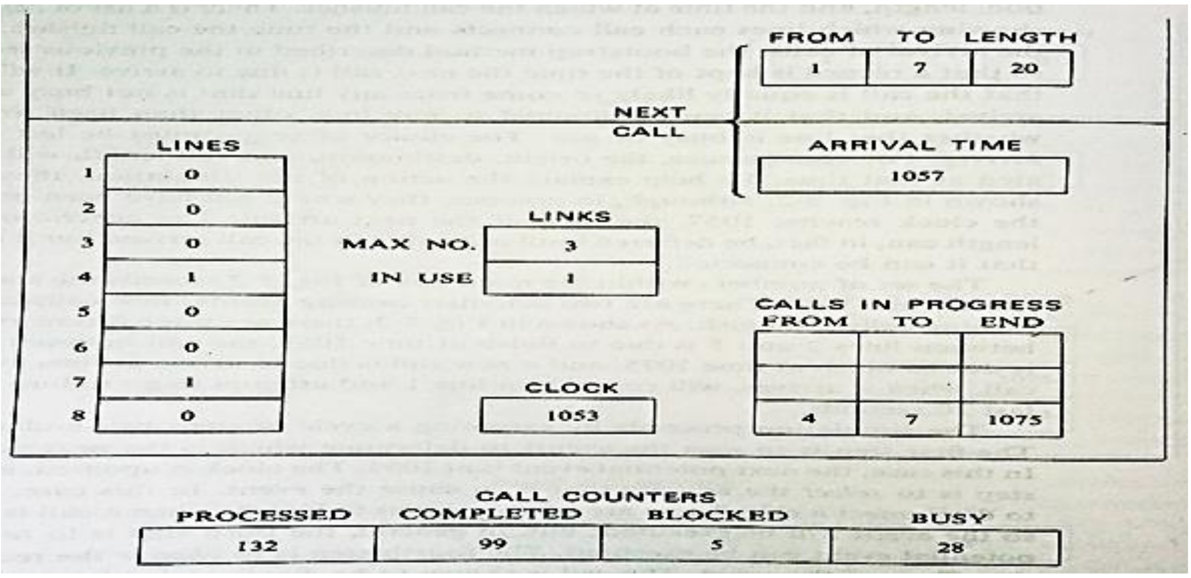


Figure 3

In the 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 the new arrival are 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, line 7 is busy so the call is lost. Both the processed call and the busy call counter are increased by one. New arrival is generated. These are shown in figure 4.

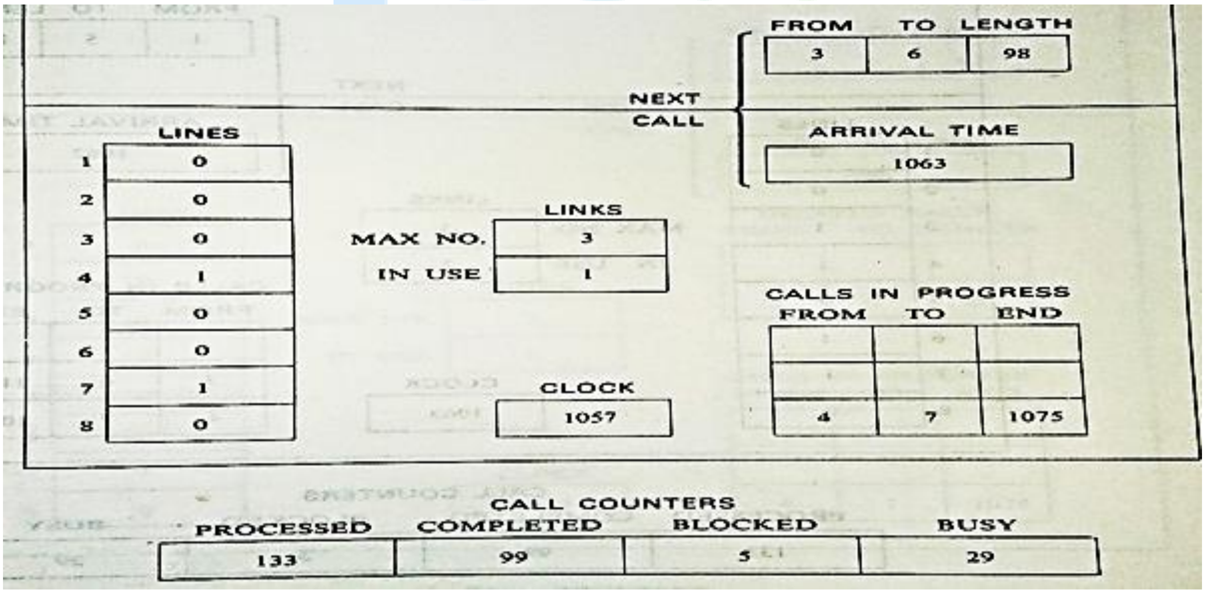


Figure 4

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 good statistics can be gathered.

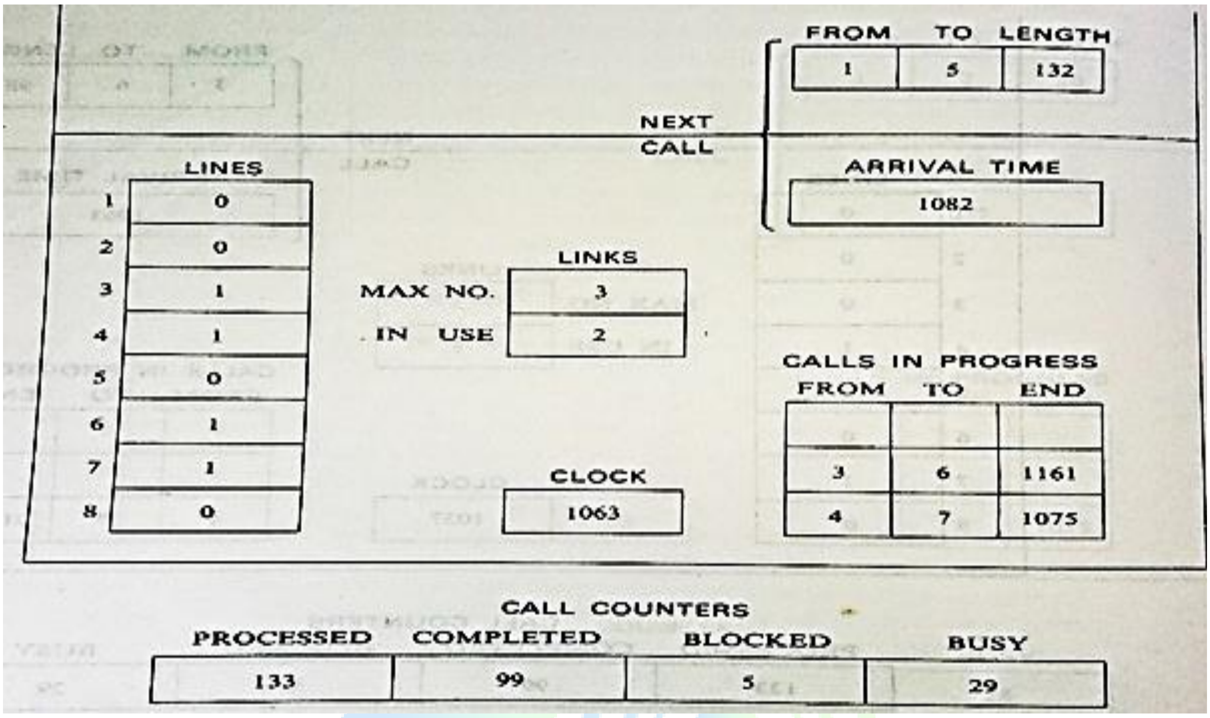


Figure 5

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 referred to as a 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 a store and forward capability.

To keep a record of the 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 figures 6, 7, and 8.

From figure 5, it is clear that the next potential event is the arrival of the 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 the below figure.

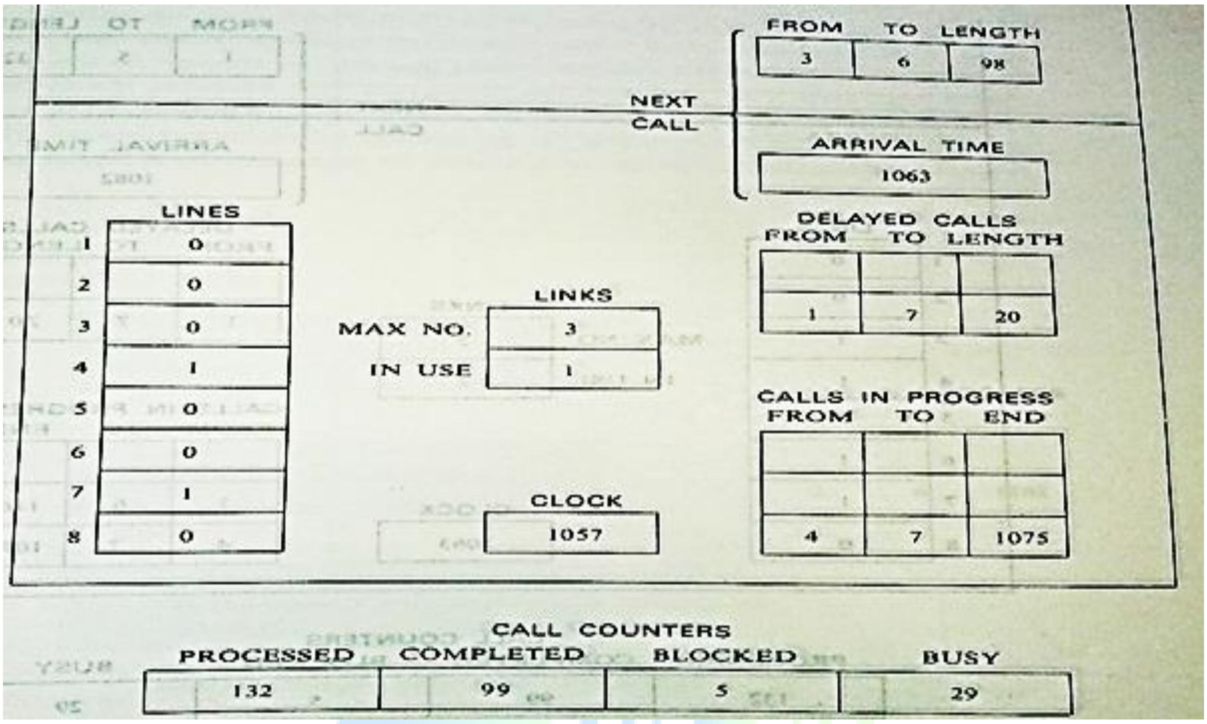


Figure 6

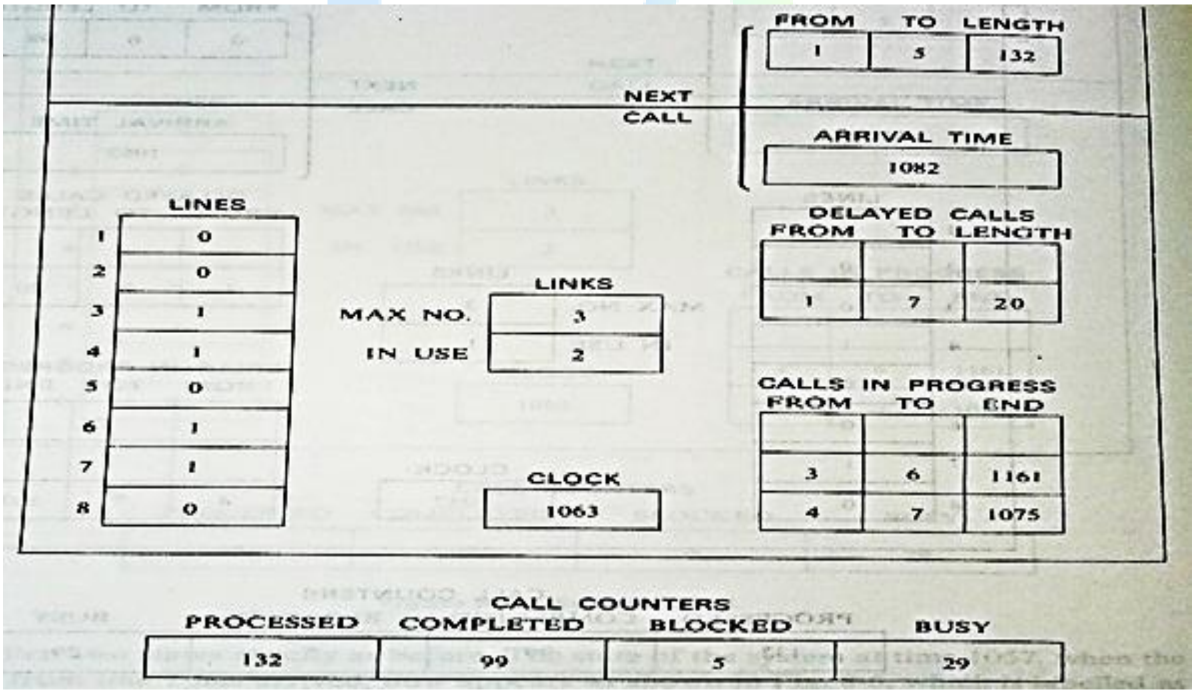


Figure 7

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

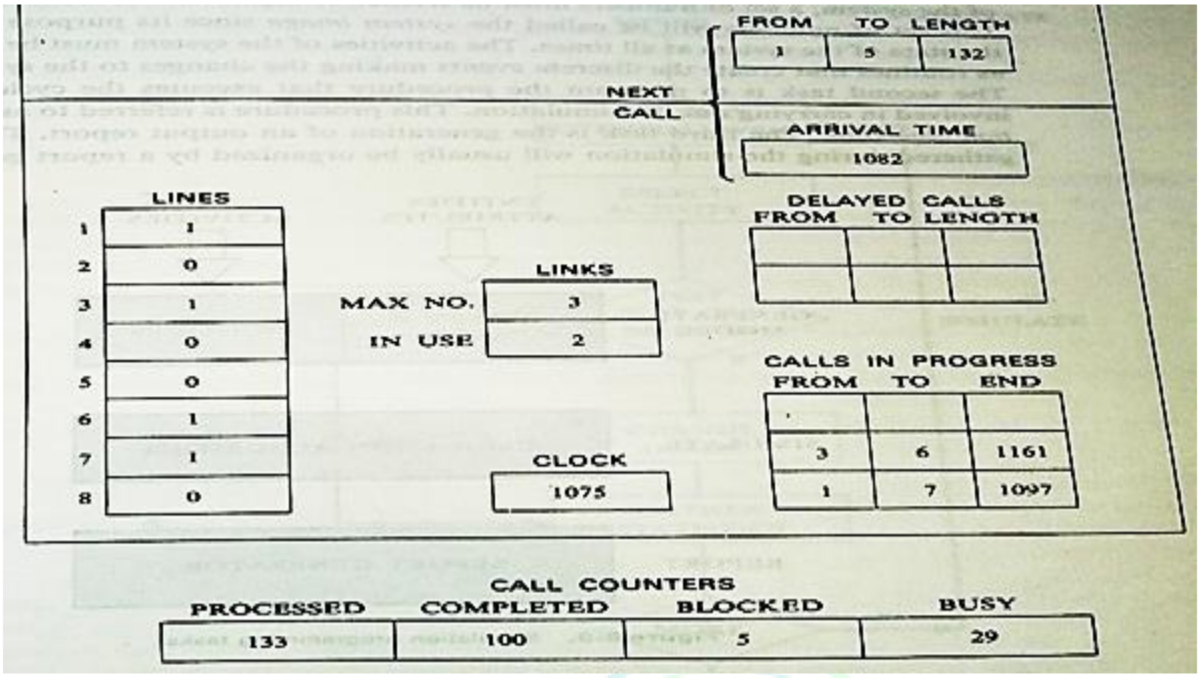


Figure 8

Simulation Programming Task:

