5 Simulation Modeling

Introduction

In many situations a modeler is unable to construct an analytic (symbolic) model adequately explaining the behavior being observed because of its complexity or the intractability of the proposed explicative model. Yet if it is necessary to make predictions about the behavior, the modeler may conduct experiments (or gather data) to investigate the relationship between the dependent variable(s) and selected values of the independent variable(s) within some range. We constructed empirical models based on collected data in Chapter 4. To collect the data, the modeler may observe the behavior directly. In other instances, the behavior might be duplicated (possibly in a scaled-down version) under controlled conditions, as we will do when predicting the size of craters in Section 14.4.

In some circumstances, it may not be feasible either to observe the behavior directly or to conduct experiments. For instance, consider the service provided by a system of elevators during morning rush hour. After identifying an appropriate problem and defining what is meant by good service, we might suggest some alternative delivery schemes, such as assigning elevators to even and odd floors or using express elevators. Theoretically, each alternative could be tested for some period of time to determine which one provided the best service for particular arrival and destination patterns of the customers. However, such a procedure would probably be very disruptive because it would be necessary to harass the customers constantly as the required statistics were collected. Moreover, the customers would become very confused because the elevator delivery system would keep changing. Another problem concerns testing alternative schemes for controlling automobile traffic in a large city. It would be impractical to constantly change directions of the one-way streets and the distribution of traffic signals to conduct tests.

In still other situations, the system for which alternative procedures need to be tested *may not even exist yet*. An example is the situation of several proposed communications networks, with the problem of determining which is best for a given office building. Still another example is the problem of determining locations of machines in a new industrial plant. The *cost* of conducting experiments may be prohibitive. This is the case when an agency tries to predict the effects of various alternatives for protecting and evacuating the population in case of failure of a nuclear power plant.

In cases where the behavior cannot be explained analytically or data collected directly, the modeler might *simulate* the behavior indirectly in some manner and then test the various alternatives under consideration to estimate how each affects the behavior. Data can then be collected to determine which alternative is best. An example is to determine the drag force on a proposed submarine. Because it is infeasible to build a prototype, we can build a scaled model to simulate the behavior of the actual submarine. Another example of this type of simulation is using a scaled model of a jet airplane in a wind tunnel to estimate the effects of very high speeds for various designs of the aircraft. There is yet another type of simulation, which we will study in this chapter. This **Monte Carlo simulation** is typically accomplished with the aid of a computer.

Suppose we are investigating the service provided by a system of elevators at morning rush hour. In Monte Carlo simulation, the arrival of customers at the elevators during the hour and the destination floors they select need to be replicated. That is, the distribution of arrival times and the distribution of floors desired on the simulated trial must portray a possible rush hour. Moreover, after we have simulated many trials, the daily distribution of arrivals and destinations that occur must mimic the real-world distributions in proper proportions. When we are satisfied that the behavior is adequately duplicated, we can investigate various alternative strategies for operating the elevators. Using a large number of trials, we can gather appropriate statistics, such as the average total delivery time of a customer or the length of the longest queue. These statistics can help determine the best strategy for operating the elevator system.

This chapter provides a brief introduction to Monte Carlo simulation. Additional studies in probability and statistics are required to delve into the intricacies of computer simulation and understand its appropriate uses. Nevertheless, you will gain some appreciation of this powerful component of mathematical modeling. Keep in mind that there is a danger in placing too much confidence in the predictions resulting from a simulation, especially if the assumptions inherent in the simulation are not clearly stated. Moreover, the appearance of using large amounts of data and huge amounts of computer time, coupled with the fact the lay people can understand a simulation model and computer output with relative ease, often leads to overconfidence in the results.

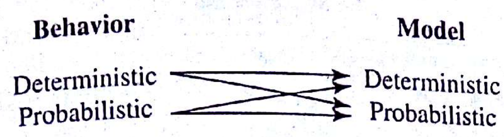
When any Monte Carlo simulation is performed, random numbers are used. We discuss how to generate random numbers in Section 5.2. Loosely speaking, a "sequence of random numbers uniformly distributed in an interval m to n" is a set of numbers with no apparent pattern, where each number between m and n can appear with equal likelihood. For example, if you toss a six-sided die 100 times and write down the number showing on the die each time, you will have written down a sequence of 100 random integers approximately uniformly distributed over the interval 1 to 6. Now, suppose that random numbers consisting of six digits can be generated. The tossing of a coin can be duplicated by generating a random number and assigning it a head if the random number is even and a tail if the random number is odd. If this trial is replicated a large number of times, you would expect heads to occur about 50% of the time. However, there is an element of chance involved. It is possible that a run of 100 trials could produce 51 heads and that the next 10 trials could produce all heads (although this is not very likely). Thus, the estimate with 110 trials would actually be worse than the estimate with 100 trials. Processes with an element of chance involved are called **probabilistic**, as opposed to **deterministic**, processes. Monte Carlo simulation is therefore a probabilistic model.

The modeled behavior may be either deterministic or probabilistic. For instance, the area under a curve is deterministic (even though it may be impossible to find it precisely). On the other hand, the time between arrivals of customers at the elevator on a particular day is probabilistic behavior. Referring to Figure 5.1, we see that a deterministic model can be used to approximate either a deterministic or a probabilistic behavior, and likewise, a Monte Carlo simulation can be used to approximate a deterministic behavior (as you will see with

■ Figure 5.1
The behavior can be either ... or probabilist...

5.1

Figure 5.1 The behavior and the model can be either deterministic or probabilistic.



a Monte Carlo approximation to an area under a curve) or a probabilistic one. However, as we would expect, the real power of Monte Carlo simulation lies in modeling a probabilistic behavior.

A principal advantage of Monte Carlo simulation is the relative ease with which it can sometimes be used to approximate very complex probabilistic systems. Additionally, Monte Carlo simulation provides performance estimation over a wide range of conditions rather than a very restricted range as often required by an analytic model. Furthermore, because a particular submodel can be changed rather easily in a Monte Carlo simulation (such as the arrival and destination patterns of customers at the elevators), there is the potential of conducting a sensitivity analysis. Still another advantage is that the modeler has control over the level of detail in a simulation. For example, a very long time frame can be compressed or a small time frame expanded, giving a great advantage over experimental models. Finally, there are very powerful, high-level simulation languages (such as GPSS, GASP, PROLOG, SIMAN, SLAM, and DYNAMO) that eliminate much of the tedious labor in constructing a simulation model.

