Proceedings of the 1989 Winter Simulation Conference
E.A. MacNair, K.J. Musselman, P. Heidelberger (eds.)

SIMULATION FOR DECISION MAKING
AN INTRODUCTION

Arne Thesen
Univeristy of Wisconsin-Madison
Madison, WI 53706, USA.

Laurel E. Travis
University of Wisconsin-Whitewater
Whitewater, WI 53190, USA

ABSTRACT

We give an overview of simulation modeling and analysis from the perspective of prospective users wanting to use simulation *as a decision aid*. Important considerations in building simulation models and analyzing their outputs are discussed. A brief overview of available software is given. At the end of this tutorial, you should have a general understanding of simulation and an understanding of its applicability to your situation.

1. INTRODUCTION

In today's competitive business climate, careful planning and analysis of alternative strategies and procedures is essential. In an effort to derive maximum benefit from available resources, engineers and business planners have made performance analysis an important part of their planning activities. Among performance analysis tools, simulation has experienced a particularly dramatic increase in popularity due to its broad range of applicability.

We frequently hear of simulation being used for tasks like driver training, rocket flight analysis, and weather prediction. These simulations describe how a system changes continuously over time in response to controls (such as the turning of the car's steering wheel) that may vary smoothly through time. In contrast, *discrete event simulation* (the topic of this tutorial) describes systems that are assumed to change instantaneously in response to certain sudden or *discrete* events. For example, if we were to model an inventory system, the arrival of a batch of raw materials could be modeled as a discrete event that caused a sudden change in the system. When we choose to model a real world system using discrete event simulation, we give up the ability to capture a degree of detail that can only be described as smooth continuous change. In return, we get a simplicity that allows us to capture the important features of many systems that are too complex to capture with continuous simulation.

These discrete event simulations are generally used to develop an understanding of the performance of a complicated system over time. For example, we may want to understand how the number of operators working at a phone bank affects the percent of callers getting a busy signal and the average time a caller spends on hold. To arrive at this understanding, we would first build a computer model representing the arrival and handling of calls. In this model, the arrival and completion of individual calls would be described as *discrete events*. The model would use random variables to replicate variability in quantities such as the time between calls, the time speaking with an operator, and the time between getting a busy signal and calling again. Then we would run the model (i.e., operate the simulated phone bank), accumulating data on the individual simulated callers. This data would then be used as a basis for determining how many operators to use.

Table 1: Some questions that can be answered by simulation.

Capacity and Feasibility Questions
How large must the factory be?
Does the building design include enough elevators?
How many robots are needed?
Will the conveyor deliver food before it is cold?
Comparing Alternatives
Should we process jobs according to first-come-first-served priority or shortest-processing-time-first?
Should we use MRP or just-in-time?
Trouble Shooting and Fine Tuning
Is the bottleneck caused by the grinder or the mill?
What is the optimal buffer capacity?
What should the inventory reorder point be?

1.1. Questions Answered by Simulation

For simulation to be effective, it must be focused on some previously defined problem (otherwise we do not know what elements of the system to include in the model and what information to collect) . Using simulation before a specific problem is articulated may lead to a large number of unfocused simulation runs that use inappropriately designed models, and produce little or no information of value.

Different questions are asked at different stages of a study, and they are answered by models with different levels of detail. For example, questions about overall

Table 2: Some recent applications of simulation

Application area	Performance Measure
Air traffic control	Delays
Bank teller scheduling	Waiting times
Electric car	Battery usage
Computer networks	Delays
Robot Scheduling rules	Throughput/day
Harbor management	Delays
Location of Fire stations	Response times
Social systems of wasps	Nest building

plant capacity are frequently asked early in the project when few details about the design are available and fairly rough answers may suffice. In this case a simple model is appropriate. On the other hand, questions about the efficiency of different scheduling rules in an automated manufacturing line can only be answered when the detailed design of the system is finalized and precise speeds, capacities and part routings are known. In this case a detailed model is required and a fairly sophisticated analysis of the simulation output is called for. Some typical questions that can be answered through simulation are given in Table 1.

While our subject is simulation, we should point out that many of the questions that arise at the early stages in a design project may be answered with sufficient precision using simpler models and specialized computer packages. For example a simple Lotus 123 model may be used to balance production lines so that each station works at approximately the same rate. Also, quick modeling tools such as Manuplan (Suri and Tomsicek (1988)) make use of analytical models and numerical approximation to estimate the average time required to get a product through a factory. The advantage of these approaches is that they are easy to use, and they provide very fast answers to "what-if" type questions in comparison to the weeks that may be required if simulation analysis is used. The disadvantage of these simpler analytic models is that their models are often inexact representations of the phenomena of interest, and even the most expertly formulated model may give results that are off by an indeterminate amount, perhaps 10-20% or more. Even these rough answers can be quite valuable, however, if obtained early in the planning of a project.

