Science and Technology, 2 (1997) 39-47

Distributed Simulation of Open Cyclic Queuing Networks

Saad Harous

Department of Computer Science, College of Science, Sultan Qaboos University, P.O.Box 36, Al-Khod 123, Muscat, Sultanate of Oman.

المحاكاة الموزعة لشبكات طابور دوائر مفتوحة

ساعد حروس

خلاصة : يقدم هذا البحث نتائج دراسة تجريبية لأداء نموذج محاكاة توزيعي دائري مفتوح لشبكة انتظار ، من خلال تعديل بسيط لخوارزمية «كاندي ومزرا» المعتمدة على اكتشاف ومعالجة مشكلة الوضع المستعصي . إن أحد العوامل الأساسية في هذه الدراسة هو تحديد تأثير فائض العمل على نماذج المحاكاة التوزيعية . علاوة على ذلك ، تم قياس نظم تمثيل محددة ومعدلة «لنسب استعجال مثالية» – وهي نسب تتجاوز مسألة التعديل على عدد المعالجات ، أي أنها تحقق مقاييس استعجالية يمكن إدراكها في نماذج محاكاة على نحو أكثر دقة .

ABSTRACT: In this paper we present the results of an experimental performance study of distributed simulation of open cyclic queuing networks using a minor variation of the deadlock detection and recovery based algorithm of Chandy and Misra. One major part of this study is to determine the effect of overhead on distributed simulation. Moreover, we measure certain refined notions of "ideal speedup ratio." These ratios are more refined than just the number of processors in the sense that they capture the potentially achievable speedups of distributed simulation more closely.

Several schemes for distributed simulation have been proposed in the literature (see: Chandy and Misra 1978, 1981, and Jefferson 1985). These schemes utilize overhead messages to handle the potential deadlock situations that may arise during the simulation. To see the possibility of deadlock if overhead messages are not used, consider the queuing network shown in Figure 1. When the distributed simulation starts *lp2* is waiting to receive from both *lp1* and *lp4*; but only *lp1* sends messages to *lp2*. *lp2, lp3, lp4* and *lp5* are permanently blocked. Even if we force the first message to be sent from *lp2* to *lp3*, still at a later stage the same problem will happen again. Unfortunately, only a few performance studies of these schemes are available (Fujimoto 1988, Reed *et al* 1988, Seethalakshmi 1979, Shorey *et al* 1994, Jha *et al* 1996). These studies have provided useful data, and have shown some positive and some negative performance results for a few combinations of the distributed simulation schemes, the system to be simulated, and the available distributed system on which the simulation is to be carried out. Several obvious questions have remained unanswered. For example, the relationship between the amount of overhead and the performance of a distributed simulation scheme is not clear. Understanding this relationship is important since this can provide useful information as to whether there is any hope in trying to improve the performance by finding variations of a scheme that aims at reducing the amount of overhead. Suppose that the distributed simulation of a particular system using the scheme by Chandy *et al* (1978) has low performance. Would it be useful to define and study variations of the scheme that would tend to reduce the number of NULL messages? A related question is how much speedup one can expect in an ideal variation of a distributed simulation scheme that would somehow totally eliminate the effect of overhead messages. Can we expect a reasonable performance in such a simulation? Also, what kind of ideal speedup can be expected if one is considering possibilities across all the well known schemes of Chandy *et al* (1978, 1981), and Jefferson (1985), and their simple variations.

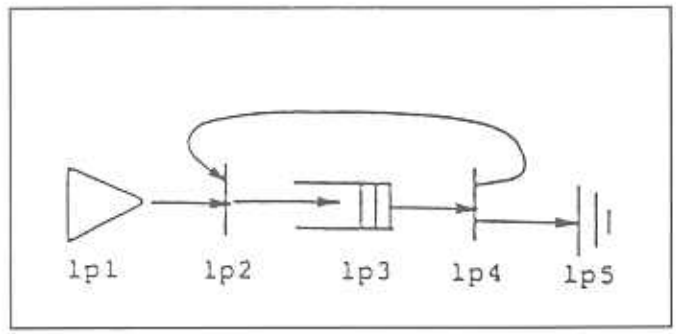

Figure 1. To show the deadlock problem.

