Proceedings of the 2010 Winter Simulation Conference
B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, eds.

ESTIMATING THE PROBABILITY OF AN EVENT EXECUTION IN QUALITATIVE DISCRETE EVENT SIMULATION

Yen-Ping Leow-Sehwail

Sehwail Consulting Group
PO Box 17858
Taheid Al-Hussein
Amman 11622, Jordan

Ricki G. Ingalls

Oklahoma State University
School of Industrial Engineering and Management
322 Engineering North
Stillwater, OK 74078, USA

ABSTRACT

Qualitative Discrete Event Simulation (QDES) is an event scheduling approach that uses the Qualitative Event Graphs (QEGs) and the Event Graphs (EGs) as a general framework to discrete event simulation modeling. In QDES, the uncertainty in event execution times is represented in a closed time interval in $\mathfrak{R}$. When two or more event execution times overlap, it results in multiple event execution sequences or threads in the QDES output. In this paper, we introduce a methodology to estimate the probability of an event execution from QDES model.

1 INTRODUCTION

QDES extends the concept of qualitative simulation to be applied particularly in discrete event systems. It uses the next event time advance approach to advance the simulation clock to the occurrence of the next event. The concepts that are involved in developing the QDES model and algorithm are discussed in Leow-Sehwail and Ingalls (2005).

As a result of the interval time representation in QDES, the future event calendar is also represented in time intervals. Even though the future event calendar in QDES is sorted according to event times, it is not a strongly ordered list. Events are sorted according to interval mathematics outlined in Allen (1983). It is likely that there would be ties on the future event calendar because of the uncertain order of events. If there is a tie, QDES would not assume a tie breaking strategy. Instead, it creates threads that make up all of the possible ordering of ties. Thus, the future event calendar in QDES collects all the event notices whose execution order is uncertain and group them in a set, called the *non-deterministically ordered set* (NOS). Each of these event notices will be executed in turns and results in a set of threads that will include all of the possible ordering of event sequences. The capability of generating all possible scenarios is achieved with the thread generation algorithm. This distinctive characteristic of QDES of generating all possible ordering of event sequences is known as coverage (Ingalls, Morrice and Whinston 2000). The coverage property ensures that all outcomes of QDES are characterized and no outcome will be missed out.

In general, the number of threads can explode exponentially making output difficult to analyze. Ingalls and Morrice (2007) propose scoring methods to rank the threads according to the relative likelihood of the thread's execution sequences. We propose a methodology in this paper to estimate the probability of an event execution for all events that are generated from the output of a QDES model. In relevance to Ingalls and Morrice (2007)'s method, we attempt to calculate the probability of an event execution which can lead to calculating the thread's probability. The threads can then be ranked according to their probability of occurrence and this will assist in the analysis of the QDES output by eliminating threads that are of low probabilities. However, this paper addresses the process of obtaining the probability of sequential events QDES model which will not include the discussion of obtaining thread probabilities.

In the next section, we will present a QDES model of a simple queuing model to illustrate our methodology of estimating the event execution probability and the thread probability. The simple queuing QDES model will be discussed in Section 2. The methodology for estimating the probability of all events executed in the simple queuing QDES model will be discussed in the following sections. Section 3 is focused on the calculation of the first NOS event's probability. Section 4 discusses the calculation for the event after the execution of the first NOS event. Section 5 describe the process of generating probability distribution for newly scheduled events. Section 6 discusses the calculation for the probability of subsequent NOS events.

2 QDES MODEL OF A SIMPLE QUEUING MODEL

We consider an example of a simple queuing model with four events, BEGIN, ENTER, START and LEAVE as shown in Figure 1. This model can be viewed as bank teller system where the system BEGINs at time [0,0], customers ENTER the bank and wait to be served by the bank teller. When the queue is empty, the next customer in line will START the service process. When the service is finished, the customer LEAVEs the bank. BEGIN is the first event to be executed and it is used to initialize the state variables. The variables in this model are the queue length (Q), status of the bank teller (S) and number of customers that have exited the system (E). The status of the server is busy if S=0. If S=1, then the bank teller is idle. Changes in the state variables occur when an event occurs. For example, when the START event occurs, the queue length is decremented by 1($Q=Q-1$) and the status of the bank teller is changed to busy ($S=0$).

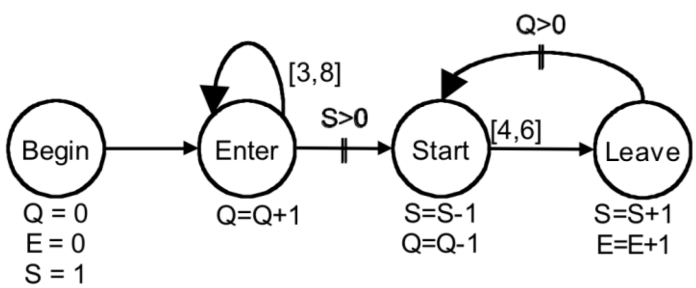

Figure 1. Event Graph of a Simple Queuing Model

The conditions on the edges of the event graph are based on the state of the system. The condition of S>0 is to check that the bank teller is available to service the next customer in line. If a customer arrives to the bank and the bank teller is not busy, the customer will be serviced immediately. The QDES model for this example is set to terminate after two customers exited. So, the terminating condition is reached when the number of exits equals two ($E=2$). A total of five threads are generated as shown in Figure 2.

