Proceedings of the 2008 Winter Simulation Conference
S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.

USING SLITHERS OF SIMULATION IN A NEW APPROACH FOR INTELLIGENT INITIALIZATION OF NON-TERMINATING SYSTEMS

Philip G. Brabazon

Nottingham University Business School
Jubilee Campus, Wollaton Road
Nottingham, NG8 1BB, UK

ABSTRACT

Reduction and even avoidance of the initial transient is known to be possible using intelligent initialization but it is not an approach that has been actively researched for some time. Using ideas from complexity science and informed by the recently developed equation-free technique a new method for identifying initializing conditions is developed. Many short bursts of simulation, called *slithers*, are used to construct the dynamic function of a system. Importantly, the function reveals the point attractor of the dynamic system to which it will evolve and the conditions at the attractor define the initializing conditions for future simulation runs. The method is demonstrated by application to a queuing network.

1 INTRODUCTION

The warm-up problem in non-terminating systems is familiar to simulation analysts due to concerns over bias in estimates of output variables from the initial transient. Sandikçi and Sabuncuoğlu (2006) classify approaches to handling the problem into three groups – truncation heuristics, intelligent initialization and other general methods. In intelligent initialization the transient is avoided (or greatly reduced) by starting a simulation in conditions that are representative of the system's steady-state. The method developed in this paper fits into this category.

Whereas truncation methods continue to be developed and new ones proposed (e.g Robinson 2007) intelligent initialization has received little attention since the work of Kelton (1989). Although it was shown to be a beneficial tactic, the conclusion was that it would be hard to apply to complex models (Law 1983). Undaunted by the pessimism an attempt is made here to develop a fresh approach for intelligent initialization.

The new approach borrows from complexity science and is inspired by the technique of equation-free modeling which uses bursts of detailed simulations to reveal empirically the macroscopic dynamics of such processes as chemical reactions and networks of neurons (Kevrekidis et al. 2004, Laing 2005).

The new approach is used to estimate the dynamic behavior of a queuing network by constructing its dynamic function from the output of many short bursts of simulation, called *slithers*. Importantly, the approach estimates a variable's attractor to which it will move. The conditions at the attractor are the initializing conditions for future simulation runs. An example of the dynamic function created by the approach is shown in Figure 1.

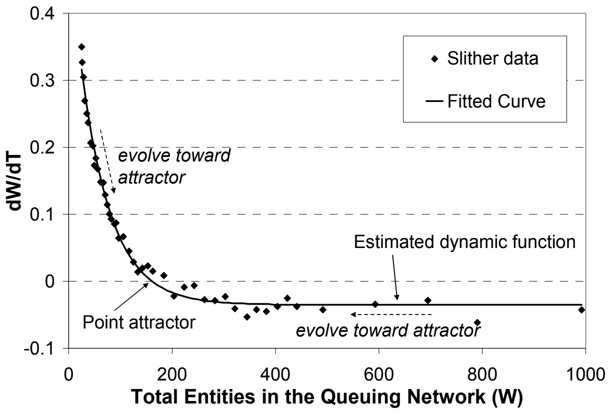

Figure 1: Estimated dynamic function

This paper introduces the approach to finding the initializing conditions and takes the reader through the process of developing the slither method by applying it to a simulation model of a queuing network. The initialization conditions identified are the number of entities in the network (*W*) and the spread of the entities across the queues. A section of the paper characterizes the simulation model's steady-state behavior to give a picture of the conditions the method is aiming to identify. The paper closes with a discussion and thoughts on future development of the slither method.

2 DESCRIPTION OF THE SLITHER APPROACH

Consider a simulation model which has an output measure W that varies over time (such as the number of orders being processed in a factory operating 24/7). In the case where the model exhibits steady-state behavior, and assuming this is when $W>0$, then if the model is re-run from empty many times (starting at $t = 0$, $W_0=0$) and its transient behavior is consistent, the many time series of W_t can be averaged to give $\overline{W_t}$. By dividing it into many small time segments, dt, the rate of change can be found, $\dot{W_t}$, and also the average W in each segment, $\overline{W_{dt}}$. By plotting $\dot{W_t}$ against $\overline{W_{dt}}$, as in Figure 2, the value of W_s at which W is stationary in the simulation can be identified since it is the intercept on the W axis. Future simulations can then be initialized at W_s.

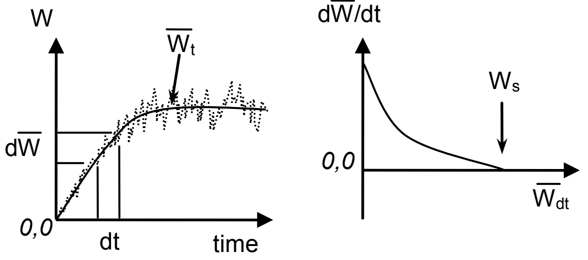

Figure 2: Conceptual Approach

In effect, the above process has constructed part of the dynamic function $dW/dt=f(W)$ but, using the method developed in this paper the full function can be pictured.

In summary, the method entails initializing a simulation model with a known value of W and then running the model for a short period only, to create a simulation *slither*. The value of W is monitored throughout the slither to enable the average W and the rate of change of W to be calculated. Therefore a slither gives a $(W,\dot{W})$ pair and by running slithers at a range of W values the full dynamic function $f(W)$ can be constructed empirically as shown in Figure 1.