The known performance studies have not addressed these issues adequately. In general, studies have provided data on the performance of distributed simulation for individual cases, but the above issues have been largely ignored. In some cases where performance of the distributed simulation turns out to be poor, some researchers have suggested that the poor performance is due to large overhead (see Chandy *et al* 1981, Reed *et al* 1988, Seethalakshmi 1979). Such a suggestion would normally mean that (i) indeed there is a large amount of overhead, and (ii) if somehow the overhead could be reduced significantly then the performance would be improved significantly. Claiming part (ii) of this statement for a specific case requires further study and such a study is reported here.

Note that it is not clear as to how to study the relationship between performance and overhead. One simple-minded approach would be to define and study different variations of a given scheme which lead to a different number of overhead messages, thereby providing some data regarding relationship between overhead and performance. Unfortunately such an approach would have several problems:

(1) In coming up with the different variations, one has to ensure that deadlocks are handled adequately, otherwise the scheme would simply be incorrect. This restricts the number of different variations one can come up with.

(2) It is usually not feasible to vary the number of overhead messages in a reasonable range so as to enable one to determine the relationship between this number and the performance. In particular, the number of overhead messages may remain large, in which case it is difficult to judge what the performance would be like if this number were somehow reduced to a small value. Note that varying the number of overhead messages in a controlled way, i.e., as an independent variable, is nearly impossible.

(3) As the different variations are being considered it is quite possible that other activities in the simulator that are not related to overhead are also changing, e.g., the algorithmic rules that determine when an event message should be computed. This makes it harder to judge whether a change in performance is simply due to a change in the number of overhead messages or whether the other factors are also influencing the result.

We discuss here a simple and new approach towards determining the relationship between amount of overhead and performance. In this approach: (i) we *simulate* the distributed simulator instead of directly implementing it on a distributed system and directly measuring its performance, and (ii) in this approach we vary the amount of communication delay and the computation time for each overhead message including the case when these values are zero. Note that this approach is quite different from trying to vary the number of overhead messages.

Regarding the issue of "ideal speedup" in distributed simulation, the usual measure for ideal speedup used in the literature is N, which is the number of processors in the simulator. This measure is too conservative in many cases, and as such does not provide much information as to how much speedup one can hope to achieve by variations in the simulation scheme or faster communication of overhead messages, etc. (for any given processing speeds for computing the event messages). In this work we present two new measures of ideal speedup that would provide more realistic bounds on what can be expected under an ideal distributed simulation.

The above ideas have also been pointed out earlier in Kumar and Harous (1993), where we studied distributed simulation of open cyclic queuing networks using the NULL messages based scheme by Chandy *et al* (1978). In this paper we extend that work by further study of the same queuing networks using a variation of the distributed simulation scheme given in Chandy *et al* (1981).

Systems Under Study

Borrowing terminology from Chandy *et al* (1979) in the following, a system to be simulated is called the *physical system*. The physical system consists of a network of *physical processes* (or *pps* for short). Each physical system is simulated by a distributed simulator called the *logical system*. The logical system is a collection of *logical processes* (or *lps* for short), each one simulating a corresponding *pp*.

In this paper, we use the term *queuing networks* to refer to networks of *pps* from the five classes - *delay*, *fork*, *merge*, *sink*, and *source*. Their definitions are similar to the ones in Chandy *et al* (1981), with the exception that here we generalize the definitions of fork and merge processes so that they may possibly have only one output or input line respectively.

Queuing networks are useful in performance modeling of various kinds of service systems such as computer systems, computer-communication networks, telecommunication systems, and manufacturing systems. Here we study the queuing networks shown in Figures 2, 3 and 4.

The Distributed Simulation Scheme Under Study

In this paper we study the scheme based on deadlock detection and recovery by Chandy *et al* (1981). We made

some minor modifications to this scheme in order to improve the degree of concurrency.

THE ORIGINAL SCHEME OF CHANDY AND MISRA: The distributed simulation algorithm described in Chandy *et al* (1981) is based on the termination detection algorithm of Dijkstra *et al* (1980). This scheme by Chandy *et al* (1981) mainly repeats the following two phases: (i) simulate the physical system until deadlock, (ii) detect the deadlock and recover from it.

The scheme by Chandy *et al* (1981) is based on the synchronous communication of Hoare's CSP. This can cause a minor performance degradation when implemented on an asynchronous system. Also, in Chandy *et al* (1981), there are unnecessarily rigid rules as to when an *lp* is allowed to send or receive tuples. In particular, an *lp* sends out a tuple on a line only when the line clock for that line becomes minimum among all clock values of all the lines adjacent to the *lp*. This can obviously affect the degree of concurrency.