3 PROBABILITY OF THE FIRST NOS EVENT EXECUTION

In this section, we will discuss the process of calculating the first NOS event's execution probability. We assume that the event delay times are uniformly distributed and the event times can be sectioned into predetermined time intervals. Without loss of generality, we assume that the time intervals are sectioned to one-unit time intervals. For example, let's say that an event A in a QDES model is uniformly distributed over time interval [1,10]. The probability of event A executing in any one-unit time interval within [1,10], i.e. [1,2], [2,3],…,[9,10] is 0.1. When a QDES model is executed, the next possible state is determined. If event A is the only next possible state to be executed, then the probability of event A executing in any one-unit time interval is still equal to 0.1. On the other hand, if there is more than one possible state, each of these states will be executed in turns and result in a set of threads that will include all of the event sequences. In this case, the probability of event A executing in a one-unit time interval within [1,10] depends on several factors. These factors includes the number of other next possible events that can be executed in the same particular time interval, scheduled execution time interval and the execution probability of these events.

As soon as the simulation starts, the simulation clock is initialized to [0,0]. There are three events scheduled to execute at time [0,0] and they are BEGIN[0,0](node 1), ENTER[0,0](node 2) and START[0,0](node 3) and in this order. At the instant ENTER[0,0] is executed, it schedules another event ENTER to be executed [3,8] time units later. Now, START[0,0] is at the top of the future event calendar, so it is executed next. The instant START[0,0] is executed, it schedules an event LEAVE to be executed [4,6] time units later. At this time, there are two events in the future event calendar, which are ENTER[3,8] and LEAVE[4,6].

If we assume that the delay time intervals are uniformly distributed, it is equally probable for LEAVE[4,6] to execute either in [4,5] or [5,6] and this probability equals to 0.5. It is also equally probable for ENTER[3,8] to execute in any of the one time unit intervals, i.e. [3,4], [4,5],.., [7,8] and this probability equals to 0.2.

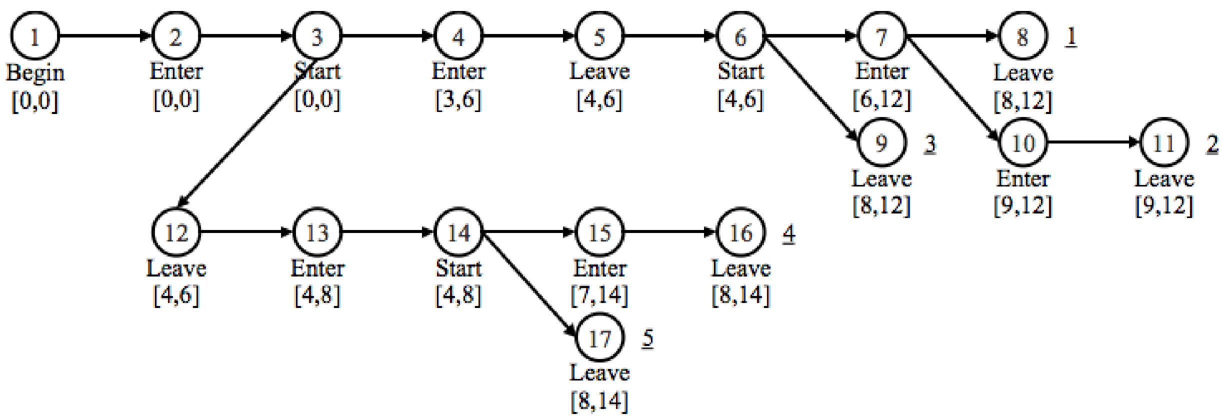

Figure 2. Event Sequences for each of the five threads generated

QDES will spawn two threads, one assumes that ENTER[3,8] is executed first and the second thread assumes that LEAVE[4,6] is executed first. If we assume that ENTER[3,8] executes first, then it must be executed in [3,6], so that the next imminent event, LEAVE[4,6] can be executed next in [4,6]. On the other hand, if LEAVE[4,6] is to be executed first in [4,6], then ENTER[3,8] can be executed next in [4,8].

The probability of LEAVE[4,6] executing first in [4,5] depends on whether ENTER[3,8] is executed in the same time interval. Assume that ENTER[3,8] is executed in [4,5] and the probability that this will happen equals to 0.2, then both LEAVE[4,6] and ENTER[3,8] are equally probable to be executed in [4,5] since these are only two NOS events that can be executed in the [4,5] time interval. In this case, the probability of LEAVE executing first in [4,5] equals to 0.5*0.5*0.2=0.05. Now, assume that ENTER is executed in other time intervals after [4,5],i.e. [5,8] (the probability that this will happen equals to 0.6) and since LEAVE is the only event that can be executed in [4,5], the probability of LEAVE executing in [4,5] equals to 0.5*1*0.6=0.3. Thus, combining the two cases, the probability of LEAVE executing first in [4,5] equals to 0.3+0.05=0.35.