1.2. Applications Areas

The military services were among the first to use simulation analysis, and simulations ranging from evaluations of maintenance policies to large scale war games are routinely used to guide defence policy. Simulation is rapidly growing in popularity and examples of successful simulation applications are found in a surprisingly eclectic range of fields (Table 2).

Table 3: Some pitfalls to avoid

1. Failure to state a clear objective
2. Failure to frame an answerable question
3. Using simulation when an analytic model would suffice
4. Analysis at an inappropriate moment
5. Inappropriate level of complexity
6. Bad assumptions in model
7. Poor output analysis
8. Budget overruns

The most recent growth in simulation applications has been in the manufacturing area. Almost all major new construction projects and many manufacturing process redesigns currently benefit from some sort of simulation analysis of the proposed design.

1.3. Some Common Pitfalls

Simulation analysis is not without drawbacks. *First*, the quality of the analysis depends on the quality of the model; model building is an art. *Second*, it is often difficult to determine if an observation made during a simulation run is due to a significant underlying relationship in the system being modeled or due to the built-in randomness of the run; simulation results are hard to interpret. *Finally*, simulation analysis is usually a time-consuming and expensive process, and an adequate analysis may not be feasible within the time available; analytic methods may be better for "quick and dirty" estimates.

Some common pitfalls are listed in Table 3. Perhaps the most important of these is the failure to clearly state the objective of the project before it is undertaken, and to be guided by this purpose throughout the life of the project.

2. BUILDING SIMULATION MODELS

When beginning a simulation, it is often tempting to build a model of phenomena that are easily observed and understood. For example, if we want to understand the effect of the reliability of one machine on overall throughput in a plant, we may be tempted to describe in detail how the machine works. However, this may be inappropriate, as the level of detail and time resolution required to completely describe machine operation is different from that of describing the general pattern of machine failure. It is therefore a good idea as a first step in the modeling process to develop the simplest possible model that provides the necessary

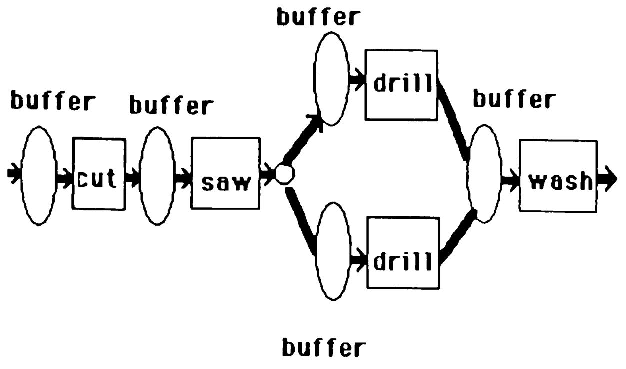


Figure 1: A simple production process

information. Starting with such a rough model enables the modeler to describe some of the important relationships in the system without excessive detail. The insights gained from this simple model can then be used to aid in the effective development of a more detailed model.

2.1. The Role of the Simulation Model

To illustrate how simulation models are used as decision aids, consider the simple production process shown in Figure 1. We want to improve the efficiency of our system by specifying appropriate buffer sizes between the different work stations. Large buffers tend to result in higher utilization as no machine is starved for work. However, there is a point beyond which additional buffers add to overall cost without significantly improving utilization. To determine the overall effectiveness of having different buffer capacities at different locations, the analyst would like to know how buffer sizes affect machine utilization. The analyst would therefore like to have a model such as the one shown in Figure 2.

Our task would be greatly simplified if we had a mathematical formula giving utilization as a function of buffer sizes and other parameters, but since we do not usually have prior knowledge of any such relationship, we use a simulation model that mimics the dynamic operation of the facility. As this model is running, we collect data about machine utilization, and at the end of the run the computer produces a report giving the average utilization for the run. So the simulation gives information about utilization for a given buffer size, but it does not compute optimal buffer size, or even give us a general rule for describing utilization as a function of buffer size. *In other words, the model does not explicitly describe the relationships that we want, rather, it describes how the system operates.* It is the job of the analyst to determine how the information needed for decision making can be obtained from the model.