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

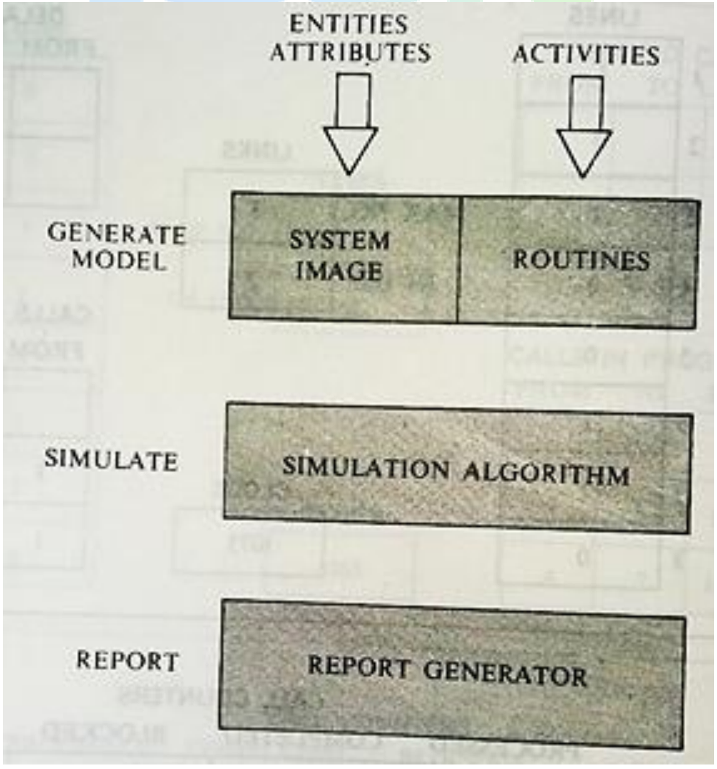


Fig: Simulation Programming Tasks

1. Generating a Model:

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

2. Simulation:

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

3. Generating Report:

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

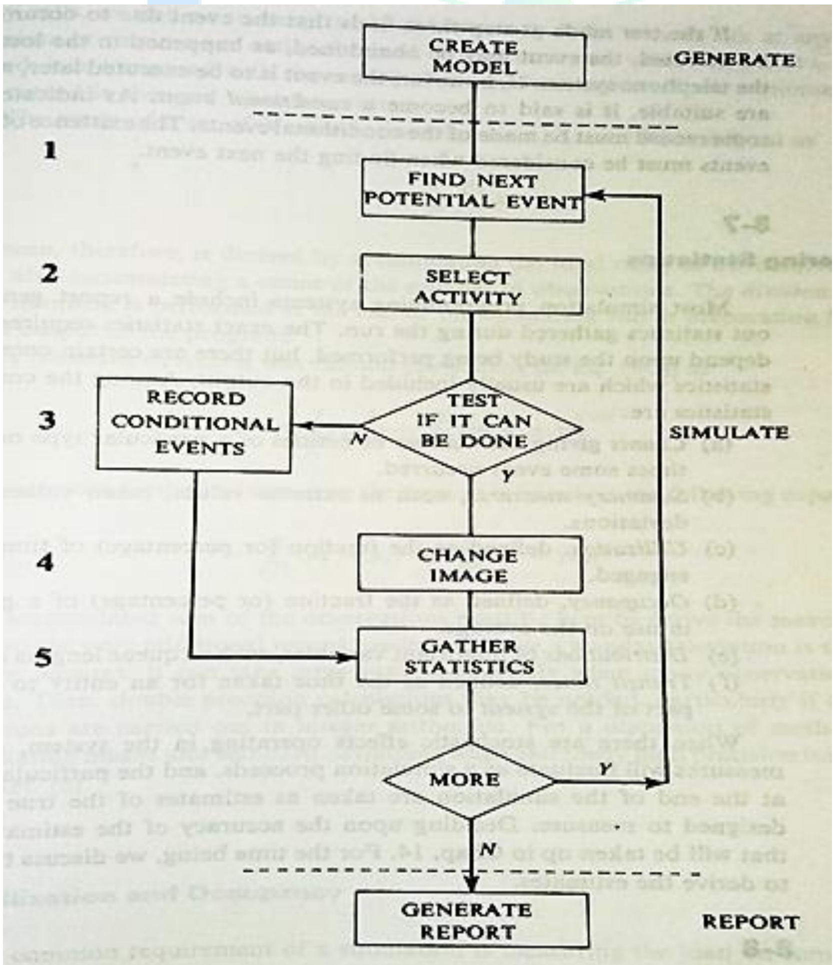


Fig: Execution of a Simulation Algorithm

The general flow of control during the execution of the simulation program is shown above. At the top of the figure is the task of generating a model which is executed once. At the bottom is the report generation task, which is usually executed once at the end of the simulation.

Carrying out the simulation algorithm involves repeated execution of five steps. These steps are:

1. Find the next potential event.
2. Select an activity.
3. Test if the event can be executed.
4. Change the system image.
5. Gather statistics.

Gathering Statistics:

To print out the statistics gathered during the run, the simulation programming system includes a report generator. The exact statistics required from a model depend upon the study being performed. Some of the commonly required statistics which is usually included in the output during the simulation are 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 infraction 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 other part.

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

Counter and Summary Statistics:

The counter which is the basis 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 the telephone system, the counter is used to record the total number of the lost call, busy call, and to keep track of how many links 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, $M = \frac{1}{N}\sum_{i=1}^{N} Xi$

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 the 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 the time, the item as engaged during the simulation. Measuring those statistics is referred to as the utilization of that equipment. The time history of an equipment usage might appear as shown in the figure below:

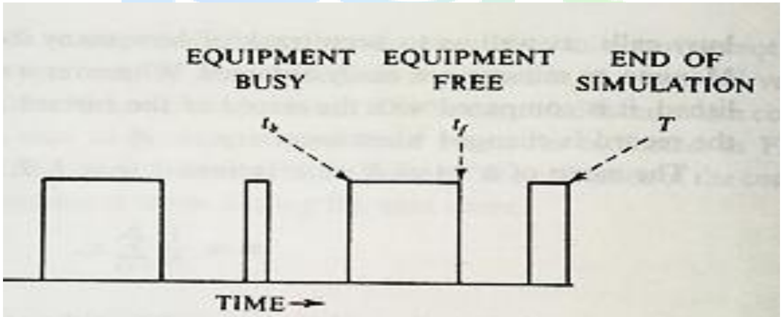


Fig: Utilization of Equipment

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

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

T =total time in the 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 a group of entities, rather than individual items, the calculation is similar, requiring that information about the number of entities involved also kept.

The figure below represents as a function of time and the number of links in the telephone system that are busy. To find an 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 ti to the value ni, then, at the time of the next change ti+1, the quantity ni(ti+1-ti) 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 several entities waiting on queues.

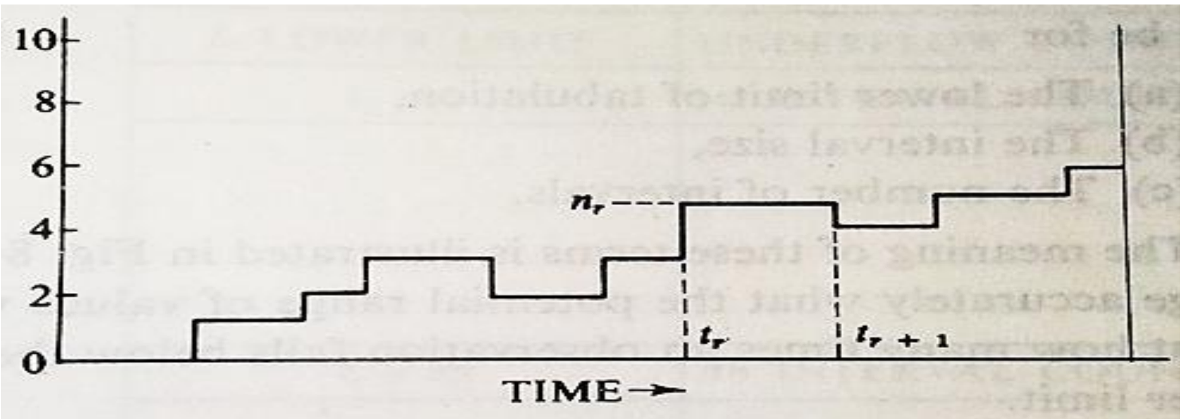


Fig: Time History of Busy Telephone Links

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

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

If M= links in a telephone

ni = number of busy in the interval ti and ti+1 then the average occupancy, assuming the number ni 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 the lower limit of the tabulation, the interval size and the number of intervals. Normally, the tabulation interval sizes are uniform.

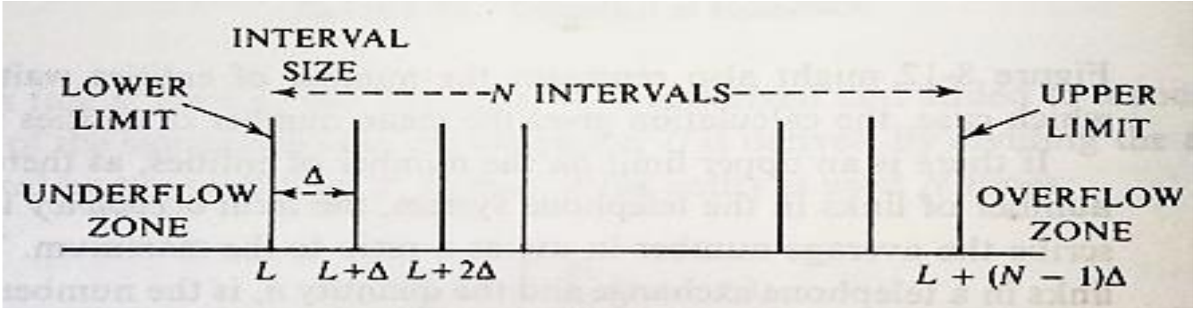


Fig: Definition of a Distribution Table

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 The Figure:

$i = 1$	L, LOWER LIMIT	UNDERFLOW COUNT
	$L + \Delta$	1st INTERVAL COUNT
	$L + i\Delta$	ith INTERVAL COUNT
$i = N - 1$	$L + (N - 1)\Delta$	$(N - 1)$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 the random variable being measured. To understand this, consider the following two examples:

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

2. 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 another commercial form as required. For example, the cumulative distribution may be given or the distribution may be resolved to express the counts as a percentage of total observation. These results are calculated at the end of the simulation.

A clock is used in the manner of the time stamp to measure transit time. When an entity reaches a point from which a measurement 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 Languages:

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

1. GPSS
2. SIMSCRIPT

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

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

Also, a good simulation 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 modelling)
b. Moving a system from state to state (Dynamic modelling), &
c. Performing relevant task such as the random number of generation, data analysis and report generation.

Classification of Discrete System Simulation Languages:

1. Flowchart Oriented Languages:

Flowchart Oriented Languages are represented by the language GPSS (General Purpose Simulation System), that exists in many versions on various computers. The user must view the dynamics of the system as a flow of the so called *Transactions* through a block diagram.

Transactions are generated, follow a path through a network of blocks, and are destroyed on exit. In blocks transactions may be delayed, processed, and passed to other blocks. Blocks are in the program represented by statements that perform the *Activities* of the model.

2. Activity Oriented Languages:

Activity oriented languages are not based on explicit scheduling of future activities. For each activity the user describes the condition under which the activity can take place (that also covers scheduling if the condition is reaching certain time).

The algorithm of the simulation control repeatedly increments time and tests conditions of all activities. The disadvantage of this approach is obvious - it is necessary to evaluate all conditions in every step that may be very time consuming.

On the other hand it is conceptually very simple and the algorithm can be easily implemented in general high level languages (there are simulation languages based on this approach, but not widely used).

3. Event Oriented Languages:

Event oriented languages are based on direct scheduling and canceling of future events. The approach is very general. The user must view the dynamics of the system simulated as a sequence of relatively independent events.

Every event may schedule and/or cancel another events. The system routine must keep record of scheduled events. That's why every event is represented by the so called Event notice that contains the time, the event type, and other user data.

Event notices are kept in the so called Calendar, where the event notices are ordered by the scheduled time. After completion of an Event routine, the system removes the event notice with the lowest time from the calendar, updates the model time by its time, and starts the corresponding routine.

This is repeated until the calendar becomes empty or the program stops because of other reason. Scheduling means inserting event notices to the calendar by the scheduled time, canceling removes them.

The approach based on explicit expressing of events is called Discrete Event Simulation that is sometimes generalized to discrete simulation as such. A typical representative of this group of languages is the language SIMSCRIPT (but its version II.5 supports also process oriented simulation).

4. Process Oriented Languages:

Process oriented languages are based on the fact, that events are not independent. An event is typically a consequence of other previous events. In other words it is often possible to define sequences of events that may be viewed as entities of a simulation model at higher level of hierarchy.

A sequence of events is called *Process*. Unlike events process has a dimension in time. Process based abstract systems are very close to reality that is always made of various objects that exist and act in parallel interfering with each other.

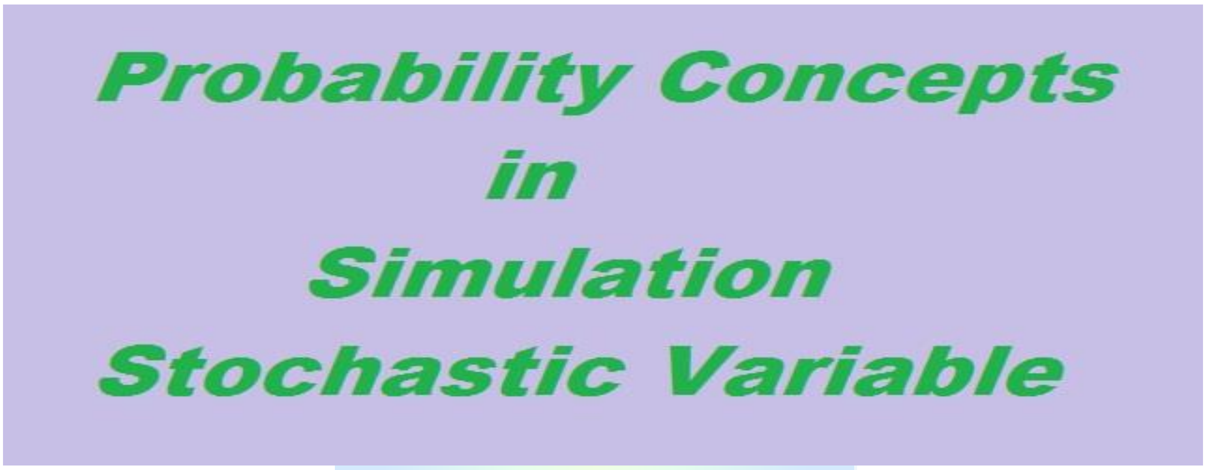
Process way of viewing system dynamics is thus very natural. Mostly a process models an activity of a real object. It is believed, that process oriented discrete simulation is the best way how to create discrete simulation models.

Typical representatives of this group of languages are MODSIM, SIMSCRIPT II.5, and the system class SIMULATION of the Simula language.

Unit V: Probability Concept and Random Number Generation - Simulation and Modeling

Probability Concepts in Simulation-Stochastic Variable:

The description of activities can be of two types deterministic and stochastic. The process on which, the outcome of an activity can be described completely in terms of its input is deterministic and the activity is said to be deterministic activity.



On the other hand, when the outcome of an activity is random, i.e. there can be various possible outcomes, the activity is said to be stochastic activity.

In case of an automatic machine, the output per hour is deterministic, while in a repair shop the number of machines repaired will vary from hour to hour, in a random fashion. The terms random and stochastic are interchangeable.

A random variable x is called discrete if the number of possible values of x (i.e. range space) is finite or countable infinite, i.e. possible values of x maybe x1, x2, …, xn.

A random variable x is called continuous if its range space is an interval or a collective of intervals. A continuous variable can assume value over a continuous range. A stochastic process is described by a probability law called Probability Density Function.