The probability of each event executing first in each one time unit interval is calculated and shown as below. Note that the probability of an event executing in interval [i,j] is denoted as P(event_name $\in$ [i,j]).

$$P(\text{LEAVE is executed first in } [4,5]) = P(\text{LEAVE} \in [4,5])\ (0.5*P(\text{ENTER} \in [4,5] + P(\text{ENTER} \in [5,8]))$$
$$= 0.5*(0.5*0.2+0.6) = 0.35$$

P(LEAVE is executed first in [5,6]) = P(LEAVE $\in$ [5,6])*(0.5*P(ENTER $\in$ [5,6])+P(ENTER $\in$ [6,8]))

$= 0.5*(0.5*0.2+0.4) = 0.25$

P(LEAVE execute first in [4,6])=P(LEAVE is executed first in [4,5])+P(LEAVE is executed first in [5,6])

$= 0.35+0.25 = 0.6$

P(ENTER is executed first in [3,4]) = P(ENTER $\in$ [3,4]) = 0.2

P(ENTER is executed first in [4,5]) = P(ENTER $\in$ [4,5])*(0.5*P(LEAVE $\in$ [4,5]+P(LEAVE $\in$ [4,5]))

$= 0.2*(0.5*0.5+0.5) = 0.15$

P(ENTER is executed first in [5,6]) = P(ENTER $\in$ [5,6])*(0.5*P(LEAVE $\in$ [5,6]))

$= 0.2(0.5*0.5) = 0.05$

P(ENTER is executed first in [3,6]) = P(ENTER is executed first [3,4]) + P(ENTER is executed first in [4,5])+ P(ENTER is executed first in [5,6])= $0.2+0.15+0.05 = 0.4$

The following figure summarizes the above calculations. Overall, the probability of ENTER[3,6] executing first is equal to 0.4, which leads us to the first branch of the event graph at node 4 in Figure 2. The probability of LEAVE[4,6] executing first is equal to 0.6 and it forms the second branch of the event graph at node 12 in Figure 2.

P{LEAVE[4,6] execute first in interval [i,j]}

[i,j]	[4,5]	[5,6]	Total
Probability	0.35	0.25	0.6

P{ENTER [3,6] execute first in interval [i,j]}

[i,j]	[3,4]	[4,5]	[5,6]	Total
Probability	0.2	0.15	0.05	0.4

Figure 3. The probability of LEAVE[4,6] and ENTER[3,8] executing first, respectively

4 PROBABILITY OF THE NEXT EVENT EXECUTION

If we assume that LEAVE[4,6] is executed first, the probability of event ENTER executing next in [4,5], [5,6], [6,7] and [7,8] can be calculated. Based on the condition of when LEAVE[4,6] is executed first, in this case, we have two conditions, namely LEAVE is executed first in [4,5] and LEAVE is executed first in [5,6], the probabilities of event ENTER executing next in [5,6], [6,7] and [7,8] are calculated and shown as below.

$$\text{P(ENTER in [4,8]|LEAVE 1}^{\text{st}}\text{ in [4,5])} = \begin{pmatrix} \text{P(ENTER 2nd in [4,5]| LEAVE 1st in [4,5])} \\ \text{+P(ENTER 2nd in [5,6]| LEAVE 1st in [4,5])} \\ \text{+P(ENTER 2nd in [6,7]| LEAVE 1st in [4,5])} \\ \text{+P(ENTER 2nd in [7,8]| LEAVE 1st in [4,5])} \end{pmatrix}$$

$$\text{P(ENTER in [5,8]|LEAVE 1}^{\text{st}}\text{ in [5,6])} = \begin{pmatrix} \text{P(ENTER 2nd in [5,6]| LEAVE 1st in [5,6])} \\ \text{+P(ENTER 2nd in [6,7]| LEAVE 1st in [5,6])} \\ \text{+P(ENTER 2nd in [7,8]| LEAVE 1st in [5,6])} \end{pmatrix}$$

If LEAVE executes in [4,5], then ENTER can execute next in [4,8]. Without considering the fact that LEAVE was executed in [4,5], we can redistribute the probability of ENTER executing in [4,8] by looking at the original probability distribution of ENTER[3,8]. Since it is equally likely for ENTER[3,8] to execute in any interval within [3,8] with a probability of 0.2, ENTER will still be equally likely to execute anywhere within [4,8] with a probability of 0.25. The probability distribution for ENTER[4,8] is shown in the first row of Figure 4. The same reasoning goes for the case if LEAVE executes in [5,6], the probability distribution for ENTER[4,8] is shown in the second row of Figure 4.