On the negative side, simulation models are typically expensive to develop and operate. They may require many hours to construct and large amounts of computer time and memory to run. Another disadvantage is that the probabilistic nature of the simulation model limits the conclusions that can be drawn from a particular run unless a sensitivity analysis is conducted. Such an analysis often requires many more runs just to consider a small number of combinations of conditions that can occur in the various submodels. This limitation then forces the modeler to estimate which combination might occur for a particular set of conditions.

5.1 Simulating Deterministic Behavior: Area Under a Curve

In this section we illustrate the use of Monte Carlo simulation to model a deterministic behavior, the area under a curve. We begin by finding an approximate value to the area under a nonnegative curve. Specifically, suppose $y = f(x)$ is some given continuous function satisfying $0 \le f(x) \le M$ over the closed interval $a \le x \le b$. Here, the number M is simply some constant that *bounds* the function. This situation is depicted in Figure 5.2. Notice that the area we seek is wholly contained within the rectangular region of height M and length $b - a$ (the length of the interval over which f is defined).

Now we select a point $P(x, y)$ at random from within the rectangular region. We will do so by generating two random numbers, x and y, satisfying $a \le x \le b$ and $0 \le y \le M$, and interpreting them as a point P with coordinates x and y. Once $P(x, y)$ is selected, we ask whether it lies within the region below the curve. That is, does the y-coordinate satisfy $0 \le y \le f(x)$? If the answer is yes, then count the point P by adding 1 to some counter.

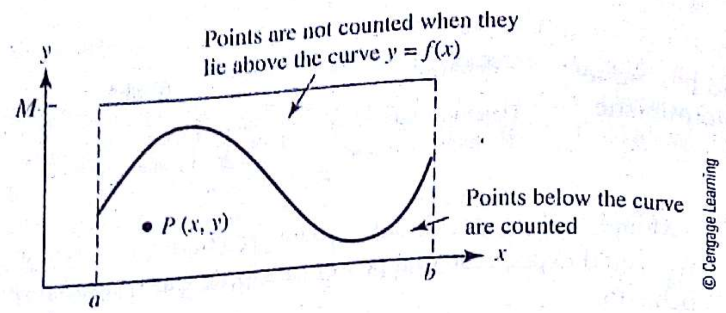


Figure 5.2
The area under the nonnegative curve $y = f(x)$ over $a \leq x \leq b$ is contained within the rectangle of height M and base length $b - a$.

Two counters will be necessary: one to count the total points generated and a second to count those points that lie below the curve (Figure 5.2). You can then calculate an approximate value for the area under the curve by the following formula:

$$\frac{\text{area under curve}}{\text{area of rectangle}} \approx \frac{\text{number of points counted below curve}}{\text{total number of random points}}$$

As discussed in the Introduction, the Monte Carlo technique is probabilistic and typically requires a large number of trials before the deviation between the predicted and true values becomes small. A discussion of the number of trials needed to ensure a predetermined level of confidence in the final estimate requires a background in statistics. However, as a general rule, to double the accuracy of the result (i.e., to cut the expected error in half), about four times as many experiments are necessary.

The following algorithm gives the sequence of calculations needed for a general computer simulation of this Monte Carlo technique for finding the area under a curve.

Monte Carlo Area Algorithm

Input Total number n of random points to be generated in the simulation.

Output AREA = approximate area under the specified curve $y = f(x)$ over the given interval $a \leq x \leq b$, where $0 \leq f(x) < M$.

Step 1 Initialize: COUNTER = 0.

Step 2 For $i = 1, 2, \ldots, n$, do Steps 3–5.

Step 3 Calculate random coordinates x_i and y_i that satisfy $a \leq x_i \leq b$ and $0 \leq y_i < M$.

Step 4 Calculate $f(x_i)$ for the random x_i coordinate.

Step 5 If $y_i \leq f(x_i)$, then increment the COUNTER by 1. Otherwise, leave COUNTER as is.

Step 6 Calculate AREA = $M(b - a)$ COUNTER/n.

Step 7 OUTPUT (AREA)

STOP

Table 5.1 gives the results of several different simulations to obtain the area beneath the curve $y = \cos x$ over the interval $-\pi/2 \leq x \leq \pi/2$, where $0 \leq \cos x < 2$.

The actual area under the curve $y = \cos x$ over the given interval is 2 square units. Note that even with the relatively large number of points generated, the error is significant. For functions of one variable, the Monte Carlo technique is generally not competitive with quadrature techniques that you will learn in numerical analysis. The lack of an error bound and the difficulty in finding an upper bound M are disadvantages as well. Nevertheless, the

Table 5.1 Monte Carlo approximation to the area under the curve $y = \cos x$ over the interval $-\pi/2 \le x \le \pi/2$

Number of points	Approximation to area	Number of points	Approximation to area
100	2.07345	2000	1.94465
200	2.13628	3000	1.97711
300	2.01064	4000	1.99962
400	2.12058	5000	2.01429
500	2.04832	6000	2.02319
600	2.09440	8000	2.00669
700	2.02857	10000	2.00873
800	1.99491	15000	2.00978
900	1.99666	20000	2.01093
1000	1.96664	30000	2.01186

Monte Carlo technique can be extended to functions of several variables and becomes more practical in that situation.

Volume Under a Surface

Let's consider finding part of the volume of the sphere

$$x^2 + y^2 + z^2 \le 1$$

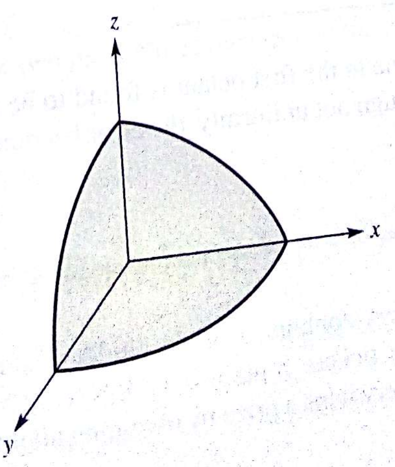
that lies in the first octant, $x > 0$, $y > 0$, $z > 0$ (Figure 5.3).

