Proceedings of the 1994 Winter Simulation Conference
ed. J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila

Service Oriented Scheduling in Time Warp

Robert Rönngren
Rassul Ayani

Royal Institute of Technology (KTH)
Dept. of Teleinformatics
Stockholm
SWEDEN

ABSTRACT

In Time Warp discrete event simulations the performance depends on the scheduling of the logical processes. Proper scheduling can drastically reduce the overheads associated with the optimistic execution approach. In some simulations the logical processes can predict the time stamps of outgoing messages resulting from the execution of future events. In this paper a scheduling method exploiting this information is presented. Experimental results show that the proposed method reduces the rollback and memory overheads and improves the performance of the simulation.

1. INTRODUCTION

In Time Warp (Jefferson 1985) parallel discrete event simulations the scheduling of logical processes is based on incomplete information, i.e. events may be processed in violation of causality constraints. If causality constraints are violated the affected logical processes (LPs) have to be rolled back in simulated time to restore an earlier state. Inadequate scheduling policies may generate a substantial overhead degrading the performance. The scheduling method should minimize the overhead by favouring the execution of safe events (i.e. events that will not be rolled back). An important aspect of a simulation that affects the scheduling is the lookahead (Fujimoto 1990, Lin and Lazowska 1990). Lookahead is the ability of an LP to predict its future behaviour. For the sake of brevity it is assumed that the reader is familiar with the basic Time Warp terminology defined in (Jefferson 1985).

The selection of events to execute is based on the view of what is the cause of rollbacks. A rollback at an LP can be viewed from two different perspectives:

(i) the LP executed the rolled back events too early
(ii) the input event causing the rollback of the LP was not delivered on time

In the first view the responsibility for the rollback lies entirely with the LP that have to rollback, whereas in the second view it is shifted towards the LPs that should have produced input events for the LP experiencing the rollback. We argue that the second view is more natural than the first view. In fact optimistic synchronization is based on the assumption that input event messages in general are delivered on time. That is, if the simulation method does not exploit more than the available (optimistic) parallelism in the model there would be no rollbacks provided that the input events were delivered on time.

An LP can not in general predict what events it is going to receive. However it may be able to give predictions on future events that it will generate. Thus, a scheduling policy based on the second view could exploit more knowledge than methods based on the first view. Consequently the first view results in more conservative scheduling policies. The first view is the most common view in optimistic discrete event simulation, whereas the second view corresponds to a common view in real world problems such as real time systems (Liu et al. 1991).

In this paper we derive a scheduling policy based on the second view of the cause of rollbacks. This method is based on presampling of the random numbers used to generate the service times of future events. In this context we also present a method that allows earlier fossil collection than methods based on GVT.

The remainder of this paper is organized as follows: Related work is presented in Section 2. The methods for scheduling and fossil collection are derived in Section 3. Section 4 presents experimental results from these methods. Finally the contributions of this paper is summarized in Section 5.

2. RELATED WORK

Several policies for scheduling LPs have been proposed, for an overview see (Preiss, MacIntyre and Loucks 1992). They differ in the knowledge exploited for the scheduling decisions. Existing methods can be classified into three categories: (i) methods using no information on the LPs such as Round-Robin;(ii) exploiting knowledge of the execution history of the LP, i.e. using the Local Virtual Time (LVT) as priority; (iii) basing decisions on the immediate future by using the time stamp of the next event to be processed as priority, MMT (Minimum Message Timestamp).

Definition 1: A scheduling policy is adequate if it produces a rollback free schedule for any simulation executed on a single processor.

It is evident that among the above mentioned methods only MMT is adequate. Experimental results indicate that MMT scheduling in general is superior to the other methods (Preiss, MacIntyre and Loucks 1992).