Discrete Probability Function:

If a random variable x can take x_i (i = 1… n) countable infinite number of values with the probability of value x_i being P(x_i) is said to be Probability Distribution or Probability Mass Function of a random variable x.

The number of $P(x_i)$ must satisfy the following two conditions:

1. $P(x_i) >= 0$ for all i
2. $\sum_{i=1}^{n} P(x_i) = 1$

Cumulative Distribution Function:

It is a function which, gives the probability of a random variable being less or equal to a random variable being less or equal to a given value. In a discrete test, the cumulative distribution function is denoted by $P(x_i)$.

This function implies that x takes values less than or equal to x_i.

Continuous Probability Function:

If the random variable is continuous and not limited to discrete values, it will have an infinite number of values in an interval. Such a variable is defined by a function f(x) called a Probability Density Function (pdf).

The probability that a variable x, falls between x and x+dx is expressed as f(x)dx and the probability that x falls in the range x1 to x2 is given as:

$$P(x) = \int_{x1}^{x2} f(x)dx$$

Random Variables:

A random variable is a rule that assigns a number to each outcome of an experiment. These numbers are called values of a random variable. Random variables are usually denoted by X.

Example:

1. If a die is rolled out, the outcome has a value from 1 through 6.
2. If a coin is tossed, the possible outcome is head 'H' or tail 'T'.

Types of Random Variables:

1. Discrete Random Variable:

A discrete random variable takes only specific, isolated numerical values. The variables which take finite numeric values are called as Finite discrete random variables and which takes unlimited values are called as Infinite discrete random variables.

The examples are shown in the table below:

Random Variables	Values	Types
Flip a coin three times; X = the total number of heads	{0, 1, 2, 3}	Discrete Finite There are only four possible values for X.
Select a mutual fund; X = the number of companies in the fund portfolio.	{2, 3, 4, ...}	Discrete Infinite There is no stated upper limit to the size of the portfolio.

Let

$X \rightarrow$ discrete random variable
$R_X \rightarrow$ possible values of X, given by range space of X.
$X_i \rightarrow$ the individual outcome in R_X.

A number $P(x_i) = P(X = x_i)$ gives the probability that the random variable equals the value of x_i. The number $P(x_i)$, I = 1, 2, 3, ... must satisfy two conditions:

1. $P(x_i) >= 0$ for all i
2. $\sum_{i=1}^{\infty} P(x_i) = 1$

The collection of pairs $(x_i, P(x_i))$ i.e. a list of probabilities associated with each of its possible values is called probability distribution of X. $P(x_i)$ is called probability mass function (pmf) of X.

Example:

Consider the experiment of tossing a single die, defining X as the number of spots on up the face of die after a toss.

Solution:

N=total number of observations = 21

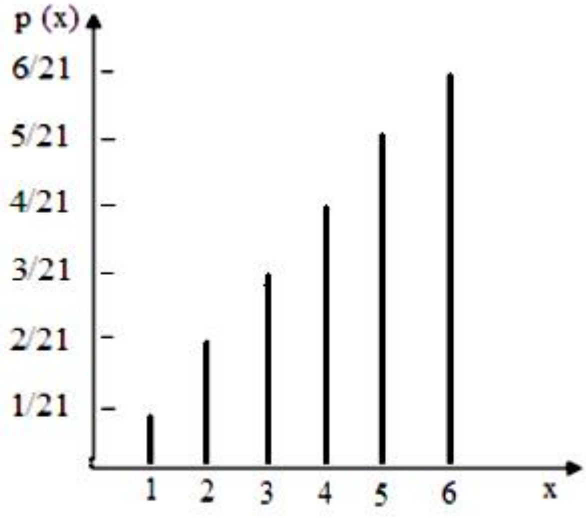
The discrete probability distribution is given by

x_i	1	2	3	4	5	6
$P(x_i)$	1/21	2/21	3/21	4/21	5/21	6/21

The conditions also are satisfied i.e.

1. $P(x_i) >= 0$, for I = 1, 2, 3, ..., 6.
2. $\sum_{i=1}^{\infty} P(x_i) = \frac{1}{21} + \frac{2}{21} + \cdots + \frac{6}{21} = 1$

The distribution is shown graphically in the figure below.



2. Continuous Random Variable:

Continuous Random Variable takes any values within a continuous range or an interval. The example is tabulated in the table below.

Random Variable	Values	Type
Measure the length of an object; X = its length in centimetres.	Any positive real number	Continuous. The set of possible measurements can take on any positive value.

For a continuous random variable X, the probability that X lies in the interval [a, b] is given by,

$$P(a \leq X \leq b) = \int_a^b f(x)dx - - - - - - i$$

The function f(x) is called Probability Density Function (pdf) of random variable X.

The pdf must satisfy the following conditions:

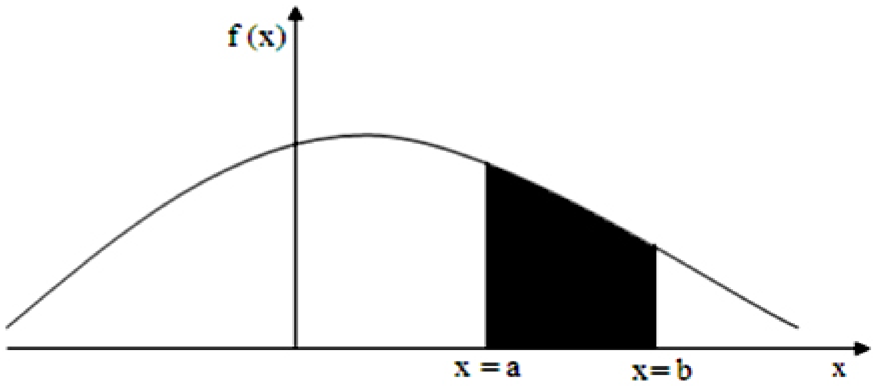
1. f(x)>=0, for all x in Rx
2. $\int_{-\infty}^{\infty} f(x)dx = 1$ (total area under graph is 1)
3. $f(x) = 0, if\ x\ is\ not\ in\ Rx$

For any specified value X_0, $P(X=x_0) = 0$, since $\int_{x_0}^{x} f(x)dx = 0$

Since $P(X = x_0) = 0$, the following equation also hold:

P(a <= X <= b) = P(a < X <= b) = P(a <= X < b) = P(a < X < b)

The graphical interpretation of equation *i* is shown in the figure below.



Random Numbers:

A random number is a number generated by a process, whose outcome is unpredictable, and which cannot be subsequentially reliably reproduced. Random numbers are the basic building blocks for all simulation algorithms.

Properties of Random Numbers:

The two important statistical properties are:

1. Uniformity
2. Independence

Each random number Ri is an independent sample drawn from a continuous uniform distribution between 0 and 1. The probability density function (pdf) is given as:

$$f(x) = \begin{cases} 1, & 0 \le x \le 1 \\ 0, & otherwise \end{cases}$$

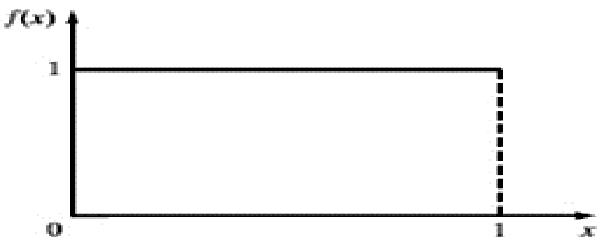


Fig: The Pdf for Random Numbers

The expected value of each Ri is given by

$$E(R) = \int_0^1 x dx = \frac{x^2}{2}\Bigg|_0^1 = \frac{1}{2}$$

The variance is given by

$$V(R) = \int_0^1 x^2\,dx - [E(R)]^2$$
$$= [x^3/3]_0^1 - (1/2)^2 = 1/3 - 1/4$$
$$= 1/12$$

The consequences of uniformity and independence properties are:

1. If the interval (0, 1) is divided into n classes or subintervals of equal length, then the expected number of observations in each interval is N / n, where N is the total number of observations.

2. The probability of observing a value in a particular interval is independent of previous values drawn.

Pseudo-Random Numbers:

Pseudo means false but here pseudo implies that the random numbers are generated by using some known arithmetic operations. Since the arithmetic operation is known and the sequence of random numbers can be repeatedly obtained, the numbers cannot be called truly random.

However, the pseudo-random numbers generated by many computer routines very closely fulfil the requirements of the desired randomness.

If the method of random number generation, i.e. the random number generator is defective, the generated pseudo-random number may have the following departures from ideal randomness:

1. The generated random numbers may not be uniformly distributed.
2. The generated random numbers may not be continuous.
3. The mean of the generated numbers may be too high or too low.
4. The variance may be too high or too low.

Generation of Random Number:

In computer simulation where a very large number of random numbers is generally required, can be obtained by the following method.

1. Random numbers maybe drawn from the random number tables stored in a memory of the computer. The process is neither practical nor economical. It is a very slow process and the number occupied considerable space of computer

memory. Above all, in the real system many time more random number than available in the table.

2. An electronic device may be constructed as a part of a digital computer to generate truly random numbers. This, however, is considered very expensive.

3. Pseudo-random numbers may be generated using some arithmetic operation. These methods must commonly specify a procedure starting with an initial number, the second number is generated and from that a third number and so on. A number of the recursive procedure are used for generating random numbers.

Qualities of an Efficient Random Number Generator:

1. It should have a sufficiently long cycle i.e. it should generate a sufficiently long sequence of random numbers before beginning to repeat the sequence.
2. The random numbers generated should be replicable i.e. by specifying the starting condition, it should numbers as and when desired. Many times common random numbers are required for the comparison of two systems.
3. The generated random numbers should fulfil the requirement of uniformity and independence.
4. The random number generator should be fast and cost-effective.
5. It should be portable to different computers and ideally to a different programming language.

Techniques for Generation of Random Numbers:

The most widely used techniques for generating random numbers are:

1. Linear Congruential Method (LCM):

The most widely used technique for generating random numbers, initially proposed by Lehmer [1951]. This method produces a sequence of integers, X1, X2 … between 0 and m-1 by following a recursive relationship:

$$X_{i-1} = (\underbrace{a}_{\text{The multiplier}}X_i + \underbrace{c}_{\text{The increment}}) \bmod \underbrace{m}_{\text{The modulus}}, \quad i = 0,1,2,\ldots$$

The initial value X0 is called seed. The selection of the values for a, c, m, and X_0 drastically affects the statistical properties and the cycle length.

a. If $c \neq 0$ in the above equation, then it is called as Mixed Congruential method.

b. If c = 0 the form is known as the Multiplicative Congruential method.

The random numbers (Ri) between 0 and 1 can be generated by

$$R_i = \frac{X_i}{m}, \quad i = 1,2,\ldots$$

Example:

Use linear congruential method to generate sequence of random numbers with X0 = 27, a = 17, c = 43, and m = 100.

Solution:

Random numbers (Ri)

The random integers (Xi) generated will be between the range 0 - 99

Equations → Xi+1 = (a Xi + c) mod m, Ri = Xi / m , i=1,2,.....

X1 = (17 * 27 + 43) mod 100 = 2, R1 = 2 / 100 = 0.02

X2 = (17 * 2 + 43) mod 100 = 77, R2 = 77 / 100 = 0.77

X3 = (17 * 77 + 43) mod 100 = 52, R3 = 52/ 100 = 0.52

Hence the numbers are generated.

The secondary properties to generate random numbers include maximum density and maximum period.

a. Maximum Density:

Maximum Density means values assumed by Ri, i = 1, 2… leave no large gaps on the interval [0, 1].

Problem: The values generated from Ri = Xi / m, is discrete on integers instead of continuous.

Solution: A very large integer for modulus m.

b. Maximum Period:

To achieve Maximum density and avoid cycling, the generator should have the largest possible period. Most digital computers use a binary representation of numbers. Speed and efficiency is aided by a modulus m, to be (or close to) a power of 2. The maximal period is achieved by proper choice of a, c, m and X0.

Different Cases Are:

1. For *m* a power of 2, say $m = 2^b$ and $c \neq 0$, the longest possible period is $P = m = 2^b$, provided that c is relatively prime to m and $a = 1 + 4k$, where k is an integer.

2. For *m* a power of 2, say $m = 2^b$ and $c = 0$, the longest possible period is $P = m / 4 = 2^{b-2}$, which is achieved provided that the seed X_0 is odd and the multiplier a, is given by $a = 3 + 8k$ or $a = 5 + 8k$, for some k = 0, 1...

3. For *m* a prime number and $c = 0$, the longest the possible period is $P = m - 1$, which is achieved provided that the multiplier a, has the property that the smallest integer k such that $a^k - 1$ is divisible by *m* is $k = m-1$.

Example:

Using the multiplicative congruential method, find the period of the generator for a = 13, $m = 2^6$ and X_0 = 1, 2, 3, and 4.

Solution:

c=0 (multiplicative congruential method), $m = 2^6 = 64$ and a=13 → (a=5+8*1=13) so 'a' is in the form 5+8k with k=1.

Therefore the maximal period p= m / 4= 64 / 4=16 for odd seeds i.e. for X_0=1 and 3

Equation à $X_{i+1} = (aX_i + c) \bmod m$

When $X_0 = 1$, i = 1, $X_2 = (13 * 1 + 0) \bmod 64 = 13 \bmod 64 = 13$

When $X_0 = 1$, i = 2, $X_3 = (13 * 13 + 0) \bmod 64 = 169 \bmod 64 = 41$

When $X_0 = 1$, i = 3, $X_4 = (13 * 41 + 0) \bmod 64 = 533 \bmod 64 = 21$

When $X_0 = 1$, i = 16, $X_{17} = (13 * 5 + 0) \bmod 64 = 65 \bmod 64 = 1$

…………………………………

…………………………………

When $X_0 = 2$, i = 1, $X_2 = (13 * 2 + 0) \bmod 64 = 26 \bmod 64 = 26$

When $X_0 = 2$, i = 2, $X_3 = (13 * 26 + 0) \bmod 64 = 338 \bmod 64 = 18$

…………………………………

…………………………………

When $X_0 = 2$, i = 8, $X_9 = (13 * 10 + 0) \bmod 64 = 130 \bmod 64 = 2$

Similarly for X_0 =3 and 4 are calculated. The values are tabulated below in the table below

Therefore

For X_0=1, 3, maximal period is 16

For X_0=2, maximal period is 8

For X_0=4, maximal period is 4

Seed

i	X_i $X_0 = 1$	X_i $X_0 = 2$	X_i $X_0 = 3$	X_i $X_0 = 4$
0	1	2	3	4
1	13	26	39	52
2	41	18	59	36
3	21	42	63	20
4	17	34	51	4
5	29	58	23	
6	57	50	43	
7	37	10	47	
8	33	2	35	
9	45		7	
10	9		27	
11	53		31	
12	49		19	
13	61		55	
14	25		11	
15	5		15	
16	1		3	

2. Combined Linear Congruential Generators (CLCG):

As computing power increases, the complexity of the system to simulate also increases. So a longer period generator with good statistical properties is needed. One successful approach is to combine two or more multiplicative congruential generators.

Theorem:

If $W_{i,1}$, $W_{i,2}$,...,$W_{i,k}$ are any independent, discrete-valued random variables and $W_{i,1}$ is uniformly distributed on integers 0 to m_1 -2, then

$$W_i = \left(\sum_{j=1}^{k} W_{i,j} \right) \bmod m_1 - 1$$

is uniformly distributed on the integers 0 to m_1 -2.

To see how this the result can be used to form combined generators,

Let $X_{i,1}$, $X_{i,2}$ ... $X_{i,k}$ be i^{th} output from k different multiplicative congruential generators, where the j^{th} generator has prime modulus m_j and multiplier a_j is chosen so that the period is m_j -1.

Then the j[th] generator is producing $X_{i,j}$ that are approximately uniformly distributed on 1 to m_j -1 and $W_{i,j} = X_{i,j}$ -1 is approximately uniformly distributed on 0 to m_j -2.

