CHAPTER 1

Overview of Computer Simulation

The wise man is one who knows what he does not know.

—Tao Te Ching

1.1 INTRODUCTION

Richmond [2003] defines thinking as "constructing mental models and then simulating them in order to draw conclusions or make decisions." Namely, he defines thinking as mental simulation. When the situation is too complex to be analyzed by mental simulation alone, we rely on computer simulation. According to Schruben [2012], simulation models provide unlimited virtual power: "If you can think of something, you can simulate it. Experimenting in a simulated world, you can change anything, in any way, at any time—even change time itself."

Fishwick [1995] defines *computer simulation* as the discipline of designing a model of a system, simulating the model on a digital computer, and analyzing the execution output. In the military, where computer simulation is extensively used in training personnel (e.g., war game simulation) and acquiring weapon systems (e.g., simulation-based acquisition), the term *modeling and simulation* (M&S) is used in place of *computer simulation*. In this book, these two terms are used interchangeably.

The purpose of this chapter is to provide the reader with a basic understanding of computer simulation. After studying this chapter, you should be able to answer the following questions:

1. What are the common characteristics that lead to a conceptual definition of system?
2. What are the three types of systems?
3. What are the three subsystems in a feedback control system?
4. What are the three types of virtual environment simulation?

Modeling and Simulation of Discrete-Event Systems, First Edition. Byoung Kyu Choi and Donghun Kang.

5. What are the three types of computer simulation?
6. What is the simulation model trajectory of a discrete-event system?
7. What is Monte Carlo simulation?
8. What is sensitivity analysis in simulation experimentation?

This chapter is organized as follows: Definitions and structures of systems are given in Section 1.2. Section 1.3 provides definitions and applications of simulation. The subsequent three sections introduce the three simulation types: discrete-event simulation in Section 1.4, continuous simulation in Section 1.5, and Monte Carlo simulation in Section 1.6. Finally, a basic framework of simulation experimentation is presented in Section 1.7.

1.2 WHAT IS A SYSTEM?

1.2.1 Definitions of Systems

Systems are encountered everywhere in the world. While those systems differ in their specifics, they share common characteristics that lead to a conceptual definition of a system. In Wu [1992], a *system* is defined as "a collection of components which are interrelated in an organized way and work together towards the accomplishment of certain logical and purposeful end." Thus, any portion of the real world may be defined as a system if it has the following characteristics: (1) it has a purpose or purposes, (2) its components are connected in an organized manner, and (3) they work together to achieve common objectives. A system consisting of people is often called a *team*. Needless to say, a mere crowd of people sharing no common objectives is not a team.

When defining a system, the concept of state variable plays a key role. A *state variable* is a particular measurable property of an object or system. Examples of state variables are the number of jobs in a buffer, status of a machine, temperature of an oven, etc. A system in which the state variables change instantaneously at discrete points in time is called a *discrete-event system*, whereas a system in which state variables change continuously over time is called a *continuous system*.

1.2.2 Three Types of Systems

Our universe, which is full of systems everywhere, may be viewed from the five levels of detail (Fig. 1.1): from the subatomic level to cosmological level. In the subatomic level, interactions among the components of a system are described using quantum mechanics, which is a physical science dealing with the behavior of matter and energy on the scale of atoms and subatomic particles. It is interesting to find that quantum mechanics is also used in modeling a system at the cosmological level [Mostafazadeh 2004]. Thus, a system in the subatomic level or cosmological level may be called a *quantum system*.

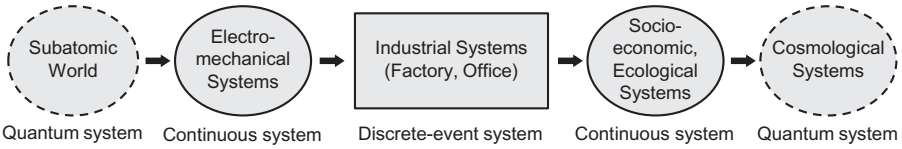

Fig. 1.1. Five levels of details of system definitions in the universe.

