Proceedings of the 1996 Winter Simulation Conference
ed. J. M. Charnes, D. J. Morrice, D. T. Brunner, and J. J. Swain

ADAPTIVE PROTOCOLS FOR PARALLEL DISCRETE EVENT SIMULATION

Samir R. Das

Division of Computer Science
The University of Texas at San Antonio
San Antonio, TX 78213-0667, USA

ABSTRACT

This paper reviews issues concerning the design of adaptive protocols for parallel discrete event simulation (PDES). The need for adaptive protocols are motivated in the background of the synchronization problem that has driven much of the research in this field. Traditional conservative and optimistic protocols and their hybrid variants are also discussed. Adaptive synchronization protocols are reviewed with special reference to their characteristics regarding the aspects of the simulation state that influence the adaptive decisions and the control parameters used. Finally, adaptive load management and scheduling strategies and their relationship to the synchronization protocol are discussed.

1 INTRODUCTION

Parallel discrete event simulation or PDES refers to parallel execution of discrete event simulation programs on a multiprocessor system or on a network of workstations. Over the past decade, there has been a considerable amount of activity in this field. There are several motivating factors. First, with the advent of technology, many application areas (e.g., simulations of computer architecture, VLSI circuits, communication networks) are getting larger and more complex. Thus speeding up simulations of such systems has become more important so that a sufficiently large design parameter space can be explored. The interest is also fueled by the common availability of multiprocessor systems (especially symmetric multiprocessors) and high-speed network based computing platforms (e.g., a cluster of workstations connected by high-speed networking hardware such as ATM or Myrinet switches). Second, it has been observed that many real simulations indeed contain a significant amount of parallelism. Third, the synchronization problem inherent in PDES is traditionally considered a challenging problem. Unlike many other areas of parallel computing (e.g., scientific computing) most discrete event simulation models are asynchronous and possess irregular, random or data dependent behavior. Thus, most of the research in PDES so far is centered around design and evaluation of synchronization protocols. For a review of the current state-of-the-art in PDES research see (Fujimoto 1990; Nicol and Fujimoto 1994; Ferscha 1995a).

There are two major classes of synchronization protocols that evolved: *conservative* and *optimistic*. The *conservative* protocols often rely on certain information on the behavior of the simulation model, such as the communication topology, or *lookahead* (i.e., the model's ability to predict the future course of events) (Fujimoto 1990) to determine which events are "safe" to process. They may also require certain implements such as message delivery in timestamp order, as in (Chandy and Misra 1979), or periodic global synchronizations as in (Lubachevsky 1989; Nicol 1993). All these make it difficult to develop general-purpose parallel simulators that can perform efficiently with little or no knowledge of the behavior of the underlying simulation model. Even if such knowledge is available, conservative mechanisms, being based on blocking, may fail to exploit much of the parallelism in the simulation model (Fujimoto 1989).

On the other hand, optimistic synchronization protocols such as *Time Warp* (Jefferson 1985) take a very different approach to resolve dependence between simulation events. They try to resolve dependence *a posteriori* by letting dependence violations to occur. If such violation is detected, erroneously executed computations are "undone" using a *rollback* mechanism. Time Warp uses *checkpointing* in addition to a *message cancellation* mechanism (based on antimessages) to implement rollback. The principal advantage of Time Warp over blocking-based, conservative protocols is that Time Warp offers the potential for greater exploitation of parallelism and, perhaps more importantly, greater transparency of the synchronization mechanism to the simulation programmer (Fujimoto

1990). The latter is due to the fact that Time Warp is less reliant on *a priori*, application specific information regarding which simulation events depend on which others.

Time Warp, however, is prone to inefficient execution in many situations because of over-optimistic behavior. For example, some LPs may execute at a much higher simulation time than others. This may lead to several performance problems, such as the possibility of long and/or cascaded rollbacks (Lubachevsky, Shwartz, and Weiss 1991) and significant memory management overheads (Das and Fujimoto 1993; Das and Fujimoto 1994). Regardless of the over-optimistic behavior, one key performance problem in Time Warp is the checkpointing related overheads.