The methodology to approximate the volume is very similar to that of finding the area under a curve. However, now we will use an approximation for the volume under the surface by the following rule:

$$\frac{\text{volume under surface}}{\text{volume of box}} \approx \frac{\text{number of points counted below surface in 1st octant}}{\text{total number of points}}$$

The following algorithm gives the sequence of calculations required to employ Monte Carlo techniques to find the approximate volume of the region.

Figure 5.3
Volume of a sphere $x^2 + y^2 + z^2 \le 1$ that lies in the first octant, $x > 0$, $y > 0$, $z > 0$



Monte Carlo Volume Algorithm

Input Total number n of random points to be generated in the simulation.

Output VOLUME = approximate volume enclosed by the specified function, $z = f(x, y)$ in the first octant, $x > 0$, $y > 0$, $z > 0$.

Step 1 Initialize: COUNTER = 0.

Step 2 For $i = 1, 2, \ldots, n$, do Steps 3–5.

Step 3 Calculate random coordinates x_i, y_i, z_i that satisfy $0 \le x_i \le 1$, $0 \le y_i \le 1$, $0 \le z_i \le 1$. (In general, $a \le x_i \le b$, $c \le y_i \le d$, $0 \le z_i \le M$.)

Step 4 Calculate $f(x_i, y_i)$ for the random coordinate (x_i, y_i).

Step 5 If random $z_i \le f(x_i, y_i)$, then increment the COUNTER by 1. Otherwise, leave COUNTER as is.

Step 6 Calculate VOLUME $= M(d-c)(b-a)$COUNTER$/n$.

Step 7 OUTPUT (VOLUME)

STOP

Table 5.2 gives the results of several Monte Carlo runs to obtain the approximate volume of

$$x^2 + y^2 + z^2 \le 1$$

that lies in the first octant, $x > 0$, $y > 0$, $z > 0$.

Table 5.2 Monte Carlo approximation to the volume in the first octant under the surface $x^2 + y^2 + z^2 \le 1$

Number of points	Approximate volume
100	0.4700
200	0.5950
300	0.5030
500	0.5140
1,000	0.5180
2,000	0.5120
5,000	0.5180
10,000	0.5234
20,000	0.5242

The actual volume in the first octant is found to be approximately 0.5236 cubic units ($\pi/6$). Generally, though not uniformly, the error becomes smaller as the number of points generated increases.

5.1 PROBLEMS

1. Each ticket in a lottery contains a single "hidden" number according to the following scheme: 55% of the tickets contain a 1, 35% contain a 2, and 10% contain a 3. A participant in the lottery wins a prize by obtaining all three numbers 1, 2, and 3. Describe

an experiment that could be us
buy to win a prize.

2. Two record companies, A and B
label, and 5% of A's new comp
is manufactured under tighter
A, so only 2% of its compact d
B recording at your local store
used to determine how many
buying two warped compact d

3. Using Monte Carlo simulatio
by considering the number of

where the quarter circle is tal

Use the equation $\pi/4 =$ are

4. Use Monte Carlo simulation
the interval $\frac{1}{2} \le x \le \frac{3}{2}$.

5. Find the area trapped betwe
y-axes.

6. Using Monte Carlo simulat
of an ellipsoid

that lies in the first octant,

7. Using Monte Carlo simula
tween the two paraboloids

Note that the two parabol

5.2 Generating Ranc

In the previous section, we c
and volumes. A key ingredie
Random numbers have a va

an experiment that could be used to determine how many tickets you would expect to buy to win a prize.

2. Two record companies, A and B, produce classical music recordings. Label A is a budget label, and 5% of A's new compact discs exhibit significant degrees of warpage. Label B is manufactured under tighter quality control (and consequently more expensive) than A, so only 2% of its compact discs are warped. You purchase one label A and one label B recording at your local store on a regular basis. Describe an experiment that could be used to determine how many times you would expect to make such a purchase before buying two warped compact discs for a given sale.

3. Using Monte Carlo simulation, write an algorithm to calculate an approximation to π by considering the number of random points selected inside the quarter circle

$$Q : x^2 + y^2 = 1, x \geq 0, y \geq 0$$

where the quarter circle is taken to be inside the square

$$S : 0 \leq x \leq 1 \text{ and } 0 \leq y \leq 1$$

Use the equation $\pi/4 = \text{area } Q/\text{area } S$.

4. Use Monte Carlo simulation to approximate the area under the curve $f(x) = \sqrt{x}$, over the interval $\frac{1}{2} \leq x \leq \frac{3}{2}$.

5. Find the area trapped between the two curves $y = x^2$ and $y = 6 - x$ and the x- and y-axes.

6. Using Monte Carlo simulation, write an algorithm to calculate that part of the volume of an ellipsoid

$$\frac{x^2}{2} + \frac{y^2}{4} + \frac{z^2}{8} \leq 16$$

that lies in the first octant, $x > 0, y > 0, z > 0$.

7. Using Monte Carlo simulation, write an algorithm to calculate the volume trapped between the two paraboloids

$$z = 8 - x^2 - y^2 \quad \text{and} \quad z = x^2 + 3y^2$$

Note that the two paraboloids intersect on the elliptic cylinder

$$x^2 + 2y^2 = 4$$

5.2 Generating Random Numbers

In the previous section, we developed algorithms for Monte Carlo simulations to find areas and volumes. A key ingredient common to these algorithms is the need for random numbers. Random numbers have a variety of applications, including gambling problems, finding an area or volume, and modeling larger complex systems such as large-scale combat operations or air traffic control situations.

In some sense a computer does not really generate random numbers, because computers employ deterministic algorithms. However, we can generate sequences of pseudorandom numbers that, for all practical purposes, may be considered random. There is no single best random number generator or best test to ensure randomness.

There are complete courses of study for random numbers and simulations that cover in depth the methods and tests for pseudorandom number generators. Our purpose here is to introduce a few random number methods that can be utilized to generate sequences of numbers that are nearly random.

Many programming languages, such as Pascal and Basic, and other software (e.g., Minitab, MATLAB, and EXCEL) have built-in random number generators for user convenience.

Middle-Square Method

The middle-square method was developed in 1946 by John Von Neuman, S. Ulm, and N. Metropolis at Los Alamos Laboratories to simulate neutron collisions as part of the Manhattan Project. Their middle-square method works as follows:

1. Start with a four-digit number x_0, called the *seed*.
2. Square it to obtain an eight-digit number (add a leading zero if necessary).
3. Take the middle four digits as the next random number.