However, there is, in general, more than one rollback free schedule for a simulation. A multitude of conservative methods for parallel discrete event simulation do, by definition, obtain rollback free simulations (Ayani and Rajaei 1992, Fujimoto 1990, Misra 1986, Nicol 1988). These methods depend on the possibility of identifying safe events that can be executed concurrently. This is based on the ability to predict the future of an LP, i.e. the lookahead (Fujimoto 1990, Lin and Lazowska 1990). A special case of conservative methods is window based schemes, such as Conservative Time Windows (CTW) (Ayani and Rajaei 1992). In CTW simulation time windows are calculated. The windows are bounded by time horizons determined for each LP. Events with timestamps smaller than the local horizon, i.e. within the window, are safe to execute. Another example of how to determine safe events is given in (Nicol 1988). In this method LPs presample the random number generators used to generate the service times. Thus lower bounds on the timestamps of outgoing event-messages can be determined before reception of the causing events. This knowledge is distributed to those LPs to which future events may be sent enabling these LPs to locally identify safe events.

Scheduling problems similar to those encountered in optimistic simulations often occur in real world problems. However, other criteria for the scheduling decisions are often used. In particular scheduling of tasks are often based on when the results of the execution of the task are needed rather than the arrival time of the task. The rationale for this service oriented scheduling is to give the receiver a better basis for its scheduling. One particular class of applications that have many similarities with optimistic discrete event simulation is real time systems. In real time systems failure to meet the deadline of a task causes a penalty to be paid (cp. rollbacks). Empirical evidence indicates that scheduling based on the deadlines of tasks (Earliest Deadline First scheduling) minimizes the penalties (Liu et al. 1991).

Thus there is a wide variety of design possibilities for scheduling policies.

3. SERVICE ORIENTED SCHEDULING

In this section we derive a scheduling policy based on the view that rollbacks are caused by late delivery of input events at the LPs experiencing rollbacks. This can be achieved given that the simulation model fulfils some conditions. In these discussions we assume a basic Time Warp mechanism such as the one defined in (Jefferson 1985).

3.1. Earliest Rollback Time

Figure 1 shows a snapshot of a simulation with 4 LPs. Informally we define the earliest rollback time (ERT) as the earliest global (simulated) time that any LP may be rolled back to. ERT at the time of the snapshot in this example is determined by the receive time of the event sent to LP4 from LP2 as a result of the execution of event e2.

In order to formally define ERT we define EOMT(LP_i), the Earliest Output Message Time of LP_i as:

Definition 2:

```
EOMT(LP_i) = if the input queue of LP_i is not empty
               then
                 if there would be any output messages
                 as a result of the execution of any of
                 the input messages then
                   minimum receive time of any output
                   message resulting from execution of
                   any message in the input queue
                 else ∞        /*no output messages*/
               else ∞
```

The Service Oriented Scheduling Time (SOST) of an LP is defined as:

Definition 3:

```
SOST(LP_i) =   if LP_i is experiencing a rollback
               then RT(LP_i)
               else EOMT(LP_i)
```

Where RT(LP_i) is the rollback time of the LP. ERT at a specific (wall clock) time of a simulation is defined as:

Definition 4:

```
ERT =          min(∀i SOST(LP_i ),
               receive time of all messages in
               transition)
```

Theorem 1: Events with timestamps less than ERT are safe to execute (i.e. can never be rolled back).

This follows directly from the definition of ERT. Thus ERT defines a horizon for safe event executions. However it does not capture all safe events. As an example we can see that event e3 in Figure 1 is a safe event though it has a time stamp greater than ERT at the time of the snap shot.

When a safe event is executed there is no need to maintain a copy of the event nor of the anti-messages. The resulting state of the execution of a safe event need only to be saved immediately before the execution of an unsafe event (all under the assumption that there are no ERT-straddling mechanisms in the Time Warp system).