MODIFICATIONS: In order to achieve a higher level of concurrency we made the following modifications: (i) We use a simple termination detection algorithm using a central process (CP), (ii) An *lp* is allowed to send messages whenever possible, rather than having to wait for inputs even when it is possible to generate the next output, (iii) An *lp* can receive input any time, except when it is able to send out more outputs, (iv) Message communication in the logical system is assumed asynchronous, (v) We assume infinite input buffers at the input port of any *lp*.

Overall Design of Our Experiments

In our experiments we have a two level simulation. A sequential simulator simulates the behavior of the distributed simulator defined above while the distributed simulator is simulating the physical system (i.e., the queuing network being simulated).

SIMULATION PARAMETERS: We first list below the various simulation parameters that were used in our experiments. Any items not listed here should be assumed to have their obvious default values (e.g., the time that an *lp* takes to receive an input message should be assumed to be zero).

(1) Length of simulation of the physical system (Z),
(2) Mean interarrival times of messages at source *pp* (IT),
(3) Service time at a delay *pp*,
(4) Number of initial jobs in the physical system (#IJ),
(5) Communication delay for NONNULL messages (NNCOMDEL),
(6) Communication delay for NULL messages, (NCOMDEL),
(7) Time to compute a NONNULL message by an *lp*,
(8) Time to compute a NULL message by an *lp* (NULLTM),
(9) Time to compute an IDLE message by an *lp* (IDLETM),
(10) Time taken by the central processor to check if there is a deadlock (and break it) when it receives an IDLE message (CPTM).

MEASURES OF IMPORTANCE: The most important measure of performance is the actual speedup obtained by distributed simulation over sequential simulation, i.e., the ratio ASR = SST/DST of total elapsed times in the two methods of simulation:

SST = Sequential simulation time, i.e., the time taken by a sequential simulator to simulate some physical systems up to time Z.

DST = Distributed simulation time, i.e., the time taken by a logical system to simulate the same physical system up to time Z.

Next we define some terms to capture various notions of "ideal speedup ratio." The first ideal speedup ratio, called ISR1, is defined to be simply N where N is the number of processors in the distributed simulator. This is the usual notion of ideal speed up ratio used in the literature. Next, we define more refined notions of ideal speedup ratio. IDST2 (ideal distributed simulation time) = the time taken by a logical system to simulate the same physical system up to time Z, assuming that:

(1) any *lp* has all input NONNULL messages available whenever it needs them in its computations; equivalently, any *lp* has all its input NONNULL messages available to it when simulation starts,

(2) Communication delay is zero for both OVERHEAD and NONNULL messages (OVCOMDEL = NNCOMDEL = 0), and time spent in any activity other than computing NONNULL message is zero.

Then we define the ideal speedup ratio, ISR2, to be SST/IDST2.

Next we define ISR3 to capture an even more refined notion of "ideal speedup ratio": IDST3 (ideal distributed simulation time) = the time taken by a logical system to simulate the physical system up to time Z with the assumptions IDLETM=0, CPTM=0 and OVCOMDEL=0.

Then we define the ideal speedup ration ISR3 by ISR3 = SST/IDST3. Thus, ISR3 is the same as ASR when communication delay for overhead messages is set to zero, IDLETM and CPTM are zero.

Further Details of the Experiments

We simulated the three open cyclic queuing networks shown in Figures 2, 3 and 4. Network 1 is a simple open cyclic queuing network. Networks 2 and 3 are borrowed from Seethalakshmi (1979).

PHYSICAL SYSTEM PARAMETERS: The service times at every delay *pp*, and the interarrival times of messages at every *source pp* in these experiments were chosen to be exponentially distributed, with the following mean values.

- For a delay *pp i*, it is 3000.0 (this value is assumed in all our experiments).
- The mean interarrival time at a source *pp* is varied so as to result in varying levels of saturation at the delay *pps* in the system. In particular, we considered three cases: (i) non-saturated delay *pps* (less than 80% utilization), (ii) nearly saturated delay *pps* (nearly 100% utilization) and (iii) over-saturated delay *pps*. Each physical system was simulated for the physical system time interval [0, Z] where Z is chosen such that a large number of jobs is generated at the source *pp* and a sufficiently large number of messages is sent through each line in the physical system.

Logical System Parameters

In all experiments, the time to compute one output NONNULL message for any given class of *lps* is: (i) delay : 300, (ii) fork : 90, (iii) merge : 120, (iv) source : 60.

