High performance N-body simulation on computational grids

Groen, D.J.

Publication date
2010

7 Conclusions

7.1 Thesis summary

In this thesis we examine the feasibility of running N-body simulations on globally distributed computers. We have run simulations of star clusters, galaxies and cosmological volumes on a globally distributed infrastructure and analyzed their performance in these wide area environments. Furthermore, we have developed time complexity models for direct and Tree/Particle-Mesh N-body simulations and applied them to predict the performance of N-body simulations across a large number of computational sites.

Our first experiments, which are described in Chapter 2, have been performed to determine whether direct-method N-body simulations can be efficiently run across a global grid. We present a time complexity model that can be applied to determine the execution time of direct N-body simulations on the grid. The model supports predictions for simulations that use either the copy or the ring communication scheme. We have run our direct-method simulations on a testbed consisting of three sites with special GRAPE hardware and measured the performance of our experiments. The overall run-time is dominated by the communication overhead in all our experiments. We find that the network response time is the primary performance bottleneck for runs over three sites that simulate fewer than 10^4 particles. When running larger simulations, the communication overhead becomes more dependent on the throughput rate of the network, which is limited in these runs due to the use of regular internet. Based on the timing results of our experiments and predictions of our performance model, we conclude that direct-method simulations are not suitable for execution on a global grid of GRAPEs, but that a national grid of GRAPEs or GPUs can efficiently be used to run N-body simulations consisting of a few million particles.

In Chapter 3 we map N-body simulations to a computational grid by adopting a dynamic code that switches between N-body integration methods and computational sites at run-time. This switching mechanism allows the application to use the resources

best suited for each integration method it applies. We implement this concept as a *l*iving simulation that switches between direct-method N-body integration on GRAPE hardware in the United States and tree code N-body integration using GPU hardware in The Netherlands. The switching between sites is not initiated by an external manager, but rather done by the application at run-time. Here, the simulation obtains grid credentials, transfers files and submits a clone of itself to the new location. We have applied the living simulation to simulate a merger between two galaxies, using up to 65538 particles. The overhead of switching between the two integrators is less than 5 percent of the total execution time for most runs.

During our initial experiments we found that simulations run more efficiently in wide area environments if they contain more particles. However, a direct N-body simulation with more than a few million particles is computationally difficult because the execution time of direct-method integrations scales with $O(N^2)$. Tree/Particle-Mesh (TreePM) simulations are used to model the structure formation of dark matter in the Universe over time and have a calculation complexity of $O(N \log N)$. Due to the lower complexity, simulations with a large number of particles require less computing time when a TreePM method is applied. By using a TreePM code it becomes possible to simulate billions of particles on a single supercomputer.

In Chapter 4 we report on our experiences in running a TreePM simulation on a globally distributed supercomputer. We have combined the GreeM cosmological N-body integrator with a custom communication library to allow parallel simulations across two sites. We describe the setup and results of experiments that concurrently use an IBM Power6 machine in Amsterdam and a Cray-XT4 machine in Tokyo. Both sites were interconnected with a 10 Gbps optical network. Our simulations have achieved a calculation efficiency of $\sim 90\%$ for production-sized problems and we predict that running the simulation across more than ten supercomputers would still provide satisfactory performance.

In Chapter 5 we analyze the performance of N-body integrators in general, and TreePM codes in particular, across multiple supercomputers. We have enhanced the cosmological code described in Chapter 4 to allow runs across any number of supercomputers. The code, named SUSHI, connects the sites in a ring topology and divides the simulation volume into one slice for each site. The workload on each site is adjusted at run-time to maintain an equal calculation time on all sites. We have tested SUSHI in parallel using up to 5 Beowulf clusters in a national grid, and up to 4 supercomputers in a global grid. Our simulations achieve an efficiency of 87% across three supercomputers when using 1024^3 particles, and an efficiency of 73% across four supercomputers when using 512^3 particles. We have developed a time complexity model for SUSHI and applied it to predict the performance of our code across a large number of supercomputers. Based on our predictions, we expect that the SUSHI scales well up to ~ 16 supercomputer sites for a simulation with 2048^3 particles and 256^3 mesh cells. To provide a comparison we also model the performance of tree and direct N-body codes across multiple supercomputers. We predict that these codes do not run efficiently across multiple sites when a block time step scheme is used. We conclude that using a widely distributed supercomputer to acquire more computing power is technically

feasible, and that the local scheduling and reservation policies are the primary obstacles for running long-lasting production simulations across supercomputers.