In spite of the above problems, both conservative and optimistic protocols had a fair amount of success in parallelizing various simulation models (Fujimoto (1993a) has a review of different PDES application areas studied). However, PDES has a very limited penetration in the general simulation modeling community (Fujimoto 1993b). One often cited reason is the current level of sophistication required to effectively exploit the technology. For an acceptable level of performance justifying the use of parallel resources, careful attention must be paid to the interplay of the synchronization mechanism with the application simulation model and the underlying systems architecture. Without an insight in all the above, it is often unclear which synchronization protocol will be the best or even how certain details of the protocol (e.g., checkpointing interval in Time Warp (Fleischmann and Wilsey 1995)) should be selected. Thus there is a recent interest in the PDES community in investigating "adaptive" or dynamic mechanisms to choose an appropriate synchronization protocol (conservative, optimistic, or some hybrid variant) or parameters (such as the checkpointing interval in Time Warp) of a possibly statically chosen protocol. This paper reviews the state-of-the-art in such adaptive mechanisms for PDES. We start with some background in the traditional conservative vs. optimistic debate in the next section and then introduce the hybrid protocols in Section 3. In Section 4, we discuss the adaptive synchronization protocols, followed by dynamic load balancing/scheduling mechanisms in Section 5. Conclusions are presented in Section 6.

2 CONSERVATIVE VS. OPTIMISTIC DEBATE

The question of relative merits of conservative or optimistic protocols has often been raised. However, it is now a widely accepted belief in the PDES community that a formulation of general rules of superiority of one protocol versus another is hard (Ferscha 1995a). Still, some analytical results tend to favor optimistic mechanisms more than their conservative counterparts. Lipton and Mizell (1990) showed that optimistic mechanisms can arbitrarily outperform conservative mechanisms. But conservative mechanisms can outperform the optimistic mechanisms by at most a constant factor. This issue may be of purely theoretical interest, as it only represents extreme case scenarios. Also, the analysis is based on simple assumptions such as constant rollback cost and zero message communication or state saving/restoration cost, which may be biased towards optimistic mechanisms. In an unrelated work, Lin and Lazowska (1990) showed that Time Warp is "optimal" (i.e., can complete in "critical path" time) under the assumption that no correct computation is rolled back by incorrect computations. This study also ignores all overheads. It is also unclear how to guarantee such an assumption except in specific simulation models so that an optimal performance can be expected.

Without any clear choice between optimistic and conservative protocols, a conventional wisdom in the PDES community has been to study *hybrid* variations that take an intermediate approach between purely conservative (block-resume) and purely optimistic (lookahead-rollback) extremes. Intuitively, these protocols use a form of "reasonable" or "calculated" optimism so that they can exploit the benefits of optimism (e.g., ability to extract more parallelism, less reliance on model-specific information) without some its liabilities (e.g., rollback overhead, large memory usage). Reynolds, Jr. (1988) was one of the first to study a framework for specifying hybrid PDES protocols. He defined the notions of "aggressiveness" and "risk." An aggressive PDES protocol may process events in out-of-timestamp order and they may be required to rollback. A risky protocol, in addition to being aggressive, may pass messages that have been generated based on aggressive processing. Thus a message generated in a risky protocol may need to be canceled. Time Warp exhibits the maximal form of aggressiveness and risk. On the other hand, conservative protocols do not employ any aggressiveness or risk. According to Reynolds, Jr. (1988), aggressiveness and risk may be considered two important design variables that may influence the performance of a PDES protocols.

3 HYBRID PROTOCOLS

Hybrid protocols belong to primarily two categories: those that impart optimism to conservative protocols and those that introduce blocking to optimistic proto-

cols. Indeed some of them is able to seamlessly switch between conservative and optimistic extremes with appropriate tuning of one or more control parameters (Rajaei, Ayani, and Thorelli 1993), and thus may be difficult to classify. The classification followed below is based more on the spirit of the protocol, rather than its ability to move seamlessly between two extremes.

3.1 Adding Optimism to Conservative Protocols

In the *speculative execution* protocol proposed by Mehl (1991) logical processes compute future events in their idle time, however such speculative computation does not modify the local state or the event queues. A scratchpad area in the memory is used as state and the events generated are stored in a private buffer. If it turns out later that the speculative execution is correct, local state can be quickly updated and the events are sent to the appropriate destinations. Thus support for rollback is not required. Dickens and Reynolds, Jr. (1990) earlier presented a similar, albeit more aggressive approach, where local state and local event queues can be modified by such speculative execution. However, events destined to remote processors are not sent out until they are determined to be correct. Thus antimessages are not required. Rollbacks are possible, but they are only "local" in the sense that there cannot be any cascades.