Therefore the combined generator of the form,

$$X_i = \left(\sum_{j=1}^{k} (-1)^{j-1} Xi,j\right) mod m_1 - 1 \; Hence, R_i = \begin{cases} \frac{X_i}{m_1}, X_i > 0 \\ \frac{m_1 - 1}{m_1}, X_i = 0 \end{cases}$$

The maximum possible period for a generator is

$$P = \frac{(m_1 - 1)(m_2 - 1)\ldots(m_k - 1)}{2^{k-1}}$$

Note: $(-1)^{j-1}$ coefficient implicitly performs the subtraction $X_{i,1} - 1$

Example:

For 32-bit computers, L'Ecuyer [1988] suggests combining k = 2 generators with m_1 = 2,147,483,563, a_1 = 40,014, m_2 = 2,147,483,399 and a_2 = 40,692. This leads to the following algorithm:

Step 1: Select Seeds

$X_{0,1}$ in the range [1 - 2,147,483,562] for the 1st generator

$X_{0,2}$ in the range [1 - 2,147,483,398] for the 2nd generator

Set i=0

Step 2: For each individual generator, evaluate

$X_{i+1,1} = 40{,}014 \, X_{i,1} \bmod 2{,}147{,}483{,}563$

$X_{i+1,2} = 40{,}692 \, X_{i,2} \bmod 2{,}147{,}483{,}399$

Step 3:

$X_{i+1} = (X_{i+1,1} - X_{i+1,2}) \bmod 2{,}147{,}483{,}562$

Step 4: Return

$$R_{i+1} = \begin{cases} \frac{X_{i+1}}{2{,}147{,}483{,}563}, & X_{i+1} > 0 \\ \frac{2{,}147{,}483{,}562}{2{,}147{,}483{,}563}, & X_{i+1} = 0 \end{cases}$$

Step 5:

Set i = i+1, go back to step 2.

The combined generator has period: $(m_1-1)(m_2-1)/2 \approx 2 \times 10^{18}$

Tests for Random Number Generation:

The two main properties of random numbers are uniformity and independence.

1. Testing for Uniformity:

The hypotheses are as follows

$H0 : Ri \sim U [0, 1]$

$H1: Ri \nsim U [0, 1]$

The null hypothesis H0, reads that the numbers are distributed uniformly on the interval [0, 1]. Rejecting the null hypothesis means that the numbers are not uniformly distributed.

2. Testing for Independence:

The hypotheses are as follows

H0: Ri ~ independently

H1: Ri ≁ independently

This null hypothesis, H0, reads that the numbers are independent. Rejecting the null hypothesis means that the numbers are not independent. This does not imply that further testing of the generator for independence is unnecessary.

For each test, a level of significance α must be stated.

$$\text{Level of significance } \alpha = \frac{\text{probability of rejecting the test}}{\text{probability of accepting the test}}$$

$= P(\text{reject } H_0 \mid H_0 \text{ true})$

Frequently, α is set to 0.01 or 0.05.

There are five types of tests. The first is concerned for testing the uniformity whereas second through five with testing for independence.

1. **Frequency Test:** Compares the distribution of a set of numbers generated to a uniform distribution by using the Kolmogorov-Smirnov or the chi-square test.

2. **Runs Test:** Tests the runs up and down or the runs above and below the mean by comparing the actual values to expected values. The statistic for comparison is the chi-square test.

3. **Autocorrelation Test:** The correlation between numbers is tested and compares the sample correlation to the expected correlation of zero.

4. **Gap Test:** Counts the number of digits that appear between repetitions of a particular digit and then uses the Kolmogorov-Smirnov test to compare with the expected size of gaps.

5. **Poker Test:** Treats the numbers grouped together as a poker hand. Then the hands obtained are compared to what is expected using the chi-square test.

Frequency Tests:

The fundamental test performed to validate a new generator is the test for uniformity. The two different methods of testing are:

1. Kolmogorov-Smirnov Test:

It compares the continuous cumulative distribution function (cdf) of the uniform distribution with the empirical cdf, of the N sample observations. The cdf of an empirical distribution is a step function with jumps at each observed value.

Notations Used:

F(x) →Continuous cdf
SN(x) → Empirical cdf
N →Total number of observations
R1, R2 ...RN → Samples from Random generator
D → Sample statistic
Dα→ Critical value

By definition,

$F(X) = x, 0 \leq x \leq 1$

$$S_N\,(x) = \frac{\text{number of } R_1, R_2 \ldots R_n, \text{which are } \leq x}{N}$$

As N becomes larger, SN(x) ≈ F(x).

Maximum deviation over the range of a random variable is given by

$D = \max | F(x) - SN(x) |$

The sampling distribution of D is known and is tabulated as a function of N in the table below.

Procedure for Testing Uniformity Using the Kolmogorov-Smirnov Test:

Step 1:

Rank the data from smallest to largest. Let R(i) denote the ith smallest observation, so that

$R(1) \le R(2) \le \ldots\ldots \le R(N)$

Step 2:

Compute

$D+ = \max \{(i / N) - R(i)\}$

$1 \le i \le N$

$D- = \max \{R(i) - [(i - 1)/ N]\}$

$1 \le i \le N$

Step 3:

Compute

$D= \max (D+, D-)$

Step 4:

Determine the critical value $D\alpha$, from the table A.8 for the specified significance level α and the given sample size N.

Step 5:

a. If $D > D\alpha$, the null hypothesis that the data are a sample from a uniform distribution is rejected.

b. If D ≤ Dα then there is no difference detected between the true distribution of {R1, R2 …RN} and the uniform distribution. So it is accepted.

Example:

Suppose 5 generated numbers are 0.44, 0.81, 0.14, 0.05, and 0.93. It is desired to perform a test for uniformity using the Kolmogorov-Smirnov test with a level of significance α = 0.05.

Solution

N=5 , i = 1, 2, 3, 4, 5

Step 1 - Arrange R_i from smallest to largest

Step 2 -

R_i	0.05	0.14	0.44	0.81	0.93
i/N	0.20	0.40	0.60	0.80	1.00
$i/N - R_i$	0.15	0.26	0.16	-	0.07
$R_i - [(i-1)/N]$	0.05	-	0.04	0.21	0.13

$D^+ = \max \{i/N - R_i\}$

$D^- = \max \{R_i - [(i-1)/N]\}$

Step 3- $D = \max(D^+, D^-) = 0.26$

Step 4- For $\alpha = 0.05$, $N = 5$

$D_\alpha = D_{0.05} = 0.565$ (from table A.8)

$D < D_\alpha \rightarrow 0.26 < 0.565$

Therefore H0 is not rejected, i.e. no difference between the distribution of generated numbers and the uniform distribution. The calculations in the above table are depicted in the figure below, where empirical cdf SN(x) is compared to uniform cdf F(x). It is seen that D+ is the largest deviation of SN(x) above F(x) and D- is the largest deviation of SN(x) below F(x).

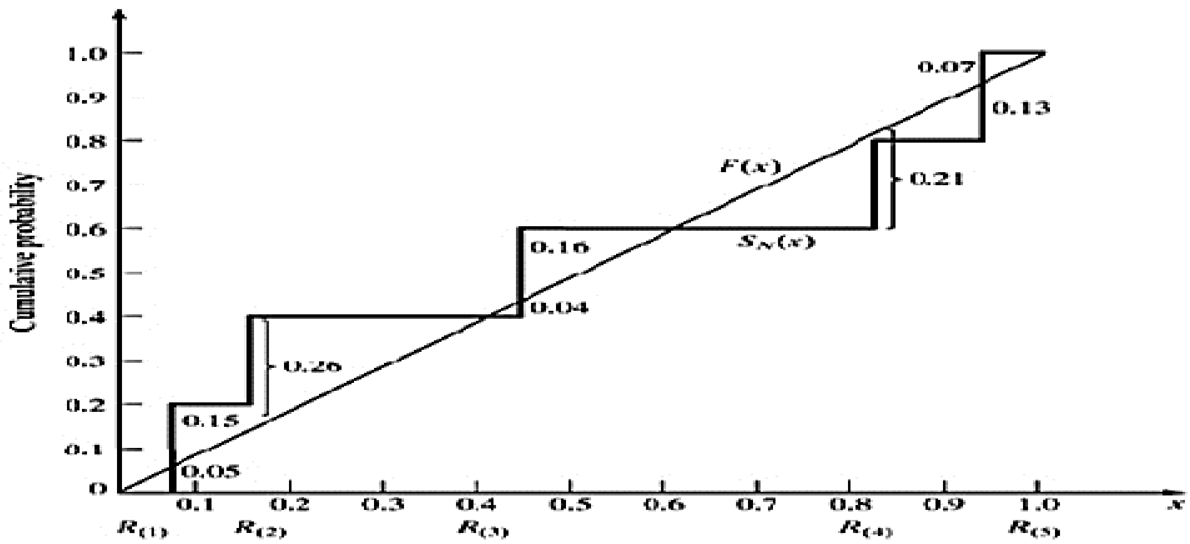


Fig: Comparison of F(x) and SN(x)

2. Chi-Square Test:

It uses the sample statistic.

$$X_0^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

Where,

Oi → observed number in ith class
Ei → expected a number in ith class
n → number of classes

For uniform distribution, Ei is given by

$$E_i = \frac{N}{n}$$

It can be shown that the sampling distribution X_0^2 is approximately the chi-square distribution with n-1 degrees of freedom (i.e.$X_0^2 \le X_{\alpha}^2, n-1$).

Example:

Use a chi-square test with α=0.05 to test whether the data shown below are uniformly distributed.

0.34	0.90	0.25	0.89	0.87	0.44	0.12	0.21	0.46	0.67
0.83	0.76	0.79	0.64	0.70	0.81	0.94	0.74	0.22	0.74
0.96	0.99	0.77	0.67	0.56	0.41	0.52	0.73	0.99	0.02
0.47	0.30	0.17	0.82	0.56	0.05	0.45	0.31	0.78	0.05
0.79	0.71	0.23	0.19	0.82	0.93	0.65	0.37	0.39	0.42
0.99	0.17	0.99	0.46	0.05	0.66	0.10	0.42	0.18	0.49
0.37	0.51	0.54	0.01	0.81	0.28	0.69	0.34	0.75	0.49
0.72	0.43	0.56	0.97	0.30	0.94	0.96	0.58	0.73	0.05
0.06	0.39	0.84	0.24	0.40	0.64	0.40	0.19	0.79	0.62
0.18	0.26	0.97	0.88	0.64	0.47	0.60	0.11	0.29	0.78

Solution:

Let n=10, the interval [0-1] divided in equal lengths, (0.01-0.10), (0.11-0.20), ---, (0.91-1.0)

N = 100

E_i=N/n=100/10=10

The calculations are tabulated below in table below

$X^2_{0.05,9}$ =16.9 (check the table A.6 –using α, n-1)

$X^2_0 < X^2_{0.05,9} = 3.4 < 16.9$

Therefore the null hypothesis of the uniform distribution is not rejected.

Interval	O_i	E_i	$O_i - E_i$	$(O_i - E_i)^2$	$X_0^2 = (O_i-E_i)^2 / E_i$
0.01 - 0.10	8	10	-2	4	0.4
0.11 - 0.20	8	10	-2	4	0.4
0.21 - 0.30	10	10	0	0	0.0
0.31 - 0.40	9	10	-1	1	0.1
0.41 - 0.50	12	10	2	4	0.4
0.51 - 0.60	8	10	-2	4	0.4
0.61 - 0.70	10	10	0	0	0.0
0.71 - 0.80	14	10	4	16	1.6
0.81 - 0.90	10	10	0	0	0.0
0.91 - 1.00	11	10	1	1	0.1
	100	100	0		$X_0^2 = 3.4$

Note:

a. In general, for any value chooses 'n' such that Ei ≥ 5.
b. Kolmogorov-Smirnov test is more powerful than the chi-square test because it can be applied to small sample sizes, whereas chi-square requires large sample, say N ≥ 50.

Runs Tests:

Run - The succession of similar events preceded and followed by a different event is called as run.

Run-length - Number of events that occur in the run.

Example: Tossing coin

Consider the sequence of tossing a coin 10 times: H T T H H T T T H T

No.	Run Length	Run
1	1	H
2	2	T T
3	2	H H
4	3	T T T
5	1	H
6	1	T

There are two possible concerns in run tests. They are

1. Number of runs- Run-up and down & Runs above and below mean
2. Length of runs

1. Runs Up And Down:

a. **Up run**-Sequence of numbers each of which is succeeded by a larger number is called as up run.
b. **Down run**-Sequence of numbers each of which is succeeded by smaller number is called as down run.
c. If a number is followed by a larger number then it denoted by '+'. If followed by a smaller number then by '-'.

To illustrate the above, consider the sequence of numbers

0.87 0.15 0.23 0.45 0.69 0.32 0.30 0.19 0.24 0.18 0.65 0.82 0.93 0.22

The up run and down run are marked as

-0.87 +0.15 +0.23 +0.45 -0.69 -0.32 -0.30 +0.19 -0.24 +0.18 +0.65 +0.82 -0.93 +0.22

The sequence of '+' and '-'are

$$\underbrace{-}\ \underbrace{+\ +\ +}\ \underbrace{-\ -\ -}\ \underbrace{+}\ \underbrace{-}\ \underbrace{+\ +\ +}\ \underbrace{-}$$

It has 7 runs, first run of length one, second run of length three, third run of length 3, and fourth run with one, fifth run with one, sixth run with three and seventh run with one.

There are three up runs and four down runs. If N is several numbers in sequence, then maximum numbers of runs are N-1 and a minimum number of runs is one. If 'a' is the total number of runs in a random sequence, Mean is given by

$$\mu_a = \frac{(2N-1)}{3}$$

Variance,

$$\sigma_a^2 = \frac{16N - 29}{90}$$

For N > 20, the distribution of 'a' is reasonably approximated by a normal distribution, N(μa, σa²). This approximation is used to test the independence of numbers from a generator. The test statistic is obtained by subtracting the mean from the observed number of runs 'a' and dividing by standard deviation, i.e. Test statistic is given by,

$$Z_0 = \frac{a - \mu_a}{\sigma_a}$$

Substituting μ_a and σ_a in above equation, we get

$$Z_0 = \frac{a - [\,(2N-1)/3\,]}{\sqrt{[\,(16N - 29)/90\,]}}$$

Where $Z_0 \sim N(0, 1)$

The null hypothesis is accepted when $-Z_{\alpha/2} \le Z_0 \le Z_{\alpha/2}$, where α is the level of significance. The critical values and rejection region is shown in the figure below.

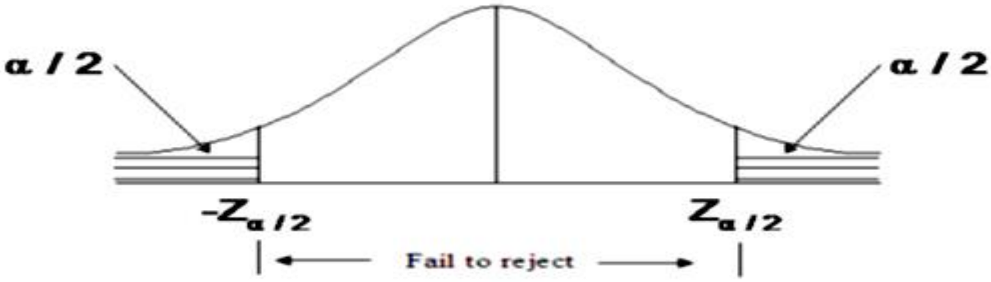


Fig: Accept the Null Hypothesis

Example:

Based on runs up and runs down, determine whether the following sequence of 40 numbers is such that the hypothesis of independence can be rejected or accepted where α = 0.05.