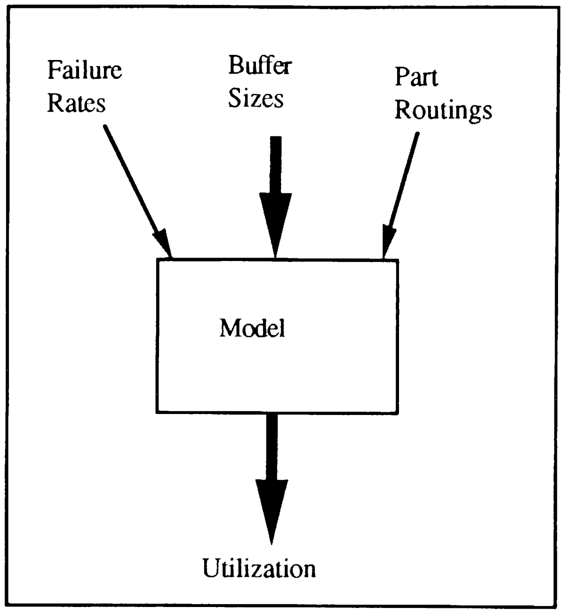


Figure 2: Model for analysis of effect of buffer size on utilization.

2.2. Elements of Simulation Modeling

Our ability to develop simulation models of a wide range of different phenomena is due to the fairly universal nature of the building blocks on which the models are based. In particular, the representation of dynamic behavior and the use of random variables are fundamental to all discrete event simulations. These two concepts are discussed in this section.

Modeling Elementary Random Processes. Regardless of the simulation package used, our goal is to replicate real life phenomena in the computer. For example, if we are studying the effect of different repair policies we need to generate intervals between machine breakdowns that model the intervals observed in the factory. Instead of carrying out a detailed analysis of the state of each machine in the system so that we can predict the exact time of breakdowns, we use random variables to represent the *pattern* of breakdowns regardless of cause. The time of any one simulated breakdown will be different from what we observe in real life but the long range pattern of breakdowns should be indistinguishable from the real life process.

Most simulation models use random variables this way to compensate for our lack of detailed knowledge of what is going to happen at any one instance in a real life process. Phenomena modeled this way include *choices*, *quantities*, *frequencies*, *intervals* and *durations*. Probability distributions describing these phenomena are readily available. Some frequently modeled random processes and their recommended distributions are given in Table 4.

Table 4: Phenomena frequently described by random variables.

Phenomena	Example	Typical Distrib.
Choice	Tossing a coin Turning right or left	Bernoulli
Frequency	Breakdowns per hour Takeoffs per day	Poisson
Quantity	Age of a victim Actual weight of product	Normal
Interval	Time between breakdowns Time between arrivals	Expon.
Durations	Time to complete a task Time to repair a lathe	Erlang Gamma

The selection of appropriate probability distributions is critical to the art of model building. We use the phrase *fitting of distributions to empirical data* to describe the process of finding a probability distribution with the property that random observations drawn from it are indistinguishable from empirical observations of a phenomenon of interest. The procedure for fitting data to distributions typically includes goodness-of-fit tests such as the Chi-square or Kolmogorov-Smirnov test and techniques for parameter estimation. Several software systems are available for this analysis (Thesen (1985), and Law and Vincent(1987)).

In addition, we are frequently asked to simulate situations about which we have limited knowledge -- we cannot fit a distribution to the data when there is no data. For example, we may be asked to evaluate the effect of different scheduling policies in a not-yet-constructed production system. In the absence of information other than the mean service time, it is convenient to specify an exponential distribution for the random variable representing service time. This choice is considered attractive because using the exponential distribution only forces the modeler to specify one parameter and hence we do not need any additional information once the mean is specified. However, exponential distributions tend to over-estimate the variability of a process. Similarly uniform distributions tend to under-estimate variability. While incorrect variability may seem like a minor oversight when the mean of the distribution is correct, it can in fact cause extremely misleading results. Systems with lower variability tend to run much more smoothly and have fewer bottlenecks than systems with higher variability, so using distributions with inappropriately high variability can lead to pessimistic models and wasteful recommendations. Selecting appropriate distributions in the absence of good data requires a great deal of experience and judgement, or the gathering of additional information.

In sum, we use random variables to replicate events in the computer. The choice of probability distributions for these random variables involves collecting data on the real world processes and fitting distributions to this data. Since the choice of these distributions has a large impact on the validity of the model, it is well worth the modeler's time and effort to collect good data.

Describing Dynamic Behavior. Discrete event simulation models are run by tracing the sequence of events that change the state of the system of interest over time. However, since we do not normally think of models of systems in terms of events and state changes, the modeler usually uses a "friendlier" representation. The computer then translates this to an event oriented approach to actually run the model. One such "friendly" approach, the transaction flow approach will be discussed here.

Many simulations describe how *transactions* flow through a *block diagram* representing a system of interest. For example, transactions may represent subassemblies and the block diagram may show how these subassemblies flow through an assembly process, or transactions may represent customers and the block diagram may show how these customers progress through multiple stages of being served. Using a limited number of standardized building blocks to describe what happens to transactions, these languages are able to represent the behavior of a wide range of different systems.