IN MEASUREMENTS OF ASR: The value of NNCOMDEL

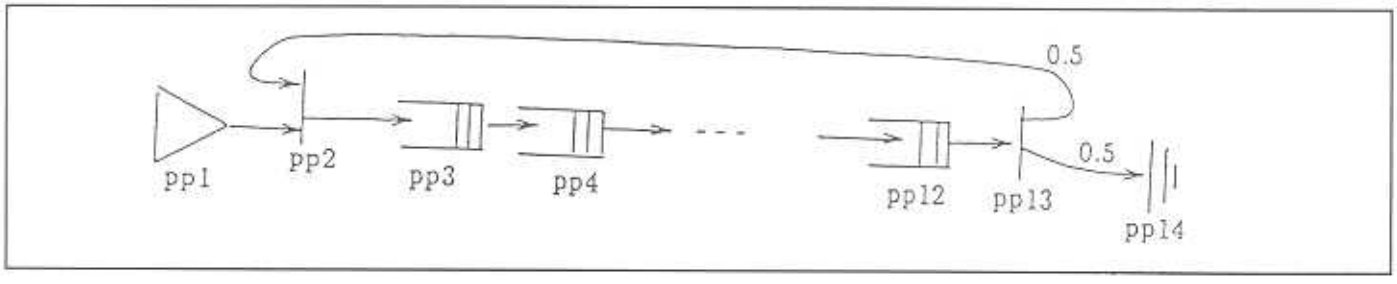

Figure 2. Queuing Network 1.

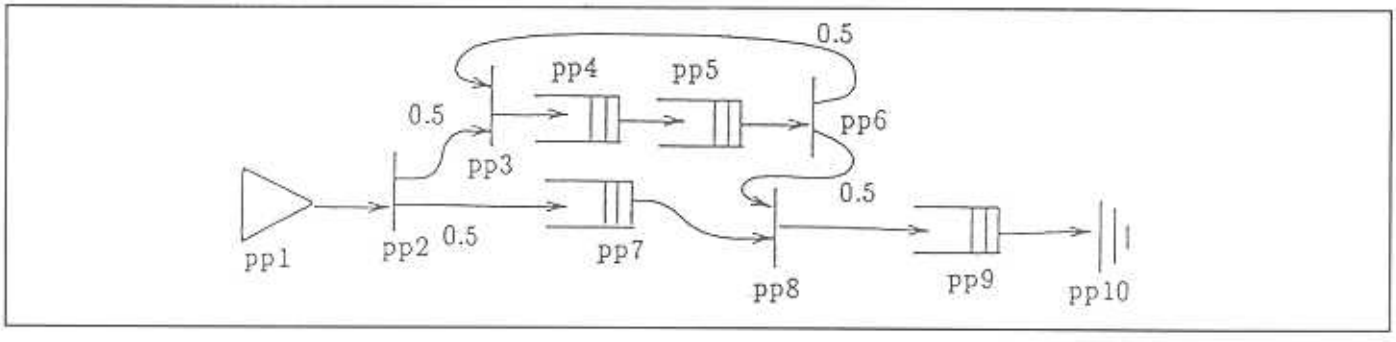

Figure 3. Queuing Network 2.