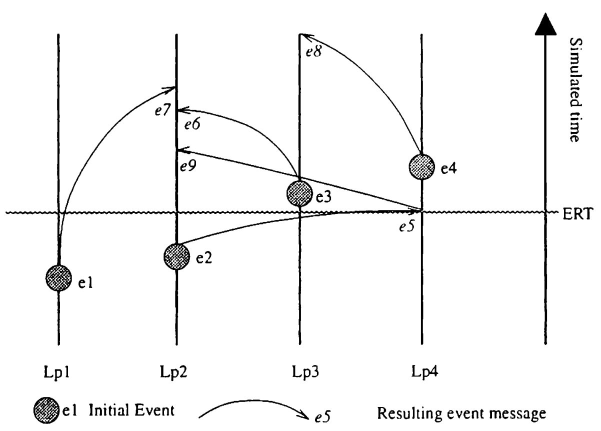

Figure 1. Snapshot of a Simulation with 4 LPs

3.2 A Service Oriented Scheduling method

Theorem 1 states that events with timestamps less than ERT are safe. Thus scheduling would be facilitated if ERT advances as rapidly as possible, increasing the number of safe events eligible for scheduling. This can be achieved by using the SOST as priority for the scheduling. This is possible for simulation models where service times can be calculated in advance without being dependent on information carried by the future input events. Examples of such simulations can be found in FCFS closed stochastic queueing networks (Nicol 1988).

Theorem 2: Any Time Warp simulation scheduled by SOST will terminate if the corresponding sequential simulation is finite.

Proof: The corresponding sequential simulation comprises a finite number of events. Scheduling by SOST will eventually schedule the LP containing the ERT regulating event. This event is a safe event and can never be rolled back. Thus the number of committed events will increase. Eventually the number of committed events in the Time Warp simulation will be equal to the number of events in the sequential simulation.

Theorem 3: SOS scheduling is adequate.

Proof: It follows from the definition of SOST that when the simulation is executed on one processor, an event scheduled for execution has a smaller time stamp than any other future event not yet generated.

We will refer to scheduling where LPs are scheduled by SOST as Service Oriented Scheduling (SOS). As an example of an execution of a simulation using SOS we consider the simulation in Figure 1. Assume that LP1 and LP2 reside on one processing element, PE1, and that LP3 and LP4 reside on another processing element, PE2. Unit execution time is assumed for an event execution and the rollback of an event. The execution histories for MMT and SOS scheduling are given in Table 1.

SOS scheduling reduces the execution with 4 cycles, i.e. two rollbacks. This example illustrates the rational for SOS. It provides information, i.e. input event messages, to the receiver as early as possible to reduce the probability (and the length) of rollbacks at the receiver. However, SOS schedules LPs more aggressively than MMT (i.e. LPs that are further into the future could be scheduled earlier). This could cause rollbacks that would not have occured using MMT, which may reduce the benefits of SOS as compared to MMT scheduling.

Table 1. Execution History of the Simulation Depicted in Figure 1 for MMT and SOS Scheduling respectively.

	MMT scheduling		SOS scheduling	
Cycle	PE1	PE2	PE1	PE2
1	Execute e1	Execute e3	Execute e2	Execute e3
2	Execute e2	Execute e4	Execute e1	Execute e5
3	Execute e6	Rollback e4	Execute e9	Execute e4
4	Execute e7	Execute e5	Execute e6	Execute e8
5	Rollback e7	Execute e4	Execute e7	
6	Rollback e6	Execute e8		
7	Execute e9			
8	Execute e6			
9	Execute e7			

3.3 Fossil Collection at ERT

SOS scheduling may delay the GVT advancement, since it will delay the execution of LPs generating output events far into the future. Delayed GVT advancement would hinder fossil collection. This is undesirable as it would increase the memory consumption, identified as a potential bottle-neck of Time Warp by many researchers (Bauer and Sporrer 1993, Lin 1990, Lin et al. 1993, Preiss, MacIntyre and Loucks 1992). Fossil collection is however not dependent on GVT. Events and states can be fossil collected as soon as they are no longer needed for state restoration purposes in rollbacks. ERT determines a lower bound on the smallest possible rollback time. Thus fossil collection can be performed up to ERT leaving one state with timestamp less than ERT at each LP (This assumes that there are no GVT/ERT straddling memory management mechanisms etc.). We refer to this as Earliest Rollback Time Fossil Collection (ERTFC). To further improve memory management the scheduler may perform a *clean-up pass* after each ERT calculation, forcing LPs with safe events to be executed first. This will enforce the advancement of GVT. The clean-up pass could preferably be performed in conjunction with fossil collection.