0.41	0.68	0.89	0.94	0.74	0.91	0.55	0.62	0.36	0.27
0.19	0.72	0.75	0.08	0.54	0.02	0.01	0.36	0.16	0.28
0.18	0.01	0.95	0.69	0.18	0.47	0.23	0.32	0.82	0.53
0.31	0.42	0.73	0.04	0.83	0.45	0.13	0.57	0.63	0.29

Solution:

The sequence of runs up and down is as follows:

+ + + - + - + - - - + + - + -
- + - + - - + - - + - + + -
- + + - + - - + + -

No. of runs → a = 26

N = 40

$\mu a = \{2(40) - 1\} / 3 = 26.33$

$\sigma a^2 = \{16(40) - 29\} / 90 = 6.79$

$Z_0 = (26 - 26.33) / \sqrt{(6.79)} = -0.13$

Critical value → $Z_{\alpha/2} \rightarrow Z_{0.025} = 1.96$ (from z - table)

$Z_{\alpha/2} \le Z_0 \le Z_{\alpha/2} \rightarrow -1.96 \le -0.13 \le 1.96$

Therefore independence of the numbers cannot be rejected, we accept the null hypothesis.

Disadvantage Of Runs Up And Down

a. Insufficient to review the independence of a group of numbers

2. Runs Above And Below The Mean

Runs are described with above/below the mean value. A '+' sign is used to indicate above mean and '-' sign for below the mean.

To illustrate the above, consider the sequence of 2-digit random numbers

0.40	0.84	0.75	0.18	0.13	0.92	0.57	0.77	0.30	0.71
0.42	0.05	0.78	0.74	0.68	0.03	0.18	0.51	0.10	0.37

Mean = (0.99+0.00)/2 = 0.495

The runs above and below mean are marked as

-0.40 +0.84 +0.75 -0.18 -0.13 +0.92 +0.57 +0.77 -0.30 +0.71

-0.42 -0.05 +0.78 +0.74 +0.68 -0.03 -0.18 +0.51 -0.10 -0.37

The sequence of '+' and '-' are

- + + - - + + + - + - - +

+ + - - + - -

There are 11 runs, of which 5 are above mean and 6 runs below mean.

Let $n_1 \rightarrow$ No. of individual observations above mean

$n_2 \rightarrow$ No. of individual observations below mean

$b \rightarrow$ Total number of runs

$N \rightarrow$ Maximum number of runs, where $N = n_1 + n_2$

The mean is given by

$$\mu_b = \frac{2n_1n_2}{N} + \frac{1}{2}$$

Variance

$$\sigma_b^2 = \frac{2n_1n_2\,(2n_1n_2 - N)}{N^2\,(N-1)}$$

For either n_1 or n_2 greater than 20, b is approximately normally distributed. The test statistic is obtained by subtracting the mean from several runs 'b' and dividing by the standard deviation i.e.

$$Z_0 = \frac{b - (2n_1n_2/N) - 1/2}{\left(\frac{2n_1n_2\,(2n_1n_2 - N)}{N^2\,(N-1)}\right)^{1/2}}$$

The null hypothesis is accepted when $-Z\alpha/2 \leq Z0 \leq Z\alpha/2$, where α is the level of significance.

Example:

Based on runs above and below mean, determine whether the following sequence of 40 numbers is such that the hypothesis of independence can be rejected or accepted where $\alpha = 0.05$.

0.41	0.68	0.89	0.94	0.74	0.91	0.55	0.62	0.36	0.27
0.19	0.72	0.75	0.08	0.54	0.02	0.01	0.36	0.16	0.28
0.18	0.01	0.95	0.69	0.18	0.47	0.23	0.32	0.82	0.53
0.31	0.42	0.73	0.04	0.83	0.45	0.13	0.57	0.63	0.29

Solution:

Mean= 0.495

The sequence of runs above and below mean is as follows:

-	+	+	+	+	+	+	+	-	-
-	+	+	-	+	-	-	-	-	-
-	-	+	+	-	-	-	-	+	+
-	-	+	-	+	-	-	+	+	-

$n_1 = 18$

$n_2 = 22$

$N = n_1 + n_2 = 40$

$b = 17$

$\mu_b = [\{2(18)\ (22)\} / 40] + (1 / 2) = 20.3$

$\sigma_b^2 = [2\ (18)\ (22)\ \{(2)\ (18)\ (22) - 40\}] / [(40)2\ (40 - 1)] = 9.54$

Since $n_2 > 20$, normal approximation is accepted.

$Z_0 = (17 - 20.3) / \sqrt{(9.54)} = -1.07$

Critical value $\rightarrow Z_{\alpha/2} \rightarrow Z_{0.025} = 1.96$ (from z - table)

$-Z_{\alpha/2} \leq Z_0 \leq Z_{\alpha/2} \rightarrow -1.96 \leq -1.07 \leq 1.96$

Therefore hypothesis of independence cannot be rejected based on this test.

Disadvantage Of Runs Above and Below Mean

a. If two numbers are below mean, two numbers are above mean and so on. Then the numbers are dependent.

3. Runs Test: Length Of Runs

Let Y_i be the number of runs of length i, in a sequence of N numbers. For an independent sequence,

The expected value of Yi for runs up and down is given by

$$E(Y_i) = \frac{2}{(i+3)!}[N(i^2 + 3i + 1) - (i^3 + 3i^2 - i - 4)], i \leq N - 2$$

$$E(Y_i) = \frac{2}{N!}, i = N - 1$$

For runs above and below mean, the expected value of Yi is given by

$$E(Y_i) = \frac{Nw_i}{E(I)}, N > 20$$

Where w_i, the approximate probability that a run has length i, is given by

$$w_i = \left(\frac{n_1}{N}\right)^i \left(\frac{n_2}{N}\right) + \left(\frac{n_1}{N}\right)\left(\frac{n_2}{N}\right)^i, N > 20$$

And E (I), the approximate expected length of a run, is given by

$$E(I) = \frac{n_1}{n_2} + \frac{n_2}{n_1}, N > 20$$

The approximate expected total number of runs (of all lengths) E (A), is given by

$$E(A) = \frac{N}{E(I)}, N > 20$$

The appropriate test is chi-square test with O_i, the observed number of runs of length i. The test statistic is given by

$$X_0^2 = \sum_{i=1}^{L} \frac{[O_i - E(Y_i)]^2}{E(Y_i)}$$

Where,

L = N – 1 for runs up and down

L = N for runs above and below mean.

If null hypothesis of independence is true then X_0^2 is approximately chi-squared distributed with L-1 degrees of freedom.

Example:

Given the sequence of numbers, can the hypothesis that the numbers are independent be rejected on the basis of length of runs up and down at α = 0.05?

0.30	0.48	0.36	0.01	0.54	0.34	0.96	0.06	0.61	0.85
0.48	0.86	0.14	0.86	0.89	0.37	0.49	0.60	0.04	0.83
0.42	0.83	0.37	0.21	0.90	0.89	0.91	0.79	0.57	0.99
0.95	0.27	0.41	0.81	0.96	0.31	0.09	0.06	0.23	0.77
0.73	0.47	0.13	0.55	0.11	0.75	0.36	0.25	0.23	0.72
0.60	0.84	0.70	0.30	0.26	0.38	0.05	0.19	0.73	0.44

Solution:

N = 60

The sequence of + and – are as follows

+	-	-	+	-	+	-	+	+	-	+	-
+	+	-	+	+	-	+	-	+	-	-	+
-	+	-	-	+	-	-	+	+	+	-	-
-	+	+	-	-	-	+	-	+	-	-	-
+	-	+	-	-	-	+	-	+	+	-	

The length of runs in the sequence is as follows

1, 2, 1, 1, 1, 1, 2, 1, 1, 1, 2, 1, 2, 1, 1, 1, 1, 2, 1, 1,

1, 2, 1, 2, 3, 3, 2, 3, 1, 1, 1, 3, 1, 1, 1, 3, 1, 1, 2, 1

Calculate O_i

Run Length, i	1	2	3	4
Observed Runs, O_i	26	9	5	0

The expected value of Yi,

For run length one,

$$E(Y_1) = \frac{2}{(1+3)!}[60(1^2 + 3(1) + 1) - (1^3 + 3(1)^2 - 1 - 4)] = 25.08$$

Run length two,

$$E(Y_2) = \frac{2}{(2+3)!}[60(2^2 + 3(2) + 1) - (2^3 + 3(2)^2 - 2 - 4)] = 10.77$$

Run length three,

$$E(Y_3) = \frac{2}{(3+3)!}[60(3^2 + 3(3) + 1) - (3^3 + 3(3)^2 - 3 - 4)] = 3.04$$

$\therefore E(Y_1) + E(Y_2) + E(Y_3) = 38.89$

We find mean (runs up and down)

$$\mu_\alpha = \frac{2N-1}{3} = \frac{2(60)-1}{3} = 39.67$$

Expected value, when $i \geq 4$

$$\mu_\alpha = \sum_{i=1}^{3} E(Y_i) = 39.67 - 38.89 = 0.78$$

To find X_0^2, the calculations and procedures are shown in the table below:

Run length (i)	Observed number of runs (O_i)	Expected number of runs $E(Y_i)$	$\frac{[O_i - E(Y_i)]^2}{E(Y_i)}$
1	26	25.08	0.03
2 3 4	9 5 0 } 14	10.77 3.82 0.78 } 14.59	} 0.02
-	40	39.67	$X_0^2 = 0.05$

$X^2_{0.05,1} = 3.84$

$X_0^2 < X^2_{0.05,1} = 0.05 < 3.84$

Example:

Given the sequence of numbers can the hypothesis that the numbers are independent be rejected on the basis of length of runs above and below mean at α = 0.05?

0.30	0.48	0.36	0.01	0.54	0.34	0.96	0.06	0.61	0.85
0.48	0.86	0.14	0.86	0.89	0.37	0.49	0.60	0.04	0.83
0.42	0.83	0.37	0.21	0.90	0.89	0.91	0.79	0.57	0.99
0.95	0.27	0.41	0.81	0.96	0.31	0.09	0.06	0.23	0.77
0.73	0.47	0.13	0.55	0.11	0.75	0.36	0.25	0.23	0.72
0.60	0.84	0.70	0.30	0.26	0.38	0.05	0.19	0.73	0.44

Solution

N = 60

Mean = (0.99+0.00)/2 = 0.495

The sequence of + and – are as follows

-	-	-	-	+	-	+	-	+	+	-	+	-
+	+	-	-	+	-	+	-	+	-	-	+	+
+	+	+	+	+	-	-	+	+	-	-	-	-
+	+	-	-	+	-	+	-	-	-	+	+	+
+	-	-	-	-	-	+	-					

$n_1 = 28$

$n_2 = 32$

$N = n_1 + n_2 = 60$

The length of runs in the sequence is as follows

4, 1, 1, 1, 1, 2, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 2, 7, 2, 2, 4, 2, 2, 1, 1, 1, 3, 4, 5, 1, 1

Calculate Oi

Run Length, i	1	2	3	≥ 4
Observed Runs, O_i	17	8	1	5

The probabilities of runs of various lengths w_i are as follows

$$w_1 = \left(\frac{28}{60}\right)^1 \frac{32}{60} + \frac{28}{60}\left(\frac{32}{60}\right)^1 = 0.498$$

$$w_2 = \left(\frac{28}{60}\right)^2 \frac{32}{60} + \frac{28}{60}\left(\frac{32}{60}\right)^2 = 0.249$$

$$w_3 = \left(\frac{28}{60}\right)^3 \frac{32}{60} + \frac{28}{60}\left(\frac{32}{60}\right)^3 = 0.125$$

The expected number of runs of various lengths is

$$E(Y_1) = \frac{N w_1}{E(I)} = \frac{60(0.498)}{2.02} = 14.79$$

$$E(Y_2) = \frac{N w_2}{E(I)} = \frac{60(0.249)}{2.02} = 7.40$$

$$E(Y_3) = \frac{N w_3}{E(I)} = \frac{60(0.125)}{2.02} = 3.71$$

The expected total number of runs is

$$E(A) = \frac{N}{E(I)} = \frac{60}{2.02} = 29.7$$

For i ≥ 4,

$$E(A) - \sum_{i=1}^{3} E(Y_i) = 29.7 - 25.9 = 3.8$$

To find X_0^2 the calculations and procedures are shown in table below:

Run length (i)	Observed number of runs (O_i)	Expected number of runs $E(Y_i)$	$\frac{[O_i - E(Y_i)]^2}{E(Y_i)}$
1	17	14.79	0.33
2	8	7.40	0.05
3	$\left.\begin{matrix}1\\5\end{matrix}\right\}6$	$\left.\begin{matrix}3.71\\3.80\end{matrix}\right\}7.51$	$\}0.30$
≥ 4			
-	31	29.70	$X_0^2 = 0.68$

$$X^2_{0.05,2} = 5.99$$

$$X_0^2 < X^2_{0.05,2} = 0.68 < 5.99$$

Therefore the hypothesis of independence is accepted.

4.Test For Autocorrelation:

The uniformity test of random numbers is only a necessary test for randomness, not a sufficient one. A sequence of numbers may be perfectly uniform and still not random.

For example the sequence 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.1, 0.2, 0.3, would give a perfectly uniform distribution with chi-square value perfectly as zero. But the sequence can be no means be regarded as random.

The numbers are not independent as the occurrence of one number say 0.3 decides the next, which is to be 0.4, etc. This defect is called serial autocorrelation of an adjacent pair of numbers.

The chi-square test for serial autocorrelation makes use of a 10 * 10 matrix. The 10 class describe in the uniformity test are represented both along the rows and columns.

If the classes are to be represented on a bar chart, 100 bars, one for each cell of a matrix will be required. To reduce the number of groups instead of 10 random numbers are divided into a smaller number of a class as 3 or 4.

Three class will be as:

a. Less than or equal to 0.33
b. Less than or equal to 0.67
c. Less than or equal to 1.0

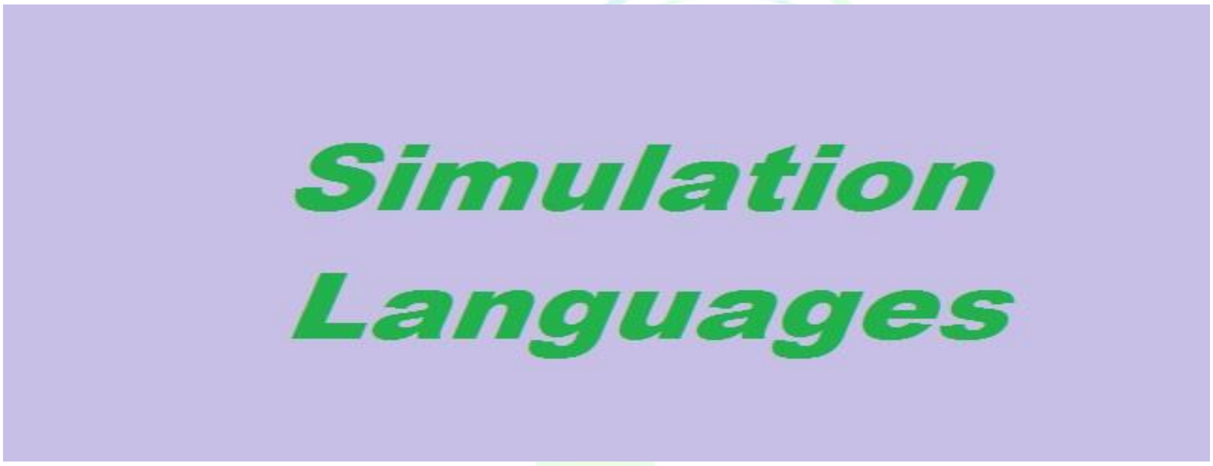
With three classes in a row and three classes in a column, there will be 9 cells.

Unit VI: Simulation Languages - Simulation and Modeling

Introduction of Simulation Languages:

Simulation languages are versatile, the general purpose of classes of simulation software that can be used to create a magnitude of modelling operation. In a sense, these languages are comparable to FORTRAN, C#, VB.NET or Java but also include specific features to facilitate the modelling process.

Some examples of modern simulation languages are GPSS/H, GPSS/PC, SLX and SIMSCRIPT III. Other simulation languages such as SIMAN have been integrated into a broader development framework. Simulation language exists for discrete, continuous and agent-based modelling paradigm.