A system in the electromechanical level usually has components whose physical dynamics are described using differential equations of effort, such as force and voltage, and flow, such as velocity and current [Karnopp et al. 2000]. The behaviors of ecological systems and socioeconomic systems are usually described using differential equations of flow [Hannon and Ruth 2001]. As a result, these systems are called a *continuous system* or a *differential equation system*.

Systems in the middle level are industrial systems which are more conveniently described in terms of discrete events, and they are discrete-event systems. An event is an instance of changes in state variables. A special type of this system is a digital system such as a computer whose states are defined by a finite number of 0s and 1s.

1.2.3 System Boundaries and Hierarchical Structure

Everything in our world is connected to everything else in some way, which is known as the small world phenomenon [Kleinberg 2000]. Thus, in order to define a system, it is first necessary to isolate the components of the system from the remaining world and to enclose them within a system boundary.

A set of isolated components of primary interest is called a *target system*. The target system may have a number of subsystems, and it may be a subsystem of a higher-level system called a *wider system*. The wider system is separated from the external environment by a boundary [Wu 1992]. In summary, a typical system consists of a target system (composed of its subsystems) and a wider system (in which the target system is included). The system of interest consisting of a target system and its wider system is often referred to as a *source system*.

Most dynamic systems in engineering and management are feedback control systems. Key subsystems in a feedback control system are operational, monitoring, and decision-making subsystems. The operational subsystem carries out the system's tasks, and the monitoring subsystem monitors system performances and reports to the decision-making subsystem. The decision-making subsystem is responsible for making decisions and taking corrective actions. The relationships among the target feedback control system, its subsystems, wider system, and external environment are shown in Fig. 1.2 [Wu 1992]. For example, if your simulation study is focused on an emergency room of a hospital, the emergency room would become the target system and the hospital the wider system.

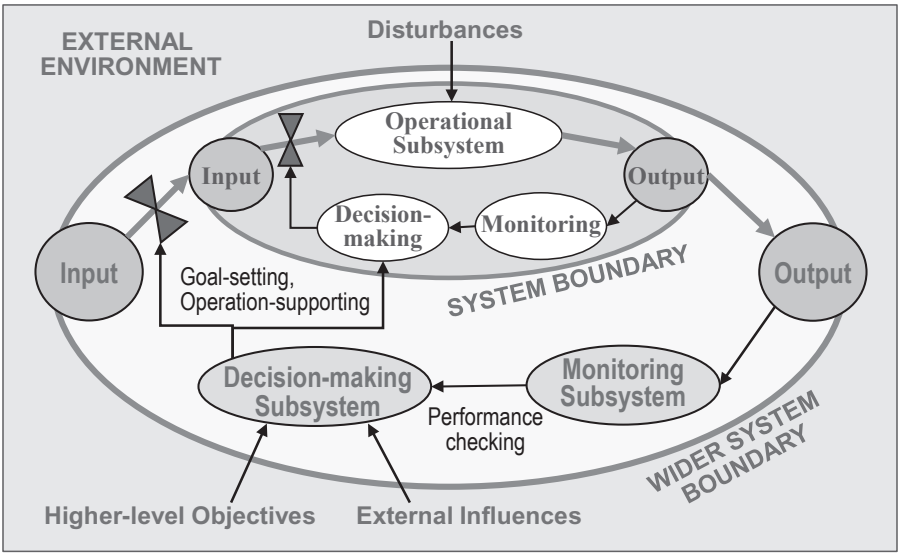

Fig. 1.2. Hierarchical structure of feedback control system.

The wider system influences the target system by setting goals, supporting operations, and checking performances. The target system is subject to disturbances from the external environment. In addition, the external environment provides the wider system with higher-level objectives and other external influences.