As ERT advances more rapidly than GVT in simulated time ERTFC makes earlier fossil collection possible and thus reduces the memory overhead. The event and state saving overhead can further be reduced by the exploitation of safe events, as observed in Section 3.1. A consequence of this is the following lemma:

Lemma 1: There exists a simulation for which Time Warp using ERTFC and knowledge of safe events can execute using less memory than the corresponding sequential simulation.

A corresponding lemma is presented in (Lin and Preiss 1991) for Chandy-Misra simulations (Misra 1986). We use the example from (Lin and Preiss 1991), Figure 2, to prove the lemma for Time Warp.

Proof: Consider the simulation depicted in Figure 2. In a sequential simulation the maximum memory consumption is 3 states and 4 events. In Time Warp using ERTFC, ERT is 3 initially. Hence the state of LP1 is not checkpointed after the execution of event e1 which generates e3 and e4. Event e1 can be immediately discarded. ERT is advanced to 5. In the next step events e2, e3 and e4 are executed in parallel generating events e5 and e6. Again there is no need to copy the states and the executed events can be immediately discarded. ERT is advanced to ∞. Finally events e5 and e6 are executed. Thus the Time Warp system executes using a maximum memory of 3 states and 3 events.

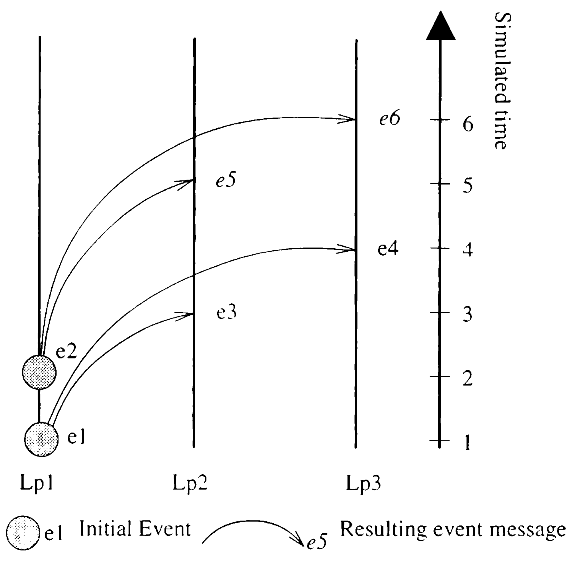

Figure 2. A Simulation using Less Memory when Executing Time Warp with ERTFC and exploitation of safe events than the Corresponding Sequential Simulation.

3.4 Relaxations on SOS scheduling

In many systems complete advance knowledge of the service times does not exist. However the service time often consists of two components: one constant and one random (both assumed to be non-negative). Digital logic circuits is one important example of such simulations. For such simulations the SOST (Definition 2) can be approximated using the known constant component. The approximated ERT will be less or equal to the true ERT. Hence ERTFC will still be correct. Using the approximated SOST as priority in the scheduling together with clean-up passes (Section 3.3) leads to a scheduling policy that guarantees GVT advancement.

EOMT, Definition 2, is defined as the minimum receive time of any output message generated from any input message present in the input queue of the LP. This indicates that presampling has to be performed for every input message present in the input queue. In practice however, it is often possible to use conditional knowledge to reduce the presampling. If for example consecutive output events have strictly increasing receive time stamps, EOMT (SOST) is determined by the service time of the first event in the input queue.

4 EXPERIMENTAL RESULTS

This section presents the experiments performed to evaluate the impact of SOS scheduling of LPs and ERTFC on a Time Warp implementation using two different (global) scheduling policies. The experiments were conducted on a shared memory bus based multiprocessor Sequent Symmetry[1] S81 equipped with 26 Intel 386i 16MHz processors. In these experiments knowledge of safe events has not been exploited to reduce event and state saving (Section 3.2) as we were mainly interested in the effects of the scheduling.