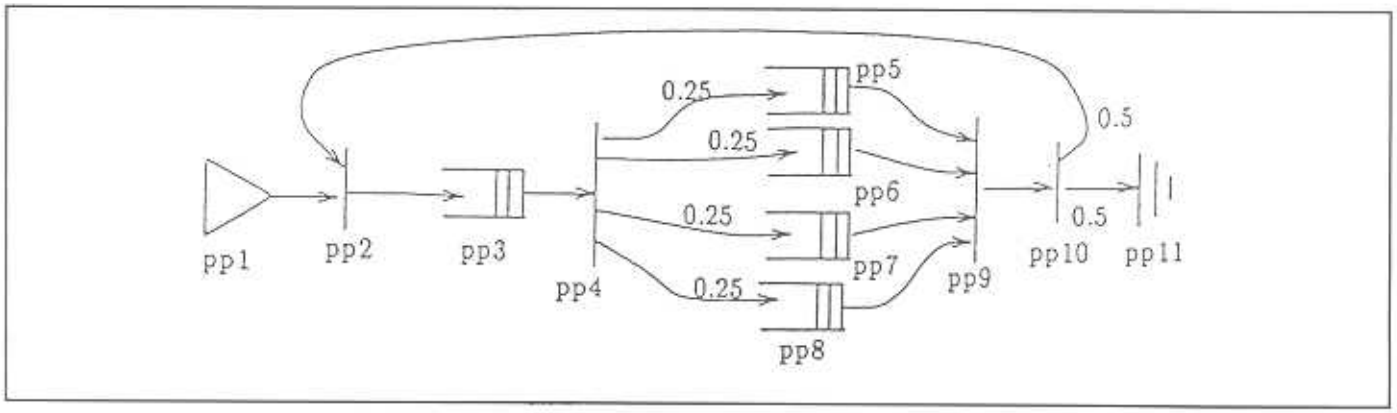

Figure 4. Queuing Network 3.

TABLE 1

Simulation Results Of Queuing Network 1 (#LJ=0).

IT	NNCOMDEL→	0	200	400	800	1600	3200	6400	ISR2
5000	ASR	8.1497	7.4609	6.7814	5.4915	3.7209	1.9589	1.1138	9.1213
	ISR3	8.3547	7.7143	7.0623	5.7778	4.0178	2.0343	1.1917	
6000	ASR	8.5207	7.6174	6.6571	5.1478	3.2265	1.7827	0.9292	9.6788
	ISR3	8.7682	7.9261	6.9708	5.4513	3.4224	1.8873	0.9852	
8600	ASR	8.5063	6.5966	5.0230	3.3565	1.9589	1.0611	0.5536	10.2013
	ISR3	8.8145	7.0266	5.4539	3.5105	2.0343	1.1040	0.5766	

is varied from 0 to 6400. The value of OVCOMDEL is same as NNCOMDEL. The values of NULLTM are: (i) delay : 150, (ii) fork : 45, (iii) merge : 60, (iv) source : 60. The values of IDLETM are: (i) delay: 15, (ii) fork : 15, and (iii) merge : 60. The value of CPTM is 150.

IN MEASUREMENTS OF ISR3: NNCOMDEL here is varied as in case of measuring ASR. By definition, OVCOMDEL, IDLETM and CPTM are zero.

IN MEASUREMENTS OF ISR2: In each experiment of ASR or ISR3, we compute ISR2. But note, as mentioned before, that the value of ISR2 is unaffected by changes in NNCOMDEL, OVCOMDEL, IDLETM or CPTM.

For simplicity no times are associated with other activities in the logical system. In particular, no time is associated with receiving a NONNULL message.

Simulation Results

QUEUING NETWORK 1: Table 1 shows the main experimental results regarding this network. The three values (in increasing order) of IT in Table 1 correspond, respectively, to oversaturated, nearly saturated, and non-saturated conditions of delay *pps* in the system. These data are summarized in Table 1.

Also, in these experiments we observed that for any given value of IT, the ratio of the total number of OVERHEAD messages to the total number of NONNULL messages increased monotonically as NNCOMDEL was increased from 0 to 6400. The range of values of this ratio is given below:

IT=5000: 512/3499 to 625/3499.
IT=6000: 490/3325 to 628/3325.
IT=8600: 452/2795 to 753/2795.

Next we state our main conclusions from the data shown in Table 1.

(1) For any value of IT, at NNCOMDEL=0 the value of ASR is fairly high; roughly one less than the number of delays *lps* in the system. As NNCOMDEL is increased, the value of ASR goes down and would become close to 1 for sufficiently large value of NNCOMDEL. Thus, for any given value of IT, there is a range of NNCOMDEL values between 0 and a fairly large value (which depends on IT), such that in this range the value of ASR is high.
(Note that in the experiments to determine ASR, the value of OVCOMDEL is the same as the value of NNCOMDEL.)

(2) As IT is increased, the range of values of NNCOMDEL which gives high ASR values is somewhat decreased.

(3) If the value of NNCOMDEL is small (say between 0 and the time it takes a delay *lp* to process a NONNULL message) and the value of IT is also small, then trying to reduce NNCOMDEL further won't significantly improve ASR value.

(4) Interestingly, for each entry of the Table the difference between ASR and ISR3 values is very small. [This is true irrespective of whether ASR values are low or high]. This shows that:
   a. Reducing OVCOMDEL, CPTM, or IDLETM would not have any significant effect on ASR.
   b. Consider variations of this distributed simulation scheme where one tries to reduce the number of overhead messages. Such variations are unlikely to significantly improve the performance.