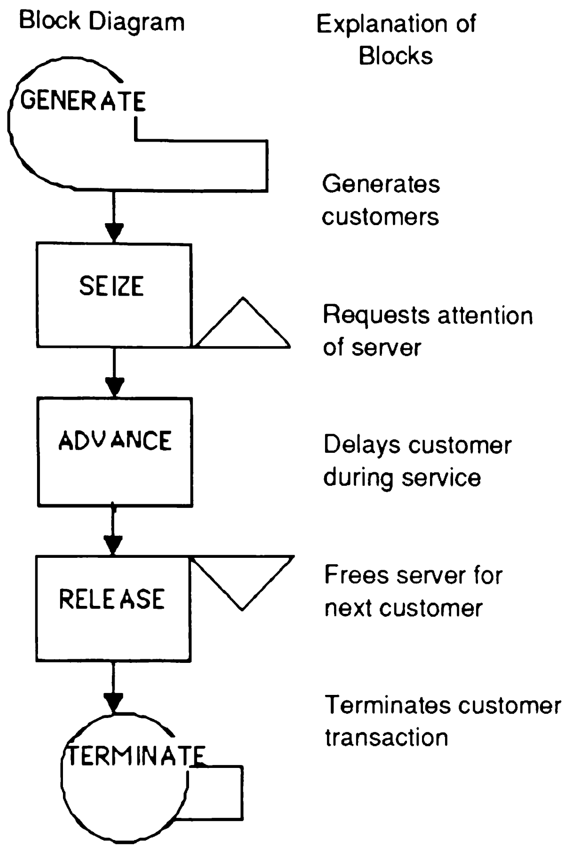
The first block (or statement) in a model generates transactions. For example, transactions representing individual customers in a waiting line or *queuing* system might be generated at random time intervals. Each transaction immediately flows through the diagram until it hits some obstacle that causes it to be delayed. Eventually, conditions change and the delayed transaction is allowed to move again. Two important mechanisms that cause the flow of transactions to be impeded are:

Explicitly specified delays - the transaction waits in a block while being served, and

Blocking - the transaction is refused entry to the next block.

Blocking usually occurs when a transaction wants to use a *resource* that is currently unavailable. For example, the transaction may want to receive the attention of a server that is busy serving somebody else. Since many transactions may be waiting for service or in service, a running model may contain a large number of transactions simultaneously.

A feature of the transaction-flow approach to model representation, is that resources are not always explicitly shown in the model. Instead, we show how the resource and the transaction interact. Accordingly, languages using the transaction-flow approach (such as GPSS) provide blocks to request the use of a resource and blocks to release control of the resource. A diagram of



Block Diagram
Explanation of Blocks
GENERATE
Generates customers
SEIZE
Requests attention of server
ADVANCE
Delays customer during service
RELEASE
Frees server for next customer
TERMINATE
Terminates customer transaction

Figure 3: Graphical representation of a single server queuing model in GPSS.

a single server queuing model written using five lines of GPSS/PC code is shown in Figure 3. Instead of explicitly showing the server, a request for service is represented by the SEIZE block and the release of the server is represented by the RELEASE block.

The resulting models would be identical if other dialects of GPSS such as GPSS/H (Henriksen and Crain (1983)) were used, and similar if any of the other major languages such as SIMSCRIPT (Markowitz et al (1987)), SIMAN (Pegden(1986)), and SLAM(Pritsker(1986) were used. These simulation languages all use similar approaches for modeling dynamic behavior. Other approaches to model representation are discussed in the following section.

Commonly Used Modeling Packages. While it often easy to describe a situation to other humans, (for example by using graphs and/or analogies), it can be exceptionally difficult to capture its detail on a computer. We therefore try, whenever possible, to design models that draw upon previously developed models and programs.

The more choices that the model builder must make, the more difficult the modeling process becomes. On one extreme, with special purpose simulation systems such as Sandie (Thesen (1986)) and StarCell (Steudel (1987)), users are shown a model *template* giving model structure and default parameters. For example the model we saw in Figure 3 is a standard feature in both Sandie and StarCell. After changing a few parameter values, it is ready to run. Although convenient if the pre-programmed model accurately reflects the problem at hand, this approach is unable to describe many special circumstances.

Table 5: Four approaches to model representation.

Model Type	Interface	User Input	Typical Tool
Template	Menus	Parameters	StarCell Sandie
Network	Graphical	Structure Parameters	XCell Simple_1
Simulation Language	Text editor/ graphical	Structure Parameters Performance measure	GPSS, SIMAN, SIMSCRIPT SLAM
Programming Language	Text editor	Structure Parameters Performance measure Time keeping	C FORTRAN GASP Modula-2 PASCAL

Users of many of the emerging interactive *network* based simulation environments with graphical user interfaces (for example Corbin (1987) and Conway (1987)) have more flexibility in specifying model structures. Mice and pop-up menus are often used for this purpose. As with the template models the accessibility of these environments comes with the disadvantage of limited flexibility. The benefit is a fairly short model building time and some reasonable assurance that the model works as intended.