Features of Simulation Languages:

Specialized features usually differentiate from a general programming language. These features are intended to free the analyst from recreating software tools and procedures used by virtually all modelling application.

Not only the development of these features be time-consuming and difficult, but without them, the consistency of a model could vary and additional debugging, validation and verification would be required.

Most simulation languages should have the following features:

1. Simulation clock or a mechanism for advancing simulated time.
2. Methods to schedule the accuracy of the event.
3. Tools to collect and analyze statistics concerning the uses of various resources and entities.

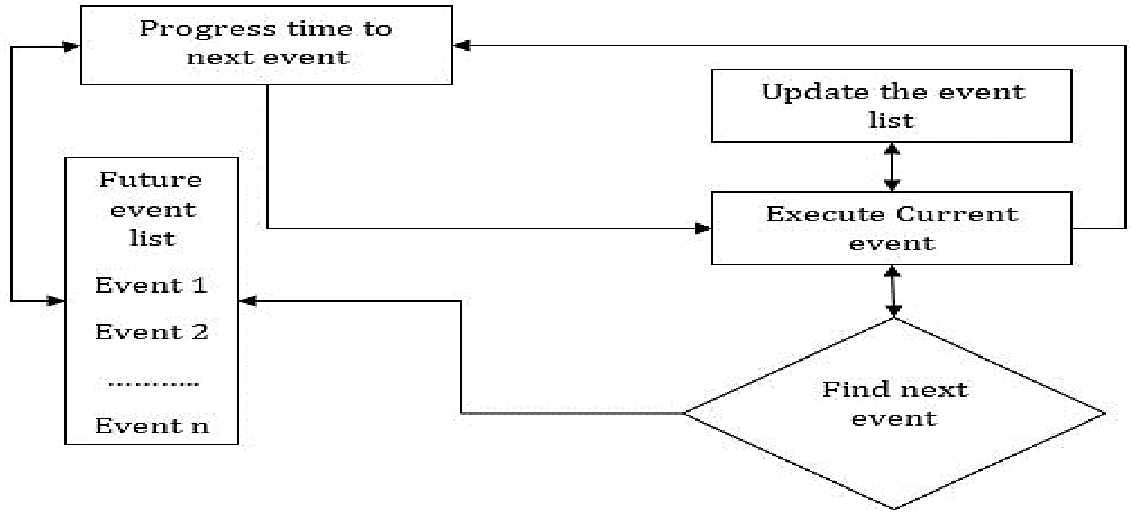
4. Tools for reporting the result.
5. Debugging and error detection facilities.
6. Random number generator and related set of tools.
7. The General framework for model creation.

Working on Simulation Language:

Most discrete event simulation language model a system by updating the simulation clock to the time that the next event is scheduled to occur.

Events and their scheduled times of occurrence are maintained automatically on one of two order list, the current event chain or future event chain. The current event chain keeps a list of all events that will occur at the present clock time.

The future event chain is a record of all events that can occur at some point in the future. A simulation clock moves to the next event on the future events chain and changes the system state of the model based on the characteristics of that event.



Merits of Simulation Language:

1. Since most of the features to be programmed are built-in simulation language takes comparatively less programming time and effort.

2. Since simulation language consists of blocks, specially constructed to simulate the common feature, they provide a natural framework for simulation and modelling.

3. Simulation models coded in simulation language can be easily changed and modified.

4. The error detection and analysis are done automatically in a simulation language.

5. Simulation models developed in simulation language, especially the specific application package called simulators are very easy to use.

Features of Simulation Software:

1. Modeling Flexibility:

The simulation must be flexible enough to adapt to change the configuration. The domain of application should be wide and the model should be equally valid over the whole domain.

2. Ease of Modeling:

It should be easy to develop the simulation model, easy to debug the program and validate the model. The software should have an in-built interactive debugger, error detector and online health.

3. Fast Execution Speed:

The speed of execution is always a very important requirement of any simulation model. It is especially an essential feature in case of simulation of a large and complex system.

4. Compatibility to Various Computer System:

The simulation language should be compatible with various computer system like microcomputer, engineering workstation minicomputers and mainframe computers.

5. Statistical Capabilities:

A simulation software should contain multiple stream random number generators and as many standards probability distributions as possible. It should have blocks for designing the statistics like the simulation run, warming up period and the number and the length of replication.

6. The Capability of Animation:

The animation capability of the simulation package is one of the main reasons for the popularity of simulation language. The animation is carried out in two modes i.e. concurrent mode and playback mode.

7. Report Presentation Capabilities:

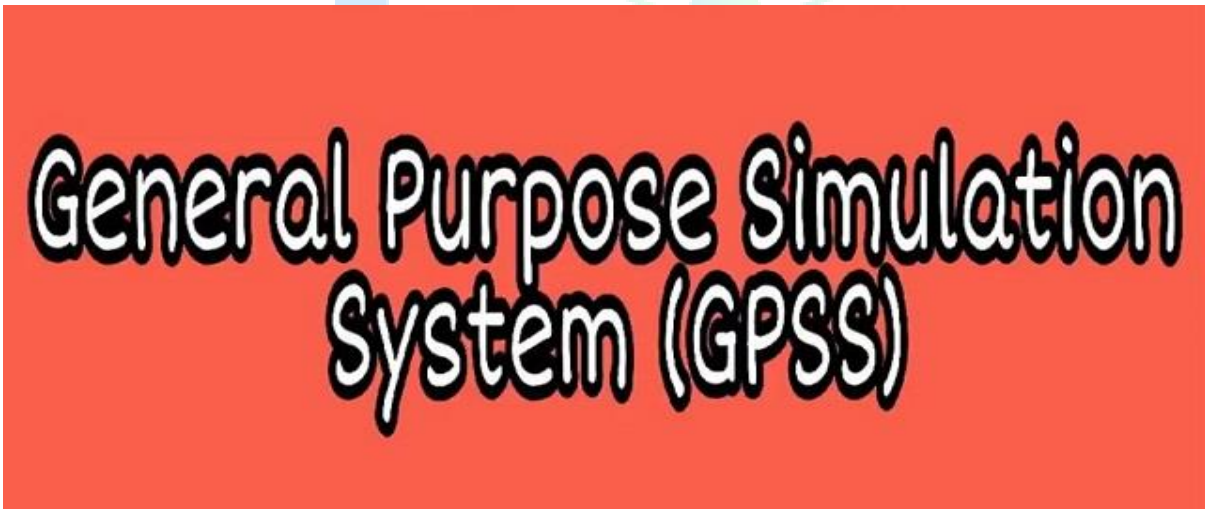
The output of the simulation the model should be well documented and illustrated. The user should be able to get the information of its interest in the easily understandable form.

Types of Simulation Languages:

A variety of simulation languages exist and are used by business, researchers, manufacturing and service companies and consultant. The two common simulation languages are GPSS and SIMSCRIPT.

1.GPSS (General Purpose Simulation System):

GPSS was originally developed by Geoffrey Gordon of IBM and released in October 1961. Following IBM release of GPSS to the public domain it became a multi-vendor simulation language and has been in continuous use since.



In general, GPSS enjoys widespread popularity due to its sensible world view and power. Its basic function can be easily learned while powerful features make it ideal for modelling complex systems.

It is used to simulate the queuing system that consists of customer entities interacting and completing a system of constraint resources. The resources are structured as a network of blocks that entities are entered and used to perform various task in the sudden amount of simulated time.

As entities move through these networks, which have been organized to represent a real-world system, statistics are collected and used to determine if the system contains bottleneck, is over or underutilization or exits other characteristics. Output data is made available for the analyst at the end of the production run.

Discrete Systems Modelling and Simulation with GPSS:

To represent the state of the system in the model, a set of numbers is used that is used in the discrete-time system. A number, state descriptor is used to represent some aspect of the system state. Some state descriptors range over values that have physical significance and some represent conditions.

The effect of the state descriptors is to change the value as the simulation proceeds. We define a discrete event as a set of circumstances that causes an instantaneous change in one or more system state descriptors.

It is possible that two different events occur simultaneously, or are modelled as being simultaneous so that not all changes of state descriptors occurring simultaneously necessarily belong to a single event.

The term simultaneous refers to the occurrence of the changes in the system, and not to when the changes are made in the model in, which, of necessity, the changes must be made sequentially.

A system simulation must contain a number representing time. The simulation proceeds by executing all the changes to the system descriptors associated with each event, as the events occur, in chronological order. How events are selected for execution (when there are simultaneous events) is an important aspect of programming simulations.

For writing discrete system simulation programs, several programming languages have been produced. These programs embody a language with which to describe the system and a programming system that will establish a system image and execute a simulation algorithm.

Each language is based upon a set of concepts used for describing the system. The term worldview has come to be used to describe this aspect of simulation programs. The user of the program must learn the world-view of the particular language he is using and to describe the system in those terms.

Given such a description, the simulation programming system can establish a data structure that forms the system image. It will also compile and sometimes supply the routines to carry out the activities.

One of the most commonly used simulation languages is GPSS. It illustrates the divergence in design consideration. GPSS has been written especially for the user with little or no knowledge of programming experience. The simplification of GPSS results in some loss of flexibility. GPSS applies to a wide variety of systems.

GPSS Programs:

The General-Purpose Simulation System language has been developed over many years, principally by the IBM Corporation. It has been implemented on several different manufacturers' machines and there are variations in the different implementations.

GPSS V is more powerful and has more language statements and facilities than GPSS/360.The GPSS V is implemented by IBM Corporation.

General Description:

The system to be simulated in GPSS is described as a block diagram in which the blocks represent the activities and lines joining the blocks indicate the sequence in which the activities can be executed. Where there is a choice of activities, more than one line leaves a block and the condition for the choice is stated at the block.

To base a programming language on this descriptive method, each block must be given a precise meaning. The approach taken in GPSS is to define a set of 48 specific block types, each of which represents a characteristic action of the systems.

Only the specified block types are allowed while drawing the block diagram of the system. Each block type is given a name that is descriptive of the block action and is represented by a particular symbol. Each block type has several data fields.

Moving through the system being simulated are entities that depend upon the nature of the system. In a communication system, the entities of concern are messages, which are moving. Meanwhile, in a road transportation system, the entities that are moving are the motor vehicles.

A data processing system is concerned with records. In the simulation, these entities are called transactions. The sequence of events in real-time is reflected in the movement of transactions from block to block in simulated time.

Transactions are created at one or more GENERATE blocks and are removed from the simulation at TERMINATE blocks. There can be many transactions simultaneously moving through the block diagram.

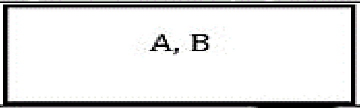
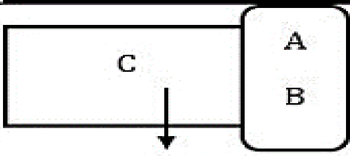
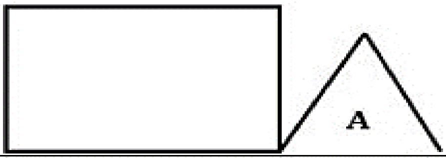
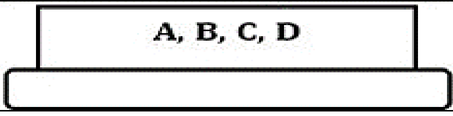
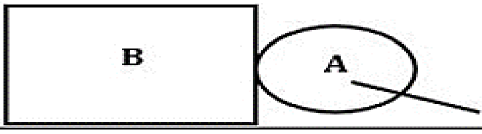
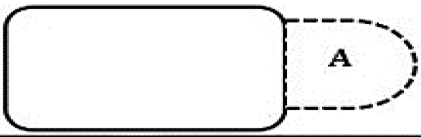
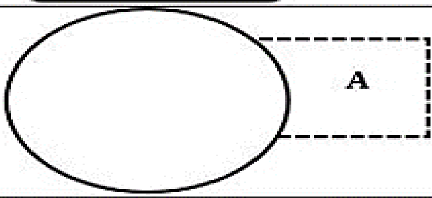
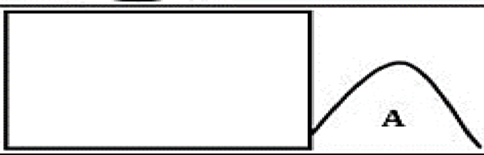
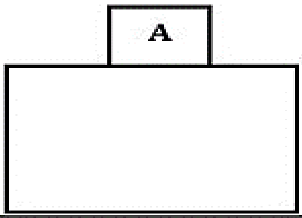
Each transaction is always positioned at a block and most blocks can hold many transactions simultaneously. The transfer of a transaction from one block to another occurs instantaneously at a specific time or when some change of system condition occurs.

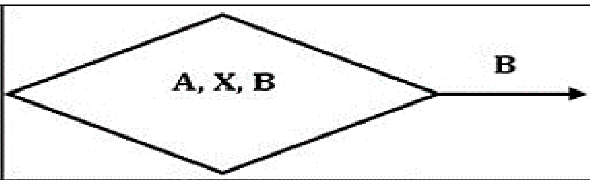
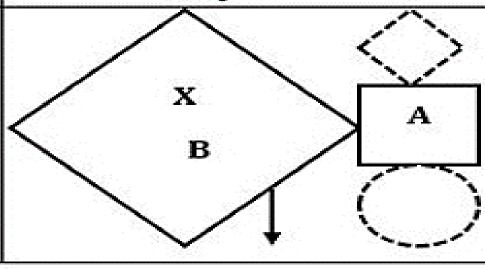
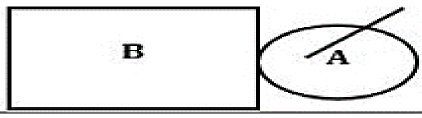
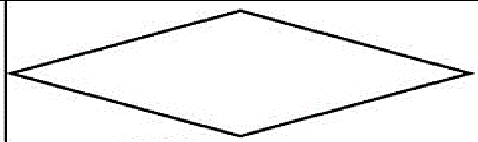
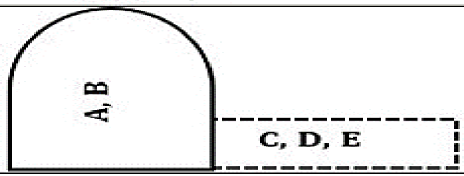
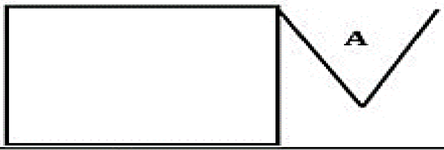
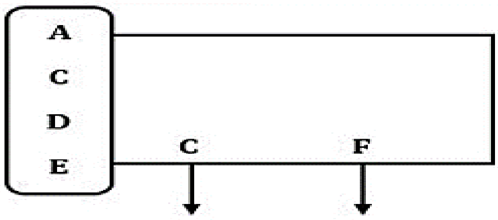
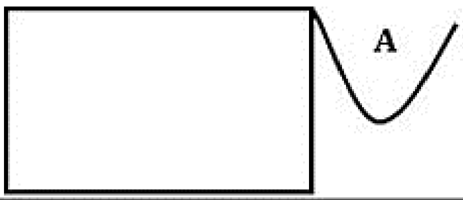
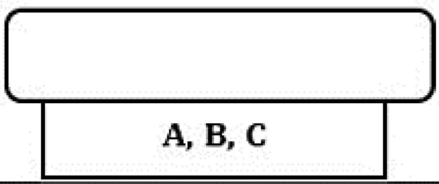
A GPSS block diagram can consist of many blocks up to some limit prescribed by the program (usually set to 1000). An identification number called a location is given to each block, and the movement of transactions is usually from one block to the block with the next highest location.

The locations are assigned automatically by an assembly program within GPSS so that, when a problem is coded, the blocks are listed in sequential order. Blocks that need to be identified in the programming of problems are given a symbolic name.

The assembly program will associate the name with the appropriate location. Symbolic names of blocks and other entities of the program must be from three to five non-blank characters of which the first three must be letters.