(4) Consider the cases where performance is poor, i.e., ASR values are near or less than 1. (As indicated above, this happens when NNCOMDEL value is sufficiently high for any given IT value). It would be wrong in these cases to claim that the poor performance is "due to a large number of overhead messages":
   a. Since in these cases ISR3 is also near or less than 1, this indicates that even if, somehow, one could reduce these

overhead messages to zero, the ASR value is unlikely to be significantly higher than *1*.

b. As observed above, the ratio of the total number of OVERHEAD messages to the total number of NONNULL messages was not large in these experiments.

(6) In many cases ISR3 is significantly less than ISR2 or ISR1. This shows that in studying variations of this scheme which try to reduce overhead, ISR3 captures the ideal speedup ratio in a more realistic manner than ISR1 or ISR2.

(7) The values of ISR2 are less than ISR1 in these experiments. This clearly shows that in studying variations of *BASIC-related* schemes, ISR2 captures the ideal speedup ratio in a more realistic manner than ISR1.

(8) In an earlier paper (Kumar *et al*, 1993) we have reported a simulation study of the same physical system simulated by the NULL message based scheme of Chandy *et al* (1979), with similar simulation parameters. We note that the values of ASR and ISR3 shown here in Table 1 are somewhat worse than the ones we obtained using the NULL messages based distributed simulation scheme.

QUEUING NETWORK 2: Table 2(a) shows the main experimental results regarding this network. The three values (in increasing order) of IT in Table 2(a) correspond, respectively, to oversaturated, nearly saturated, and non-saturated conditions of *pp* 4 and *pp* 5. These data are summarized in Table 2(a).

Also, in these experiments we observed that for any given value of IT, the ratio of the total number of OVERHEAD messages to the total number of NONNULL messages increased monotonically as NNCOMDEL was increased from 0 to 6400. The range of values of this ratio is given below:

IT=2000: 713/5142 to 859/5142.
IT=3000: 733/4016 to 941/4016.
IT=4300: 953/2718 to 934/2718.

Next we state our main conclusions from the data shown in Table 2(a).

(1) For any value of IT, at NNCOMDEL=0 the value of ASR is fairly high. As NNCOMDEL is

TABLE 2(a)

Simulation Results Of Queuing Network 2 (#LJ=0).

IT	NNCOMDEL→	0	200	400	800	1600	3200	6400	ISR2
2000	ASR	5.5416	5.3357	5.1133	4.5978	3.6550	2.4698	1.4091	5.6780
	ISR3	5.5734	5.4009	5.2074	4.7534	3.8312	2.6225	1.5057	
3000	ASR	4.7184	4.1316	3.4761	2.4309	1.4175	0.7628	0.3965	5.0253
	ISR3	4.7707	4.2210	3.5805	2.5306	1.4777	0.7965	0.4145	
4300	ASR	2.8910	1.6643	1.1381	0.6954	0.3905	0.2081	0.1076	5.0142
	ISR3	3.1067	1.8393	1.2798	0.7937	0.4504	0.2415	0.1253	

TABLE 2(b)

Simulation Results Of Queuing Network 2 (IT=3000).

IT	NNCOMDEL→	0	200	400	800	1600	3200	6400	ISR2
2	ASR	4.6886	4.0645	3.3930	2.3729	1.3789	0.7454	0.3885	5.0010
	ISR3	4.7544	4.1763	3.5238	2.4898	1.4541	0.7883	0.4115	
4	ASR	4.7951	4.2518	3.5562	2.4612	1.4482	0.7881	0.4123	4.9915
	ISR3	4.8344	4.3149	3.6281	2.5243	1.4890	0.8115	0.4248	
8	ASR	4.7836	4.2111	3.5319	2.4296	1.4221	0.7724	0.4036	4.9990
	ISR3	4.8196	4.2726	3.6083	2.4931	1.4608	0.7946	0.4155	
16	ASR	4.9061	4.8427	4.5693	3.4916	2.1208	1.1603	0.6065	4.9912
	ISR3	4.9211	4.8680	4.6016	3.5233	2.1423	1.1726	0.6130	

increased, the value of ASR goes down - and would become <1 for sufficiently large value of NNCOMDEL. Thus, for any given value of IT, there is a range of NNCOMDEL values between 0 and a positive value (which depends on IT), such that in this range the value of ASR is high.

(2) This point is the same as point 2 in the case of network 1.

(3) Obviously in several cases NNCOMDEL has a significant impact on ASR (the impact is higher when NNCOMDEL is large or when IT is large). Therefore in these cases a good way to improve performance is to cut down on NNCOMDEL.