Steinman (1992a) proposed the *Breathing Time Bucket* algorithm, which also has similar, aggressive, but risky-free approach. Here, logical processes process events in cycles. In each cycle the maximum number of independent events are processed in the following way. Each LP processes events in timestamp order until the timestamp (called the local "event horizon") of the earliest new event it generates in the current cycle. New events generated are accumulated in local buffers and are not actually passed to the destination processes. The LPs synchronize (at least implicitly) at the end of the cycle to compute the minimum of all local event horizons, This minimum is the global event horizon. The messages generated by events with timestamp less than or equal to the global event horizon are then sent to their appropriate destinations. This may cause rollbacks, which, however, are only local. Analytical modeling as well as experimental results showed that this approach can be quite efficient if enough events can be processed in each cycle.

Lubachevsky, Shwartz, and Weiss (1989) proposed an extension to the *bounded lag* protocol (Lubachevsky 1989) to incorporate optimism to a purely conservative mechanism. The bounded lag protocol is a conservative protocol that relies on a conservative estimate of the minimum simulation time distance (called *lag*) between a pair of logical processes as a basis for determining events safe to process. In the extension, called *filtered rollback*, the simulation time distance is more aggressively estimated and the protocol runs the risk of overestimation (possibly rarely). Thus dependence violations are possible which are taken care of by rollbacks and event cancellation just like Time Warp. The protocol is thus both aggressive and risky.

3.2 Throttling Optimistic Protocols

Many protocols try to control the optimism in Time Warp. One key technique often used is to limit all event computations within a window of simulated time beyond the *global virtual time* or GVT (GVT in Time Warp denotes a lower bound on the timestamps of all future rollbacks and thus defines a commitment horizon (Jefferson 1985)). This limits length of rollbacks, thus preventing long cascades. This also bounds memory usage, a major concern in Time Warp simulations. The original work exploiting this idea was by Sokol, Briscoe, and Wieland (1988), where they proposed the *moving time window* (MTW) protocol. A key problem here is the determination of the appropriate size of the time window (as observed by Reiher, Wieland, and Jefferson (1989)) Too narrow a window will reduce rollbacks, but will admit only a small amount of parallelism. Too large a window can potentially exploit more parallelism, but rollbacks may increase as well. A similar idea is also explored by Turner and Xu (1992) in the *bounded Time Warp* (BTW) protocol, where no events are processed beyond a bound in simulation time until all processes have reached that bound, when a new bound is established.

In the MIMDIX system Madisetti, Hardaker, and Fujimoto (1993) explored the idea of *probabilistic resynchronization*. All processes in the simulation are rolled back to close to GVT at probabilistically chosen intervals of real time. Special processes, called *genies*, broadcast special SYNC messages with timestamp slightly larger than the current GVT to induce such rollbacks. This scheme was shown to effectively eliminate over-optimistic behavior for a suitable choice of the resynchronization interval.

Steinman (1993) proposed an extension to his breathing time bucket algorithm by allowing it to take risks. The resulting protocol is called *breathing Time Warp* and is essentially a mixture of Time Warp and breathing time bucket. The idea is to release the events generated by the first N (say) events beyond GVT in each cycle. This is similar to Time Warp, as these messages may need to be canceled later. The protocol switches back to the risk-free breathing time

bucket from the $N + 1$ event in the cycle.

In Composite ELSA protocol (Arvind and Smart 1992) a node can switch between conservative and optimistic modes by using additional information regarding whether an event is safe. Both local and cascaded rollbacks are allowed. Rajaei, Ayani, and Thorelli (1993) explored a *local Time Warp* approach, where logical processes use Time Warp within clusters and use a synchronous, time window based conservative protocol across clusters. This hierarchical, cluster-based approach limits the progress of erroneous computations within a cluster. All inter-cluster messages are correct messages.

Several protocols have also been suggested to stop the spread of incorrect computation as soon as possible. Examples include "Wolf calls" (Madisetti, Walrand, and Messerschmitt 1988) and "Filter" (Prakash and Subramanian 1991). They broadcast or multicast special correction messages as soon there is a rollback in any LP. These correction messages stop the spread of incorrect computation in remote processes quicker than regular antimessages as they are sent directly instead of an indirect, recursive fashion as antimessages. However, these techniques are reactive, rather than proactive as action is taken only after the primary rollback (an observation made by Srinivasan and Reynolds (1996)).