2.1 Slither method for a queuing simulation

For a queuing system with many queues (at servers), if W is the number of entities in the model, the initialization problem is not solely a question of identifying W_s but also of determining how the entities are distributed across the m queues $(w_1,w_2,\ldots,w_m)$. Both W_s and the distribution must be appropriate for the simulation model to be in steady-state and therefore the method should be capable of estimating W_s and w_i.

In a system with m servers (with queues), a set of initialization conditions J are predefined. Each initialization condition j in the set constitutes a distribution vector $\mathbf{F_j}=(f_{j1},f_{j2},\ldots,f_{jm})$ which specifies the *fraction of entities at each queue*, for which $\sum_{i=1}^{m} f_{ji} = 1$, and the *initial number of entities in the system* W_j. The distribution vector $\mathbf{F_j}$ is an estimate of the steady-state distribution and therefore is kept constant across the set of J initialization conditions.

To initialize the server queues in a condition j, the entities at a server is the product of the total W_j and the initial fraction for that server as specified in the vector $\mathbf{F_j}$. To illustrate, if $W_j = 150$ and $f_{j3} = 0.13$, the entities at server 3 is the rounded down value of *19*. One entity is put into the server and the remaining 18 are queued.

Once initialized, a slither k is run for T units of time (Figure 3). At each time increment the total entities in the system is logged to create a dataset of T observations for the slither. The average number of entities throughout the slither, $\overline{W_k}$, and rate of change of entities along the slither, $\dot{W_k}$, are determined from the dataset, the latter found using linear regression.

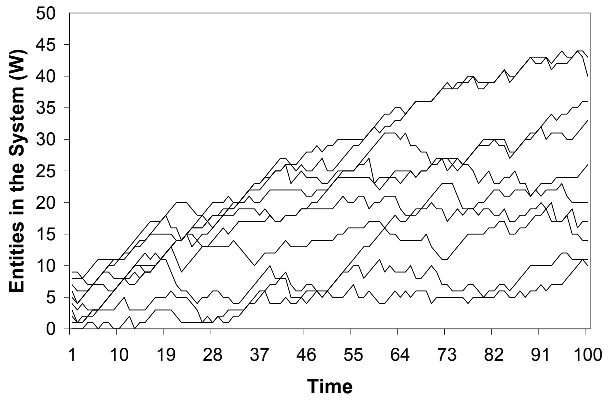

Figure 3: A simulation slither (as a stack chart of entities at each of 10 servers)

At each initial condition j the model is run for N slithers. Once a slither terminates, the entities are removed and the model reinitialized with the $\mathbf{F}_j$ distribution and re-run for the next slither, and so on. Differences between the N slithers are created by the randomness in the model, e.g. inter-arrival times and serving times. Note, a pseudo random number stream must not be restarted between each slither of condition j as this would make all slithers identical, but can be restarted between initializing conditions if so wished.

The output from each condition j is a pair of points - the average number of entities and the average gradient:

$$\overline{W_j} = \frac{1}{N}\sum_{k=1}^{N}\overline{W_k} \qquad \dot{W_j} = \frac{1}{N}\sum_{k=1}^{N}\dot{W_k}$$

Once the set of J conditions have been run a plot is made of the ($\overline{W}_j,\dot{W}_j$) pairs and a curve fitted to give the empirical estimate of *f(W)*. Figure 1 presents an estimated *f(W)* which shows that at low W there is a net increase of entities in the queuing system. The rate declines as W increases and is zero at the intercept on the axis, this being the estimate of W_s, the total entities in the system at which the model is stationary (as long as the vector **F** is representative of steady-state conditions). When $W > W_s$ the rate of change is negative and so the system has a net outflow of entities. The rate tends toward an asymptote which is the difference between average arrival rate and the average processing rate.

Like all simulation based analysis there are statistical issues to address, such as the number of slither replications at each W in order to improve the resulting estimate of the macro function.

3 QUEUING MODEL FOR TESTING THE SLITHER METHOD

To develop and test the approach it is applied to the simulation model of a queuing system pictured in Figure 4. The system has 10 identical servers arranged in sequence. Each can service one entity at a time and has an unlimited queue. All have negative exponential service times with average service rate $\mu = 1$. Entities enter the system at a fixed rate of $\lambda = 1/1.05$.

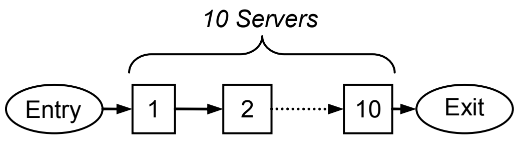

Figure 4: Test Model

3.1 Performance of the test model

In the absence of theoretical quantification of the model's performance, a single very long run is used to estimate its steady-state metrics. No formal test for determining the length of the transient is applied but the run, which started from empty, is truncated by a generous 400,000 time units. From this point a set of 5,000 observations of the number of entities at each server (which includes the entity being served) are taken at intervals of 500 time units to construct the histograms in Figures 5 and 6.

The mean, mode and median of the observations are 168.8, 157 and 166 respectively. For the data, 90% of observations lie in the range from 122 to 225 and 95% in the range 115 to 240.