Continuing in this manner, we obtain a sequence that appears to be random over the integers from 0 to 9999. These integers can then be scaled to any interval a to b. For example, if we wanted numbers from 0 to 1, we would divide the four-digit numbers by 10,000. Let's illustrate the middle-square method.

Pick a seed, say $x_0 = 2041$, and square it (adding a leading zero) to get 04165681. The middle four digits give the next random number, 1656. Generating 13 random numbers in this way yields

n	0	1	2	3	4	5	6	7	8	9	10	11	12
x_n	2041	1656	7423	1009	0180	0324	1049	1004	80	64	40	16	2

We can use more than 4 digits if we wish, but we always take the middle number of digits equal to the number of digits in the seed. For example, if $x_0 = 653217$ (6 digits), its square 426,692,449,089 has 12 digits. Thus, take the middle 6 digits as the random number, namely, 692449.

The middle-square method is reasonable, but it has a major drawback in its tendency to degenerate to zero (where it will stay forever). With the seed 2041, the random sequence does seem to be approaching zero. How many numbers can be generated until we are almost at zero?

Linear Congruence

The linear congruence method was introduced by D. H. Lehmer in 1951, and a majority of pseudorandom numbers used today are based on this method. One advantage it has over other methods is that seeds can be selected that generate patterns that eventually cycle (we illustrate this concept with an example). However, the length of the cycle is so large that the pattern does not repeat itself on large computers for most applications. The method requires the choice of three integers: a, b, and c. Given some initial seed, say x_0, we generate a sequence by the rule

$$x_{n+1} = (a \times x_n + b)\,\text{mod}(c)$$

where c is the modulus, a is the multiplier, and b is the increment. The qualifier mod(c) in the equation means to obtain the remainder after dividing the quantity $(a \times x_n + b)$ by c. For example, with $a = 1$, $b = 7$, and $c = 10$,

$$x_{n+1} = (1 \times x_n + 7)\,\text{mod}(10)$$

means x_{n+1} is the integer remainder upon dividing $x_n + 7$ by 10. Thus, if $x_n = 115$, then $x_{n+1} = \textit{remainder}\ (\frac{122}{10}) = 2$.

Before investigating the linear congruence methodology, we need to discuss **cycling**, which is a major problem that occurs with random numbers. Cycling means the sequence repeats itself, and, although undesirable, it is unavoidable. At some point, all pseudorandom number generators begin to cycle. Let's illustrate cycling with an example.

If we set our seed at $x_0 = 7$, we find $x_1 = (1 \times 7 + 7)\,\text{mod}(10)$ or $14\,\text{mod}(10)$, which is 4. Repeating this same procedure, we obtain the sequence

$$7, 4, 1, 8, 5, 2, 9, 6, 3, 0, 7, 4, \ldots$$

and the original sequence repeats again and again. Note that there is cycling after 10 numbers. The methodology produces a sequence of integers between 0 and $c - 1$ inclusively before cycling (which includes the possible remainders after dividing the integers by c). Cycling is guaranteed with at most c numbers in the random number sequence. Nevertheless, c can be chosen to be very large, and a and b can be chosen in such a way as to obtain a full set of c numbers before cycling begins to occur. Many computers use $c = 2^{31}$ for the large value of c. Again, we can scale the random numbers to obtain a sequence between any limits a and b, as required.

A second problem that can occur with the linear congruence method is lack of statistical independence among the members in the list of random numbers. Any correlations between the nearest neighbors, the next-nearest neighbors, the third-nearest neighbors, and so forth are generally unacceptable. (Because we live in a three-dimensional world, third-nearest neighbor correlations can be particularly damaging in physical applications.) Pseudorandom number sequences can never be completely statistically independent because they are generated by a mathematical formula or algorithm. Nevertheless, the sequence will appear (for practical purposes) independent when it is subjected to certain statistical tests. These concerns are best addressed in a course in statistics.

a. Construct a Monte Carlo simulation for the back order submodel. If you have a calculator or computer available, test your submodel by running 1000 trials and comparing the number of occurrences of the various cancellations with the historical data.

b. Consider the algorithm you modified in Problem 1. Further modify the algorithm to consider back orders. Do you think back orders should be penalized in some fashion? If so, how would you do it?

5.4

PROJECTS

1. Complete the requirements of UMAP module 340, "The Poisson Random Process," by Carroll O. Wilde. Probability distributions are introduced to obtain practical information on random arrival patterns, interarrival times or gaps between arrivals, waiting line buildup, and service loss rates. The Poisson distribution, the exponential distribution, and Erlang's formulas are used. The module requires an introductory probability course, the ability to use summation notation, and basic concepts of the derivative and the integral from calculus. Prepare a 10-min summary of the module for a classroom presentation.

2. Assume a storage cost of $0.001 per gallon per day and a delivery charge of $500 per delivery. Construct a computer code of the algorithm you constructed in Problem 4, and compare various order points and order quantity strategies.

5.5 Queuing Models

EXAMPLE 1 *A Harbor System*

Consider a small harbor with unloading facilities for ships. Only one ship can be unloaded at any one time. Ships arrive for unloading of cargo at the harbor, and the time between the arrival of successive ships varies from 15 to 145 min. The unloading time required for a ship depends on the type and amount of cargo and varies from 45 to 90 min. We seek answers to the following questions:

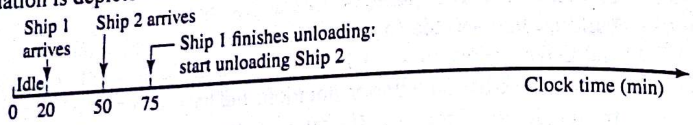
1. What are the average and maximum times per ship in the harbor?
2. If the *waiting time* for a ship is the time between its arrival and the start of unloading, what are the average and maximum waiting times per ship?
3. What percentage of the time are the unloading facilities idle?
4. What is the length of the longest queue?

To obtain some reasonable answers, we can simulate the activity in the harbor using a computer or programmable calculator. We assume the arrival times between successive ships and the unloading time per ship are uniformly distributed over their respective time intervals. For instance, the arrival time between ships can be any integer between 15 and 145, and any integer within that interval can appear with equal likelihood. Before giving a general algorithm to simulate the harbor system, let's consider a hypothetical situation with five ships.