Exercise 1.1. Give an example of a feedback control system involving people and identify all the components of the system.

1.3 WHAT IS COMPUTER SIMULATION?

1.3.1 What Is Simulation?

A dictionary definition of *simulation* is "the technique of imitating the behavior of some situation by means of an analogous situation or apparatus to gain information more conveniently or to train (or entertain) personnel." "Some situation" in the definition corresponds to a source system, and an *apparatus* is a simulator. As elaborated in the definition, there are two types of simulation objectives: one is to gain information and the other is to train or entertain personnel. The former is often called an *analytic simulation* and the latter a *virtual environment simulation* [Fujimoto 2000].

The main purpose of an analytic simulation is the quantitative analysis of the source system based on "exact" data. Thus, the simulation should be executed in an as-fast-as-possible manner and be able to precisely reproduce the event sequence of the source system. An analytic simulation is often referred

Fig. 1.3. Examples of virtual environment simulation.

to as a *time-stamp simulation*. A virtual environment simulation is executed in a scaled real-time while creating virtual environments, and it is often referred to as a *time-delay simulation*. Shown in Fig. 1.3 are scenes from a war-game simulation and from a computer game.

An analytic simulation with human interaction is called a *constructive simulation*, and one without human interaction an *autonomous simulation.* If humans interact with the simulation as a participant, it is referred to as human-in-the-loop (HIL) simulation; if machines or software agents interact with the simulation, it is called a machine-in-the-loop (MIL) simulation. A virtual environment simulation without HIL/MIL is often called a *virtual simulation*; one with HIL only a *constructive simulation*; one with both HIL and MIL a *live simulation*. Figure 1.4 shows the classification of computer simulation.

1.3.2 Why Simulate?

Modeling and simulation is the central part of our thinking process. When the situation is too complex to be analyzed by mental simulation alone, we use a computer for simulating the situation. Let's consider the following situations:

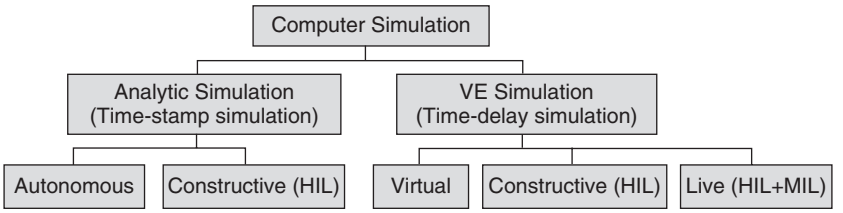

Fig. 1.4. Classification of computer simulation.

1. Finding optimal dispatching rules at a modern 300-mm semiconductor Fab
2. Evaluating alternative designs for hospitals, post offices, call centers, etc.
3. Designing the material handling system of a 3 billion dollar thin film transistor–liquid crystal display (TFT-LCD) Fab
4. Planning a wireless network for a telecommunication company
5. Evaluating high-tech weapons systems for a simulation-based acquisition
6. Designing or upgrading the urban traffic system of a big city
7. Evaluating anti-pollution policies to control pollutions in river systems
8. Evaluating risks in project schedules and financial derivatives

For the above real-life situations, simulation may be the only means to tackle the problems. In practice, simulation may be needed because experimenting with the real-life system is not feasible; your budget does not allow you to acquire an expensive prototype; a real test is risky; your customer wants it "yesterday"; your team wants to test several solutions and to compare them; you would like to keep a way to reproduce its performances later.

The simulation of a discrete-event system is called a *discrete-event simulation*, and that of a continuous system a *continuous simulation*. A class of computational schemes that rely on repeated random sampling to compute their results is referred to as *Monte Carlo simulation*. Among the above situations, Situations 1–6 are concerned with a discrete-event simulation. Situation 7 is concerned with a continuous simulation and Situation 8 with a Monte Carlo simulation.