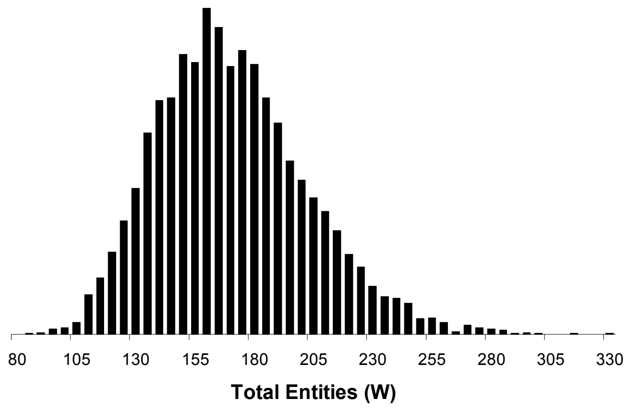

Figure 5: Distribution of Total Entities in the system

The average proportion of entities at each of the 10 servers is plotted in Figure 6 (with the variance shown). This figure describes the steady-state distribution the slither method will try to estimate.

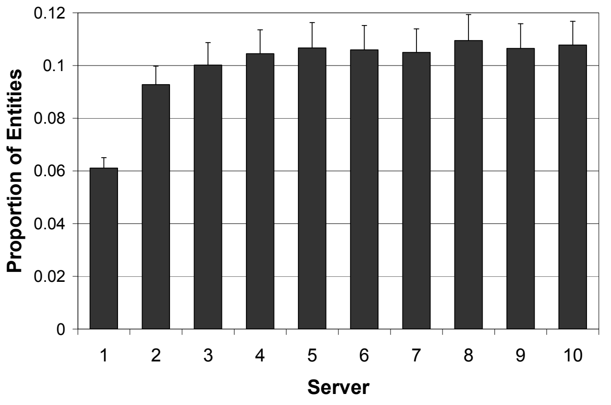

Figure 6: Average proportion of Entities at each server (with variance shown)

To appreciate the range of entity distributions across the servers experienced during steady-state, the first 250 observations are plotted as cumulative curves in Figure 7.

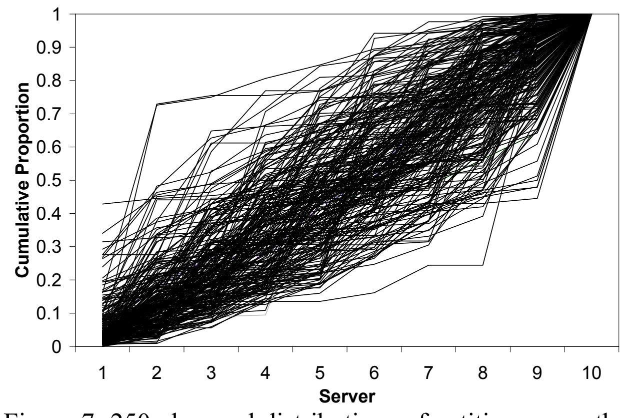

Figure 7: 250 observed distributions of entities across the servers

4 STAGE 1: IDENTIFYING W_S

The slither method identifies the initializing conditions in two stages. The first finds W_s, and the second estimates $\mathbf{F}_s$ (in section 5). However, finding W_s requires a guess at $\mathbf{F}_s$. It would be ideal if the slither method can tolerate an error in $\mathbf{F}_s$ and its ability to do so is studied in section 4.5. But to begin, I look at how the slither method estimates W_s when using good guesses of $\mathbf{F}_s$ and I do this by borrowing the 250 distribution vectors pictured in Figure 7 which were observed in the long run.

The four parameters controlling the method are in Table 1 along with their settings. A *J* set comprises 46 *j* conditions, with the first *j* condition having up to 10 entities initially in the system, and the last *j* condition having up to 1000. Alternative settings for the number and duration of slithers are explored later.

Table 1: Slither method parameters.

Parameter	Setting(s)
Size of *J* set	250
Initial # of entities in the system (W_j)	From 10 to 1000 in 46 steps
Initial distribution vectors ($\mathbf{F}_j$)	Figure 7
#of slithers at each condition *j*	100
Slither duration at each condition *j*	100

If a server is busy at the start of a slither, the randomized serving time is reduced for this entity by a fraction between 0 and 1, randomly sampled from a uniform distribution. The same approach is used for first entity entering the queuing system.

4.1 Using a static distribution of entities

In the first implementation of the method all slithers in each *j* condition are re-initialized with the $\mathbf{F}_j$ distribution.

The form of curve fitted to the 46 *j* conditions that comprise a *J* set is an exponential, $y = Ae^{-Bx} + C$ (as in Figure 1). Of the resulting 250 curves in Figure 8, 51 have a positive asymptote and hence no intercept on the *W* axis. The average intercept for the remainder is 101.4.

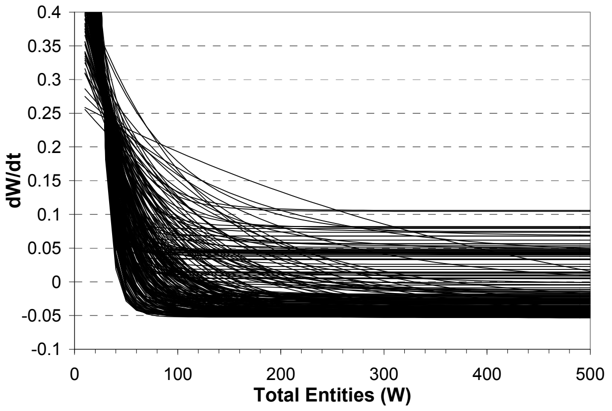

Figure 8: Exponential curves for 250 conditions

The intercept distribution is plotted in Figure 9. The average is at the low end of the value observed in the long run and the spread is wider and noticeably more positively skewed than the observed spread (see Figure 5).