4 ADAPTIVE PROTOCOLS

Many of the hybrid protocols can be adjusted in the continuum between the conservative and optimistic extremes. For example, the width of the time window in the MTW protocol can be tightened to equal to the *lag* and MTW will behave as the conservative bounded lag protocol. On the other hand if the width is infinite, MTW is equivalent to Time Warp. Similarly, the resynchronization interval in MIMDIX or the value of N in breathing Time Warp can be controlled to impart some form of extreme nature to the protocol. Not unexpectedly, experimental results indicate that overall performance of the protocol is sensitive to the choice of such control parameters. Best performance usually is obtained by setting such parameters somewhere between its extremes.

However, it is difficult for a simulation modeler, who is not intimately familiar with PDES protocols and the underlying parallel architecture, to set these parameters appropriately for an optimal performance. Furthermore, many simulation models are very dynamic in their runtime characteristics. The synchronization mechanism must adapt itself to a change in the model characteristics to optimize performance. This observation triggered a recent emergence of hybrid protocols that automatically "adapt" themselves in the continuum between purely conservative and purely optimistic strategies. The key idea is to monitor the state of the parallel simulation to estimate the appropriate trade-off between the conservatism (blocking or lost opportunity cost) and optimism (rollback and memory management costs) and accordingly adjust the control parameters.

The initial ideas about the usefulness of such adaptive protocols date back to the work by Reynolds, Jr. (1988), where he defined "adaptability" as the ability of logical processes to dynamically change one or more control parameters based on "knowledge of selected aspects of the state of the simulation." The objective is to minimize the execution time of the parallel simulation. The key issues are, of course, what control parameter(s) is (are) appropriate, and which aspects of the simulation state influence the binding of such control parameter(s) and how. Any adaptive protocol must take appropriate design decisions that address the above issues. As predicted by Reynolds, Jr. (1988), there are indeed quite a few very reasonable design choices. In the following we describe the adaptive protocols described in recent literature by broadly classifying them according to whether the adaptive decisions are taken based on purely the local state of the each LP or the global state of all LPs.

4.1 Protocols Based on Local State

One of the earliest attempts to develop adaptive protocols was made by Reiher, Wieland, and Jefferson (1989), where they proposed a *penalty based throttling* mechanism, where the logical processes that experienced rollbacks in the recent past are penalized. The penalty is represented by reduced scheduling quanta. This method was implemented in the JPL Time Warp Operating System (TWOS) but failed to produce any significant performance gain over Time Warp.

In the *Adaptive Time Warp* or ATW (Ball and Hoyt 1990) a logical process waits for an interval of real time (called *blocking window* or BW) between processing successive events. The BW is adjusted such that the sum total of the CPU time spent in blocked state and in faulted state (i.e., undoing incorrect computation) is minimized. The time spent in blocked state in directly proportional to the width the BW. The time spent in faulted state is modeled by a logistic response function. A numerically efficient approximate method is used to compute the minima. Ferscha and Lüthi (1995) proposed a similar, but more sophisticated method based on *probabilistic cost expectation function* or PCEF. An explicit cost model involving the probability and cost of rollback is used instead of a logistic response function. The probability of rollback is assumed to be dependent on

both real time and simulation time difference between the next scheduled event and the last message arrival. In order to compute this probability each LP monitors the timestamp and real time instants of arrival of every message as well as the rollback behavior. The cost model is used to evaluate the optimal real time blocking window. This method was shown to outperform Time Warp in experiments involving simulation of stochastic petri nets. Similar cost model has also been proposed for evaluating an optimal simulation time window rather than a real time blocking window (Das 1996). Here, simulation time distance from GVT is used as a parameter for evaluating the probability of rollback. This, however, depends on global state as GVT is needed.