The Time Warp implementation uses aggressive direct cancellation (Fujimoto 1989). Inexpensive GVT/ERT calculations are performed once every second followed by fossil collection. The two scheduling strategies implemented are static and dynamic scheduling (Ahmed, Rönngren and Ayani 1994). In static scheduling the LPs are statically assigned to processors and each processor maintains its own scheduling queue. In dynamic scheduling two centralized scheduling queues are used for LPs that are to rollback and for LPs ready for execution, respectively. Priority is given to rollbacks. The centralized scheduling queues are implemented by parallel access skew heaps (Jones 1989, Sleator and Tarjan 1985). The dynamic scheduling drastically reduces the number of rollbacks (Ahmed, Rönngren and Ayani 1994). This more than compensates for the overhead introduced by the centralized data structures in dynamic scheduling. The default LP scheduling policy is MMT (Preiss, MacIntyre and Loucks 1992).

Two different closed stochastic queueing models were selected for the experiments. The first model is a simple PHOLD model (Fujimoto 1989). It is a fully connected net with 64 nodes where a constant message population circulates among the LPs. The timestamp increments are taken from an exponential distribution with mean 1 and messages are equally likely to be forwarded to any other process.

The second model, Hot Spot, is a modification of the first. It has 8 hot spots to which 50% of all messages are routed. The hot spots move randomly among the nodes. The Hot Spot experiment is designed as a simple model of a mobile phone system with 64 base stations and 8 mobile phones.

The average number of messages per node (message density) was 10 in all experiments. In both models exact calculations of the SOST are performed.

The measured entities are: event rate, the number of committed events per second; efficiency, the number of committed events to the number of processed events (events re-executed in coast forwards excluded); and the memory consumption in pages (the page size is 4k).

Figures 3 and 4 show the efficiency and the event rate for the PHOLD experiment. In these experiments there are no significant difference (the curves are indistinguishable) between MMT and SOS scheduling. Figure 5 however, shows that the memory consumption could be improved by the more aggressive ERTFC fossil collection.