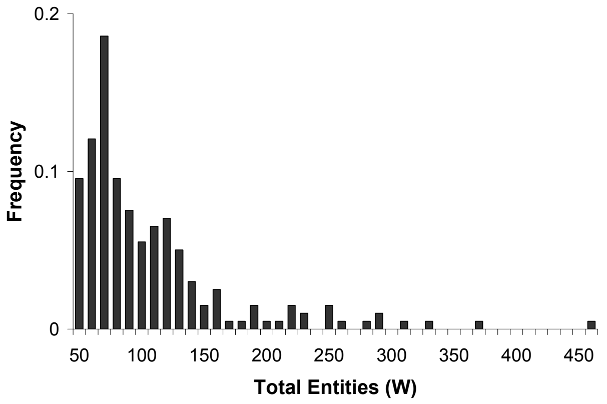

Figure 9: Intercept distribution from 199 conditions

4.2 Using an evolving distribution of entities

The method is modified to allow the entity distribution to evolve across the slithers of each *j* condition. The first slither is initialized using vector $\mathbf{F}_j$ but from then on the distribution at the end of a slither is used as the initial distribution for the next slither. The typical impact on a *J* set is illustrated in Figure 10 which shows that the intercept is shifted by this modification.

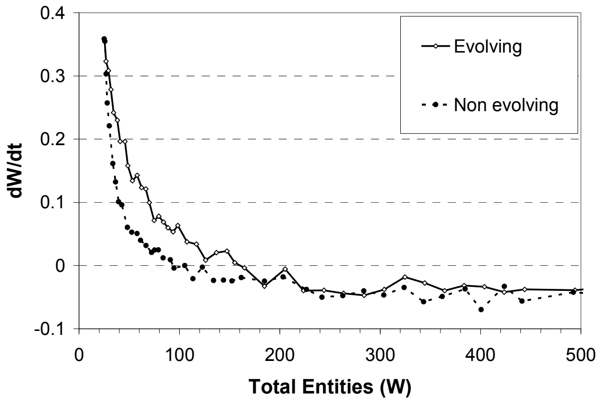

Figure 10: Evolving and non evolving *J* sets having the same initial conditions

In the set of curves for all 250 distributions, shown in Figure 11, the number of outliers with a positive asymptote is reduced to 2. For the 248 that do cross the axis the average intercept is higher at 140.7 the spread of which is narrowed as can be seen by comparing Figure 12 to Figure 9.

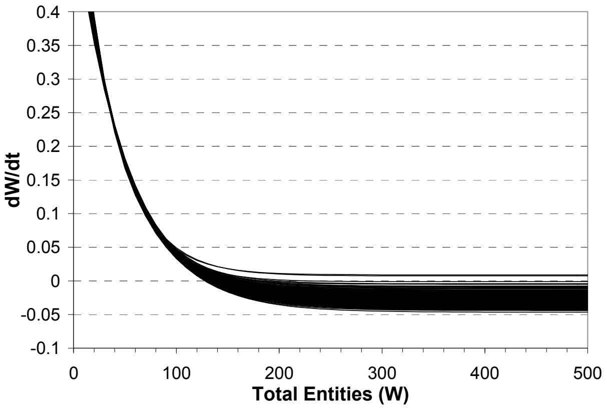

Figure 11: Exponential curves for 250 conditions using the evolving method

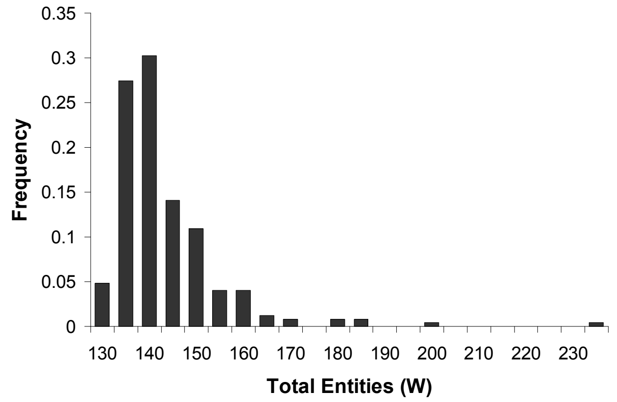

Figure 12: Intercept distribution for 248 conditions, using the evolving method

4.3 Alternative curve function

The choice of function to fit to the data alters the estimate of the intercept. This is apparent by comparing the intercepts from fitting a Rodbard function to the values from the exponential function. The Rodbard function has an horizontal asymptote and can form a similar shape to the exponential, but its 'elbow' is less severe. Its form is:

$$y = d + \frac{(a-d)}{\left(1+(x/c)^b\right)}$$

The Rodbard curves in Figure 13 which are fitted to the 250 J sets look similar to the exponential curves. It is also the case that the same two J sets that had no intercept with the exponential function have none with the Rodbard. Other than for these two cases the Rodbard curves are better fits, having higher R^2 values (Figure 14).