We have the following data for each ship:

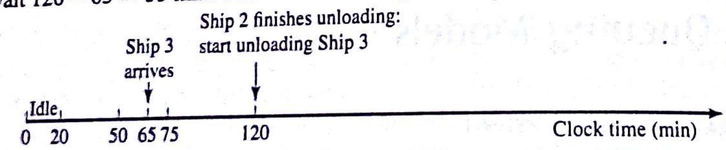
	Ship 1	Ship 2	Ship 3	Ship 4	Ship 5
Time between successive ships	20	30	15	120	25
Unloading time	55	45	60	75	80

Because Ship 1 arrives 20 min after the clock commences at $t = 0$ min, the harbor facilities are idle for 20 min at the start. Ship 1 immediately begins to unload. The unloading takes 55 min; meanwhile, Ship 2 arrives on the scene at $t = 20 + 30 = 50$ min after the clock begins. Ship 2 cannot start to unload until Ship 1 finishes unloading at $t = 20 + 55 = 75$ min. This means that Ship 2 must wait $75 - 50 = 25$ min before unloading begins. The situation is depicted in the following timeline diagram:



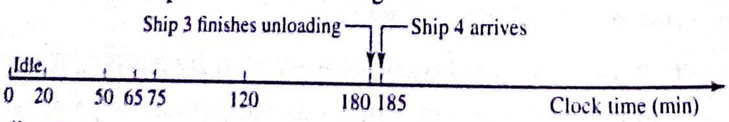
Timeline 1

Now before Ship 2 starts to unload, Ship 3 arrives at time $t = 50 + 15 = 65$ min. Because the unloading of Ship 2 starts at $t = 75$ min and it takes 45 min to unload, unloading Ship 3 cannot start until $t = 75 + 45 = 120$ min, when Ship 2 is finished. Thus, Ship 3 must wait $120 - 65 = 55$ min. The situation is depicted in the next timeline diagram:



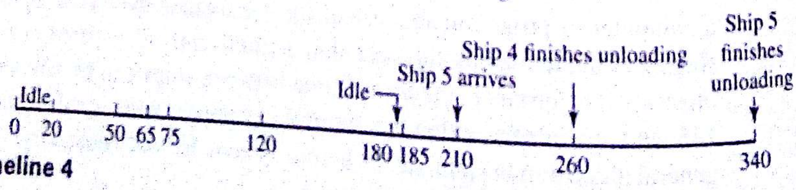
Timeline 2

Ship 4 does not arrive in the harbor until $t = 65 + 120 = 185$ min. Therefore, Ship 3 has already finished unloading at $t = 120 + 60 = 180$ min, and the harbor facilities are idle for $185 - 180 = 5$ min. Moreover, the unloading of Ship 4 commences immediately upon its arrival, as depicted in the next diagram:



Timeline 3

Finally, Ship 5 arrives at $t = 185 + 25 = 210$ min, before Ship 4 finishes unloading at $t = 185 + 75 = 260$ min. Thus, Ship 5 must wait $260 - 210 = 50$ min before it starts to unload. The simulation is complete when Ship 5 finishes unloading at $t = 260 + 80 = 340$ min. The final situation is shown in the next diagram:



Timeline 4

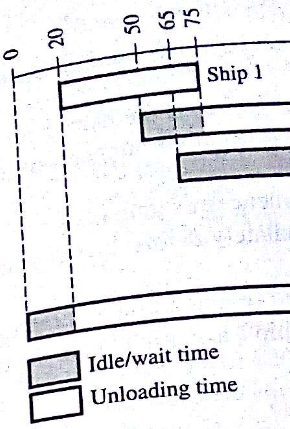


Figure 5.9
Idle and unloading tir

In Figure 5.9, we summ
ship arrivals. In Table
hypothetical ships. No
is 130 min. This waiti
dissatisfaction with th
25 min of total idle
approximately 93% c

Suppose the own
they are providing a
whether improveme
evaluating the quali
harbor is 130 min by
are very sensitive to
spent waiting for a
Some customers ar
the longest queue i
statistics to assess

Table 5.14 Summary of the harbor

Ship no.	Random time between ship arrivals	Arrival time
1	20	20
2	30	50
3	15	65
4	120	185
5	25	210
Total (if appropriate):		
Average (if appropriate):		

Note: All times are given in minutes after th

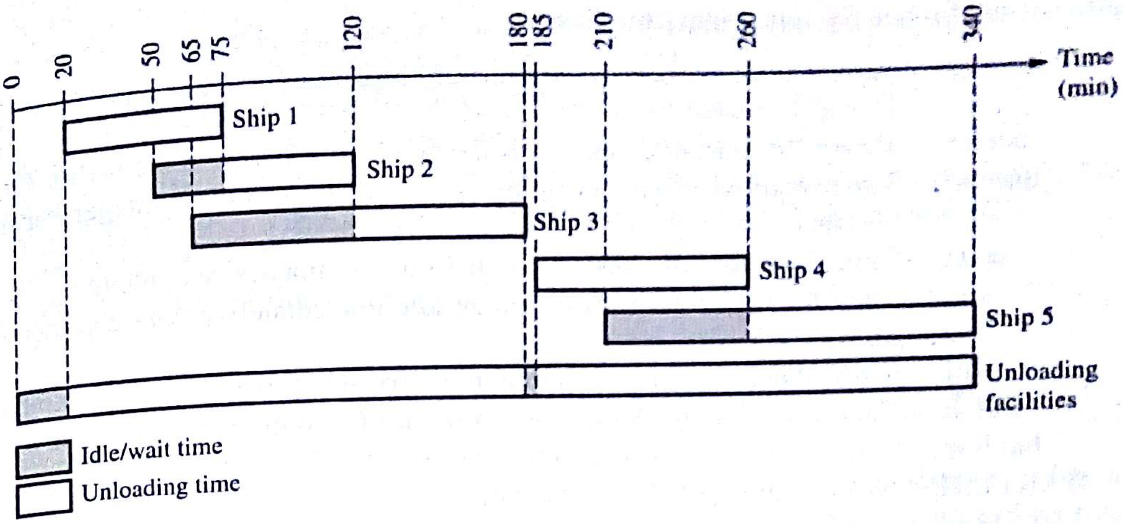


Figure 5.9
Idle and unloading times for the ships and docking facilities

In Figure 5.9, we summarize the waiting and unloading times for each of the five hypothetical ship arrivals. In Table 5.14, we summarize the results of the entire simulation of the five hypothetical ships. Note that the total waiting time spent by all five ships before unloading is 130 min. This waiting time represents a cost to the shipowners and is a source of customer dissatisfaction with the docking facilities. On the other hand, the docking facility has only 25 min of total idle time. It is in use 315 out of the total 340 min in the simulation, or approximately 93% of the time.

