Proceedings of the 1986 Winter Simulation Conference
J. Wilson, J. Henriksen, S. Roberts (eds.)

AN INTERACTIVE RUN LENGTH CONTROL FOR SIMULATIONS ON PCs

Ralph R. Duersch
General Electric Company
Schenectady, New York

Lee W. Schruben
Cornell University
Ithaca, New York

ABSTRACT

This paper describes an interactive procedure implementing a new confidence interval estimation technique that determines how long or how many times a simulation should be run to produce results that satisfy a relative precision criterion. It also identifies initial transients and ignores transient data points when determining run length. This technique has been implemented for use with personal computers (PCs) as an interactive program that notifies the user of estimated computation times, thus permitting the scheduling of runs or the modification of run lengths as desired.

1. INTRODUCTION

The user of simulation always has to answer the question of how many times to re-run a simulation to get "correct" answers. In many cases the user is looking for steady-state results and must first remove a transient portion of the simulation and then extend the length of the simulation run until enough steady-state data has been collected. Authors of simulation texts usually propose heuristic techniques to detect the end of transient periods, and they discuss the confidence interval approach to determine run lengths. Practitioners commonly determine transient periods by plotting the data and finding the end of the transient period by inspection. They increase run length until changes in the averages of the result variables become negligible.

Recent results (Schruben, 1983 and 1985; Schruben et al., 1983) have made it possible to construct algorithms that find the end of a transient period at the beginning of a string of output data from simulations. The remaining data values can then be used to estimate the total number of data points needed using a special confidence interval test. These algorithms can be implemented in a procedure that calls for additional data points until a relative precision criterion is met with a given confidence. Thus, this procedure can be completely automated.

Today simulations are often carried out on personal computers (PCs). Simulations of any complexity tend to have long running times, and therefore the PC user may wish to determine how long such a simulation might take before committing the PC to this use. Thus, for PC simulations, the algorithms are imbedded in an interactive procedure that keeps the user informed of the expected running time and any needs for additional simulation time that must first be approved by the user. This interactive procedure is described below.

2. INTERACTIVE RUN CONTROL PROCEDURE

2.1 Background

Simulations can be of two types, terminating or steady-state. Terminating simulations represent situations that have fixed starting and ending times and may contain transients that are part of the system's normal operation. For these types of simulations, getting "correct" results means repeatedly running the simulation for the specified period. For each run, a performance variable must be computed, yielding one data point that will be saved in an array and later used by the run control algorithm. Performance variables are briefly discussed below.

Steady-state simulations represent situations where the simulator is interested only in the long-term performance of a system. These simulations are often started from an empty and idle state and the length of the run is open-ended. Because of the difference between the initial states and the (unknown) steady states, an initial transient is present. This initial transient "dies out" if the system is stable and steady state is said to be reached. For these types of simulations, getting "correct" results means making a run that is sufficiently long. During the run, the value of a performance variable must be computed at appropriate times yielding a series of data points that are saved in an array and later used by the run control algorithm.

The computation of a performance variable must be added to the simulation. It should be a function of one or more variables of the simulation, being representative of the simulation's dynamic behavior. A possible performance variable for a terminating simulation is the average Time in System of the most important entity type flowing through the simulation during a run. For steady-state simulations, a possible performance variable is the waiting time in a queue, computed every time an entity leaves the queue. No attempt has been made in this work to determine how to construct efficient performance variables.

2.2 Approach

The procedure starts by asking the user to specify the largest period of time that can be spent in arriving at the simulation results. The program computes a small number of data values to determine the computation rate for the simulation, which it reports to the user. It then proceeds with the simulation, but for only 20% of the time limit given, similar to the procedure reported in Heidelberger and Welch (1983). When this time is reached, a new confidence interval estimation procedure for the mean of the data values is used with a specified allowable relative error to estimate the additional simulation time required.