As we saw in the previous section, users writing their simulations in a *simulation language* use model building blocks such as GENERATE, ADVANCE, TERMINATE, SEIZE, to specify the flow and logic of the model. While statements in a simulation language correspond to activities in the system of being modeled (WAIT, QUEUE) rather than to activities in the computer (multiply, divide), these languages have much of the structural flexibility of programming languages. Simulation languages therefore are appropriate when the modeler requires more flexibility than provided by template packages, and has some programming expertise. Some of these languages can be used with graphical user interfaces and animation, and although these features radically change what the model builder views on the computer screen, they do not change the underlying logic or level of detail of the model. Graphical interfaces and animation may cause model building time may be shortened (or perhaps in the case of animation lengthened), but the modeler still requires the same detailed knowledge of the model.

Occasionally a situation is so unique that it cannot be effectively modeled using any of the approaches listed above. For example, this might occur when complex material handling systems using fairly elaborate control schemes are simulated. In these cases general purpose *programming languages* must be used. Thesen (1987) gives an overview of how such programs can be written in Pascal. Approximately 1000 lines of Pascal code would be required to implement the simple model depicted in Figure 3.

Finally there are certain simulation systems that fall outside the classification scheme listed above. For example simulation models can be integrated with a factory's production scheduling system to aid in day-to-day scheduling by running simulations using current information from the shop floor. (MacFarland (1987)).

3. OUTPUT ANALYSIS

3.1. Analysis of a Single Design

Simulations are used to develop some sense of the performance of a system over time. We may want to know the average waiting time in a facility or we may want to know the expected demand during a stock-out period. While this type of summary information is not usually *used* in the simulation model, one of our tasks is to accumulate it during the run.

For example, we can easily record the length-of-stay for each individual customer during a run, and at the end of the run we can compute the *average* length-of-stay. Since random variates are used to replicate real life in service times and arrival times, the resulting average length-of-stay will also be a random variable. In other words simulation follows the RIRO principle -- random input, random output. It is essential to keep this in mind when interpreting your results. If we were to run the same simulation model again, we would in all likelihood observe a different average length-of-stay. We would like to draw conclusions and make decisions based on the average length-of-stay, but every time the model is run a different value results. We are observing values of a random variable whose distribution we do not know (see Figure 4). *The purpose of most simulation runs is to estimate the true mean of the distribution of these averages and to develop an understanding of its variability*

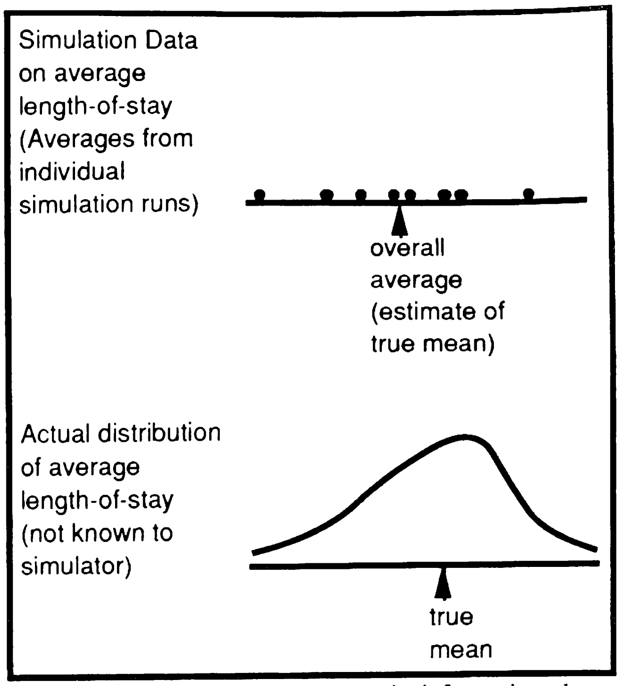

Simulation Data on average length-of-stay (Averages from individual simulation runs)
overall average (estimate of true mean)
Actual distribution of average length-of-stay (not known to simulator)
true mean

Figure 4. Simulation is used to gain information about an unknown distribution.

3.2. Comparing Designs

Simulations are often used to compare the performance of different potential solutions to a problem. This requires careful planning. Since the observed value of a performance measure for any single simulation run may be thought of as a random variable, the observed difference between the performance of multiple systems will also be a random variable. If other words, say we are interested in comparing system A to system B, and we run a computer simulation of each. If the simulated performance of system A is slightly better than that of system B, can we conclude that this difference is due to a genuine difference between the systems, or might it be a chance event caused by the inherent randomness of our simulation runs? Careful analysis is therefore necessary to see if this performance difference is statistically significant. This type of analysis can answer questions such as the following:

One decision variable
- Which *system* has the fastest response time?
- Is throughput increased if the *buffer size* is increased 10%?