1.3.3 Types of Computer Simulation

As depicted earlier in Fig. 1.1, the dynamic systems in the universe can be classified into five levels and three types. The three types of dynamic systems are: (1) discrete-event systems, (2) continuous systems, and (3) quantum systems. Thus, it is conceivable that there is one type of computer simulation for each system type. Discrete-event simulation and continuous simulation are widely performed on computers, but the direct simulation of quantum systems

on classical computers is very difficult because of the huge amount of memory required to store the explicit state of the system [Buluta and Nori 2009].

Continuous simulation is a numerical evaluation of a computer model of a physical dynamic system that continuously tracks system responses over time according to a set of equations typically involving differential equations. Let $\mathbf{Q}(t)$ and $\mathbf{X}(t)$ denote the system state and input trajectory vectors, respectively. Then, a linear continuous simulation is a numerical evaluation of the linear state transition function $d\mathbf{Q}(t)/dt = \mathbf{AQ}(t) + \mathbf{BX}(t)$, where $\mathbf{A}$ and $\mathbf{B}$ are coefficient matrices.

Discrete-event simulation is a computer evaluation of a discrete-event dynamic system model where the operation of the system is represented as a chronological sequence of events. In state-based modeling (see Chapter 9), the system dynamics is described by an internal state-transition function (δ_{int}: $Q \rightarrow Q$) and an external state-transition function (δ_{ext}: $Q \times X \rightarrow Q$), where Q is a set of system states and X is a set of input events. Thus, discrete-event simulation can be regarded as a computer evaluation of the internal and external transition functions.

Another type of popular computer simulation is the Monte Carlo simulation, which is not a dynamic system simulation. It is a class of computational algorithms that rely on repeated random sampling to compute the numerical integration of functions arising in engineering and science that are impossible to evaluate with direct analytical methods. In recent years, Monte Carlo simulation has also been used as a technique to understand the impact of risk and uncertainty in financial, project management, and other forecasting models.

1.4 WHAT IS DISCRETE-EVENT SIMULATION?

Figure 1.5 depicts a single server system consisting of a machine and a buffer in a factory. The dynamics of the system may be described as follows: (1) a job arrives at the system with an inter-arrival time of t_a, and the job is loaded on the machine if it is idle; otherwise, the job is put into the buffer; (2) the loaded job is processed for a service time of t_s and unloaded; (3) when a job is unloaded, the next job is loaded if the buffer is not empty. In Fig. 1.5, the state variables of the system are q and m, where q is the number of jobs in the buffer

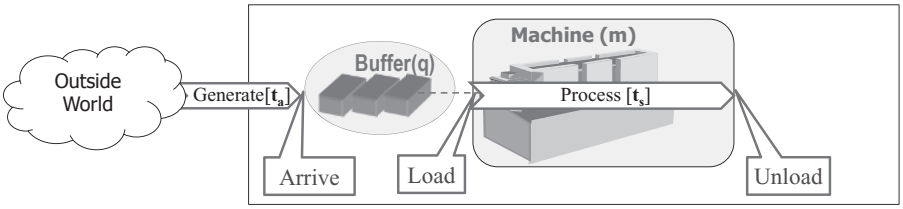

Fig. 1.5. A single server system model.

and m denotes the status (Idle or Busy) of the machine, and the events are Arrive, Load, and Unload.

1.4.1 Description of System Dynamics

Using the state variables and events, the system dynamics of the single server system may be described more rigorously as follows: (1) when an Arrive event occurs, q is increased by one, the next Arrive event is scheduled to occur after t_a time units, and a Load event is scheduled to occur immediately if m ≡ Idle(=0); (2) when a Load event occurs, q is decreased by one, m is set to Busy(=1), and an Unload event is scheduled to occur after t_s time units; (3) when an Unload event occurs, m is set to Idle and a Load event is scheduled to occur immediately if q > 0. The dynamics of the single server system may be described as a graph as given in Fig. 1.6, which is called an *event graph*.