	[4,5]	[5,6]	[6,7]	[7,8]
If LEAVE executes in [4,5]	0.25	0.25	0.25	0.25
If LEAVE executes in [5,6]		0.333333	0.333333	0.333333

Figure 4. The probability of occurrence for ENTER[4,8]

If LEAVE [4,6] executes in [4,5], then the probability for ENTER to execute in [4,5] is equal to 0.125 (half of 0.25). Since ENTER has a 0.25 probability to be in [4,5], the probability of ENTER to execute in [4,5], given that LEAVE executes in [4,5] is equal to 0.5*0.25 = 0.125. Thus, P{ENTER execute second in [4,5]/LEAVE executed in [4,5]}=0.125/(0.125+0.25*3) $\cong$ 0.14286

P(ENTER execute in [4,5] | LEAVE executed first in [4,5])
= P(ENTER in [4,5])

$$= \frac{\frac{1}{2} * 0.2}{\left(\frac{1}{2} * 0.2 + 3 * 0.2\right)} \cong 0.14286$$

P(ENTER execute in [5,6] | LEAVE executed first in [4,5])
= P(ENTER did not execute in [4,5] out of [4,8])*P(ENTER in [5,6] out of [5,8])

$$= (1 - 0.14286) * \frac{0.2}{(3 * 0.2)} \cong 0.28571$$

P(ENTER execute in [6,7] | LEAVE executed first in [4,5])
= P(ENTER did not execute in [4,5] out of [4,8] and [5,6] out of [5,8])*P(ENTER in [6,7] out of [6,8])

$$= (1 - 0.14286) * \left[1 - \frac{0.2}{(3 * 0.2)}\right] * \frac{0.2}{(2 * 0.2)} \cong 0.28571$$

P(ENTER execute in [7,8] |LEAVE executed first in [4,5])
= P(ENTER did not execute in [4,5] out of [4,8], [5,6] out of [5,8] and [6,7] out of [6,8])*P(ENTER in [7,8] out of [7,8])

$$= (1 - 0.14286) * \left[1 - \frac{0.2}{(3 * 0.2)}\right] * \left[1 - \frac{0.2}{(2 * 0.2)}\right] * 1 \cong 0.28571$$

P(ENTER execute in [5,6] |LEAVE execute first in [5,6])
= P(ENTER in [5,6] out of [5,8])

$$= \frac{\frac{1}{2} * 0.2}{\left(\frac{1}{2} * 0.2 + 2 * 0.2\right)} = 0.2$$

P(ENTER execute in [6,7] / LEAVE execute first in [5,6])
= P(ENTER did not execute in [5,6] out of [5,8])* P(ENTER in [6,7] out of [6,8])

$$= (1 - 0.2) * \frac{0.2}{2 * 0.2} = 0.4$$

P(ENTER execute in [7,8] / LEAVE execute first in [5,6])

= P(ENTER did not execute in [5,6] out of [5,8] and [6,7] out of [6,8])* P(ENTER in [7,8] out of [7,8])

$$= (1-0.2)*\left[1-\frac{0.2}{2*0.2}\right]*1 = 0.4$$

Figure 5 shows the calculations we have so far. With all the calculation we have so far, we can now calculate the probability of ENTER execute next in [4,5], [5,6], [6,7] and [7,8]. The probability distribution of ENTER [4,8] is given in Figure 6.

P{ENTER [4,8] execute second in interval (i,j) | LEAVE [4,6] execute first in (p,q)}

(p,q) \ **(i,j)**	**(4,5)**	**(5,6)**	**(6,7)**	**(7,8)**	**TOTAL**
(4,5)	0.14286	0.28571	0.28571	0.28571	**1**
(5,6)	0	0.2	0.4	0.4	**1**

Figure 5. Probability of occurrence for ENTER[4,8] given that LEAVE[4,6] was executed

P(ENTER execute in [4,5])
= P(ENTER execute in [4,5] / LEAVE execute first in [4,5]) * P(LEAVE execute first in [4,5])

$$= 0.142857*\frac{0.35}{(0.35+0.25)} \cong 0.08333$$

P(ENTER execute in [5,6])
= {P(ENTER execute in [5,6] / LEAVE execute first in [4,5]) * P(LEAVE execute first in [4,5])} + {P(ENTER execute in [5,6] / LEAVE execute first in [5,6]) * P(LEAVE execute first in [5,6])}

$$= \left(0.28571*\frac{0.35}{(0.35+0.25)}\right)+\left(0.2*\frac{0.25}{(0.35+0.25)}\right) \cong 0.25$$

P(ENTER execute in [6,7])
= {P(ENTER execute in [6,7] / LEAVE execute first in [4,5]) * P(LEAVE execute first in [4,5])} + {P(ENTER execute in [6,7] / LEAVE execute first in [5,6]) * P(LEAVE execute first in [5,6])}

$$= \left(0.28571*\frac{0.35}{(0.35+0.25)}\right)+\left(0.4*\frac{0.25}{(0.35+0.25)}\right) \cong 0.3333$$

P(ENTER execute in [7,8])
= {P(ENTER execute 2^{nd} in [6,7] / LEAVE execute first in [4,5]) * P(LEAVE execute first in [4,5])} + {P(ENTER execute 2^{nd} in [7,8] / LEAVE execute first in [5,6]) * P(LEAVE execute first in [5,6])}

$$= \left(0.285714*\frac{0.35}{(0.35+0.25)}\right)+\left(0.4*\frac{0.25}{(0.35+0.25)}\right) \cong 0.3333$$

P{ENTER [4,8] execute in interval (i,j)}

(i, j)	**(4,5)**	**(5,6)**	**(6,7)**	**(7,8)**	**TOTAL**
	0.083333	0.25	0.333333	0.333333	**1**