Two decision variables
- Does throughput change when scheduling *rules* and *buffer sizes* change?
- What is the best combination of *order-point* and *order-quantity* ?

Many decision variables
- What *factors* affect throughput?
- What *staffing pattern, truck routing* and *fire station territories* minimize response time?

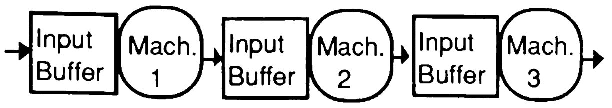


Figure 5: Three station Flow Shop

An Example. A simulation study was conducted to compare two possible decision rules for determining the order in which jobs were processed in a simple three station job-shop (Figure 5). The goal was to select a processing rule to minimize mean time in system for the jobs. The *First Come First Served* (FCFS) and the *Shortest Processing Time First* (SPT) dispatching rule were compared. The two models were run 10 times each. The following results are shown in Table 6.

Table 6: Mean time in the system for 10 replications of two different scheduling policies for a three station flow-shop.

Average Time in System		
Replication	FCFS	SPT
1	113.9	115.2
2	121.7	112.8
3	117.5	114.1
4	121.4	110.8
5	114.5	117.5
6	118.6	114.3
7	118.8	114.9
8	113.2	113.9
9	113.9	118.1
10	118.1	113.9
Overall Average	117.2	114.6
Standard Deviation	3.1	2.1

The observed difference in average throughput time is 117.2 - 114.6 = 2.6 time units. If we were to repeat these simulations, we would be likely to observe a different difference. However, if we are justified in assuming identical variability in the process producing the two data sets, then we can use classical statistical inference to estimate the 90% confidence interval about the difference between the two true means as

$$-2.00 \leq (\mu_1 - \mu_2) \leq 7.24 \ .$$

Since this interval includes zero, we are unable to reject the hypothesis that the two service policies result in identical performance.

This inconclusiveness is either due to the fact that there indeed is no difference in performance, or, it is due to the fact that the variance in the observed data was too high (i.e., the data was too "noisy" for us to extract useful information). If the observed difference were due to the built-in randomness of the model, then we could use either preform many more (perhaps expensive) simulation runs or we could use *variance reduction* techniques to reduce the variability in the data and improve the conclusiveness of the study.

Table 7: Differences between mean time in the system for 10 replications of two different scheduling policies for a three station flow-shop using identical arrival times for each pair of replications.

Average Time in System			
Replication	FCFS	SPT	Difference
1	113.9	110.8	3.1
2	121.7	117.5	4.2
3	117.5	114.3	3.2
4	121.4	114.9	6.5
5	114.5	113.9	0.6
6	118.6	112.8	5.8
7	118.8	114.1	4.7
8	113.2	115.2	-2.0
9	113.9	118.1	-4.2
10	118.1	113.9	4.2
Average			2.61
Standard Deviation			1.86

One approach to reducing variability is to operate the two different models under identical random conditions. For example we could generate a single set of random numbers to represent arrival times of jobs, and use this set for *both* models. This process would be repeated several times. Performance differences between the models in any one replication would then not be due to random differences in arrival times. We then make *pair-wise* comparisons between the models for each replication. The resulting differences are of course random variables, and their true means must be the same as the difference between the true means estimated before. However, since we have eliminated a source of variability, it is likely that their variance is smaller. As we show in Table 7 this indeed is the case for our example.

The corresponding confidence interval is

$$0.59 \leq (\mu_1 - \mu_2) \leq 4.68 \ .$$

Since this confidence interval does not include zero, we reject the hypothesis that the two policies result in identical performance, enabling us to recommend one policy over the other. Using this *common random numbers technique* we reduced the inherent randomness in the modeling process by employing common random number streams in the two scenarios. The resulting lower variance enabled us to gain more information from our simulation. Techniques such as this one can often save hundreds of lengthy computer runs by allowing more information to be extracted from less data. A brief survey of other variance reduction techniques is given in Nelson(1987).

So we have seen that the RIRO principle (random input, random output) has implications critical to the interpretation of simulation results. Observed differences between individual runs of two simulated systems do not necessarily imply actual performance differences between the two models. Many replications of the simulation runs and careful data analysis are generally required to get conclusive results. There are several special techniques (such as common random numbers) available to reduce the time and effort required to gain information.