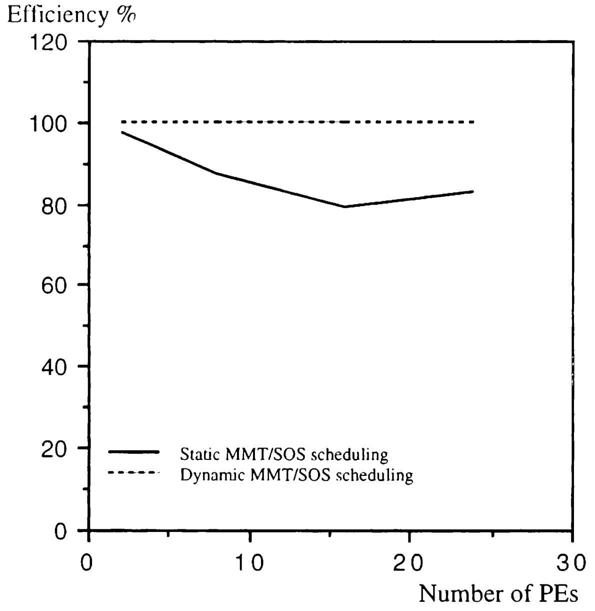

Figure 3. Efficiency for PHOLD experiments, message density 10

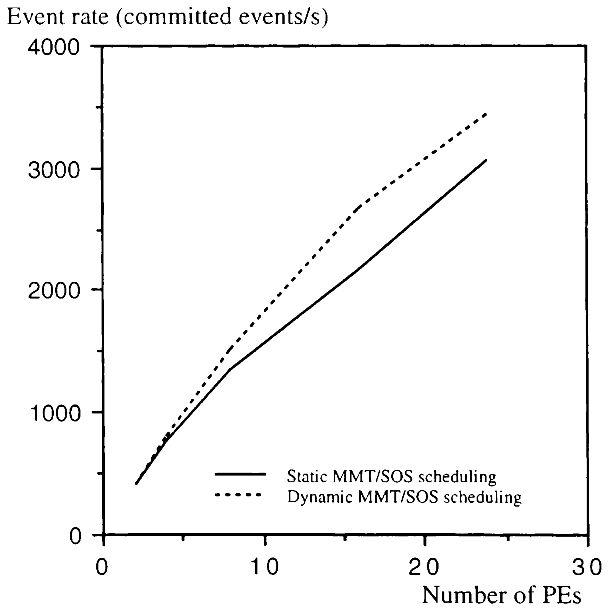

Figure 4. Event rate for PHOLD experiment, message density 10

[1] Symmetry is a trade mark of Sequent Computer Systems, Inc.

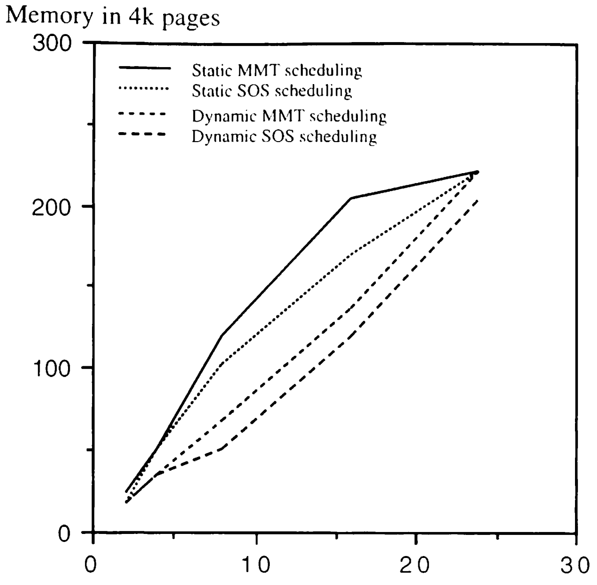

Figure 5. Memory consumption PHOLD experiment, message density 10

Figures 6 - 8 depicts the performance for the Hot Spot experiments. These experiments have a lower degree of available parallelism and are subject to higher rollback frequencies. Thus they are harder to schedule.

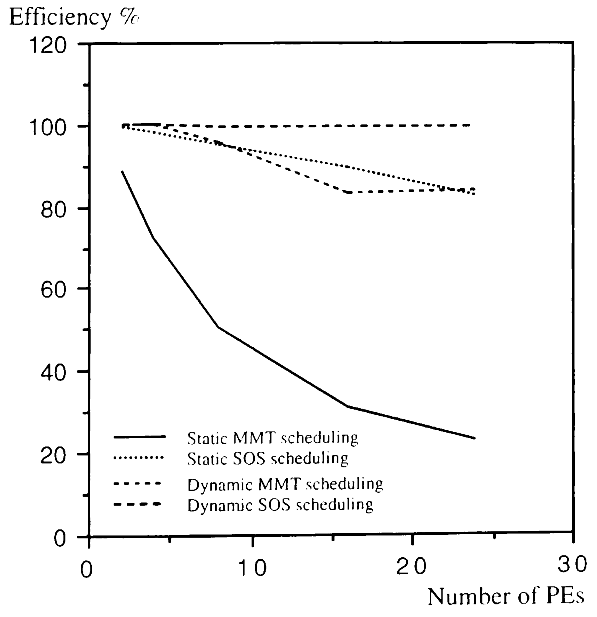

Figure 6. Efficiency for Hot Spot experiment, message density 10

In the Hot Spot experiments the number of rollbacks is drastically reduced with SOS scheduling and the execution speed is increased with up to 70%. When SOS scheduling is combined with ERTFC fossil collection memory consumption is reduced with up to 50%. However the increased efficiency (i.e. decreased rollback frequency) opens possibilities for further improvements of memory management (i.e. state saving overhead) as the optimal checkpoint interval is proportional to $(1/r)^{1/2}$, where r is the rollback frequency (Lin 1990, Lin et al. 1993). We also note that SOS scheduling scales better than MMT scheduling.