If this first run produces an acceptable estimate, the run is stopped. If additional data points are needed, a time estimate of the additional runtime is given to the user for approval. The user may accept, reject, or choose an intermediate time period. This process continues until acceptable results are obtained, or until the time limit is reached without obtaining the desired accuracy. In the latter case, the user is notified of the accuracy achieved and has the option of extending the time limit.

2.3 Run Length Estimation

The discussion and procedures described in this section are based on the work by Schruben (1983) on confidence interval estimation. Because classical statistics requires independent, identically distributed observations, while simulation data are generally correlated (Law and Kelton, 1982), the classical confidence interval formulas cannot be used. Schruben has developed a new approach based on standardized time series that can be used with virtually any simulation output data. All data points used are considered to be subdivided into b batches containing m data points, so that the total number of data points are $n=bm$. In Section 2 of Schruben (1983), the following classical/sum confidence interval estimator is developed:

$$I_{csum} = \overline{\overline{Y}} \pm H_{csum} = \overline{\overline{Y}} \pm t_{CL,df}\, \hat{\sigma}/\sqrt{n} \qquad (1)$$

where $\overline{\overline{Y}} = (1/n)\sum_{i=1}^{n} Y_i = (1/b)\sum_{i=1}^{b} \overline{Y}_{i,m}$

$\overline{Y}_{i,m}$ = average of the i^{th} batch of size m

t = value of the Student t distribution

CL = confidence level

df = degrees of freedom = $2b-1$

$\hat{\sigma}$ = estimate of σ, given by $\hat{\sigma}^2 = A/(2b-1)$ (2)

where $A = \sum_{i=1}^{b} [(12A_i^2)/(m^3-m) + m(\overline{\overline{Y}} - \overline{Y}_{i,m})^2]$

or $A = [12/(m^3-m)]\sum_{i=1}^{b} A_i^2 + m\sum_{i=1}^{b} \overline{Y}_{i,m}^2 - mb\overline{\overline{Y}}^2$

and A_i can be computed by the recursion

$$A_i(k+1) = A_i(k) + 1/2(kY_{i,k+1} - \sum_{j=1}^{k} Y_{i,j}), \quad k = 1, m$$

with $Y_{i,j}$ = j^{th} observation of the i^{th} batch,
and $A_i(o) = 0$.

If we assume an acceptable errorband around the computed mean, $y(1 \pm \epsilon)$, where we let the user choose ϵ, such as 0.05, then the equation for the halfwidth lets us estimate the total number of data points needed for the computed mean to be within this band:

$$n_t = [t_{CL,df}\, \hat{\sigma} / (E\overline{\overline{Y}})]^2 \qquad (3)$$

Any additional data points needed can be computed and the computation time can be estimated.

2.4 Truncation Point Detection

The development of a test procedure to detect the presence of initialization bias has been described in a paper by Schruben, Singh, and Tierney (1983). The truncation procedure used here differs from that only in using a weighting of k instead of $(1-k/n)$ as given in equation (3) of Schruben et al. (1983) and using the data points in reverse order. For a run of j observations, this changes the test statistic $\hat{T}$, given by equation (5) in Schruben et al., to

$$\hat{T}(j) = [(45/j^5)/\hat{\sigma}]\sum_{k=1}^{j} k^2(\overline{Y}_j - \overline{Y}_k) \ .$$

Now $Var[\hat{T}(j)] = j^5\sigma^2/45$, using $\hat{\sigma}$ as described in the previous section. $\hat{T}(j)$ can be computed by recursion:

$$\hat{T}(j+1) = \hat{T}(j) + [(2j+1)/6]\ [jY_{j+1} - \sum_{i=1}^{j} Y_i]$$

with $\hat{T}(o) = 0$.

This statistic is computed proceeding from the last data point toward the first. Once half the data points have been used, the scaled value of this statistic is compared to $3\hat{\sigma}$ after each data point is added.

$$\text{If } |(45/j^5)^{1/2}\,\hat{T}| < 3\hat{\sigma} \ ,$$

then no transient has been detected. When this test is failed for the last time, the data point just added and all data points that precede it in simulated time will be truncated. The last data point added before the test failed is the truncation point.