(4-7) These points are the same as points 4-7 in the case of network 1.

(8) For some cases (when values of NNCOMDEL and IT are both low), the values of ASR are close to ISR2. Thus, for these cases, the distributed simulation scheme considered is nearly optimal with respect to all *BASIC-related schemes*.

(9) The values of ASR and ISR3 obtained in these experiments are less than the ones obtained in our earlier experiments (Kumar *et al* 1993) with the distributed simulation scheme based on NULL messages, but the difference is small.

Variations: In addition to the above experiments, we also did the following experimentation with network 2. We placed a certain number of jobs (say #IJ) initially at the input of *pp*4. The other parameters were the same as above. As expected, this had negligible effect on the values of ASR, ISR2, and ISR3. We did this experimentation for #IJ varying between 2 and 16. The results are shown in Table 2(b).

QUEUING NETWORK 3: Table 3(a) shows the main experimental results regarding this network. The three values (in increasing order) of IT in Table 3(a) correspond, respectively, to oversaturated, nearly saturated, and non-saturated condition of *pp*3. These data are also shown in Table 3(a).

Also, in these experiments we observed that for any given value of IT, the ratio of the total number of OVERHEAD messages to the total number of NONNULL messages increased monotonically as NNCOMDEL was increased from 0 to 6400. The range of

TABLE 3(a)

Simulation Results Of Queuing Network 3 (#IJ=0).

IT	NNCOMDEL→	0	200	400	800	1600	3200	6400	ISR2
5500	ASR	1.5760	1.2020	0.9517	0.6577	0.3902	0.0745	0.1104	3.5196
	ISR3	2.5861	1.7968	1.3513	0.8872	0.5190	0.1116	0.1457	
6000	ASR	1.2292	0.7741	0.5430	0.3371	0.1892	0.0999	0.0513	3.5052
	ISR3	1.8914	1.0947	0.7621	0.4696	0.2636	0.1401	0.0724	
8570	ASR	0.7417	0.3668	0.2352	0.1368	0.0745	0.0390	0.0200	3.5007
	ISR3	1.0624	0.5146	0.3395	0.2020	0.1116	0.0589	0.0303	

TABLE 3(b)

Simulation Results Of Queuing Network 2 (IT=6000).

IT	NNCOMDEL→	0	200	400	800	1600	3200	6400	ISR2
2	ASR	1.2205	0.7590	0.5333	0.3301	0.1851	0.0977	0.0503	3.5044
	ISR3	1.8676	1.0756	0.7475	0.4602	0.2583	0.1373	0.0709	
4	ASR	1.2146	0.7531	0.5292	0.3271	0.1833	0.0967	0.0497	3.5046
	ISR3	1.8618	1.0698	0.7427	0.4570	0.2564	0.1363	0.0704	
8	ASR	1.2824	0.7531	0.5912	0.3711	0.2099	0.1111	0.0571	3.5065
	ISR3	1.9894	1.0698	0.8303	0.5158	0.2906	0.1547	0.0800	
16	ASR	1.2401	0.7914	0.5501	0.3406	0.1913	0.1011	0.0519	3.5049
	ISR3	1.9001	1.0967	0.7622	0.4691	0.2631	0.1399	0.0723	

values of this ratio is given below:

IT=5500: 13987/17755 to 14443/17755.
IT=6000: 24611/21243 to 25038/21243.
IT=8570: 26959/11614 to 26962/11614.

Next we state our main conclusions from the data shown in Table 3(a).

(1) In all cases here we find that ASR is less than or roughly equal to 1 (even when NNCOMDEL=0).

(2) Let us look at the differences between ASR and ISR3 values.Notice that in all cases there is some difference between ASR and ISR3 values. Therefore:
   a. Reducing OVCOMDEL, CPTM or IDLETM would somewhat improve ASR. (however, see point 3 below).
   b. There is a small potential for improving the speedup by devising variations of this distributed simulation scheme where one tries to reduce the number of overhead messages. (However, see point 3 below).

(3) Note that in almost all cases, both ASR and ISR3 are low (less than or roughly equal to 1). It would be wrong in these cases to claim that the poor performance is "due to a large number of overhead messages." Since in these cases ISR3 is small (i.e., less than or roughly equal to 1), this indicates that even if, somehow, one could reduce these overhead messages to zero, the ASR value is unlikely to be significantly higher than 1. (As seen above, the number of overhead messages is not very large.)