GPSS Block Diagram Symbols:

S.N.	Representation/Symbols	Meaning
1		Advance
2		Link
3		Seize
4		Assign
5		Logic
6		Tabulate
7		Depart
8		Mark
9		Terminate
10		Enter
11		Priority

12		Test
13		Gate
14		Queue
15		Transfer
16		Generate
17		Release
18		Unlink
19		Leave
20		Save Value

Basic Concepts:

Action Times:

Clock Time is represented by tan integral number, with the interval of real-time corresponding to a unit of time chosen by the program user. The unit of time is not specifically stated but is implied by giving all the time in terms of the same unit.

One block type called ADVANCE is concerned with representing the expenditure of time. The program computes an interval of time called action time for each transaction as it enters an ADVANCE block and the transaction remains at the block for this interval of a simulated time before attempting to proceed.

The only other block type that employs action time is the GENERATE block, which creates transactions. The action time at the GENERATE block controls the interval between successive arrivals of transactions.

The action time may be a fixed interval or a random variable, and it can be made to depend upon conditions in the system in various ways. Action time is defined by giving a mean and modifier as the A and B fields for the block.

If the modifier is zero, the action time is a constant equal to the mean. If the modifier is a positive number (<=mean), the action time is an integer random variable chosen from the range (mean ± modifier), with equal probabilities given to each number in the range.

By specifying the modifier at an ADVANCE or GENERATE block to be a function, the value of the function controls the action time. The action time is derived by multiplying the mean by the value of the function.

Various types of input can be used for the functions, allowing the functions to introduce a variety of relationships among the variable of a system. In particular, by making the function an inverse cumulative probability distributed, and using as input a random number which is uniformly distributed, the function can provide a stochastic variable with a particular non-uniform distribution.

The GENERATE block normally begins creating transactions from zero time and continues to generate them throughout the simulation. The C field, however, can be used to specify an offset time as the time when the first transaction will arrive.

If the D field is used, it specifies a limit to the total number of transactions that will come from the block. Transactions have a priority level and they carry items of data called parameters. The E field determines the priority of the transactions at the time of creation. If it is not used, the priority is of the lowest level.

Parameters can exist in four formats. They can be signed integers of the full word, half word, or byte size, or they can be signed floating-point numbers. If no specific assignment of parameter type is made, the program creates transactions with 12 half word parameters. The C, D and E fields, also, will not be needed.

The Succession of Events:

The program maintains records of when each transaction in the system is due to move. It proceeds by completing all movements that are scheduled for execution at a particular instant of time and that can logically be performed.

Where there is more than one transaction due to move, the program processes transactions in the order of their priority class, and on a first-come, first-served basis within a priority class.

Once the program has begun moving a transaction it continues to move the transaction through the block diagram until one of the several circumstances arises.

The transaction may enter an ADVANCE block with a non-zero action time, in which case, the program will turn its attention to other transactions in the system and return to that transaction when the action time has been expended.

The conditions in the systems may be such that the action the transaction is attempting to execute by entering a block cannot be performed at the current time. The transaction is said to be blocked and it remains at the block it last entered. The program will automatically detect when the blocking condition has been removed and will start to move the transaction again at that time.

A third possibility is that the transaction enters a TERMINATE block, in which case it is moved from the simulation. A fourth possibility is that a transaction may be put on a chain.

When the program has moved one transaction as far as it can go, it turns its attention to any other transactions due to move at the same time instant. If all such movements are complete, the program advances the clock to the time of the next most imminent event and repeats the process of executing events.

Choice of Paths:

The TRANSFER block allows some location other than the next sequential location to be selected. The choice is normally made between two blocks referred to as next blocks A and B. The method used for choosing is indicated by a selection factor in the field A of the TRANSFER block.

It can be set to indicate one of the nine choices. Next blocks A and B are placed in fields B and C, respectively. If no choice is to be made, the selection factor is left blank. An unconditional transfer is then made to next block A.

A random choice can be made by setting the selection factor, S, to a three-digit decimal fraction. The probability of going to next block A is then 1-S, and to the next block B it is S. a conditional mode, indicated by setting field A to BOTH, allows a transaction to select an alternate path depending upon existing conditions.

The transaction goes to next block A if this move is possible, and to the next block B if it is not. If both moves are impossible the transaction waits for the first to become possible, giving preference to A in the event of simultaneity.

GPSS Programs Application:

Let us consider a scenario representing a machine tool in a manufacturing shop which is turning out parts at the rate of every 5 minutes. The parts are then examined by the inspector who takes 4 3 minutes for each part and rejects about 10% of the parts. We can represent each part by one transaction, and the time unit selected for the problem will be 1 minute. A block diagram representing the system is given below:

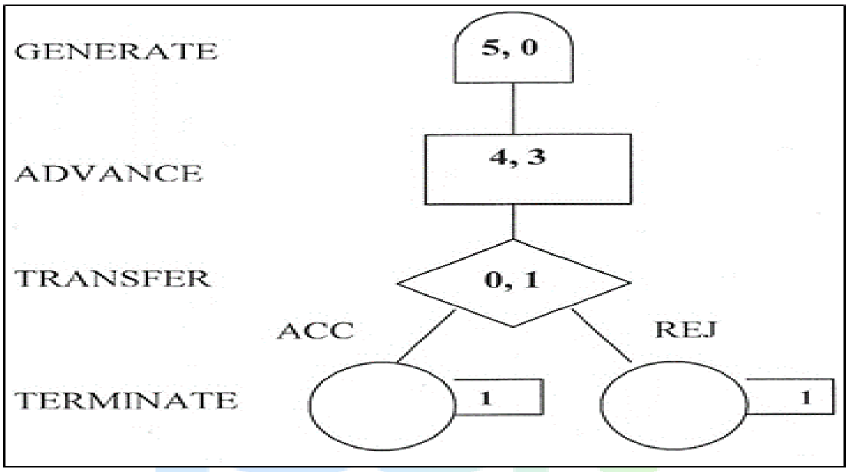


Fig: Manufacturing Shop – Model 1

In the block diagram, the block location is placed at the top of the block, the action time is indicated in the center in the form T = a, b where a is the mean MD b is the modifier and the selection factor is placed at the bottom of the block.

A GENERATE block is used to represent the output of the machine by creating one transaction every five units of time. An ADVANCE block with a mean of 4 and modifier of 3 is used to represent an inspection.

The time spent on inspection will, therefore, be any one of the values 1,2, 3, 4, 5, 6 or 7, with equal probability given to each value. Upon completion of the inspection, transactions go to a TRANSFER block with a selection factor of 0.1, so that 90% of the parts go to the next location called ACC, to represent accepted parts and l0% go to another location called REJ to represent rejects. Both locations reached from the TRANSFER block are TERMINATE blocks.

The problem can be coded. Column 1 is only used for a comment card. A field from columns 2 to 6 contains the location of the block where it must be specified. The GPSS program will automatically assign sequential location numbers as it reads the statements, so it is not usually necessary for the user to assign locations.

The TRANSFER block, however, will need to make reference to the TERMINATE blocks to which it sends transactions, so these blocks have been given symbolic location names ACC and REJ.

The second section of the coding, from columns 8 to 18, contains the block type name, which must begin in column 8. Beginning in column 19, a series of fields may be present, each separated by commas and having no embedded blanks. Anything following the first blank is treated as a comment. The meaning of the fields depends upon the block type.

The program also accepts input in a free format, in which the location field, if used, begins in column 1; but none of the other fields has a fixed starting position. Instead a single blank mark the transition from the location field to the operation field and from the operation held to the operands. If no location is specified, an initial blank is used.

For the TRANSFER block, the first field is the selection factor, and the B and C fields are exits 1 and2, respectively. In this case, exit 1 is the next sequential block, and it would be permissible to omit the name ACC in both the TRANSFER field B and the location field of the first TERMINATE block.

It should also be noted that when a TRANSFER block is used in an unconditional mode, so that field A is not specified, a comma must still be included to indicate the field.

The program runs until a certain the count is reached as a result of transactions terminating. Field A of the TERMINATE block carries a number indicating by how much the termination count should be incremented when a transaction terminates at that block.

The number must be positive and it can be zero, but there must be at least one TERMINATE block that has a non-zero field A. in this case, both TERMINATE blocks have 1; so the terminating counter will add up the total number of transactions that terminate, in other words, the total number of both good and bad parts inspected.

A control statement called START indicates the end of the problem definition and contains, in field A, the value the terminating counter is to reach to end the simulation. In this case, the START Statement is set to stop the simulation at a count of 1,000. Upon reading a START statement, the program begins executing the simulation.

When the simulation is completed, the program automatically prints an output report, in a prearranged format, unless it has been instructed otherwise.

The problem input is printed first, with the locations assigned by the problem listed to the left, and a sequential statement number on the right. The first line of the output following the listings gives the time at which the simulation stopped.

The time is followed by a listing of block counts. Two numbers are shown for each block of the model. On the left is a count of how many transactions were in the block at the time the simulation stopped, and on the right is a figure showing the total number of transactions that entered the block during the simulation.

Facilities and Storage:

Associated with the system being simulated are many permanent entities, such as items of equipment, which operate on the transactions. Two types of permanent entities are defined in GPSS to represent system equipment.

A facility is defined as an entity that can be engaged by a single transaction at a time. Storage is defined as an entity that can be occupied by many transactions at a time, up to some predetermined limit.

A transaction controlling a facility, however, can be interrupted or preempted by another transaction. Also, both facilities and storages can be made unavailable, as would occur if the equipment they represent breaks down, and can be made available again, as occurs when a repair has been made.

There can be many instances of each type of entity to a limit set by the program (usually 300), Individual entities are identified by number, a separate number sequence being used for each type. The number 0 for these and all other GPSS entities is illegal.

Block types SEIZE, RELEASE, ENTER and LEAVE are concerned with using facilities and storages. Field A in each case indicates which facility or storage is intended, and the choice is usually marked in the flag attached to the symbols of the blocks.

The SEIZE block allows a transaction to engage a facility if it is available. The RELEASE block allows a transaction to disengage the facility. An ENTER block allows a transaction to occupy a pace in storage if it is available, and the LEAVE block allows it to give up space.

If the fields B of the ENTER and LEAVE blocks are blank, the storage contents are changed by 1. If there is a number (>=1), then the contents change by that value. Any number of blocks may be placed between the points at which a facility is seized and released to simulate the actions that would be taken while the transaction has control of a facility. Similar arrangements apply for making use of storages.

To illustrate the use of these block types, consider again the manufacturing shop. Since the average inspection time is 4 minutes and the average generation rate is one every 5 minutes, there will normally be only one part inspected at a time.

Occasionally, however, a new part can arrive before the previous part has completed its inspection. This situation will result in more than one transaction being at the ADVANCE block at one time.

Assuming that there is only one inspector, it is necessary to represent the inspector by a facility, to simulate the fact that only one part at a time can be inspected. A SEIZE block and a RELEASE block needs to be added to simulate the engaging and disengaging of the inspector.

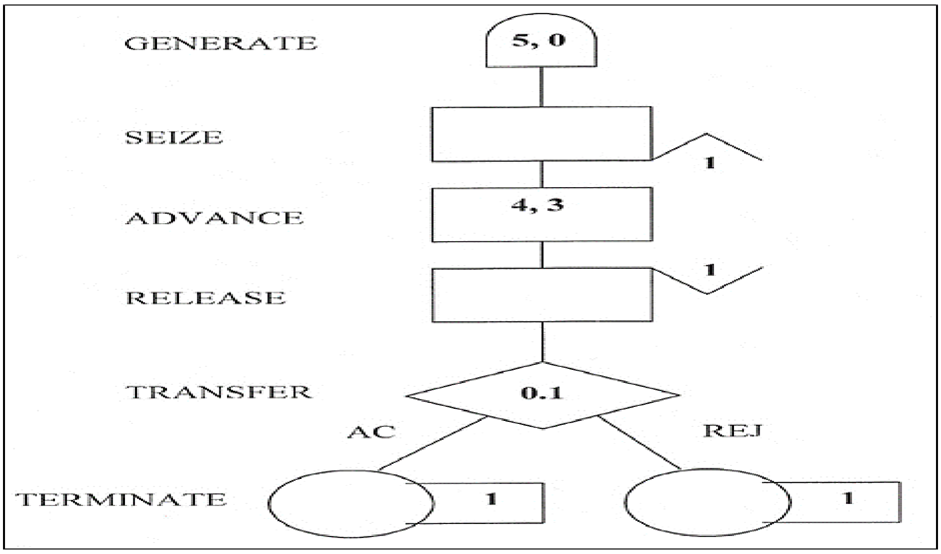


Fig: Manufacturing Shop – Model 2

If more than one inspector is available, they can be represented as a group by storage with a capacity equal to the number of inspectors. The SEIZE and RELEASE blocks of the previous model are then replaced by an ENTER and a LEAVE block, respectively. Suppose, for example, the inspection time were three times as long as before; three inspectors would be justified.

The case of a single inspector can be modelled by using storage with capacity 1 instead of the facility. An important logical difference between the two possible representations is that, for a facility, only the transaction that seized the facility can release it, whereas, for storage, entering and leaving can be separate actions carried out independently by different transactions.

2.SIMSCRIPT III:

This language is a direct descendant of the original SIMSCRIPT language produced at Rand Corporation in 1960's. SIMSCRIPT III has constructs that allow a modular to approach a problem from either a process or an event-oriented world view.

It Offers Unique Features Which Are:

a. Object Oriented Programming
b. Modularity
c. SIMSCRIPT III development studio
d. Object Oriented SIMSCRIPT III graphics
e. Database Connectivity (SDBC)

In general, SIMSCRIPT III is a free form of language with English like a statement. This syntax allows the code in the system to become self-documenting. Model components can be programmed clearly enough to provide an excellent representation of the organization and logic of the system being simulated.

SIMSCRIPT III

SIMSCRIPT Program:

SIMSCRIPT is a very widely used language for simulating the discrete system. The language can be considered as more than just a simulation language since it can be applied to general programming problems. The description of the language given is organized in five levels.

1. Beginning with simple teaching, level to introduce the concept of programming.
2. The description add level corresponding to a specific programming language.
3. A general-purpose language.
4. A list processing language, needed for creating the data structure of a simulation model.
5. The simulation-oriented function needed to control the simulation and gathering statistics.

SIMSCRIPT System Concept:

The system to be simulated is considered to consist of entities having attributes that interact with activities. The interaction causes events that change the state of the system. In the system, SIMSCRIPT uses the term entities and attributes. To make the program efficiency it must distinguish between temporary and permanent entities and attributes.

The temporary entities represent those entities are created and destroyed during the execution of a simulation while permanent entities represent that entities remain during the run.

A special emphasis is placed on how temporary entities form sets. The user can define sets and facilities are provided for entering and removing entities into and from sets.

Activities are considered as extending overtime with their beginning and being marked as events occurring instantaneously. Each type of event is described by an event routine, each of which is given a name and a program as a separate closed routine.

A distinction is made between the indigenous (internal) event, which arises from action within the system and the exogenous (external) event, which arises from the action in the system environment.

An indigenous event is caused by a scheduling statement in some event routine. The event marking the beginning of activity will usually schedule the event that marks the end of the activity.

If the beginning or end of activity implies the beginning of some activity, then the event routine marking the beginning or end of the first activity will schedule the beginning of the second.

An exogenous event requires the reading of data supplied by the user. Among the data is time at which, the event occurs. There can be many data sets representing a different set of external events.

Events routines are needed to execute the changes that result when an external event becomes due for execution. Part of the automation initialization procedure of SIMSCRIPT is to prepare the first exogenous event for each data set.

In practice, external event, such as arrivals are often generated using the bootstrap method. Given the statistical distribution of the inter-arrival time, an indigenous event routine continuously creates one arrival from its predecessor. An exogenous event routine is essential only when specific data are needed.