In the conservative-optimistic or CO-OP protocol, (Steinman 1992b) LP scheduling on a processor uses some knowledge of the probability of the next event being correct. He also suggested a local CO-OP protocol where conditions on the input channels of the LPs control a variable that influence the LP scheduling decisions on a processor. (Ferscha and Chiola 1994) proposed a *probabilistic distributed simulation protocol* where Time Warp delays event computation artificially event computations based on the probability of an external message arrival inducing a rollback. Hamnes and Tripathi (1994b) proposed a *local adaptive protocol* or LAP, which uses input channel specific information for each LP to determine a real time blocking window. An LP blocks if it estimates that an event arriving later in one of the empty input channels will cause a rollback. This estimate is based on average inter-arrival times (both in simulation time and real time) of messages in each input channel in recent past. Null messages are used to prevent deadlock and to preempt unnecessary blocking. LAP was shown to outperform both conservative and optimistic methods in a performance study with closed queuing systems simulations (Hamnes and Tripathi 1994a).

Ferscha (1995b) presented a *probabilistic adaptive direct optimism control* or PADOC protocol, which is similar in spirit, but adds a probabilistic component and computes the blocking window incrementally. Each logical process maintains a history record of the simulation time differences of the arriving messages in the recent past. This record is used to forecast the timestamps of forthcoming messages. Several forecasting methods have been explored, such as methods based on central tendencies as well as autoregressive moving average (ARMA) models. Other models, such as neural networks, are also deemed possible. In addition to the forecast about the timestamp of the next arriving message, a confidence (ξ) in the forecast is also computed. The LP is blocked for a small predetermined period (same as the average event granularity) with probability $P(\xi)$ and advances to the next event in the event list with probability $1 - P(\xi)$. Then a new forecast is computed (in case there are new message arrivals in between) and this cycle continues. $P(\xi)$ was assumed to be a sigmoid function dependent on ξ as well as the difference between the local simulation time and the timestamp forecast for the next arriving message. PADOC was shown to outperform Time Warp for stochastic petri net simulations, especially under load imbalance.

Palaniswamy and Wilsey (1993) exploited an *adaptive bounded time window* approach where width of the time window (similar to MTW) is controlled dynamically, based on the concept of *useful work* done by an LP. Useful work is a measure of the productive work done by the LP and is a function of a number of parameters such as the ratio of the number of events committed to the number of events executed, number of rollbacks, average rollback length, number of antimessages sent etc. Each LP has its own value of window which is increased or decreased periodically depending on the change in useful work. Experiments with digital logic simulations demonstrated superiority of this method over ordinary Time Warp.

4.2 Protocols Based on Global State

Some protocols rely primarily on some aspects of the global state of the simulation for making adaptive decisions. Important examples are the *Adaptive Memory Management* protocol by Das and Fujimoto (1994) and the *Near Perfect State Information* (NPSI) protocols by Srinivasan and Reynolds (1995). In the adaptive memory management protocol an indirect approach is used for adaptation. It has been observed that over-optimistic Time Warp not only incurs high rollback costs, but also high memory management costs, because of loss of locality and possibly other virtual memory management overheads. Also, artificial memory limitation automatically restricts Time Warp's optimism. Thus the total amount of memory used by the simulation is used as the control parameter, by exploiting Time Warp's ability to continue simulation using any amount of memory larger than a lower bound. The cancelback protocol (Jefferson 1990) is used to restrict Time Warp within the allocated amount of memory. This scheme reduces both rollback and memory related costs and attempts to run Time Warp with just the sufficient amount of memory required for the best overall performance. Several monitored information about the rollback behavior, frequency of fossil collection and cancelback is used to decide the right amount of memory to be allocated.

NPSI protocols are a class of adaptive protocols relying on the availability of "near-perfect" information on the global state of the simulation. NPSI protocols use a quantity called *error potential* EP_i associated with each LP_i. The value of EP_i is used to control LP_i's optimism by, for example, introducing artificial blocking. Note that an inherent assumption in NPSI protocols is that the NPSI is available with minimal cost. This limits the use of such protocols to shared memory multiprocessors or distributed memory systems with parallel reduction network support. The usefulness of the latter for PDES has been demonstrated in (Reynolds, Jr., Pancerella, and Srinivasan 1993).

Different NPSI protocols can be envisaged for different choices for EP and different mechanisms to control optimism. A specific NPSI protocol called the *Elastic Time Algorithm* or ETA is described in (Srinivasan and Reynolds 1995). In ETA, the difference between the next event time of LP_i and the GVT is used as EP_i. More recently, the definition of EP_i is refined to replace GVT by the minimum of the next event time of all LPs that can send events to LP_i and timestamps of any messages in transit that are destined to LP_i (Srinivasan and Reynolds 1996). Optimism is controlled by blocking for an amount of real time proportional to EP_i between processing of successive events. The proportionality constant s (called the *scale factor*) requires some tuning for the optimal performance, which can also be adapted (Srinivasan and Reynolds 1996).