3. IMPLEMENTATION

The procedure was initially implemented for use with simulations written in FORTRAN and was written as a set of FORTRAN subroutines for an IBM PC. Assuming that a simulation program exists, the user must define a performance variable and store its values in a Y array in unlabeled COMMON. An outer loop has to be added to the simulation program and the run control subroutines IRC1 and IRC2 must be called before and after the loop as indicated:

```
    Call IRC1(N0,N,IPATH)
10  CONTINUE
    DO 100 I=N0,N
    .
    .     (BODY OF SIMULATION PROGRAM)
    .
100 CONTINUE
    CALL IRC2(N0,N,IPATH)
    IF (IPATH.EQ.1) GOTO 10
```

Subroutine IRC1 is used to initialize the procedure and to determine the type of problem to be solved. Therefore, the user is asked the following three questions:

"Enter the longest acceptable run time (min):"
"Is this a steady state(1) or terminating(2) simulation?:"
"Enter the relative stopping accuracy:"

The questions regarding the longest time and the type of simulation are easily answered. The relative stopping accuracy refers to the fraction of the average of the performance variable which sets the allowable error in that average, as used by equation (3). This requires specifying a confidence level that is assumed to be 90% in these routines. Also, the number of batches is fixed at 15.

After these initial runs have been made, IRC2 is called to estimate the number of data points required for a run of 20% of the longest acceptable run time. A number of checks are also carried out to prevent unreasonable parameters for the run control logic. A message tells the user the number of data points to be computed and the estimated time. The simulation program then takes over to compute these data values.

When these calculations are completed, IRC2 takes control again and calls one of two confidence interval estimating subroutines. For terminating simulations, the standard deviation is estimated using the classical confidence interval technique. The allowable relative error is computed and equation (3) is solved for the total number of runs needed. From that total number, the runs made so far are subtracted to determine the additional points needed.

For steady-state simulations, the data values are first checked for an initial transient and to determine a truncation point. If no such point is found, the logic attempts to double the available number of data points by requesting the simulation to continue its run. If a truncation point was found, only the steady-state data is used to determine any additional data points needed. The calculation of $(\hat{\sigma})$ is given by equation (2).

For either type of simulation, the programs also print the confidence interval computed. Before the simulation run is resumed, the run control logic performs a few checks and arbitrarily reduces the number of additional data points by 20%. It also uses the latest computation rate to estimate the time required to make these computations, and it prints the following message for the user:

"Additional runs will take an est. ... min.—Is that ok? (y/n):"

If the user answers with a 'y' the run proceeds; if it is 'n' the program asks

"Enter an acceptable run time (min):"

If the user enters a nonzero time, it will be converted into the needed number of data points and the simulation resumed. If zero, the message

"Evaluations aborted by user"

appears and the simulation is stopped. If more data points are needed, IRC2 causes the simulation program to compute the number needed.

When the total number of data points needed as calculated by equation (3) is less than the number computed already, the desired precision has been achieved and a message to that effect will be printed.

Each time a confidence interval is computed, the data array must be accessed. For terminating simulations, the data are needed twice to calculate $\hat{\sigma}$; for steady-state simulations, the data are accessed three times, once to determine the truncation point, and twice to determine $\hat{\sigma}$. A number of timing runs were made to estimate the time spent in the run-control subroutines. For steady-state simulations, these calculations took less than 3 sec per 1000 data points on an IBM AT with coprocessor.

4. INTERACTIVE OPERATION

The purpose of the interactive run control program is to provide the interactive user with information and control of the real-time execution of the simulation. The software written for the PC includes a clock printout before and after the simulation for runlength documentation. During the simulation, whenever the software evaluates past runs and estimates the need for additional simulations, the user is notified of additional real time needed for more simulations. He can modify that time as desired, except initially when the program begins to execute the first 20% of the time allocated by the user. The data representing the performance variables for the examples below are random numbers or M/M/1 queue waiting time data.