Another Pitfall. Most conventional statistical data analysis techniques require that the data be independent and identically distributed random variables. Simulation data almost never satisfies this assumption. For example, the best predictor of the waiting time for customer 23 in a busy system may be the waiting time experienced by customer 22. Intuitively, if adjacent values in a data set tend to be similar, the data is said to be positively autocorrelated. Autocorrelated data does not satisfy the assumption of independence. Hence blind application of conventional statistical techniques will lead to misleading results whenever the real-life application shows autocorrelation.

Changes occur more slowly in positively autocorrelated data. If we ignore this, and blindly compute the sample variance from a simulation data set, this estimate of variance is usually less than the true variance of the system. Confidence intervals based on low variance estimators are too narrow, and they may lead us to believe that our simulation results include much less error than they actually do. This type of error could lead an analyst to believe that a given change in the system would cause a large improvement in performance when, actually, the improvement observed in the model was simply random variation. The resulting policy recommendation could be an expensive mistake.

Several methods have been developed to deal with autocorrelated data. Many practitioners use *batch means analysis* to develop confidence limits about the mean for autocorrelated data. The original data set is replaced with a reduced data set containing only the means of contiguous batches of the original observations. The variability in this set is then used to estimate the desired confidence interval. This technique is based on the hope that the means formed by these contiguous groups (or batches) of observations are uncorrelated. The autocorrelation in the data is not completely eliminated by this technique, but it can be reduced by increasing total sample size and changing the batch sizes . Hence the accuracy of a confidence interval increases as the length of the run increases. While beyond the scope of this tutorial, more sophisticated transformations are generally more efficient than the batch means technique, in that they yield more information from a data set of the same size. Hence shorter, less costly runs may suffice for the more sophisticated techniques.

In sum, since performance measures generated by simulations are random, statistical inference must be used to estimate their true means. This is complicated, however, by the fact that simulation data often does not satisfy the assumptions underlying the most common statistical techniques. Blind application of inappropriate statistical techniques may lead to misleading conclusions and hence expensive errors in policy. Consequently, a great deal of effort has been devoted to the development of valid, efficient techniques for extracting information from simulation data.

4. FINAL REMARKS

We have attempted to provide an introduction to the uses of simulation, the underlying concepts, and the types of computer packages available to the analyst. While some of these tools require significant expertise and experience, others are quite accessible to the novice.

We have highlighted a few guidelines for the beginning analyst:

1) Define your objectives before simulating.
2) Use the correct level of detail -- begin with a simple model.
3) Select software that is appropriate for your problem, level of experience, and time frame.
4) Remember that simulation results are observations of random variables, and interpret your results accordingly.

We have also pointed to a few of the many technical considerations involved in effective simulation, but, needless to say, in this short tutorial, the list of subjects that we treated very lightly or omitted completely is very long. Discussions on many such subjects can be found elsewhere in this volume. In addition, we identify sources of additional information in the bibliography section below.

BIBLIOGRAPHY

General Texts

Banks, Jerry and John S. Carson II (1984) ***Discrete-Event System Simulation***, Prentice-Hall, Englewood Cliffs,N.J.,1984

Bradey, Paul, Bennett L. Fox, and Linus E. Schrage (1987) ***A Guide to Simulation, 2nd edition.*** Springer-Verlag, NY.

Bulgren, William G. (1982) ***Discrete System Simulation***, Prentice-Hall,Englewood Cliffs, NJ.

Curry, Guy., Bryan L. Deuer Meyer and Richard M. Feldman (1989) ***Discrete Simulation,***

Fundamentals and Microcomputer Support, Holden Day, Oakland, CA.

Fishman,G.S. (1973) ***Concepts and Methods in Discrete Event Digital Simulation***, Wiley, NY.

Fishman, G.S. (1978) ***Principles of Discrete Event Simulation***, Wiley, NY.

Gottfried Byron S. (1984) ***Elements of Stochastic Process Simulation***, Prentice-Hall, Englewood Cliffs, NJ.

Gordon, Geoffrey (1978) ***System Simulation***, 2nd ed.,Prentice-Hall, Englewood Cliffs, NJ.

Graybeal, W. J., and U. W. Pooch (1980) ***Simulation: Principles and Methods***, Winthrop, Cambridge, MA.

Johnson, M.E. (1987) ***Multivariate Statistical Simulation,*** John Wiley, NY.

Law, Averill M., and W. David Kelton (1982) ***Simulation Modeling and Analysis***, McGraw-Hill, NY.

Matloff, Norman S. (1988) ***Probability Modeling and Computer Simulation,*** PWS-Kent, Boston, MA.

Nayor,T.H. (1971) ***Computer Simulation Experiments with Models of Economic Systems***, Wiley, NY

Payne, James A. (1982) ***Introduction To Simulation: Programming Techniques and Methods of Analysis***, McGraw-Hill, NY.

Shannon, Robert E. (1975)***System Simulation: The Art and Science***, Prentice-Hall, Englewood Cliffs, NJ.