Suppose the owners of the docking facilities are concerned with the quality of service they are providing and want various management alternatives to be evaluated to determine whether improvement in service justifies the added cost. Several statistics can help in evaluating the quality of the service. For example, the maximum time a ship spends in the harbor is 130 min by Ship 5, whereas the average is 89 min (Table 5.14). Generally, customers are very sensitive to the amount of time spent waiting. In this example, the maximum time spent waiting for a facility is 55 min, whereas the average time spent waiting is 26 min. Some customers are apt to take their business elsewhere if queues are too long. In this case, the longest queue is two. The following Monte Carlo simulation algorithm computes such statistics to assess various management alternatives.

Table 5.14 Summary of the harbor system simulation

Summary of Harbor System Algorithm Terms

$between_i$	Time between successive arrivals of Ships i and $i-1$ (a random integer varying between 15 and 145 min)
$arrive_i$	Time from start of clock at $t=0$ when Ship i arrives at the harbor for unloading
$unload_i$	Time required to unload Ship i at the dock (a random integer varying between 45 and 90 min)
$start_i$	Time from start of clock at which Ship i commences its unloading
$idle_i$	Time for which dock facilities are idle immediately *before* commencement of unloading Ship i
$wait_i$	Time Ship i waits in the harbor after arrival before unloading commences
$finish_i$	Time from start of clock at which service for Ship i is completed at the unloading facilities
$harbor_i$	Total time Ship i spends in the harbor
HARTIME	Average time per ship in the harbor
MAXHAR	Maximum time of a ship in the harbor
WAITIME	Average waiting time per ship before unloading
MAXWAIT	Maximum waiting time of a ship
IDLETIME	Percentage of total simulation time unloading facilities are idle

Harbor System Simulation Algorithm

Input Total number n of ships for the simulation.

Output HARTIME, MAXHAR, WAITIME, MAXWAIT, and IDLETIME.

Step 1 Randomly generate $between_1$ and $unload_1$. Then set $arrive_1 = between_1$.

Step 2 Initialize all output values:

$$\text{HARTIME} = unload_1, \qquad \text{MAXHAR} = unload_1,$$
$$\text{WAITIME} = 0, \qquad \text{MAXWAIT} = 0, \qquad \text{IDLETIME} = arrive_1$$

Step 3 Calculate finish time for unloading of Ship_1:

$$finish_1 = arrive_1 + unload_1$$

Step 4 For $i = 2, 3, \ldots, n$, do Steps 5–16.

Step 5 Generate the random pair of integers $between_i$ and $unload_i$ over their respective time intervals.

Step 6 Assuming the time clock begins at $t = 0$ min, calculate the time of arrival for Ship_i:

$$arrive_i = arrive_{i-1} + between_i$$

Step 7 Calculate the time difference between the arrival of Ship_i and the finish time for unloading the previous Ship_{i-1}:

$$timediff = arrive_i - finish_{i-1}$$

Step 8 For nonnegative timediff, the unloading facilities are idle:

$$idle_i = timediff \quad \text{and} \quad wait_i = 0$$

For negative timediff, Ship_i must wait before it can unload:

$$wait_i = -timediff \quad \text{and} \quad idle_i = 0$$

Step 9 Calculate the start time for unloading Ship_i:

$$start_i = arrive_i + wait_i$$

Step 10 Calculate the finish time for unloading Ship_i:

$$finish_i = start_i + unload_i$$

Step 11 Calculate the time in harbor for Ship_i:

$$harbor_i = wait_i + unload_i$$

Step 12 Sum $harbor_i$ into total harbor time HARTIME for averaging.

Step 13 If $\text{harbor}_i > \text{MAXHAR}$, then set $\text{MAXHAR} = \text{harbor}_i$. Otherwise leave MAXHAR as is.

Step 14 Sum wait_i into total waiting time WAITIME for averaging.

Step 15 Sum idle_i into total idle time IDLETIME.

Step 16 If $\text{wait}_i > \text{MAXWAIT}$, then set $\text{MAXWAIT} = \text{wait}_i$. Otherwise leave MAXWAIT as is.

Step 17 Set $\text{HARTIME} = \text{HARTIME}/n$, $\text{WAITIME} = \text{WAITIME}/n$, and $\text{IDLETIME} = \text{IDLETIME}/\text{finish}_n$.

Step 18 OUTPUT (HARTIME, MAXHAR, WAITIME, MAXWAIT, IDLETIME) STOP

Table 5.15 gives the results, according to the preceding algorithm, of six independent simulation runs of 100 ships each.

Now suppose you are a consultant for the owners of the docking facilities. What would be the effect of hiring additional labor or acquiring better equipment for unloading cargo so that the unloading time interval is reduced to between 35 and 75 min per ship? Table 5.16 gives the results based on our simulation algorithm.

You can see from Table 5.16 that a reduction of the unloading time per ship by 10 to 15 min decreases the time ships spend in the harbor, especially the waiting times. However, the percentage of the total time during which the dock facilities are idle nearly doubles. The situation is favorable for shipowners because it increases the availability of each ship for hauling cargo over the long run. Thus, the traffic coming into the harbor is likely to increase. If the traffic increases to the extent that the time between successive ships is reduced to between 10 and 120 min, the simulated results are as shown in Table 5.17. We can see from this table that the ships again spend more time in the harbor with the increased traffic, but now harbor facilities are idle much less of the time. Moreover, both the shipowners and the dock owners are benefiting from the increased business.

Table 5.15 Harbor system simulation results for 100 ships

Average time of a ship in the harbor	106	85	101	116	112	94
Maximum time of a ship in the harbor	287	180	233	280	234	264
Average waiting time of a ship	39	20	35	50	44	27
Maximum waiting time of a ship	213	118	172	203	167	184
Percentage of time dock facilities are idle	0.18	0.17	0.15	0.20	0.14	0.21

Note: All times are given in minutes. Time between successive ships is 15–145 min. Unloading time per ship varies from 45 to 90 min.