5 DYNAMIC LOAD BALANCING AND SCHEDULING

It is interesting to note how dynamic load balancing can interact with the synchronization mechanism. In particular, optimistic mechanisms introduce "a new wrinkle" to dynamic load balancing: high processor utilization may not imply good performance as processor may be busy incorrect computation that will be undone later (Nicol and Fujimoto 1994). Several load balancing algorithms have been proposed for Time Warp. (Reiher and Jefferson 1990) used a metric called *effective processor utilization* which is defined as the fraction of time a processor is executing correct computations (which are not later rolled back). Based on this metric, processes are migrated from processors with high effective utilization to processors with low utilization. Glazer and Tropper (1993) proposed an allocating *virtual time slices* to LPs based on their observed rate of progress in simulation time.

It is conceivable that a load distribution strategy that targets equalization of simulation clocks thus imparting some form of temporal (in the sense of simulation time) locality in the simulation system will potentially reduce rollbacks (Jefferson 1985). One (extreme) way of doing it is to migrate LPs to processors in such a way that the n (n being the number of processors) slowest (in simulation time) LPs in the system are all assigned to different processors (Ahmed, Rönngren, and Ayani 1994). If the above condition is strict, then there is a possibility of LP migration after processing each simulation event. Such strict condition on LP scheduling improves temporal locality at the expense of spatial locality, as different events on an LP is processed at different processors. Also, use of a centralized scheduling queue may introduce bottlenecks. A milder technique was explored by Burdorf and Marti (1993) that may improve spatial locality. It is expected for most systems a tradeoff between temporal and spatial locality will yield the best performance. However, in our knowledge no such tradeoff has been evaluated for PDES systems.

6 CONCLUSIONS

Even though the adaptive protocols differ in many details, they all (i) use some aspect of the simulation state, local or global, for making adaptive decisions, (ii) use one or more control parameters to tune the dynamics of the simulation, and (iii) provide a mapping that specifies the binding of the control parameter depending on the state. Past rollback behavior or past message arrival pattern is frequently used as the state information if the protocol relies on local state alone. Availability of global state information (as in NPSI protocols) yields more complete knowledge about the dynamics of the simulation as the relative progress of all LPs or overall memory usage can be known. The control parameters commonly used are scheduling quanta, real time blocking window, simulation time window, or amount of allocated memory. The sole use of the control parameters is to selectively reduce the simulation time progress of some LPs per unit real time by starving it of processor or memory resources. The mapping between the state information and the control parameter is often linear, but may be non-linear (as in the PADOC protocol). Some protocols (such as the ETA) even use tuning of the parameters of the mapping function.

One view of any adaptive protocol is that of a feedback based control system (Palaniswamy and Wilsey 1994), and there is an inherent possibility of instability in such systems, as the feedback is always applied with a lag in real time. Instability is very probable if the characteristics of the simulation model (e.g., model parallelism) changes too fast thus not giving the protocol sufficient time to react to the changes. Study of stability properties is impor-

tant because of the dynamic nature of many simulation models. With the initial success in developing performance-efficient adaptive PDES protocols, we feel that this is one area that needs to be explored in future.

REFERENCES

Ahmed, H., R. Rönngren, and R. Ayani 1994. Impact of event scheduling on performance of Time Warp parallel simulations. In *Proceedings of the 27th Annual Hawaii Intl. Conf. on System Sciences.*

Arvind, K. and C. Smart 1992. Hierarchical parallel discrete event simulation in composite ELSA. In 6^{th} *Workshop on Parallel and Distributed Simulation*, pp. 147–158.

Ball, D. and S. Hoyt 1990. The adaptive Time-Warp concurrency control algorithm. *Proceedings of the SCS Multiconference on Distributed Simulation 22*(1), 174–177.

Burdorf, C. and J. Marti 1993. Load balancing strategies for Time Warp on multi-user workstaions. *The Computer Journal 36*(2), 168–176.