1.4.2 Simulation Model Trajectory

An executable model of a system is called a *simulation model*, and the trajectory of the state variables of the model is called the *simulation model trajectory*. Let $\{a_k\}$ and $\{s_k\}$ denote the sequences of inter-arrival times (t_a) and service times (t_s), respectively. Then, the simulation model trajectory of the single server system would look like Fig. 1.7, where $\{t_i\}$ are event times, X(t) is input trajectory, and Q(t) = {q(t), m(t)} denotes the trajectory of the system

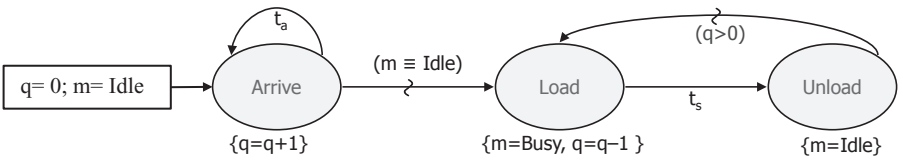

Fig. 1.6. Event graph describing the system dynamics of the single server system.

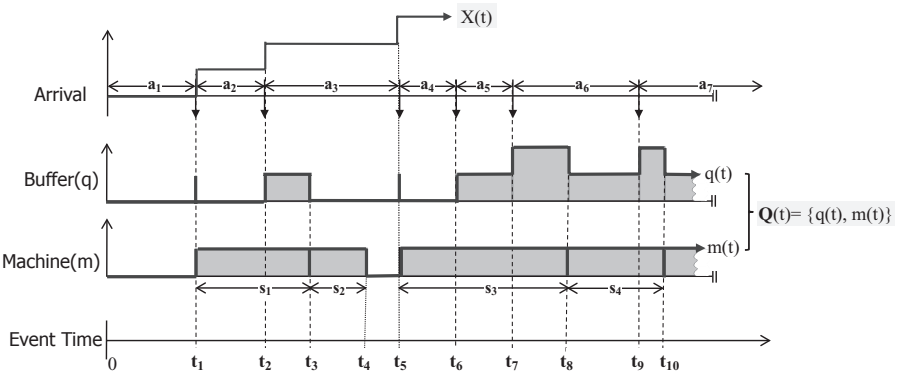

Fig. 1.7. Simulation model trajectory of the single server system.

state variables. The "time" here means a simulation time, which is a logical time used by the simulation model to represent physical time of the target system to be simulated.

At time t_1 (=a_1), a job J_1 arrives at an empty system and is loaded on the idle machine to be processed for a time period of s_1. In the meantime, another job J_2 arrives at time t_2 (=$a_1 + a_2$), which will be put into the buffer since the machine is busy. Thus, the buffer will have one job during the time period [t_2, t_3], which is denoted as a shaded bar in the buffer graph q(t) of Fig. 1.7. At t_3 (=$t_1 + s_1$), the first job J_1 is unloaded and the job J_2 in the buffer is loaded on the machine. At t_4 (=$t_3 + s_2$), J_2 is finished and unloaded, which will make the system empty again. Thus, the machine is busy during the time period [t_1, t_4]. At time t_5 (=$a_1 + a_2 + a_3$), another job J_3 arrives at the system and is loaded on the machine, and so on.

1.4.3 Collecting Statistics from the Model Trajectory

When simulating a service system, one may be interested in such items as (1) queue length, (2) waiting time distribution, (3) sojourn time, (4) server utilization, etc. In the case of the single server system, the following statistics can be collected from the model trajectory.