5 CONCLUSIONS

In this paper we presented a method for scheduling LPs in Time Warp simulations. The method exploits presampling of service times. Based on this information LPs are scheduled to increase the probability that the results of their execution are produced on time for the receiving LPs. Experimental results indicate that the proposed scheduling method improves performance of the Time Warp system for some applications. In particular it can reduce the number of rollbacks and increase the execution speed for simulations with a low degree of available parallelism. The reduction of rollbacks could be exploited to reduce the state saving overhead by allowing larger checkpoint intervals. A method that allows more aggressive fossil collection than methods based on GVT calculations was also proposed. The more aggressive fossil collection reduces the memory overhead and can be used independently of the scheduling scheme. The proposed methods are only applicable when the service times of the logical processes can be presampled or approximated in advance. However many important classes of simulations, such as digital logic simulation and some closed stochastic queueing networks, have this property.

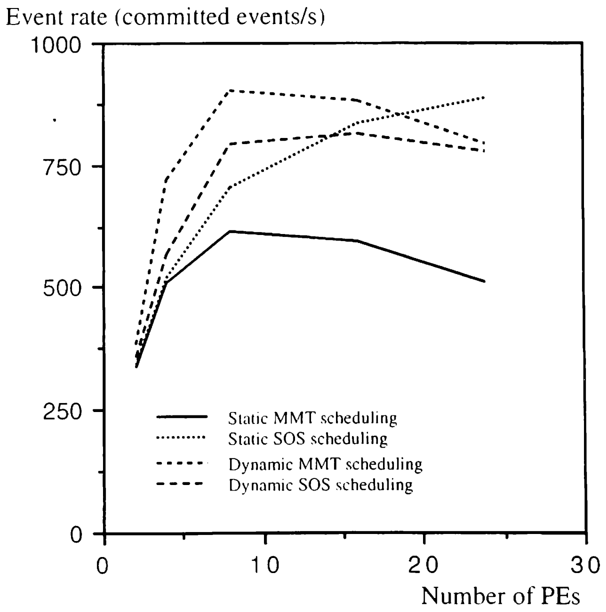

Figure 7. Event rate for Hot Spot experiment, message density 10

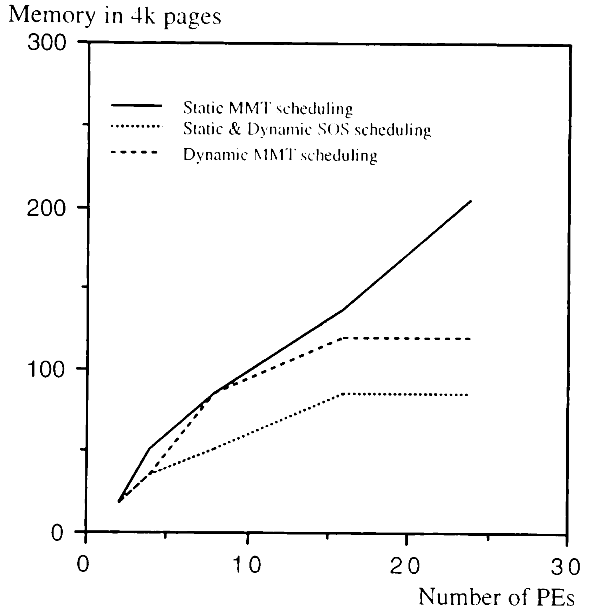

Figure 8. Memory consumption for Hot Spot experiment, message density 10

ACKNOWLEDGEMENTS

This work is part of a research project on Advanced Simulation Techniques and Tools funded by the Swedish National Board for Industrial and Technical Development (NUTEK) under contract number 93-3319.