4.1 Terminating Runs

The information provided is slightly different between the steady-state and terminating cases. Figure 1 is an example of the information printed on the screen for terminating runs. In Figure 1, the user had allocated 1 minute maximum time for the runs. The printout shows the start time in minutes since midnight, and shows the computation rate as determined by evaluating the user's simulation several times. It then told the user that the first 20% of the allowable number of runs, 110, would require an estimated .18 minutes. When these runs were completed, evaluation of the confidence interval showed that the required precision had already been reached and the run was stopped.

```
    WAITING FOR EVALUATION OF TIMING RUNS STARTED AT   500.67 MIN.
         THE EST. COMPUTATION RATE IS    550.5 OBS./MIN.
FOR AN INITIAL   110 OF A MAX. OF    550 OBSERVATIONS, THE ESTIMATED
            COMPUTATION TIME IS     .18 MIN.
AFTER RUN   110, 90% CI =   .995 +/-   .045. ADD     0 RUNS.***
           SATISFACTORY RUN-LENGTH ACHIEVED AT   500.87 MIN.
             *** EVALUATIONS COMPLETE AT   500.87 MIN. ***
Stop - Program terminated.
```

Figure 1: Example of Screen Display

In the example shown in Figure 2, the user specified a 10-minute time limit for the runs. The run length control software estimated that this time would allow 4190 runs to be made and started with 20% or 838 runs. After completing these runs it determined that an additional 7014 runs would be needed. Since there is only time for a total of 4190, it evaluates these and reports on the confidence limits and the need for more runs.

```
    WAITING FOR EVALUATION OF TIMING RUNS STARTED AT   502.18 MIN.
         THE EST. COMPUTATION RATE IS    419.1 OBS./MIN.
FOR AN INITIAL   838 OF A MAX. OF   4190 OBSERVATIONS, THE ESTIMATED
            COMPUTATION TIME IS    1.97 MIN.
AFTER RUN   838, 90% CI =  1.114 +/-   .344. ADD  7014 RUNS.***
 ADDITIONAL RUNS WILL TAKE AN EST.   7.53 MIN. - IS THAT OK? (Y/N): y
AFTER RUN  4190, 90% CI =  1.395 +/-   .176. ADD  2464 RUNS.***
              TIMELIMIT REACHED WITHOUT DESIRED ACCURACY
             *** EVALUATIONS COMPLETE AT   511.48 MIN. ***
Stop - Program terminated.
```

Figure 2: Example of Insufficient Computation Time

The last example of controlling the length of terminating simulations (Figure 3) shows a case where the user had allocated 20 minutes for the runs. After 1690 runs, it is estimated that an additional 5414 runs should be made, which would require another 9.73 minutes. Figure 3 shows that the user agreed to this extension. After these runs have been made, the procedure estimates another extension of .28 minutes is required to make 158 more runs to reach the desired precision. After the user agreed, the runs were successfully completed.

```
    WAITING FOR EVALUATION OF TIMING RUNS STARTED AT   513.94 MIN.
         THE EST. COMPUTATION RATE IS    422.6 OBS./MIN.
FOR AN INITIAL  1690 OF A MAX. OF   8451 OBSERVATIONS, THE ESTIMATED
            COMPUTATION TIME IS    3.97 MIN.
AFTER RUN  1690, 90% CI =  1.348 +/-   .277. ADD  5414 RUNS.***
 ADDITIONAL RUNS WILL TAKE AN EST.   9.73 MIN. - IS THAT OK? (Y/N): y
AFTER RUN  6021, 90% CI =  1.385 +/-   .140. ADD   158 RUNS.***
 ADDITIONAL RUNS WILL TAKE AN EST.    .28 MIN. - IS THAT OK? (Y/N): y
AFTER RUN  6147, 90% CI =  1.384 +/-   .129. ADD     0 RUNS.***
           SATISFACTORY RUN-LENGTH ACHIEVED AT   528.48 MIN.
             *** EVALUATIONS COMPLETE AT   528.48 MIN. ***
Stop - Program terminated.
```

Figure 3: Example of User Interaction

4.2 Steady-State Runs

The interaction of the runlength control program with the user during steady-state simulations is very similar to that in the terminating case. Since the procedure must estimate a truncation point and then disregard all data points prior to that point as unwanted transient data, the truncation point is given each time a confidence interval is computed.