Zeigler, Bernard (1984) ***Multifaceted Modelling and Discrete Event Simulation,*** Academic Pres.

Environments and Languages

Cobbin, Philip (1988) **The SIMPLE_1 Simulation Environment,** *Proceedings of the 1988 Winter Simulation Conference, M.Abrams, P.Haig and J.Comfort (eds.)* pp 141-145.

Conway, R. and W. Maxwell (1987) **Modeling Ascyncronous Material Handling in XCELL+** , *Proceedings of the 1987 Winter Simulation Conference, A. Thesen, H. Grant and W.David Kelton (eds.)* pp 202 - 206.

Cox, S. (1986) ***GPSS/PC Users Manual*** , Minute Software, Stow, MA.

Henriksen, J.O et al (1988) ***GPSS/H Users Manual 3rd ed.*** Wolverine Software Corp, Annandale, VA.

MacFarland, Douglas G. (1987) ***Scheduling Manufacturing Systems with FACTOR,*** *Proceedings of the 1987 Winter Simulation Conference, A. Thesen, H. Grant and W.David Kelton (eds.) pp 235-237.*

Markowits, H.M. and P.J.Kiviat, and R. Villanueva (1987) ***Simscript II.5 Programming Language.*** CACI. Los Angeles, CA.

Pegden, C.C. (1986) ***Introduction to SIMAN,*** Systems Modeling Corporation, State College, PA.

Pritsker, A.A.B, C.Elliot Sigal and R.D.Jack Hammesfahr (1988)***SLAM II Network Models for Decision Support,*** Prentice Hall, NJ.

Pritsker, A.A.B (1986) ***Introduction to Simulation and SLAM III, 3rd edition,*** Systems Publishing Corp.

Schriber, Thomas J. (1974) ***Simulation Using GPSS***, John Wiley New York, NY.

Standridge, C.R., and A.A.B. Pritsker (1987) ***TESS: The Extended Simulation System***, Halsted Press.

Steudel, H.J. and T.Park (1987) ***STAR*CELL: A Flexible Manufacturing Cell Simulator,*** *Proceedings of the 1987 Winter Simulation Conference, A. Thesen, H. Grant and W.David Kelton (eds.)* pp230-234.

Suri,R. and M.Tomsicek (1988) ***Rapid Modeling Tools for Manufacturing Simulation and Analysis,*** *Proceedings of the 1988 Winter Simulation Conference.*

Thesen, Arne (1987) ***Writing Simulations from Scratch: Pascal Implementations,*** *Proceedings of the 1987 Winter Simulation Conference, A. Thesen, H. Grant and W.David Kelton (eds.)* pp152-164.

Analysis

Box, G.E.P, W.G.Hunter and J.S.Hunter (1978) ***Statistics for Experimenters,*** Wiley and Sons, NY.

Johnson, M.E. (1987) ***Multivariate Statistical Simulation,*** John Wiley, NY.

Kleijnen, J.P.C. (1987) ***Statistical Tools for Simulation Practitioners,*** Marcel Dekker, *nc, New York, NY.*

Meketon, Marc S. (1987) ***Optimization in Simulation: A Survey of Recent Results.*** *Proceedings of the 1987 Winter Simulation Conference, A. Thesen, H. Grant and W.David Kelton (eds.)* pp 58-67.

Nelson, B.L. (1987) ***Variance Reduction for Simulation Practitioners***, *Proceedings of the 1987 Winter Simulation Conference, A. Thesen, H. Grant and W.David Kelton (eds.)* pp43-51.

Sargent, R. (1987) ***An Overview of Verification and Validation of Simulation Models,*** *Proceedings of the 1987 Winter Simulation Conference, A. Thesen, H. Grant and W.David Kelton (eds.)* pp 33-39.

Thesen, Arne (1989), ***Introduction to SANDIE,*** *Proceedings of the 1989 Winter Simulation Conference*

AUTHORS BIOGRAPHY

ARNE THESEN, a Professor of Industrial Engineering and Computer Sciences at the University of Wisconsin-Madison, wrote his first simulation program in 1964. His current research interests are in the area of expert scheduling systems. He is the co-author with Professor Travis of a forthcoming text on Simulation for Decision Making (Academic Press 1990).

Arne Thesen
Department of Industrial Engineering
1513 University Ave,
Madison, WI 53706
(608) 262-3960.

LAUREL E. TRAVIS is an Assistant Professor in the Management Department at the University of Wisconsin-Whitewater. She is the co-author with Professor Thesen of a forthcoming text on Simulation for Decision Making (Academic Press 1990).

Laurel E. Travis
Management Department
University of Wisconsin-Whitewater
Whitewater, WI 53190
(414) 472-5478