Chandy, K. M. and J. Misra 1979. Distributed simulation: A case study in design and verification of distributed programs. *IEEE Transactions on Software Engineering SE-5*(5), 440–452.

Das, S. R. 1996. Estimating the cost of throttled execution in Time Warp. In *Proceedings of the 10th Workshop on Parallel and Distributed Simulation*, pp. 186–189.

Das, S. R. and R. M. Fujimoto 1993. A performance study of the cancelback protocol for Time Warp. *Proceedings of the 7th Workshop on Parallel and Distributed Simulation 23*(1), 135–142.

Das, S. R. and R. M. Fujimoto 1994. An adaptive memory management protocol for Time Warp parallel simulation. In *Proceedings of the 1994 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, pp. 201–210.

Dickens, P. M. and P. F. Reynolds, Jr. 1990. SRADS with local rollback. *Proceedings of the SCS Multiconference on Distributed Simulation 22*(1), 161–164.

Ferscha, A. 1995a. Parallel and distributed simuation of discrete event systems. In A. Y. H. Zomaya (Ed.), *Parallel and Distributed Computing Handbook*, Chapter 35. McGraw-Hill.

Ferscha, A. 1995b. Probabilistic adaptive direct optimism control in Time Warp. In *Proceedings of the 9th Workshop on Parallel and Distributed Simulation*, pp. 120–129.

Ferscha, A. and G. Chiola 1994. Self adaptive logical processes: The probabilistic distributed simulation protocol. In *Proceedings of the 27th Annual Simulation Symposium*, pp. 78–88.

Ferscha, A. and J. Lüthi 1995. Estimating rollback overhead for optimism control in Time Warp. In *Proceedings of the 28th Annual Simulation Symposium.*

Fleischmann, J. and P. A. Wilsey 1995. Comparative analysis of periodic state saving techniques in Time Warp simulators. In *Proceedings of the 9th Workshop on Parallel and Distributed Simulation*, pp. 50–58.

Fujimoto, R. M. 1989. Performance measurements of distributed simulation strategies. *Transactions of the Society for Computer Simulation 6*(2), 89–132.

Fujimoto, R. M. 1990. Parallel discrete event simulation. *Communications of the ACM 33*(10), 30–53.

Fujimoto, R. M. 1993a. Parallel and distributed simulation: Algorithms and applications. In *1993 Winter Simulation Conference Proceedings*, pp. 106–114.

Fujimoto, R. M. 1993b. Parallel discrete event simulation: Will the field survive? *ORSA Journal of Computing 5*(3), 213–230.

Glazer, D. W. and C. Tropper 1993. On process migration and load balancing in Time Warp. *IEEE Transactions on Parallel and Distributed Systems 3*(4), 318–327.

Hamnes, D. O. and A. Tripathi 1994a. Evaluation of a local adaptive protocol for distributed discrete event simulation. In *Proceedings of the 1994 International Conference on Parallel Processing*, pp. III:127–134.

Hamnes, D. O. and A. Tripathi 1994b. Investigations in adaptive distributed simulation. In *Proceedings of the 8th Workshop on Parallel and Distributed Simulation*, pp. 20–23.

Jefferson, D. R. 1985. Virtual time. *ACM Transactions on Programming Languages and Systems 7*(3), 404–425.

Jefferson, D. R. 1990. Virtual time II: The cancelback protocol for storage management in distributed simulation. In *Proc. 9th Annual ACM Symposium on Principles of Distributed Computing*, pp. 75–90.

Lin, Y.-B. and E. D. Lazowska 1990. Optimality considerations of "Time Warp" parallel simulation. *Proceedings of the SCS Multiconference on Distributed Simulation 22*(1), 29–34.

Lipton, R. J. and D. W. Mizell 1990. Time Warp vs. Chandy-Misra: A worst-case comparison. *Proceedings of the SCS Multiconference on Dis-*

tributed Simulation 22(1), 137–143.

Lubachevsky, B. D. 1989. Efficient distributed event-driven simulations of multiple-loop networks. *Communications of the ACM 32*(1), 111–123.

Lubachevsky, B. D., A. Shwartz, and A. Weiss 1989. Rollback sometimes works ... if filtered. In *1989 Winter Simulation Conference Proceedings*, pp. 630–639.

Lubachevsky, B. D., A. Shwartz, and A. Weiss 1991. An analysis of rollback-based simulation. *ACM Transaction on Modeling and Computer Simulation 1*(2), 154–193.