Table 5.16 Harbor system simulation results for 100 ships

Average time of a ship in the harbor	74	62	64	67	67	73
Maximum time of a ship in the harbor	161	116	167	178	173	190
Average waiting time of a ship	19	6	10	12	12	16
Maximum waiting time of a ship	102	58	102	110	104	131
Percentage of time dock facilities are idle	0.25	0.33	0.32	0.30	0.31	0.27

Note: All times are given in minutes. Time between successive ships is 15–145 min. Unloading time per ship varies from 35 to 75 min.

Suppose now that we are not satisfied with the assumption that the arrival time between ships (i.e., their interarrival times) and the unloading time per ship are uniformly distributed over the time intervals $15 \le \text{between}_i \le 145$ and $45 \le \text{unload}_i \le 90$, respectively. We decide to collect experimental data for the harbor system and incorporate the results into our model, as discussed for the demand submodel in the previous section. We observe (hypothetically) 1200 ships using the harbor to unload their cargoes, and we collect the data displayed in Table 5.18.

Following the procedures outlined in Section 5.4, we consecutively add together the probabilities of each individual time interval between arrivals as well as probabilities of each individual unloading time interval. These computations result in the cumulative histograms depicted in Figure 5.10.

Next we use random numbers uniformly distributed over the interval $0 \le x \le 1$ to duplicate the various interarrival times and unloading times based on the cumulative histograms. We then use the midpoints of each interval and construct linear splines through adjacent data points. (We ask you to complete this construction in Problem 1.) Because it is easy to calculate the inverse splines directly, we do so and summarize the results in Tables 5.19 and 5.20.

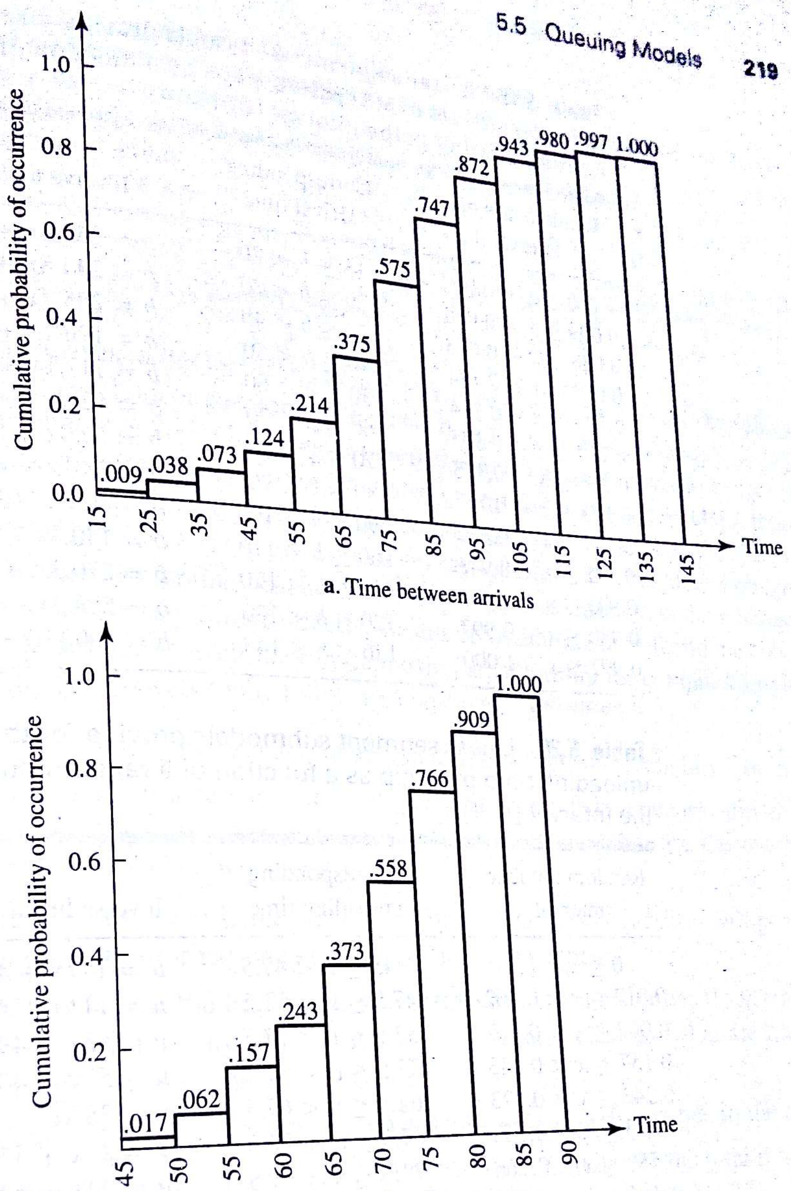
Table 5.17 Harbor system simulation results for 100 ships

Average time of a ship in the harbor	114	79	96	88	126	115
Maximum time of a ship in the harbor	248	224	205	171	371	223
Average waiting time of a ship	57	24	41	35	71	61
Maximum waiting time of a ship	175	152	155	122	309	173
Percentage of time dock facilities are idle	0.15	0.19	0.12	0.14	0.17	0.06

Note: All times are given in minutes. Time between successive ships is 10–120 min. Unloading time per ship varies from 35 to 75 min.

Table 5.18 Data collected for 1200 ships using the harbor facilities

Figure 5.10
Cumulative histograms of the time between ship arrivals and the unloading times, from the data in Table 5.18



a. Time between arrivals

b. Unloading time

Finally, we incorporate our linear spline submodels into the simulation model for the harbor system by generating between_i and unload_i for $i = 1, 2, \ldots, n$ in Steps 1 and 5 of our algorithm, according to the rules displayed in Tables 5.19 and 5.20. Employing these submodels, Table 5.21 gives the results of six independent simulation runs of 100 ships each.

EXAMPLE 2 *Morning Rush Hour*

In the previous example, we initially considered a harbor system with a single facility for unloading ships. Such problems are often called *single-server queues*. In this example, we consider a system with four elevators, illustrating *multiple-server queues*. We discuss the problem and present the algorithm in Appendix B.

Table 5.19 Linear segment submodels provide for the time between arrivals of successive ships as a function of a random number in the interval [0, 1].