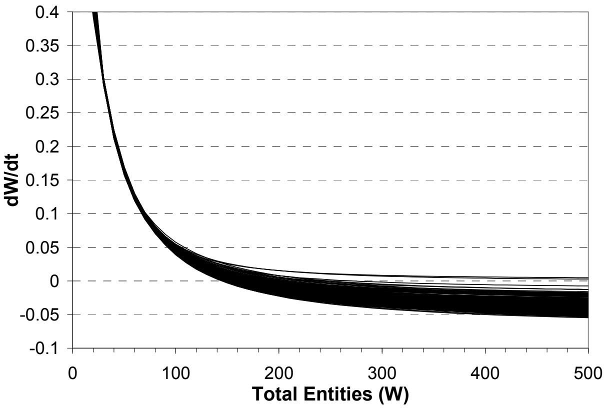

Figure13: Rodbard curves for 250 conditions

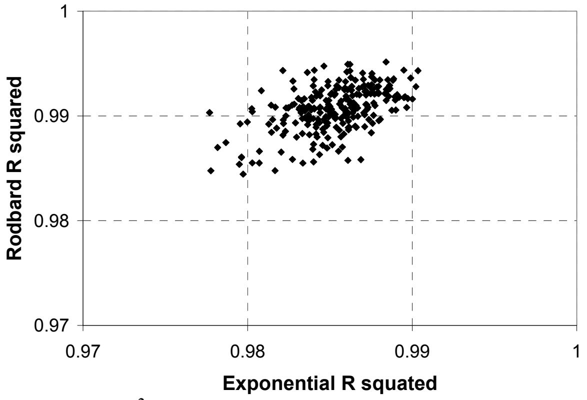

Figure 14: R^2 pairs for Rodbard and Exponential curves

The average intercept from the 248 Rodbard curves is 169.7 and their distribution is in Figure 15.

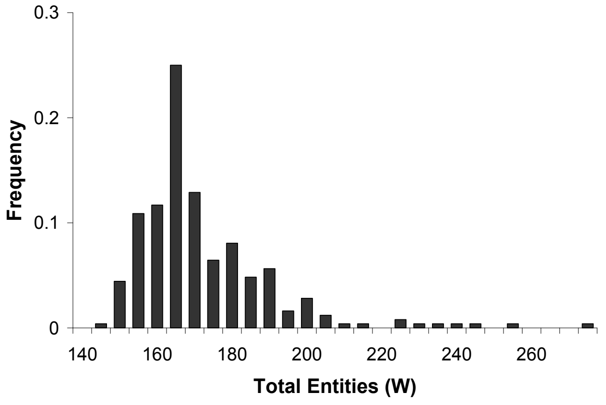

Figure 15: Distribution of intercept values for the Rodbard function

4.4 Different settings

An important issue is whether the method is sensitive to its parameters, specifically the number of slithers and their duration. In Figure 16 the results from three J sets are plotted which differ only in the duration of the slithers (i.e. they are initialized by the same distribution vector $\mathbf{F_j}$). A conclusion from eyeballing these plots is that the points in a J set become smoother as slither duration increases and there is no obvious shift in the intercept.

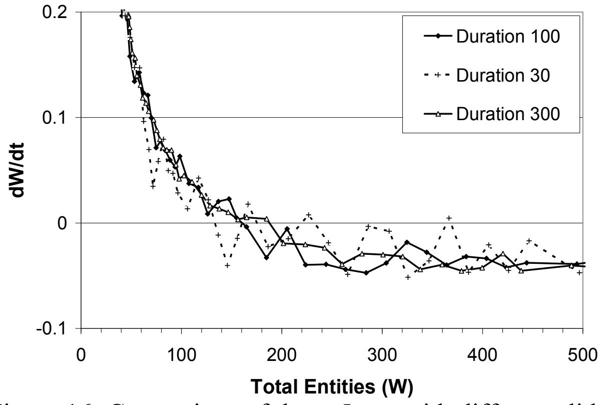

Figure 16: Comparison of three J sets with different slither durations

When Rodbard curves are fitted to these three plots, the intercept values are:

- Slither Duration 300: intercept = 162.7
- Slither Duration 100: intercept = 161.0
- Slither Duration 30: intercept = 149.5

The slither data in the region of the intercept is enlarged in Figure 17 and when eyeballing the plots there is no obvious shift due to slither duration. Therefore the lower Rodbard intercept can be judged to be an artifact of the curve fitting process. The conclusion is that slither length (and number of slithers) alters the confidence in the estimate of W_s and greater noise in the data may cause the curve fitting to introduce a systematic bias (Table 2). This is an issue for further study.

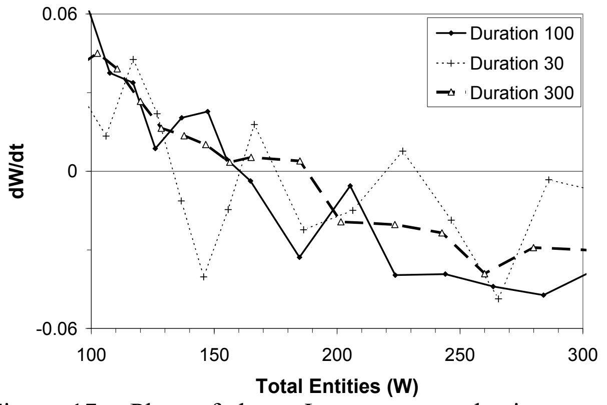

Figure 17: Plots of three J sets near to the intercept, showing vertical confidence intervals

Table 2: Slither parameters and Rodbard intercepts.

Slithers in *j* condition		Rodbard Intercept		
Number	Duration	# Curves without	Mean	Std. Dev.
100	300	0	160.4	3.7
100	100	2	169.7	11.5
100	30	43	178.3	54.1
30	100	12	181.7	50.7

4.5 Out-of-envelope initializations

The above results show the slither method can make a good estimate of W_s when initialized with valid $\mathbf{F}_s$, but can it do so when initialized with unrepresentative distributions from outside of the steady-state envelope?