Organization of SIMSCRIPT Program:

Since the event routines are closed routines, some means must be provided for control between them. The transfer is affected by the use of event notice which is created when it is determined that an event is scheduled. At all time, an event notice exists for every indigenous event scheduled to occur either at current clock time or in the future event.

Each event notice records the time is due to occur and the event routine that is to execute the event. If the event is to involve one of the temporary entities, the event notice will usually identify which one is involved.

The event notice is filed in chronological order. When all events that can be executed at a particular time have been processed, the clock is updated to the time of the next event notice and control is passed to the event routine identified by the notice. Their actions are automatic and do not need to be programmed and event notice do not usually go to more than one activity.

If the event executed by a routine result in another event, either at the current clock time or in future, the routine must create a new event notice and file it with the other notice.

In the case of the exogenous event, a series of exogenous event statement are created one for each event. All exogenous event statements are sorted to chronological order and they are read by the programs when the time for the event is due to occur.

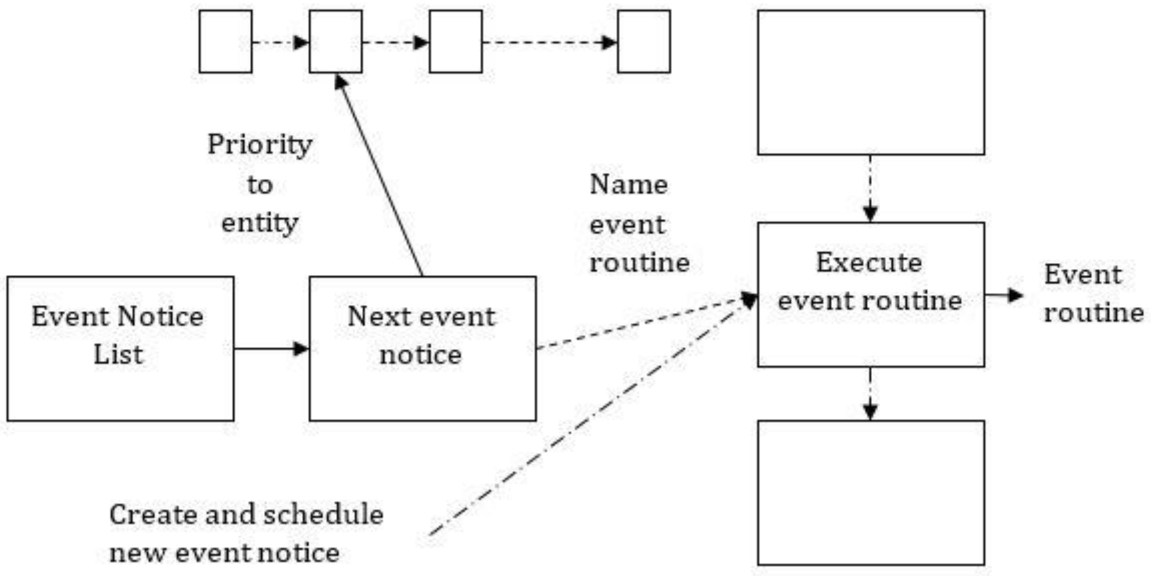
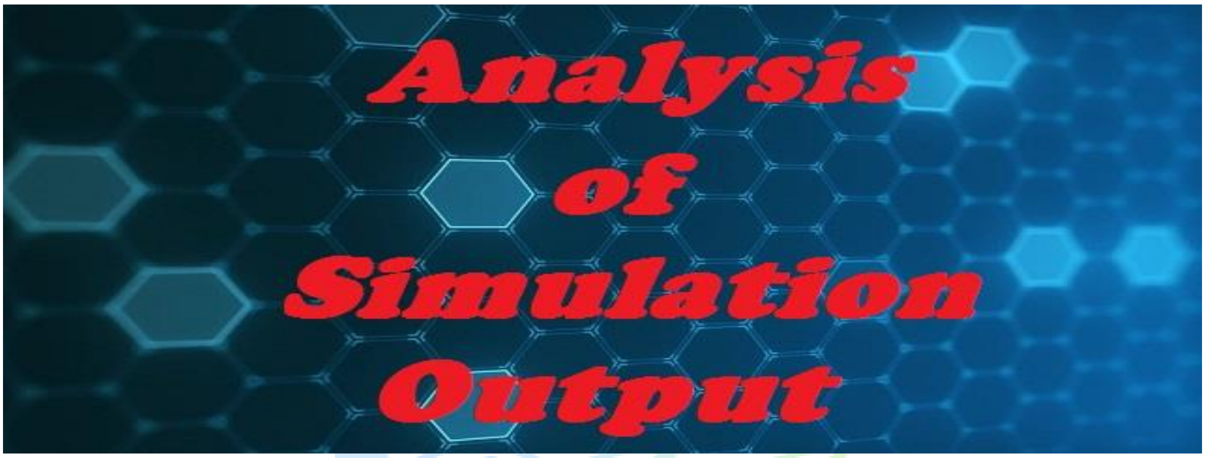


Fig: Organization of SIMSCRIPT Program

Unit VII: Analysis of Simulation Output - Simulation and Modeling

Analysis of Simulation Output:

The goal in analyzing output data from running a simulation model is to make a valid statistical inference about the initial and long-term average behavior of the system based on the sample average from N replicate simulation runs.



Output Analysis is the analysis of data generated by a simulation run to predict system performance or compare the performance of two or more system designs. In stochastic simulations, multiple runs are always necessary. The output of a single run can be viewed as a sample of size 1.

Output analysis is needed because output data from a simulation exhibits random variability when random number generators are used. i.e., two different random number streams will produce two sets of output which (probably) will differ. The statistical tool mainly used is the confidence interval for the mean.

For most simulations, the output data are correlated and the processes are non-stationary. The statistical (output) analysis determines:

a. The estimate of the mean and variance of random variables.
b. The number of observations required to achieve a desired precision of these estimates.

Nature of the Problem:

Once a stochastic variable has been introduced into a simulation model, almost all the system variables describing the system behavior also become stochastic. The values of most of the system variables will fluctuate as the simulation proceeds so that no one measurement can be taken to represent the values of a variable.

Instead many observations of the variable values must be made in order to make a statistical estimate of its true values. Some statement must also be made about the probability of the true value falling within the given interval about the estimated value. Such a statement defines a confidence interval, without it simulation result are of little value to the system analyst.

A large body statistical method has been developing over the years to analyze results in science, engineering and other fields where experimental observation is made. So, because of the experimental measurements of the system of simulation for these statistical methods can be adapted to simulation results to analyze.

Newly Developing Statistical Methodology Concerns:

1. To ensure that the statistical estimates are consistent, meaning that as the sample size increases the estimate tends to a true value.

2. To control biasing in measure of both new values of variance. Bias causes the distinction of an estimate to differ significantly from the true population statistics, even though the estimate may be consistent.

3. To develop sequential testing methods, to determine how long a simulation should be run in order to obtain confidence in its return.

Estimation Method:

Statistical methods are commonly used on the random variable. Usually, a random variable is drawn from an infinite population with a finite mean 'μ' and finite variance 'σ2'. These random variables are independently and identically distributed (i.e. IID variables).

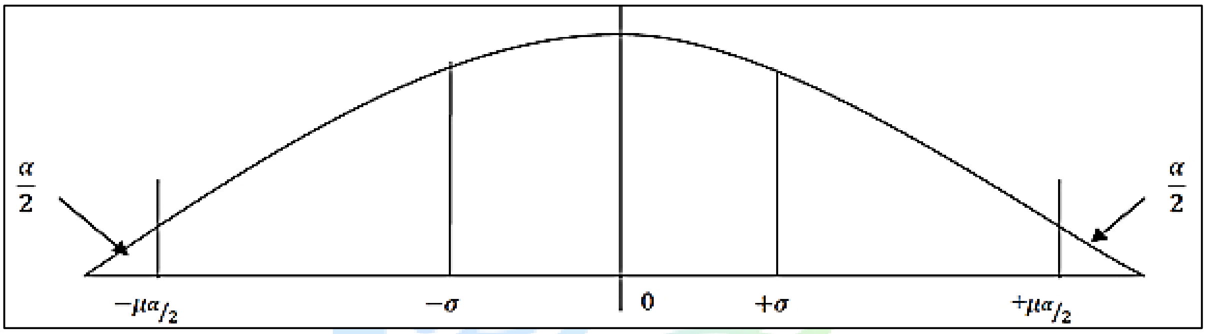
Let, xi=iid random variables. (i = 1, 2…, n), then according to central limit theorem and applying transformation, approximate normal variance,

$$Z = \frac{\sum_{i=1}^{n} x_i - n\mu}{\sqrt{n}\sigma}$$

$$Z = \frac{\bar{x} - \mu}{\sigma/\sqrt{n}}$$

Where, $\bar{x}$ = sample mean

a. It can be shown to be a consistent estimator for the mean of the population from which the sample is done.
b. Since the sample mean is some of the random variables, it is itself a random variable. So, a confidence interval about its computed value needs to be established.
c. The probability density function on the standard normal variable (Z) is shown in the figure below.



The integral formed $-\infty$ to μ is the probability that Z is less than or equal to μ and is denoted by $\emptyset(u)$.

Let us consider the value of $u(u_{\alpha/2})$ such that $\emptyset(u) = 1 - \alpha/2$ where $\alpha =$ some constant < 1. Then probability of Z for $Z > u_{\alpha/2} = \alpha/2$.

Probability of Z for $\left(-u_{\frac{\alpha}{2}} \leq Z \leq +u_{\frac{\alpha}{2}}\right) = 1 - \alpha$.

In terms of sample mean μ, the probability statement can be written as:

Prob $\left\{\bar{x} + \frac{\sigma}{\sqrt{n}} u_{\frac{\alpha}{2}} \geq \mu \geq \bar{x} - \frac{\sigma}{\sqrt{n}} u_{\frac{\alpha}{2}}\right\} = 1 - \alpha.$

Here, the constant $1 - \alpha$ is a confidence level (usually expressed in percent (%)) and the interval $\bar{x} \pm \frac{\sigma}{\sqrt{n}} u_{\alpha/2}$ is the confidence interval.

Estimation population variance s^2 (not σ^2 which is actual population variance) is given as: $S^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2$.

Replacing $u_{a/2}$ by $t_n - 2, {}^{\alpha}/_{2}$, then the estimated variance s^2, the confidence interval for $\bar{x}$ is given as: $C = \bar{x} \pm \frac{s}{\sqrt{n}} t_n - 1, {}^{a}/_{2}$

Simulation Run Statistics:

On every simulation run, some statistic is measure based on some assumption; for example: on establishing confidence interval it is assumed that the observation is mutually independent and distinction from which they are drawn is stationary. But many statistics are interesting in simulation don't meet this condition.

Let us illustrate the problems that arise in measuring statistic from a simulation run with the example of a single server system.

Consider the occurrence of arrivals has a Poisson distribution:

a. The service time has an exponential distribution.
b. The queuing discipline is FIFO
c. The inter-arrival time is distributed exponentially
d. System has a single server.

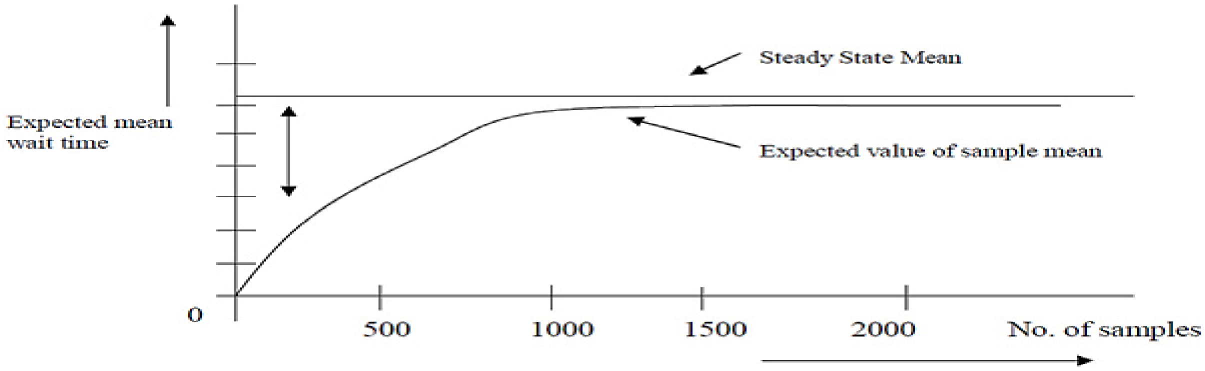
Then in a simulation run, the simplest way to estimate the mean waiting is to accumulate the waiting time of n successive entities and dividing it by 'n'. This gives sample mean denoted by $\bar{x}(n)$. If x_i (i =1, 2, 3… n) are the individual waiting times, then $\bar{x}(n) = \frac{1}{n}\sum_{n-1}^{n} x_i$

The 1st problem, here is that, the waiting times measures this way are not independent because whenever a waiting line forms, the waiting time of each entity on the line depends upon the waiting time of its predecessor (i.e. the entities are auto co-related).

The usual formula for estimating the mean value of the distribution remains on satisfactory estimate for the mean of auto co-related data. However, the variance of auto-correlated data is not related to the population variance by simple expression $\frac{\sigma^2}{n}$ as occurs for independent data.

The 2nd problem is that the distribution may not be stationary; it is because a simulation run is started with the system in some initial state, frequently the idle state, in which no service is being given and no entities are waiting, thus the early arrivals have a more probability of obtaining service quickly.

So, a sample means that includes the early arrivals will be biased. As the length of the simulation run extended and the sample size increases, the effect of bias will be minimum. This is shown in the figure below.



Replication of Runs:

One problem in measuring the statistic in the simulation run is that the results are dependent. But it is required, in simulation, to get the independent result. The one way of obtaining the independent result is to repeat the simulation.

Repeating the experiment with different random numbers for the same sample size 'n' gives a set of an independent determination of sample mean $\overline{x}$ (n).

Even though the distribution of the sample means depends upon the degree of autocorrelation, this independent determination of sample mean can be used to estimate the variance of the distribution.

Suppose,

Experiment is repeated p-times with independent random numbers. $x_{ij} = i^{th}$ observation of j^{th} run. Then,

Estimates for sample mean $\overline{x}(n) = \frac{1}{n}\sum_{i=1}^{n} x_{ij}$

Estimate for variance for j^{th} run $S^2j(n)$ is $s^2j(x) = \frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \overline{x}_n(n))^2$

Now, combining the results of 'p' independent measurement gives the following estimate for the mean waiting time $\overline{x}$ and variance s^2:

$$\overline{x} = \frac{1}{p}\sum_{i=1}^{p} \overline{x}_j\,(n)$$

$$s^2 = \frac{1}{p}\sum_{j=1}^{p} s^2j(n)$$

Here, the value of x¯ is an estimate for mean waiting time and the value of s^2 can be used to establish the confidence of intervals.

Elimination of Initial Bias:

There two general approaches that can be used to remove the initial bias:

1. The system can be started in more representative states rather than in the empty state.
2. The first part of the simulation run can be ignored.

In the first approach, it is necessary to know the steady-state distinction for the system and we then select the initial state distinction. In the study of simulation, particularly the existing system, there may be information available on the expected condition which makes it feasible to select a better initial condition and thus eliminating the initial bias.

The second approach that is used to remove the initial bias is the most common approach. In this method, the initial section of the run which has a high bias (simulation) result is eliminated.

First, the run is started from an idle state and stopped after a certain period of time (the time at which the bias is satisfactory). The entities existing in the system at that are left as they are and this point is the point of a restart for other repeating simulation runs.

Then the run is restarted with statistics being gathered from the point of the restart. These approaches have the following difficulties:

1. No simple rules can be given to deciding how long an interval should be eliminated. For this, we have to use some pilot run starting from the ideal state to judge how long the initial bias remains. These can be done by plotting the measured, statistics against the run length.

2. Another disadvantage of eliminating the first part of the simulation run is that the estimate of variance will be based on less information affecting the establishment of confidence limit. These will then cause to increase in confidence internal size.