(4,5) These points are the same as points 6 and 7 in the case of network 1.

(6) In all the cases shown in the table 3(a), the values of ASR are significantly less than ISR2. This shows the possibility that ASR values may be increased significantly by considering other *BASIC-related* schemes.

(7) The values of ASR and ISR3 obtained in these experiments are much smaller than the corresponding values obtained in our earlier study (Kumar *et al* 1993) using the distributed simulation scheme based on NULL messages. For example, at NNCOMDEL=400, the ASR values were 2.3840, 1.3472, 0.2623 for the three IT values respectively.

Variations: In addition to the above experiments, we also did the following experimentation with this network. We placed a certain number of jobs (#IJ) initially at the input of *pp*3. The other parameters were the same as the above case when IT was equal to 6000. As #IJ is varied (between 2 and 16), it produced negligible effect on ASR and ISR3; this is expected since the initial jobs would leave the system after some initial period of simulation. the results are shown in Table 3(b).

Discussion

In this paper, we have presented a framework to address the following three important issues (discussed below) in studying performance of distributed simulation schemes. (Here we are considering only *BASIC-related* schemes, i.e., there is a one to one correspondence between the *pps* and the *lps* simulating them.)

(1) How to study the relationship between overhead and performance in a controlled manner? This is a difficult problem if one tries to see the relationship between the *number* of overhead messages and the performance. In our approach, one would vary the *time parameters* related to computation and communication of overhead messages (e.g., IDLETM, CPTM and OVCOMDEL).

(2) What kind of ideal speedup one could expect via considering algorithmic variations for the simulation scheme? Here we are given a physical system and characteristics of the hardware on which simulations are executed in terms of processing times of the event messages and their communication delays.

(3) How does performance of distributed simulation depend on hardware characteristics of the implementation? Again, our approach of *simulating* the logical system is very helpful in this regard. In particular, we illustrated how to see the effect of varying NNCOMDEL. Obviously using our approach of *simulating* the logical system, one can vary other parameters of the logical system as well in a straightforward manner.

References

CHANDY, K. M. and J. MISRA, 1978. Distributed Simulation: A Case Study In Design And Verification of Distributed Programs. *IEEE Transactions on Software Eng.* **5**, pp 440-452.

CHANDY, K. M. and J. MISRA, 1981. Asynchronous Distributed Simulation Via a Sequence of Parallel Computations. *Communications of the ACM*, **24**, pp198-205, 1981.

DIJKSTRA, E. W. and C. S. SCHOLTEN, 1980. Termination Detection for Diffusing Computation. Information Processing Letters, **11**, 1.

FUJIMOTO, R. M., 1988. Performance Measurements of Distributed Simulation Strategies. *Proceedings of the SCS Multiconference on Distributed Simulation*, 3-5 February 1988. U.S.A.

HAROUS, S., 1991. Studies in Distributed Simulation. Ph.D. Dissertation, Department of Computer Engineering and Science, Case Western Reserve University, Cleveland, Ohio, USA.

HOARE, C. A. R., 1978. Communicating Sequential Processes. *Communications of the ACM*. **21,** pp 666-777.

JEFFERSON, D. R., 1985. Virtual Time. *ACM Transactions on Programming Languages and Systems*. **7,** pp 404-425.

JHA, V. and R. BAGRODIA, 1996. A Performance Evaluation Methodology for Parallel Simulation Protocols. Proceeding of the Tenth Workshop on Parallel and Distributed Simulation. 22-24 May 1996. U.S.A.

KUMAR, D. and S. HAROUS, 1993. A Study of Achievable Speedup in Distributed Simulation Via NULL messages. *IEEE Transaction on Parallel and Distributed System*. **4,** pp 347-354.

REED, D. A., A. D. MALONY, and B. D. MCCREDIE, 1988. Parallel Discrete Event Simulation Using Shared Memory. *IEEE Transactions on Software Engineering*. **14,** pp 541-553.

SEETHALAKSHMI, M., 1979. A Study And Analysis of Performance of Distributed Simulation. Master's Report, Dept. of Computer Sciences, University of Texas, Austin, Texas , U.S.A.

SHOREY, R. and A. KUMAR, 1994. Stability and Performance of Distributed Simulators for Open Queueing Networks. Proceeding of the First International Workshop on Parallel Processing. 26-31 December 1994, India.

Received 6 January 1996
Accepted 10 March 1997