REFERENCES

Ahmed H., Rönngren R. and Ayani R. 1994. Impact of Event Scheduling on Performance of Time Warp Parallel Simulations. In *Proceedings of the 27th Annual Hawaii International Conference on System Sciences*, Vol II, 455-462

Ayani R. and Rajaei H. 1992. Parallel Simulation using Conservative Time Windows. In *Proceedings of the 1992 Winter Simulation Conference*, 709-717

Bauer H. and Sporrer C. 1993. Reducing Rollback Overhead in Time-Warp Based Distributed Simulation with optimized Incremental State Saving. In *Proceedings of the 26th Annual Simulation Symposium*, 12-20, March 1993

Fujimoto R. 1989. Time Warp on a Shared Memory Multiprocessor. *Transactions of the Society for Computer Simulation*, Vol. 6, No. 3, pp. 211-239, July 1989.

Fujimoto R. 1990. Parallel Discrete Event Simulation. *Communications of the ACM*, Vol. 33, No. 10, pp 30-53, October 1990

Jeffersson D. 1985. Virtual Time. *ACM trans. on Programming Languages and Systems* Vol. 7, No. 3, pp. 404 - 425, July 1985.

Jones D.W. 1989. Concurrent Operations on Priority Queues. *Communications of the ACM*, Vol. 32, No. 1, pp. 132-137, Jan. 1989.

Lin Y. 1990. Understanding the Limits of Optimistic and Conservative Parallel Simulation. PhD thesis, Dept. of Computer Science and Engineering, University of Washington, Tech. Report 90-08-02, August 1990

Lin Y. and Lazowska E. D. 1990. Exploiting lookahead in Parallel Simulation. *IEEE Transactions on Parallel and Distributed Computing Systems*, Vol. 1, No. 4, pp 457-469, October 1990

Lin Y. and Preiss B. 1991. Optimal Memory Management for Time Warp Parallel Simulation. *ACM Transactions on Modeling and Computer Simulation*, Vol. 1, No. 4, pp 283-307, October 1991

Lin Y., Preiss B., Loucks W. M. and Lazowska, E. D. 1993. Selecting the Checkpoint Interval in Time Warp Simulation. In *Proceedings of the 7th workshop on Parallel and Distributed Simulation (PADS93)* pp. 3-10, Vol 23, No 1, July 1993

Liu J.W.S, Lin K., Shih W., Yu A. C., Chung J. and Zhao W. 1991. Algorithms for Scheduling Imprecise Computations. *IEEE Computer*, pp 58-68, May 1991

Misra J., 1986. Distributed Discrete-Event Simulation. *ACM Computing Surveys*, Vol. 18, No. 1, pp 39-65, March 1986

Nicol D. M., 1988. Parallel Discrete Event Simulation Of FCFS Stochastic Queueing Networks, *ACM SIGPLAN Symposium on Parallel Programming: Experience and Applications, Languages and Systems*, pp 124-137, 1988

Preiss B., MacIntyre I. D. and Loucks W. M. 1992. On the Trade-off Between Time and Space in Optimistic Parallel Discrete-Event Simulation. In *Proceedings of the 6th Workshop on Parallel and Distributed Simulation*, pp. 33-42, Vol 24, No 3, January 1992

Sleator D. D. and Tarjan R. E., 1985. Self-Adjusting Binary Search Trees. *Journal of the ACM*, Vol. 32, No. 3, pp. 652-686, Jul. 1985.

AUTHOR BIOGRAPHIES

Robert Rönngren Is a PhD candidate at the Department of Teleinformatics at the Royal Institute of Technology (KTH) in Stockholm, Sweden. His research interests focus on sequential and parallel discrete event simulation. He received a MS degree in engineering physics from the Royal Institute of Technology in 1986.

Rassul Ayani Is an Associate Professor at the Department of Teleinformatics, Royal Institute of Technology (KTH) in Stockholm, Sweden. He received his Dipl. Ing. degree from Technische Hochschule in Vienna, Austria (1970) and his MS and Ph.D. degree from KTH. His research interests are in parallel architecture, parallel algorithms and parallel simulation. He is Associate Editor of the ACM Transactions on Modeling and Computer Simulation (TOMACS).