As an example, assume the user set a time limit of 4 minutes on the simulation run time. The printout shown in Figure 4 informs the user of the computation rate and the time required to compute the first 20% of the allowable data points. When 192 data points are calculated, the truncation point (TP) and the confidence interval are given. The program also requests more time (.12 minutes) to compute additional data points. These evaluations lead to a successful conclusion of the simulation. The data values generated for this case are plotted in Figure 5.

```
    WAITING FOR EVALUATION OF TIMING RUNS STARTED AT   727.89 MIN.
         THE EST. COMPUTATION RATE IS    240.8 OBS./MIN.
FOR AN INITIAL   192 OF A MAX. OF    963 OBSERVATIONS, THE ESTIMATED
            COMPUTATION TIME IS      .59 MIN.
    OBS.    192  TP=    103  90% CI =   .9472 +/-    .0566
 ADDITIONAL RUNS WILL TAKE AN EST.    .12 MIN. - IS THAT OK? (Y/N): y
    OBS.    222  TP=    103  90% CI =   .9387 +/-    .0436
            SATISFACTORY RUN-LENGTH ACHIEVED AT   728.86 MIN.
              *** EVALUATIONS COMPLETE AT   728.86 MIN. ***
Stop - Program terminated.
```

Figure 4: Screen Messages During Simulation

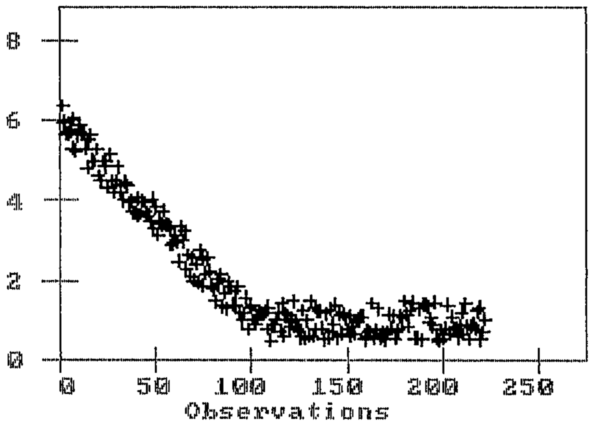

Figure 5: Data with Transient Used in Figure 4.

Another example shows the messages generated by the program to indicate that the time limit set by the user was not adequate to estimate the confidence within the desired accuracy. The printout is shown in Figure 6. Here, as in the preceding section, the logic computes until the allowed time is exhausted and then reports the truncation point and the confidence interval. The first 2000 data values used in the computations are shown in Figure 7.

5. SUMMARY AND CONCLUSIONS

This paper has described an approach and implementation to control simulation run length and has demonstrated how this can be done interactively. The procedure works both for terminating simulations and for steady-state simu-

```
    WAITING FOR EVALUATION OF TIMING RUNS STARTED AT   735.85 MIN.
         THE EST. COMPUTATION RATE IS    433.7 OBS./MIN.
FOR AN INITIAL   867 OF A MAX. OF   4336 OBSERVATIONS, THE ESTIMATED
            COMPUTATION TIME IS     1.88 MIN.
    OBS.    867  TP=    418  90% CI =  1.4952 +/-    .4604
 ADDITIONAL RUNS WILL TAKE AN EST.   6.91 MIN. - IS THAT OK? (Y/N): y
    OBS.   3920  TP=    336  90% CI =  1.5568 +/-    .3112
 ADDITIONAL RUNS WILL TAKE AN EST.    .94 MIN. - IS THAT OK? (Y/N): y
    OBS.   4336  TP=    347  90% CI =  1.4940 +/-    .2638
              TIMELIMIT REACHED WITHOUT DESIRED ACCURACY
             *** EVALUATIONS COMPLETE AT   745.79 MIN. ***
```