To establish the envelope boundary the 5000 observations from the long run are searched for the *minimum cumulative fraction* observed at server 1, then at server 2 and so on up to server 9 (as the value is 1 for server 10). Then the set is searched for the *maximum cumulative fraction* observed at each server. The resulting 18 points are connected to form the upper and lower boundaries of the envelope, shown in Figure 18.

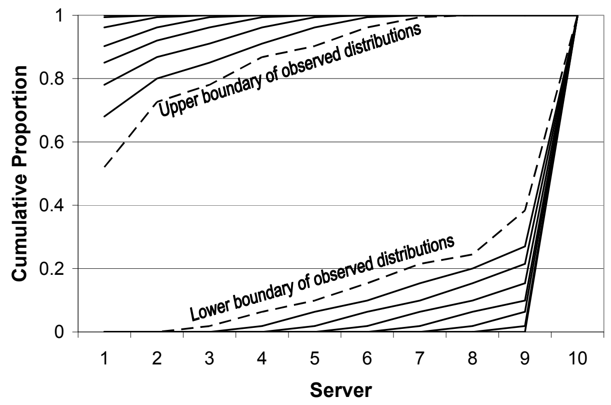

Figure 18: Distribution Envelope and out-of-envelope distributions

Also in Figure 18 are 14 artificially created out-of-envelope distributions which can be considered as forming two groups – the *front loaded* group and the *back loaded* group. In the 7 front loaded distributions (upper left) the entities are distributed over the upstream servers and in the extreme case all entities are allocated to server 1. Conversely, in the back loaded group the most extreme distribution has all entities allocated to server 10. Note other forms of out-of-envelope distributions can be imagined, such as having all entities at any of the servers.

The Rodbard curves for the 14 out-of-envelope distributions and both upper and lower boundary distributions are shown in Figure 19. The two boundary and seven back loaded distributions intercept the W axis in a tight cluster but none of the front loaded group do so.

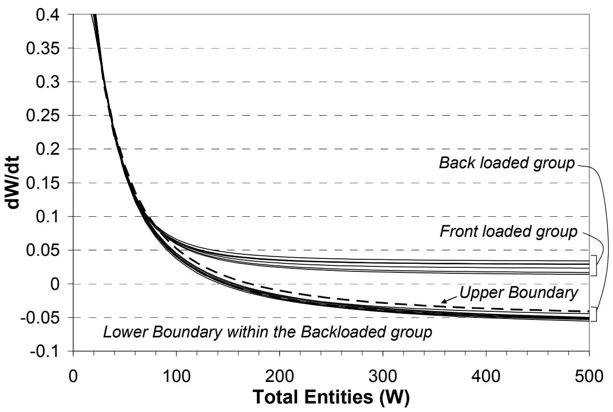

Figure 19: Rodbard curves for boundary and out-of-envelope distributions

The curves for the two groups are similar when the system is initialized with a low number of entities. As the number of entities rises, the queue at the server with the highest fraction becomes long, to the point when this server is choked throughout the set of slithers. The choking of the first server has a greater impact than when the last server is choked, hence the divergence of the curves for the two groups.

It was noted earlier (Table 2) that not all of the 250 legitimate distributions gave intercepts. Therefore a conclusion is the first stage of slither method to estimate W_s cannot cope with all distributions, but it can produce good estimates with some out-of-envelope cases so does not rely on an accurate guess of $\mathbf{F}_s$.

5 STAGE 2: IDENTIFYING THE STATIONARY DISTRIBUTION OF ENTITIES

Having estimated W_s the second task is to estimate the steady-state distribution, $\mathbf{F_s}$. The slither method lends itself to doing this in two ways.

5.1 Evolving distribution

It appears the evolution process within the method causes the distribution to move towards the long run average described in Figure 6. This is evident when the method is initialized with 170 entities and run for 100 slithers. The cumulative distribution across the servers at the end of each slither is observed and, after 100 slithers, the average is calculated. The resulting average cumulative distributions, having initialized with the 250 distribution vectors, are plotted in Figure 20.

Repeating this process at several entity volumes shows the shape of the cumulative distribution to have some sensitivity to the initial volume, but the resulting distributions are well within the upper and lower boundaries determined from the long run (Figure 21).

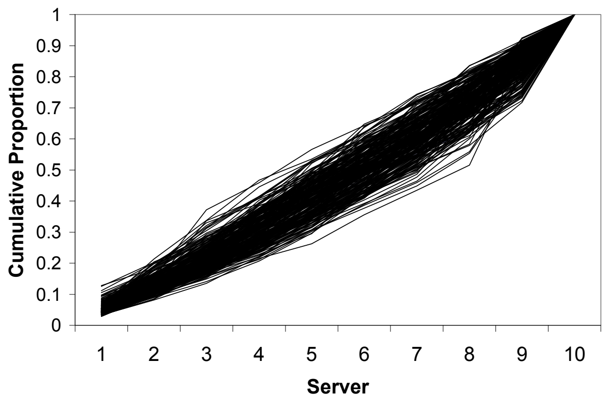

Figure 20: 250 evolved cumulative distributions of entities across the servers (initialized with W=170)