Figure 6. Probability of occurrence for ENTER[4,8]

5 GENERATING A NEW PROBABILITY DISTRIBUTION

Let's assume that the simulation proceeds with the current simulation clock at [4,8], ENTER[4,8] has just executed. As soon as ENTER [4,8] has executed, it schedules a new event START to happen immediately since the server is available there is no other customer waiting for the server. START[4,8] could execute immediately in [4,8] with the same probability distribution as ENTER[4,8] (refer to node 13 in Figure 2). A new event ENTER is also scheduled by ENTER[4,8] to happen [3,8] time units later, which leads to ENTER[7,16]. START[4,8] schedules a LEAVE event to execute [4,6] time units after time interval [4,8], which then leads to LEAVE[8,14]. Events ENTER[7,16] (refer to node 15 in Figure 2) and LEAVE[8,14] (refer to node 17 in Figure 2) are currently on the future event calendar. Because their execution times overlap, they create another NOS.

ENTER[7,16] is scheduled to occur [3,8] time units after [4,8], which leads to time interval [7,16]. The delay of [3,8] is assumed to follow uniform distribution, but the time interval [4,8] from ENTER[4,8] is not uniform. Table 1 shows the probability distribution for ENTER[4,8] and the uniform distribution [3,8]. The probability distribution for ENTER[7,16] can be obtained by "adding" the distribution of ENTER[4,8] to the distribution of Uniform[3,8].

Table 1. Probability distributions for events ENTER[4,8] and Uniform[3,8]

	ENTER[4,8]		**UNIFORM[3,8]**
[4,5]	0.083333333	[3,4]	0.2
[5,6]	0.25	[4,5]	0.2
[6,7]	0.333333333	[5,6]	0.2
[7,8]	0.333333333	[6,7]	0.2
		[7,8]	0.2

Adding these two distributions is equivalent to the result of drawing one sample from each distribution and add the values of the samples together to get the final value, x. The addition of ENTER[4,8] and Uniform[3,8] probability distributions creates the probability distribution for this final value, x.

To get the probability distribution of (ENTER[4,8]+Uniform[3,8]), we can simply multiply the ENTER[4,8] column (refer to Table 1) with the Uniform[3,8] column. For example, P(E[4,5] + U[3,4]) = P(ENTER[7,9]) = 0.083333333* 0.2 ≅0.016666.

Table 2 shows the resulting probability distribution. E[4,5]+U column holds the probability values for ENTER[7,9], [8,10], [9,11], [10,12], and [11,13], which are the results of adding the probability of ENTER[4,5] with Uniform probability of [3,4], [4,5], [5,6], [6,7] and [7,8], respectively.

In order to get the probability of ENTER executing in a certain interval, the sum of probabilities attributed to that interval is computed. For example, P(ENTER[8,10]) = { P(E[4,5]+U[4,5])=[8,10]) } + { P(E[5,6]+U[3,4])=[8,10]) } ≅ 0.016666667 + 0.05 ≅ 0.066666667

Table 2. Probability distribution of ENTER[7,16]

E[4,5] +U	**P{E[4,5]+U}**	**E[5,6]+U**	**P{E[5,6]+U}**	**E[6,7]+U**	**P{E[6,7]+U}**	**E[7,8]+U**	**P{E[7,8]+U}**
[7,9]	0.016666667	[8,10]	0.05	[9,11]	0.06666667	[10,12]	0.06666667
[8,10]	0.016666667	[9,11]	0.05	[10,12]	0.06666667	[11,13]	0.06666667
[9,11]	0.016666667	[10,12]	0.05	[11,13]	0.06666667	[12,14]	0.06666667
[10,12]	0.016666667	[11,13]	0.05	[12,14]	0.06666667	[13,15]	0.06666667
[11,13]	0.016666667	[12,14]	0.05	[13,15]	0.06666667	[14,16]	0.06666667

The resulting interval will be of length 2, which are [7,9], [8,10], [9,11], [10,12] and [11,13] with the same probability of 0.016666 , respectively. If we leave the interval length as it is, as the simulation continues the length of the interval will keep increasing every time we add probability distributions. However, we can reduce the interval to one time unit by assuming that the each of these resulting intervals of [7,9], [8,10], [9,11], [10,12] and [11,13] follows a uniform distribution. If we assume that the probability of ENTER[8,10] which equals to 0.06666 is evenly distributed throughout [8,10], then ENTER could be executed in [8,9] and [9,10] each with the probability of $\frac{0.06666666}{2} \cong 0.0333333$. The same assumption is applied to all other intervals, including ENTER[7,9] (which has a probability of 0.0166666), then we can find the probability of ENTER[8,9] by adding half of the probability from [8,10] and half of the probability from [7,9], as shown below. Table 3 shows the probability distribution of ENTER[7,16], after the reduction of the interval length to one time unit. The probability distribution of LEAVE[8,14] can be obtained using similar approach.