1. Queue length q(t) statistics during $t \in [t_0, t_{10}]$: AQL (average queue length)
   - AQL = $\{(t_3 - t_2) + (t_7 - t_6) + 2(t_8 - t_7) + (t_9 - t_8) + 2(t_{10} - t_9)\}/t_{10}$
2. Waiting time $\{W_j\}$ statistics for the first four jobs: AWT (average waiting time)
   - AWT = $\{W_1 + W_2 + W_3 + W_4\}/4 = \{0 + (t_3 - t_2) + 0 + (t_8 - t_6)\}/4 = (t_3 - t_2 + t_8 - t_6)/4$
3. Sojourn time $\{S_j\}$ statistics for the first four jobs: AST (average sojourn time)
   - AST = AWT + Average service time = AWT + $(s_1 + s_2 + s_3 + s_4)/4$
4. Server utilization during $t \in [t_0, t_{10}]$: U (utilization)
   - U = $\{(t_4 - t_1) + (t_{10} - t_5)\}/t_{10}$

1.5 WHAT IS CONTINUOUS SIMULATION?

As mentioned in Section 1.3.3, continuous simulation is a numerical evaluation of a computer model of a physical system that continuously tracks system responses over time, $\mathbf{Q}(t)$, according to a set of equations typically involving differential equations like $d\mathbf{Q}(t)/dt = f[\mathbf{Q}(t), \mathbf{X}(t)]$, where $\mathbf{X}(t)$ represents controls or input trajectory.

As an example, consider a Newtonian cooling model [Hannon and Ruth 2001]. Let $\sigma(t)$ be the cooling rate, then the temperature T(t) changes as

$dT(t)/dt = -\sigma(t)$. The cooling rate is expressed as $\sigma(t) = \kappa^*[T(t) - T_a]$, where κ is cooling constant and T_a is ambient temperature.

1.5.1 Manual Simulation of the Newtonian Cooling Model

The governing differential equation may be approximated by the following difference equation:

$$T(t + \Delta t) = T(t) - \sigma(t)^*\Delta t = T(t) - \kappa^*[T(t) - T_a]^*\Delta t, \text{ for } t = 0, \Delta t, 2\Delta t, 3\Delta t$$

Let's assume $T(0) = 37°C, T_a = 10°C, \kappa = 0.06$, and $\Delta t = 0.1$, then the temperature curve T(t) may be evaluated as follows:

$$T(0.1) = T(0) - 0.06^*[T(0) - 10]^*0.1 = 37 - 0.06^*(37 - 10)^*0.1 = 37 - 0.162$$
$$= 36.838$$
$$T(0.2) = T(0.1) - 0.06^*[T(0.1) - 10]^*0.1 = 36.838 - 0.06^*(36.838 - 10)^*0.1$$
$$= 36.677$$
…

1.5.2 Simulation of the Newtonian Cooling Model Using a Simulator

The cooling model may be simulated by using a commercial simulator such as STELLA®, as depicted in Fig. 1.8. In STELLA®, the level of state variable is regarded as a *stock* and the change in state variable as *flow*. In Fig. 1.8, TEMPERATURE is a stock and COOLING-RATE is a flow. COOLING CONSTANT and AMBIENT TEMPERATURE are parameters. These and other data are provided to the simulator via dialog boxes.

1.6 WHAT IS MONTE CARLO SIMULATION?

Monte Carlo simulation methods are a class of computational algorithms that rely on repeated random sampling to compute their results. They were developed for performing numerical integration of functions arising in engineering and science that were difficult to evaluate with direct analytical methods. In recent years, Monte Carlo simulation has also been used as a technique to understand the impact of risk and uncertainty in financial, project management, and other forecasting models.

1.6.1 Numerical Integration via Monte Carlo Simulation

As an example of numerical integration, consider the problem of finding the value of π via simulation. I am sure you have memorized the value of π as 3.14159. . .: but, for the moment, assume that you do not remember the value.