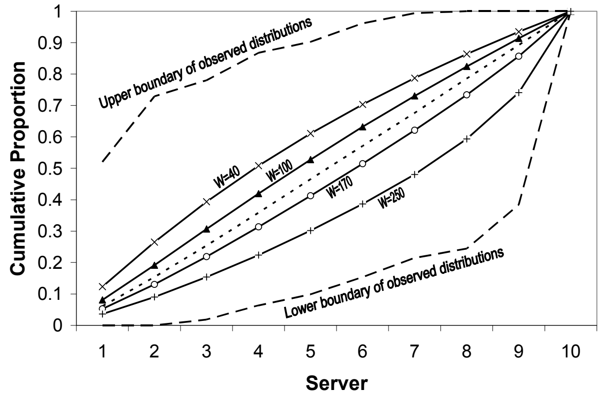

Figure 21: Cumulative entity distributions at various initialization W (each averaged from 250 cases)

Although the approach is resilient to the initial volume of entities, the tests with out-of-envelope cases (section 4.5) found front loaded distributions unable to evolve when the number of entities is high, due to server choking. When the entity level is not too high, a set of 100 slithers is sufficient for the out-of-envelope distributions to evolve as shown in Figure 22 in which the model is initialized with 40 entities.

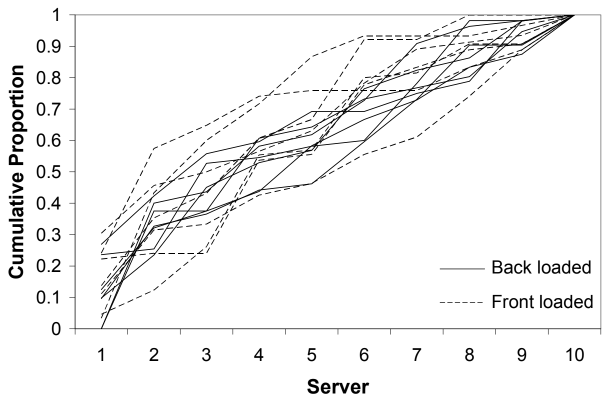

Figure 22: Cumulative entity distributions for 14 out-of-envelope distributions, initialized at W=40

5.2 Dynamic function of the distribution

Rather than use the evolution process to estimate the steady state distribution, an alternative is to analyze the distribution as a dynamic process and estimate it empirically in a similar way to estimating *f(W)*. The process describe in section 2 is adjusted for the purpose of estimating the dynamic function for the cumulative fraction at each server, $dcf_i/dt=f(cf_i)$.

As earlier, a set *J* of initializing condition is predefined with each condition, *j*, constituting a distribution across the servers and an initial number of entities in the system. A key difference from above is the initial number of entities is kept constant across the *j* conditions, since this is the estimate of the steady-state volume in the system, W_s.

A difficulty of adapting the approach is the coupling between servers. This prevents simple control over the cumulative fraction at one specific server, and therefore it is not obvious how to study a server across a range of cumulative fractions in a systematic way. The approach taken is to use the 250 observed distributions even though this can be expected to add noise to the results.

At each time increment during a slither for condition *j* the cumulative fraction at each server is logged, cf_i. At the end of the slither the data set is used to calculate the pair ($\overline{cf}_{ji}, \dot{cf}_{ji}$) for each server. All *J* conditions are run and all pairs for a server are plotted so as to estimate $f(cf_i)$.

The modification of the evolving distribution across slithers is not used in this analysis so every slither is reset to the initial distribution vector $\mathbf{F_j}$. This is done because the aim is to estimate the rate of change of the cumulative fraction at a specific fraction.

From the plot for the first server a function is apparent (Figure 23) but it is unclear what form of curve to fit to assist in estimating the intercept. From eyeballing the data, the intercept would be less than 0.5 which is below the long run value for the first server (Figure 6).

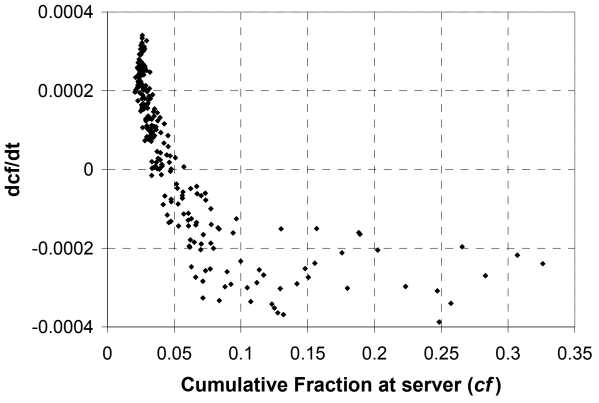

Figure 23: Slither data for server 1 (system initialized at *W=170*)

The plots for servers 2 and 9 hint at a function but the scatters are high (Figure 24). The plot for the fifth server (not shown) has no discernable pattern.

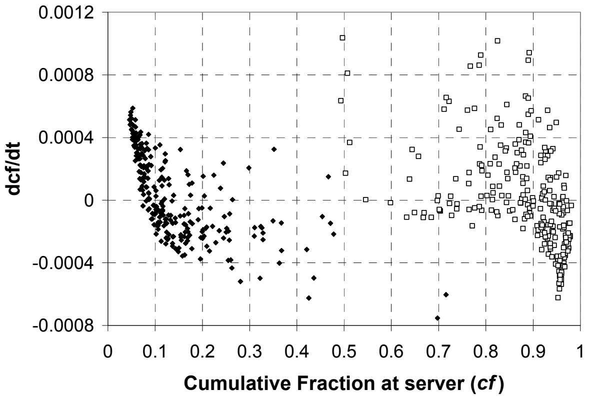

Figure 24: Slither data for servers 2 and 9 (system initialized at *W=170*)

6 DISCUSSION

Simulation models are dynamic systems and it is appropriate to apply complexity theory to interrogate and interpret their behavior. The concept of steady-state used in discrete event simulation is synonymous with the complexity concept of attractor.

