Distributed Event-driven Simulation- Scheduling Strategies and Resource Management

Overeinder, B.J.

Publication date
2000

Chapter 8

Summary and Conclusions

The White Rabbit put on his spectacles. "Where shall I begin, please your Majesty?" he asked. "Begin at the beginning," the King said gravely, "and go on till you come to the end: then stop."

—Lewis Carroll

The increasing understanding of the complexity of the world around us, both in natural sciences and computer science, necessitates realistic models reflecting the complexity of systems under study. The realistic models, and consequently more complex and larger simulations, require vast amounts of execution time. We are particular interested in real-world and theoretical systems that are characterized by heterogeneous spatial and temporal behavior. Systems with this behavior are very common in, for example, population dynamics, immunology, statistical physics, and in computer science. The heterogeneous spatial and temporal behavior is most exactly mapped to asynchronous models.

Experimentation with complex simulations and interpretation of vast amounts of generated data is a challenging endeavor. In this respect, virtual environments that integrate the visualization of data and the interaction with the simulation are becoming more and more important. Consider for example a virtual laboratory where scientists interact with a simulation in a virtual environment (such as a CAVE Automatic Virtual Environment). As the scientist interactively makes changes to the model or to the parameters of the model, the running simulation must be stopped or even rolled back to a previous state, such that the changes can be sustained. Furthermore, support for combined continuous simulation and discrete event simulation is necessary, as applications used in virtual environments can be of both types.

The combination of complex simulations, asynchronous execution mechanisms, and virtual environments, cumulates in large applications that require sufficient computational power to allow for interactive experimentation with the simulation model. Such high performance computing demands cannot be easily fulfilled by a single supercomputer (also not by the distributed nature of the virtual laboratory application). Distributed and parallel computing techniques are a viable solution to the virtual laboratory application, where the aggregated computing resources are bundled to provide a high performance computing platform.

In our endeavor to distributed and parallel computing, we exploit the locality of data and processing by event scheduling methods from parallel discrete event simulation. The overall balance of workload over the distributed and parallel processing nodes is accomplished by the Dynamite dynamic process migration environment. The combined parallel scheduling of simulation events and the load balance of computational work over the processing nodes is a very complex task. In this thesis, we have approached the two (sub-) problems independently, but generically as both components must be integrated in one environment.

The design and implementation of a simulation environment for parallel discrete event simulation resulted in the APSIS system. The core of APSIS is the Time Warp optimistic simulation kernel. Central issues in the design were the efficient support for large, data-intensive scientific simulations with dynamic behavior. Data-intensive simulations put efficient memory management requirements to the Time Warp simulation kernel. The Time Warp method must regularly checkpoint the simulation state in order to recover from erroneous, optimistic processing of simulation events. We proposed, implemented, and validated an incremental state saving mechanism that *only* saves the changes to the state, instead of the complete state vector. The dynamic behavior of the system asks for the scheduling *and* retraction of simulation events. Event retraction is originally not included in the Time Warp method, but *is* included in the APSIS environment.

The APSIS environment is complemented with the APSE parallelism analysis methodology. The effectiveness and dynamic behavior of the Time Warp method is extensively evaluated using a prototypical application from statistical physics, namely the Ising spin system. The APSE parallelism analysis and the performance experiments with the Ising spin simulation show that the APSIS environment can efficiently schedule the events over the parallel processors.

The experiments with the Ising spin simulation showed also the need for optimism control. In general, Time Warp is quite robust, i.e., performs fairly well without specific adaptations to the simulation application. However, unexpected long turnaround times are observed near the phase transition (or critical phase) in the Ising spin system due to an increase in length and frequency of cascaded rollbacks. The length and frequency of cascaded rollbacks can be bounded by limiting the optimism in the Time Warp method, which is shown by the experiments. The non-trivial interference between application and Time Warp method is conjectured to show self-organized critical behavior. Here the computational complexity of the Time Warp method and the physical complexity of the application are entangled and contribute both to turnaround time and rollback behavior in a non-linear way.

The Dynamite environment incorporates our ideas on dynamic load balancing of computational work over the distributed and parallel processing nodes. We have realized a runtime support system for dynamic load balancing of parallel programs by supporting task migration in the PVM system. To allow for task migration of parallel running tasks, a number of issues had to be taken care of: consistent checkpointing, a migration protocol, and packet routing. To implement dynamic load balancing by task migration, the runtime support system must be able to create an image of the running process, the so-called checkpoint. A complicating factor with checkpointing communicating PVM tasks, is that the state of the process also includes the communication status. To prohibit the creation of process checkpoints during communication, we apply the notion of critical sections and embed all interprocess communication operations in such sections. A transparent migration protocol allows for the movement of tasks without affecting the operation of other tasks in the system. To provide transparent and correct message routing with migrating tasks, the task identifiers must be made location independent. This imposes additional requirements on the runtime support system in order to route the messages to their correct destination.

The dynamic load balancing facilities of the Dynamite environment are validated on a cluster of workstations. The ability of the Dynamite environment to adapt to dynamically changing environments is assessed by a number of NAS Parallel Benchmark kernels and two large finite element simulation runs (typically time-driven simulations). One of the finite element simulation models is GRAIL: a large antenna for the detection of gravitational waves. Other finite element simulation experiments are realized with PAMCRASH, a finite element package to evaluate safety issues in car crashes. The results from the experiments showed the potential of Dynamite to dynamically balance the computational load over the parallel or distributed processors.

Future research directions for the APSIS environment include further development of efficient memory management strategies, such as hybrid state saving where copy state saving and incremental state saving techniques are combined and adaptively engaged according to the behavior of the application. Another important feature that is prominent on our wish list is adaptive optimism control. The strategy to approach adaptive optimism control is still an open research question. The difference in predictive quality of computational intensive and complex statistical methods, and fast and simple statistical methods to control the optimism in Time Warp, does not give a marked off advantage to complex methods. Also the influence of self-organized critical behavior in optimistic simulation has to be studied in relation to optimism control, and might lead to new insights.

One of the future developments in Dynamite is the support of task migration for MPI. This work will be a collaboration with Mississippi State University and incorporates their work on Hector. Furthermore, global resource management strategy research will be conducted in context with the Polder metacomputer.

Finally, and maybe in the context of this thesis most important, the Dynamite environment will be used as a runtime support system for the APSIS simulation environment. The main research issue for Dynamite with APSIS is the study of load balancing strategies for optimistic parallel simulations. In optimistic parallel simulation there is a difference between computational work or load, and the notion of progress. For example, event rollback is computational work but there is no progress of the simulation. The essential question is to find a compact set of system and Time Warp parameters, and an effective strategy to balance the parallel simulation.