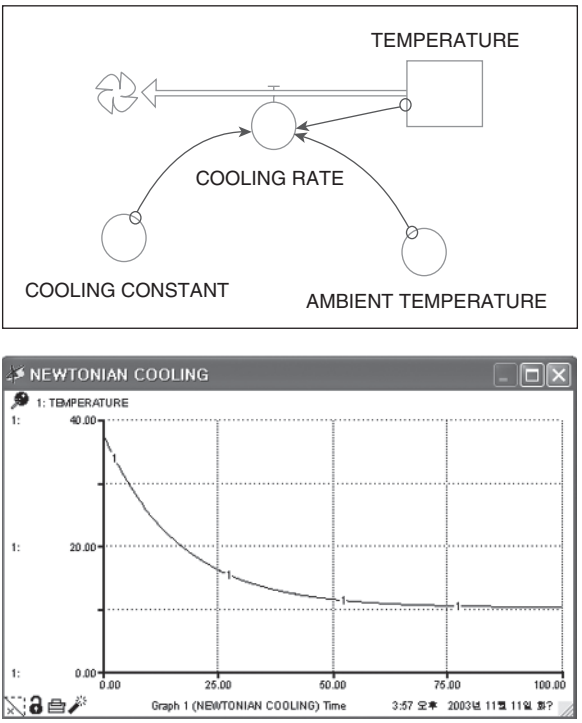

Fig. 1.8. STELLA® block-diagram modeling and output plot of the cooling system.

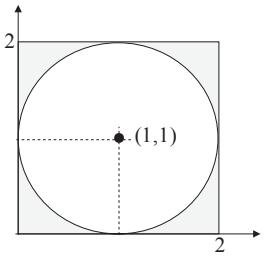

Fig. 1.9. A circle of unit radius to compute the value of π via Monte Carlo simulation.

In order to obtain the value of π via a Monte Carlo simulation, let's consider the circle shown in Fig. 1.9. It is a circle with a unit radius ($r = 1$) and its center is located at (1, 1). Uniform random variables with a range of [0, 2] are generated in pairs and are used as coordinates of points inside the square. Let n = total number of points generated (i.e., inside the square) and m = number of points inside the circle, and let A_c and A_s denote the areas of the circle and square, respectively. Then, the value of m/n approaches to the ratio A_c/A_s for

a large n. Since we know that $A_c = \pi r^2 = \pi$ and $A_s = 4$, we can compute π from the following relation: $m/n = A_c/A_s = \pi/4 \rightarrow \pi = 4\ m/n$ [Pidd 2004].

For the reader who may be curious about the execution of the simple Monte Carlo simulation, Java codes for (1) generating uniform random numbers and (2) computing the value of π are given below.

(1) Java code for generating uniform random number U ~ Uniform[0, 1]

```
double U = Math.random(); // Java function //
```

(2) Java code for finding the value of pi:

```
double m = 0, n = 0;
double max = 10000; // total number of sampling
while (n < max) {
   double u1 = Math.random();
   double u2 = Math.random();
   double x = 2.0 * u1;
   double y = 2.0 * u2;
   if ( ((x - 1) * (x - 1) + (y - 1) * (y - 1)) <= 1) m++;
   n++;
} // end of while
double phi = 4.0*m/n;
```

Exercise 1.2. Modify the above Monte Carlo simulation program (Java code) to compute the shaded area under the piece-wise linear function in Fig. 1.10.

1.6.2 Risk Analysis via Monte Carlo Simulation

Consider a project consisting of three tasks[1]: Task1, Task2, and Task3. Estimates of the time durations for the individual tasks are given in Table 1.1. We are interested in estimating the risk (or chance) of failing to meet a given project duration, say 15 months.

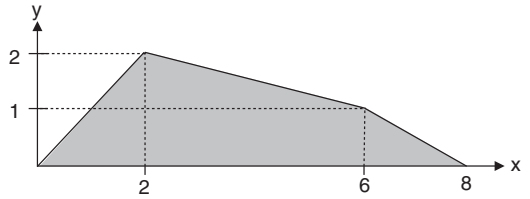

Fig. 1.10. Area under a piece-wise linear function.