It is useful to know the dynamic function for output variables since it not only identifies the attractor, but also indicates the rate at which the simulation approaches steady-state. It is known that a model started from empty can reach equilibrium more quickly than when initialized with more than the mean number of entities (Pawlikowski 1990) and examination of the dynamic function for this model predicts this would be the case (see Figure 1). Pawlikowski (1990) also notes that different variables tend to steady-state with different rates and constructing the dynamic function for each variable would alert the analyst to this fact.

The aim has been to use slithers of simulation to estimate empirically the dynamic function *f(W)* and so identify the point attractor for the number of entities in a queuing system, W_s, with which to initialize future runs. From the evidence presented above, it can be concluded the approach succeeds, particularly when the distribution of entities is allowed to evolve across a slither set. When this is enabled the estimates of W_s, even when initialized with unrepresentative distributions of entities across the servers, are close to the average of 168.9 observed in a long run, and since there is no theoretical analysis of the queuing system the long run statistics are assumed to be correct. As the slither length is increased the estimate moves down (to 160.4 for a slither duration of 300) and perhaps it is tending towards the long run mode (of 157) rather than the mean. This is a question for further investigation.

The attempt to use the slither method to construct a dynamic function for the distribution of entities across the servers is less successful and requires further consideration. Perhaps a general conclusion to be drawn is that, in applying the method to a model, there will be the challenge of framing the dynamic function in a form for which initializing conditions can be controlled and varied systematically.

The development of the slither method laid out in this paper tackled the identification of W_s as the first stage before estimating the steady-state distribution across servers. These two steps cannot be fully separated as the construction of the dynamic function to reveal W_s requires an estimate of the entity distribution and vice versa. However, the insights from section 5.1 suggest the estimation of distribution is not strongly sensitive to the value of W_s and legitimate distributions can be estimated from a wide range of initial W values. Therefore the conclusion is, when applying the slither method to a queuing system, the stages should be reversed and the stages should be:

1. across a range of W values and a range of entity distributions, run the model to evolve the distribution. Identify the envelope for $\mathbf{F}_s$.
2. use distributions from within the envelope in the process of estimating W_s.
3. After estimating W_s, rerun the evolving model to finalize the estimate of $\mathbf{F}_s$.

The suspicion raised in section 4.4 that the curve fitting process may be biased by noise in the slither data is an issue for further study. It would be ideal if the method could generate valid initializing conditions using a small set of short slithers.

There are aspects of the method to be further explored and refined, particularly if the method is to be formalized as a generic approach. Here it has been developed and demonstrated on one queuing model and its translation and adaptation to simulation models of other types of operations requires thought and development.

It must be acknowledged that although the method shows promise in intelligently identifying initializing conditions, the test of whether these conditions shorten or eliminate the initial transient has not been performed. A systematic test across a range of models of the efficacy of the slither method as a technique for intelligent initialization is for future work.

Research on intelligent initialization techniques appears to have reached a dead-end nearly twenty years ago, but maybe the complexity perspective taken in this study can open a new avenue. A perceived major problem with using intelligent initialization for multiple response variables in complex models was that it would require the estimation of multivariate distributions which is a challenging task (Law 1983). However, if the problem can be reshaped as one of determining the dynamic function for one or more variables, it may be more tractable.

ACKNOWLEDGEMENTS

My thanks to Jon Shapiro, Alan McKane and David Broomhead for organizing the EPSRC funded summer school *From Individual to Collective Behaviour in Large-Scale, Complex Systems,* for which a bursary to attend was received from the *Journal of Statistical Mechanics: theory and experiment*. The approach was first conceived within a group project at the event involving Panagiota Petkaki and Dave Broomhead.

REFERENCES

Kelton, W. D. 1989. Random Initialization Methods in Simulation. *IIE Transactions* 21:355-367.

Kevrekidis, I. G., C. W. Gear, and G. Hummer. 2004. Equation-free: The computer-aided analysis of complex multiscale systems. *AIChE Journal* 50:1346-1355.

Laing, C. 2006. On the application of "equation-free modelling" to neural systems. *Journal of Computational Neuroscience* 20:5-23.

Law, A. M. 1983. Statistical Analysis of Simulation Output Data. *Operations Research* 31:983-1029.

Pawlikowski, K. 1990. Steady-state Simulation of Queueing Processes: A Survey of Problems and Solutions. *ACM Computing Surveys* 22: 123-170.

Robinson, S. 2007. A statistical process control approach to selecting a warm-up period for a discrete-event simulation. *European Journal of Operational Research* 176:332-346.

Sandikci, B., and I. Sabuncuoglu. 2006. Analysis of the behavior of the transient period in non-terminating simulations. *European Journal of Operational Research* 173:252-267.

AUTHOR BIOGRAPHY

PHILIP G. BRABAZON is a Senior Research Fellow in the Operations Management Division at Nottingham University Business School. Philip graduated in Mechanical Engineering and has been a manufacturing development engineer and an industrial risk analyst. Philip received his Ph.D. from the University of Nottingham in the area of Mass Customization and he continues to research this strategy, with the objectives of investigating its fundamental challenges and developing quantitative and qualitative operational templates. Philip's e-mail and web page are:
`<philip.brabazon@nottingham.ac.uk>`
`<http://www.nottingham.ac.uk/business/lizpgb.html>`