Table 3. Probability distribution for ENTER[7,16] with interval width of one time unit

ENTER[7,16]	Calculated Probability
[7,8]	0.008333333
[8,9]	0.041666667
[9,10]	0.1
[10,11]	0.166666667
[11,12]	0.2
[12,13]	0.191666667
[13,14]	0.158333333
[14,15]	0.1
[15,16]	0.033333333

6 PROBABILITY OF AN EVENT EXECUTION

The probability of ENTER[7,16] executing in any given time interval (i.e. either of [7,8],…, [15,16]) depends on the condition of two things:

a) In which time interval for which ENTER[7,16]'s scheduling event is executed.

If given that its scheduling event, ENTER[4,8] is executed in [4,5], then ENTER[7,16] can be executed in [7,9], [8,10], [9,11], [10,12] and [11,13] with probability of 0.2, respectively. This distribution with overlapping time interval represents the probability distribution of ENTER[7,11], given the time interval of ENTER[4,5]. Note that it has the same shape as the delay distribution of [3,8]. Recall that the delay distribution of [3,8] has a uniform probability distribution of 0.2 in each of its one-unit time interval. We know that P(ENTER[7,16] is in [7,9]) = P(E[4,5] + U[3,4]) = P(ENTER[7,9]) $\cong$ 0.083333333* 0.2 = 0.016666… and P(ENTER[4,8] is in [4,5]) $\cong$ 0.083333333 (refer to Figure 6).

Since P(ENTER[7,16] is in [7,9]) = P([ENTER[7,16] is in [7,9]/ENTER[4,8] is in [4,5])* P(ENTER[4,8] is in [4,5]). Thus, P([ENTER[7,16] is in [7,9]/ENTER[4,8] is in [4,5])= P(ENTER[7,16] is in [7,9])/ P(ENTER[4,8] is in [4,5]) $\cong$ (0.083333333* 0.2)/ 0.083333333= 0.2.

If we assume that the probability of all the two-unit time intervals are evenly distributed within their time intervals, using the same reasoning as above, the probability distribution of ENTER[7,16] executing in [7,8], [8,9], [9,10], [10,11], [11,12] and [12,13] given that its scheduling event ENTER[4,8] was executed in [4,5] is equal to 0.1, 0.2, 0.2, 0.2, 0.2 and 0.1 respectively.

b) Whether ENTER[7,16] intersects with its immediate preceding event.

The probability distribution that was obtained earlier only takes into account the effect of ENTER[4,8]'s execution time on the probability of occurrence for ENTER[7,16]. If the immediate preceding event for ENTER[7,16] is ENTER[4,8], there will not be any intersection of time intervals when calculating the conditional probability. Meanwhile, the probability for ENTER[7,16] will be lesser than the amount that has been calculated for the case if there is an intersection with the immediate preceding event. This is likely to be the case if ENTER[7,16] is one of the NOS events in the NOS set.

Even though the immediate preceding event for ENTER[7,16] is not its scheduling event, ENTER[4,8], but START[4,8] inherits the same probability distribution from ENTER[4,8]. This means that there will not be any intersection of time intervals when calculating the probability for ENTER[7,16].

Thus, the probability for ENTER[7,16] that is calculated earlier will not be affected. Figure 7 shows the probability distribution for ENTER[7,16].

P{ENTER [7,16] execute in interval (i,j) / START[4,8] execute in (p,q)}

(p,q) \ (i,j)	(7,8)	(8,9)	(9,10)	(10,11)	(11,12)	(12,13)	(13,14)	(14,15)	(15,16)
[4,5]	0.1	0.2	0.2	0.2	0.2	0.1			
[5,6]		0.1	0.2	0.2	0.2	0.2	0.1		
[6,7]			0.1	0.2	0.2	0.2	0.2	0.1	
[7,8]				0.1	0.2	0.2	0.2	0.2	0.1

Figure 7. The probability distribution of ENTER[7,16]

For all the events that are to be executed after the first non-zero execution time event, they will carry information about the timing of all preceding events. Each combination of the timing of these preceding events forms a condition for the probability distribution of the event that we are evaluating.

The conditions for the probability of ENTER[7,16] are shown in the left column of Figure 9. The interval [5,6] of condition “[4,5]-[5,6]” represents the time interval [5,6] of the immediate preceding event of ENTER[7,16] (i.e. START[4,8] and ENTER[4,8]), since they both have exactly the same probability distribution. If START is executed in [4,5], ENTER will also be executed in [4,5]. The interval [4,5] of condition “[4,5]-[5,6]” represents the time interval [4,5] of the second last event, which is LEAVE[4,6]. The same conditions are generated for LEAVE[8,14]. The probability for LEAVE[8,16] is shown in Figure 9.