[1]This example was taken from www.riskamp.com.

TABLE 1.1. Range Estimates for Individual Tasks

Task	Min (most optimistic)	Most likely	Max (most pessimistic)
Task1	4 months	5 months	7 months
Task2	3 months	4 months	6 months
Task3	4 months	5 months	6 months
Total	11 months	14 months	19 months

TABLE 1.2. Results of 500 Simulation Runs

Time duration (months)	12	13	14	15	16	17	18
# of on-time finishes	1	31	171	394	482	499	500
% of on-time finishes	0%	6%	34%	79%	96%	100%	100%

It is well accepted that the duration times are assumed to follow beta distribution (see Chapter 3). In the Monte Carlo simulation, values for the task duration times are randomly generated from respective beta distributions. The results of 500 simulation runs are summarized in Table 1.2, from which one may conclude that the risk of failing to finish the project within 15 months is about 20%. In recent years, Monte Carlo methods are quite popular in financial derivatives and option pricing evaluations.

1.7 WHAT ARE SIMULATION EXPERIMENTATION AND OPTIMIZATION?

The rules that govern the behavior of the system are called *laws*, while the rules under our control are called *policies*. When we experiment to determine the effects of changing the parameters of laws, we are doing a sensitivity analysis. When we experiment with changes in the control factors of policies, we are doing optimization [Schruben and Schruben 2001]. Both the parameters of laws and control factors of policies become handles of simulation experimentation. Both the optimization and sensitivity analysis may be performed in a simulation study. A simulation study should be carried out with

1. clear objectives of the study together with a set of performance measures;
2. output variables that can be mapped into the performance measures;
3. well-defined handles with which the simulation runs are to be controlled.

An experimental frame is a specification of the conditions under which the simulator is experimented with [Zeigler et al. 2000], and it is concerned with simulation optimization. As shown in Fig. 1.11, an experimental frame for simulation optimization consists of five steps: (1) an initial value of each

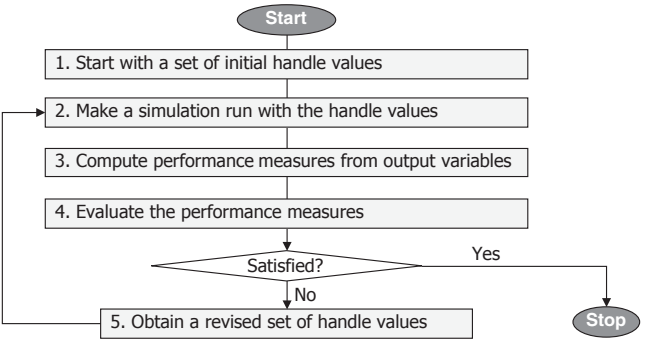

Fig. 1.11. Experimental frame for simulation optimization.

handle is generated; (2) a simulation run is made to compute values of the output variables; (3) performance measures are computed from the output variables; (4) the performance measures are evaluated to see if the results are acceptable; (5) if the results are not acceptable, go back to Step 2 with a revised set of handle values. Steps 3, 4, and 5 are often called *transducer*, *acceptor*, and *generator*, respectively.

1.8 REVIEW QUESTIONS

1.1. What are the common characteristics that lead to a conceptual definition of system?

1.2. Give a definition of a team based on the concept of system.

1.3. What is the difference between a source system and a target system?

1.4. What are the three key subsystems in a feedback control system?

1.5. What is an analytic simulation?

1.6. What is time-stamp simulation?

1.7. What would be the two popular areas where virtual environment simulation is used?

1.8. What is constructive simulation?

1.9. What is the main output from a continuous simulation?

1.10. In simulation, a rule under our control is called a policy. What is a law?

1.11. What is sensitivity analysis in simulation experimentation?

1.12. What is simulation optimization?

1.13. What is the role of the acceptor in an experimental frame?