Random number interval	Corresponding arrival time	Inverse linear spline
$0 \le x < 0.009$	$15 \le b < 20$	$b = 555.6x + 15.0000$
$0.009 \le x < 0.038$	$20 \le b < 30$	$b = 344.8x + 16.8966$
$0.038 \le x < 0.073$	$30 \le b < 40$	$b = 285.7x + 19.1429$
$0.073 \le x < 0.124$	$40 \le b < 50$	$b = 196.1x + 25.6863$
$0.124 \le x < 0.214$	$50 \le b < 60$	$b = 111.1x + 36.2222$
$0.214 \le x < 0.375$	$60 \le b < 70$	$b = 62.1x + 46.7080$
$0.375 \le x < 0.575$	$70 \le b < 80$	$b = 50.0x + 51.2500$
$0.575 \le x < 0.747$	$80 \le b < 90$	$b = 58.1x + 46.5698$
$0.747 \le x < 0.872$	$90 \le b < 100$	$b = 80.0x + 30.2400$
$0.872 \le x < 0.943$	$100 \le b < 110$	$b = 140.8x - 22.8169$
$0.943 \le x < 0.980$	$110 \le b < 120$	$b = 270.3x - 144.8649$
$0.980 \le x < 0.997$	$120 \le b < 130$	$b = 588.2x - 456.4706$
$0.997 \le x \le 1.000$	$130 \le b \le 145$	$b = 5000.0x - 4855$

Table 5.20 Linear segment submodels provide for the unloading time of a ship as a function of a random number in the interval [0, 1].

Random number interval	Corresponding unloading time	Inverse linear spline
$0 \le x < 0.017$	$45 \le u < 47.5$	$u = 147x + 45.000$
$0.017 \le x < 0.062$	$47.5 \le u < 52.5$	$u = 111x + 45.611$
$0.062 \le x < 0.157$	$52.5 \le u < 57.5$	$u = 53x + 49.237$
$0.157 \le x < 0.243$	$57.5 \le u < 62.5$	$u = 58x + 48.372$
$0.243 \le x < 0.373$	$62.5 \le u < 67.5$	$u = 38.46x + 53.154$
$0.373 \le x < 0.558$	$67.5 \le u < 72.5$	$u = 27x + 57.419$
$0.558 \le x < 0.766$	$72.5 \le u < 77.5$	$u = 24x + 59.087$
$0.766 \le x < 0.909$	$77.5 \le u < 82.5$	$u = 35x + 50.717$
$0.909 \le x \le 1.000$	$82.5 \le u \le 90$	$u = 82.41x + 7.582$

Consider an office building with 12 floors in a metropolitan area of some city. During the morning rush hour, from 7:50 to 9:10 a.m., workers enter the lobby of the building and take an elevator to their floor. There are four elevators servicing the building. The time between arrivals of the customers at the building varies in a probabilistic manner every 0–30 sec, and upon arrival each customer selects the first available elevator (numbered 1–4). When a person enters an elevator and selects the floor of destination, the elevator waits 15 sec before closing its doors. If another person arrives within the 15-sec interval, the waiting cycle is repeated. If no person arrives within the 15-sec interval, the elevator departs to deliver all of its passengers. We assume no other passengers are picked up along the way. After delivering its last passenger, the elevator returns to the main floor, picking up no passengers on the way down. The maximum occupancy of an elevator is 12 passengers.

Table 5.21 Harbor system simulatio

Average time of a ship in the harbor
Maximum time of a ship in the harbor
Average waiting time of a ship
Maximum waiting time of a ship
Percentage of time dock facilities are idle

Note: Based on the data exhibited in Table 5.

When a person arrives in the lobby
are transporting their load of passe

The management of the buildin
and is interested in exactly what s
have to wait too long in the lobby b
too much time riding the elevator,
in the lobby during the morning r
resolve these complaints by a mo

We wish to simulate the ele
tation that will give answers to t

1. How many customers are ac
2. If the *waiting time* of a pers
arrival at the lobby until e
maximum times a person w
3. What is the length of the l
management with informa
4. If the *delivery time* is the
in the lobby, including a
and maximum delivery ti
5. What are the average an
6. How many stops are ma
hour time is each elevat

An algorithm is prese

5.5 PROBLEMS

1. Using the data from
cumulative plots of
Figure 5.7). Calcula
Compare your resul
2. Use a smooth poly
times. Compare res
3. Modify the ship h
in the queue.

Table 5.21 Harbor system simulation results for 100 ships

Average time of a ship in the harbor	108	95	125	78	123	101
Maximum time of a ship in the harbor	237	188	218	133	250	191
Average waiting time of a ship	38	25	54	9	53	31
Maximum waiting time of a ship	156	118	137	65	167	124
Percentage of time dock facilities are idle	0.09	0.09	0.08	0.12	0.06	0.10

Note: Based on the data exhibited in Table 5.18. All times are given in minutes.

When a person arrives in the lobby and no elevator is available (because all four elevators are transporting their load of passengers), a queue begins to form in the lobby.

The management of the building wants to provide good elevator service to its customers and is interested in exactly what service it is now giving. Some customers claim that they have to wait too long in the lobby before an elevator returns. Others complain that they spend too much time riding the elevator, and still others say that there is considerable congestion in the lobby during the morning rush hour. What is the real situation? Can the management resolve these complaints by a more effective means of scheduling or utilizing the elevators?

We wish to simulate the elevator system using an algorithm for computer implementation that will give answers to the following questions:

1. How many customers are actually being serviced in a typical morning rush hour?
2. If the *waiting time* of a person is the time the person stands in a queue—the time from arrival at the lobby until entry into an available elevator—what are the average and maximum times a person waits in a queue?
3. What is the length of the longest queue? (The answer to this question will provide the management with information about congestion in the lobby.)
4. If the *delivery time* is the time it takes a customer to reach his or her floor after arrival in the lobby, including any waiting time for an available elevator, what are the average and maximum delivery times?
5. What are the average and maximum times a customer actually spends in the elevator?
6. How many stops are made by each elevator? What percentage of the total morning rush hour time is each elevator actually in use?

An algorithm is presented in Appendix B.

5.5 PROBLEMS

1. Using the data from Table 5.18 and the cumulative histograms of Figure 5.10, construct cumulative plots of the time between arrivals and unloading time submodels (as in Figure 5.7). Calculate equations for the linear splines over each random number interval. Compare your results with the inverse splines given in Tables 5.19 and 5.20.
2. Use a smooth polynomial to fit the data in Table 5.18 to obtain arrivals and unloading times. Compare results to those in Tables 5.19 and 5.20.
3. Modify the ship harbor system algorithm to keep track of the number of ships waiting in the queue.