Condition	[7,8]	[8,9]	[9,10]	[10,11]	[11,12]	[12,13]	[13,14]	[14,15]	[15,16]
[4,5]-[4,5]	0.1	0.2	0.2	0.2	0.2	0.1			
[4,5]-[5,6]		0.1	0.2	0.2	0.2	0.2	0.1		
[4,5]-[6,7]			0.1	0.2	0.2	0.2	0.2	0.1	
[4,5]-[7,8]				0.1	0.2	0.2	0.2	0.2	0.1
[5,6]-[5,6]		0.1	0.2	0.2	0.2	0.2	0.1		
[5,6]-[6,7]			0.1	0.2	0.2	0.2	0.2	0.1	
[5,6]-[7,8]				0.1	0.2	0.2	0.2	0.2	0.1

Figure 8.The probability of ENTER[7,16] under all possible combinations of conditions

Condition	[8,9]	[9,10]	[10,11]	[11,12]	[12,13]	[13,14]
[4,5]-[4,5]	0.25	0.5	0.25			
[4,5]-[5,6]		0.25	0.5	0.25		
[4,5]-[6,7]			0.25	0.5	0.25	
[4,5]-[7,8]				0.25	0.5	0.25
[5,6]-[5,6]		0.25	0.5	0.25		
[5,6]-[6,7]			0.25	0.5	0.25	
[5,6]-[7,8]				0.25	0.5	0.25

Figure 9. The probability of LEAVE[8,14] under all possible combinations of conditions

As can be seen from Figures 8 and 9, the probability for ENTER[7,16] and LEAVE[8,14] is not affected by what happens before its scheduling event, ENTER[4,8]. The delay distribution and the scheduling event's probability distribution determine the timing of the newly scheduled event and its probability distribution. The probability of occurrence that is obtained for the new event is lesser for the case when there is an intersection of time intervals with the immediate preceding event. If the immediate preceding event for the new event happens to be its scheduling event, there will not be any intersection of time intervals. In this case, the probability of occurrence that is obtained from the delay distribution and scheduling event's distribution will remain the same.

The probability of each condition occurring could be obtained by normalizing the conditional probability in Figure 5. The result is show in Table 4. The probability of each condition occurring is needed when calculating the probability of a NOS event executing first.

Table 4. Probability of the conditions

Condition	Probability
[4,5]-[4,5]	0.083333
[4,5]-[5,6]	0.166667
[4,5]-[6,7]	0.166667
[4,5]-[7,8]	0.166667
[5,6]-[5,6]	0.083333
[5,6]-[6,7]	0.166667
[5,6]-[7,8]	0.166667

The probability of ENTER[7,16] executing first and LEAVE[8,14] executing first are computed and are given in Figure 10 and Figure 11, respectively. ENTER[7,16] has a probability of 0.00833 to execute first in the time interval [7,8], 0.03958 to execute first in [8,9], 0.08542 to execute first in [9,10] and so on. These probabilities are obtained from the sum product of the probability of the condition in Table 4 with the probability of ENTER[7,14] in each time interval. Combining the probabilities from all these time intervals, ENTER[7,14] has a probability of 0.4 to execute first in [7,14]. On the other hand, LEAVE[8,14] has a probability of 0.6 to execute first in [8,14]. Notice that the addition of ENTER[7,14] and LEAVE[8,14]'s probability equals to one, as they are exclusive events.

ENTER[7,14]	[7,8]	[8,9]	[9,10]	[10,11]	[11,12]	[12,13]	[13,14]	Node 25
[4,5]-[4,5]	0.1	0.175	0.1	0.025				
[4,5]-[5,6]		0.1	0.175	0.1	0.025			
[4,5]-[6,7]			0.1	0.175	0.1	0.025		
[4,5]-[7,8]				0.1	0.175	0.1	0.025	
[5,6]-[5,6]		0.1	0.175	0.1	0.025			
[5,6]-[6,7]			0.1	0.175	0.1	0.025		
[5,6]-[7,8]				0.1	0.175	0.1	0.025	
P(Execute 1st)	0.0083	0.0396	0.0854	0.1188	0.0979	0.0417	0.0083	0.4
Normalized p	0.0208	0.099	0.2136	0.2969	0.2448	0.1042	0.0208	1

Figure 10: The probability of ENTER[7,16] executing first

LEAVE[8,14]	[8,9]	[9,10]	[10,11]	[11,12]	[12,13]	[13,14]	Node 31
[4,5]-[4,5]	0.2	0.3	0.1				
[4,5]-[5,6]		0.2	0.3	0.1			
[4,5]-[6,7]			0.2	0.3	0.1		
[4,5]-[7,8]				0.2	0.3	0.1	
[5,6]-[5,6]		0.2	0.3	0.1			
[5,6]-[6,7]			0.2	0.3	0.1		
[5,6]-[7,8]				0.2	0.3	0.1	
P(Execute 1st)	0.0167	0.075	0.15	0.1917	0.1333	0.0333	0.6
Normalized p	0.0278	0.125	0.25	0.3195	0.2222	0.0556	1

Figure 11: The probability of LEAVE[8,14] executing first

The probability distribution for the timing of an event can be obtained from normalizing the probability of an NOS event executing first. For the case where there is one event in the NOS set, the probability distribution for the timing of an event is already in the normalized form. The "Normalized p" rows in

Figures 10 and 11 are the probability distributions for the timing of ENTER[7,16] and LEAVE[8,14], respectively, as the result of normalization from the probability in the "P(Execute 1st)" row.