Madisetti, V., J. Walrand, and D. Messerschmitt 1988. Wolf: A rollback algorithm for optimistic distributed simulation systems. In *1988 Winter Simulation Conference Proceedings*, pp. 296–305.

Madisetti, V. K., D. A. Hardaker, and R. M. Fujimoto 1993. The MIMDIX operating system for parallel simulation and supercomputing. *Journal of Parallel and Distributed Computing 18*(4), 473–483.

Mehl, H. 1991. Speedup of conservative distributed discrete-event simulation methods by speculative computing. *Proceedings of the Multiconference on Advances in Parallel and Distributed Simulation 23*(1), 163–166.

Nicol, D. M. 1993. The cost of conservative synchronization in parallel discrete event simulations. *Journal of ACM 40*(2), 304–333.

Nicol, D. M. and R. M. Fujimoto 1994. Parallel simulation today. *Annals of Operations Research 53*, 249–286.

Palaniswamy, A. C. and P. A. Wilsey 1993. Adaptive bounded time windows in an optimistically synchronized simulator. In *Great Lakes VLSI Conference*, pp. 114–118.

Palaniswamy, A. C. and P. A. Wilsey 1994. Scheduling time warp processes using adaptive control techniques. In *Proceedings of the 1994 Winter Simulation Conference*, pp. 731–738.

Prakash, A. and R. Subramanian 1991. Optimistic distributed simulations. In *Proceedings of the 24th Annunal Simulation Symposium*, pp. 123–132.

Rajaei, H., R. Ayani, and L.-E. Thorelli 1993. The local Time Warp approach to parallel simulation. In *Proceedings of the 7^{th} Workshop on Parallel and Distributed Simulation*, pp. 119–126.

Reiher, P. L. and D. Jefferson 1990. Dynamic load management in the Time Warp Operating System. *Transactions of the Society for Computer Simulation 7*(2), 91–120.

Reiher, P. L., F. Wieland, and D. R. Jefferson 1989. Limitation of optimism in the Time Warp Operating System. In *Proceedings of the 1989 Winter Simulation Conference*, pp. 765–770.

Reynolds, Jr., P. F. 1988. A spectrum of options for parallel simulation. In *1988 Winter Simulation Conference Proceedings*, pp. 325–332.

Reynolds, Jr., P. F., C. M. Pancerella, and S. Srinivasan 1993. Design and performance analysis of hardware support for parallel simulation. *Journal of Parallel and Distributed Computing 18*(4), 435–453.

Sokol, L. M., D. P. Briscoe, and A. P. Wieland 1988. MTW: a strategy for scheduling discrete simulation events for concurrent execution. *Proceedings of the SCS Multiconference on Distributed Simulation 19*(3), 34–42.

Srinivasan, S. and P. F. Reynolds 1995. NPSI adaptive synchronization algorithms for PDES. In *1995 Winter Simulation Proceedings*, pp. 658–665.

Srinivasan, S. and P. F. Reynolds 1996. Elastic time. *ACM Transactions on Modeling and Computer Simulation*. Submitted.

Steinman, J. 1992a. SPEEDES: A multiple-synchronization environment for parallel discrete event simulation. *International Journal of Computer Simulation 2*(3), 251–286.

Steinman, J. 1992b. SPEEDES: A unified approach to parallel simulation. In *6th Workshop on Parallel and Distributed Simulation*, pp. 75–84.

Steinman, J. S. 1993. Breathing Time Warp. In *Proceedings of the 7th Workshop on Parallel and Distributed Simulation*, pp. 109–118.

Turner, S. J. and M. Q. Xu 1992. Performance evaluation of the bounded Time Warp algorithm. *Proceedings of the SCS Multiconference on Parallel and Distributed Simulation 24*(3), 117–126.

AUTHOR BIOGRAPHY

SAMIR R. DAS is an assistant professor in the Division of Computer Science at The University of Texas at San Antonio. He received his ME in computer science and engineering from the Indian Institute of Science, Bangalore, in 1988 and PhD in computer science from the Georgia Institute of Technology, Atlanta, in 1994. His research interests include parallel and distributed systems, performance evaluation and discrete event simulation. In the recent past he worked on performance aspects of Time Warp based parallel simulation systems.