Figure 6: Messages for Example with Insufficient Data

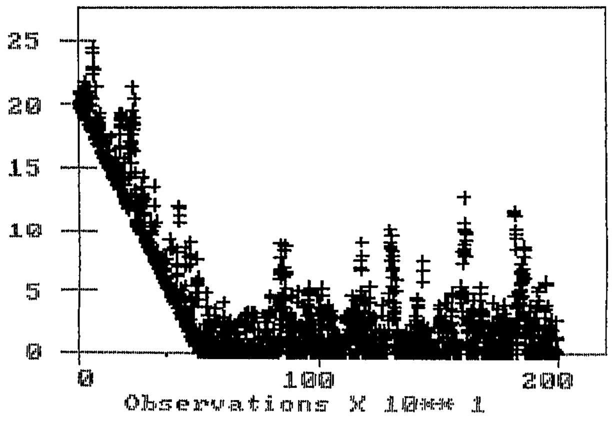

Figure 7: Data Used in Example of Figure 6.

lations where transients must be first detected and removed from the rest of the data.

The FORTRAN implementation consists of less then 250 lines of code and can be added to a simulation using six statements, defining a performance variable, and storing its values as they are computed into a vector in unlabeled COMMON. When the procedure determines the confidence interval, it reads this vector once to estimate the truncation point and two more times to estimate the standard deviation. The time required to make the computations to determine the proper run length is insignificant.

A Random Number model and M/M/1 data were used to demonstrate the procedure. For steady-state simulations, transients were introduced to show that the procedure can find the end of a transient period. The results show that the user can control the running of simulations and produce simulation results of known precision. This interactive aspect can be very helpful to simulators in scheduling the use of their own time, since the estimate of computation rate makes it possible to predict the length of a run or an added increment.

REFERENCES

Heidelberger, P. and Welch, P.D. (1983). Simulation run length control in the presence of an initial transient, *Operations Research* **31,6**, 1109-1144.

Law, A.M. and Kelton, W.D. (1982). *Simulation Modeling and Analysis.* McGraw-Hill, Inc. ISBN 0-07-036696-9, 1982.

Schruben, L. (1983). Confidence interval estimation using standardized time series, *Operations Research* **31,6**, 1090-1108.

Schruben, L. (1985). Overview of standardized time series. In: *Proceedings of the 1985 Winter Simulation Conference,* December 11-13, 1985, 115-118.

Schruben, L., Singh, H., and Tierney, L. (1983). Optimal test for initialization bias in simulation output, *Operations Research* **31,6**, 1167-1178.

AUTHORS' BIOGRAPHIES

RALPH R. DUERSCH is involved in improving simulation languages by using interactive graphics and developing automatic programming software and dynamic workflow representations. Dr. Duersch has been active in simulation and Decision Support Systems during most of his career. He is familiar with a number of simulation languages and has used them in numerous applications. He is the author of over 50 papers and internal reports. Dr. Duersch is a licensed professional engineer in the State of New York and a Senior Member of the IEEE. He also serves as an Adjunct Professor in the Institute of Industrial Administration and Management at Union College.

General Electric Company
Corporate Research and Development
P.O. Box 8, K1-5B32
Schenectady, NY 12301
(518) 387-6406

LEE W. SCHRUBEN received his undergraduate degree in engineering from Cornell University and a Masters degree from the University of North Carolina. His Ph.D. is from Yale University. Prof. Schruben has been on the Cornell faculty in Operations Research and Industrial Engineering for the past nine years. Prior to coming to Cornell he was the Associate Director of the Health Systems Research Division of the Medical School at the University of Florida. His research interests are in the statistical design and analysis of large scale simulation experiments. His consulting activities have been primarily in the area of manufacturing systems simulation.

Lee W. Schruben
S.O.R.I.E.
342 Upson Hall
Cornell University
Ithaca, NY 14853
(607) 255-9133