7 RESULT

In order to compare the event execution probabilities that are calculated using the method discussed in the previous sections, a simulation model of the Simple Queuing System was created using Microsoft Excel. 10,000 timings of all the scheduled events are randomly generated. Figure 12 shows the event probability distribution for the events from thread 4. It shows that the calculated probabilities are very close to the simulated probabilities as the simulation progresses. It also exhibits the ability of the proposed calculation to capture the shape of the event timing probability distribution.

LEAVE[8,14] (Node 16)			
Interval	**Simulated Prob.**	**Calculated Prob.**	**Difference**
[8,9]	0.0090090	0.0104167	0.0014077
[9,10]	0.0735387	0.0729167	-0.0006220
[10,11]	0.2150988	0.1979167	-0.0171822
[11,12]	0.3167819	0.3020833	-0.0146986
[12,13]	0.2771842	0.2916667	0.0144825
[13,14]	0.1084875	0.1250000	0.0165125
ENTER[7,14] (Node 15)			
Interval	**Simulated Prob.**	**Calculated Prob.**	**Difference**
[7,8]	0.1431664	0.0208333	-0.1223331
[8,9]	0.0955374	0.0989583	0.0034209
[9,10]	0.2224173	0.2135417	-0.0088757
[10,11]	0.3094490	0.2968750	-0.0125740
[11,12]	0.2479223	0.2447917	-0.0031307
[12,13]	0.0970738	0.1041667	0.0070928
[13,14]	0.0132691	0.0208333	0.0075642
ENTER[4,8] (Node 13,14)			
Interval	**Simulated Prob.**	**Calculated Prob.**	**Difference**
[4,5]	0.0786368	0.0833333	0.0046966
[5,6]	0.2602137	0.2500000	-0.0102137
[6,7]	0.3306097	0.3333333	0.0027237
[7,8]	0.3305398	0.3333333	0.0027935
LEAVE[4,6] (Node 12)			
Interval	**Simulated Prob.**	**Calculated Prob.**	**Difference**
[4,5]	0.5869823	0.5833333	-0.0036490
[5,6]	0.4130177	0.4166667	0.0036490

Figure 12: Simulated vs. Calculated Probabilities

The approach in this paper is repeatable for all the events that are generated from the QDES output. Thus it can be extended and developed into an algorithm.

8 CONCLUSION AND FUTURE RESEARCH

This work is significant in the area of qualitative discrete event simulation because it gives a probabilistic map of the output space of the simulation. The work was started with a uniform distribution assumption because of the ease of calculating probabilities. Even though the work was done under that assumption, the resulting structure will allow any discrete distribution as an input for the delay and time between creation delays.

Also, this work has the problem of computational effort, in that it the combinations explode with time. However, since this method gives the complete probabilistic view of the output space, there is no need for iterations as you would have to do in a traditional discrete-event simulation.

The research team continues to look for algorithms and approaches that exploit this work. The research lies in two primary areas: (1) improving the computational efficiency of the algorithm and (2) exploiting the output space information for the purpose of simulation optimization.

REFERENCES

Allen, J.F. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832-843.

Ingalls, R.G., D.J. Morrice and A.B. Whinston. 2000. The implementation of temporal intervals in qualitative simulation graphs. *ACM Transactions on Modeling and Computer Simulation*, 10(2): 215-240.

Ingalls, R.G. and D.J. Morrice. 2007. Alternative thread scoring methods in qualitative event graphs. In *Proceedings of the 2007 Winter Simulation Conference*., ed. S.G. Henderson, B. Biller, M.-H. Hsieh,. J. Shortle, J.D.Tew, and R.R. Barton. 667-675. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Leow-Sehwail, Y-P and R.G. Ingalls. 2005. Qualitative discrete event simulation. In *Proceedings of the 2005 Winter Simulation Conference*., ed. M.E. Kuhl, N.M. Steiger, F.B. Armstrong, and J. A. Joines. 648-653. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHIES

YEN-PING LEOW-SEHWAIL is currently a Senior Consultant in Sehwail Consulting Group. She has a B.S. in Mathematical and Computer Science from the University of Adelaide, South Australia, Australia, a M.S. in Industrial Engineering and Management and PhD in Industrial Engineering and Management from Oklahoma State University, Stillwater, Oklahoma. She is a member of IIE, Alpha Pi Mu Industrial Engineering Honor Society, American Society for Quality and INFORMS. Her research interests include qualitative discrete event simulation, simulation methodology and analysis, probability and statistics. Her e-mail address is <carmenleow@yahoo.com>.

RICKI G. INGALLS is an Associate Professor and Site Director of the Center for Engineering Logistics and Distribution (CELDi) in the School of Industrial Engineering and Management at Oklahoma State University. He was also the Co-Editor of the *Proceedings of the 2004 Winter Simulation Conference* and Program Chair of the *2009 Winter Simulation Conference*. He joined OSU in 2000 after 16 years in industry with Compaq, SEMATECH, General Electric, and Motorola. He has a B.S. in Mathematics from East Texas Baptist College (1982), a M.S. in Industrial Engineering from Texas A&M University (1984) and a Ph.D. in Management Science from the University of Texas at Austin (1999). His research interests include the supply chain design issues and the development and application of qualitative discrete event simulation. He is a member of INFORMS and ACM. His e-mail address is <ricki.ingalls@okstate.edu>.