In Chapter 6 we present the MPWide communication library, which we have developed to perform message passing over long-distance optical networks. MPWide is a light-weight library that has few dependencies and can be quickly installed on different supercomputer platforms. It can be used to combine several local MPI applications into an application that runs across multiple sites. Connections established with MPWide can be customized individually without modifying the settings of the underlying system. We have tested MPWide on a local network, between two sites in a national grid, and between two supercomputers. The supercomputers were respectively located in Amsterdam (The Netherlands) and Helsinki (Finland). We achieved a sustained bandwidth of up to 4.8 Gbps on the 10 Gbps shared network between Amsterdam and Helsinki. In addition, we applied the library to run cosmological simulations across supercomputers.

7.2 Conclusions and Recommendations

In the introduction we asked whether it would be possible to map high performance N-body simulations to infrastructures that span up to several thousands of kilometers, and how we could make efficient use of such widely distributed resources. In this section we first review the performance characteristics of N-body simulations across a globally distributed computer, and then comment on the challenges faced when creating and using a planet-wide distributed supercomputer. We conclude with a discussion on future perspectives and applications of globally distributed (super)computing.

7.2.1 Simulations on a planet-wide distributed supercomputer

We have mapped star cluster, galaxy and cosmological simulations to a globally distributed infrastructure, combining computational resources from multiple sites to perform one simulation. By interconnecting computational sites to form a distributed system, we are able to run a simulation using multiple architectures and obtain more computing power than we could otherwise get from a single site.

One class of programs that efficiently uses global infrastructures consists of living N-body simulations, which dynamically combine multiple N-body solvers to tackle a complex N-body problem. Living simulations are used on heterogeneous grids, where each solver is best run on a different computational site. The program is then able to detect when a different solver is required to advance the simulation, dynamically switch to that solver and migrate to an architecture best suited to run it. A living simulation does not require the intervention of an external manager to switch between sites, and can be used to run hybridized simulations on grids without occupying any excess nodes. We have measured a switching overhead of only a few percent of the total runtime for our hybrid galaxy merger simulation, which has been presented in Chapter 3.

Another way to use the additional resources provided by a global computer is to run one simulation distributed over multiple sites. Such runs are able to use more

compute power, if the communication overhead introduced by wide area networks does not dominate the overall run-time. The overhead introduced by wide area networks is considerably larger than that introduced by local area networks, because the round-trip time of one message is much higher. For example, a light path between Amsterdam and Tokyo has a round-trip time of about 0.27 s, at least three orders of magnitude higher than that of a local supercomputer network. In addition, the bandwidth capacity of light paths is about one order of magnitude smaller than that of local supercomputer networks, and ranges between 1 and 10 Gbps at the time of writing. Due to the inferior performance characteristics of long distance networks, simulations run over a global grid are less efficient than if they were run on a local site, except in rare cases where local networks are prone to congestion [13].

The efficiency of N-body simulations run on a globally distributed machine can be improved in part by adjusting the parameters of the simulation. Many of the adjustments that reduce the relative communication overhead on a single site, also improve the efficiency of simulations run over a planet-wide distributed supercomputer. Examples of these adjustments are:

- Increasing the number of particles in the simulation.
- Decreasing the opening angle θ (if using tree integration).
- Distributing the particle load such that all processes spend an equal amount of time on calculations.

However, a simulation optimized for parallel execution on a single site does not necessarily run efficiently on a global grid. The increased response time of wide area networks cause each communication call to spend additional time. This especially reduces the efficiency of simulations that require a large number of communications with small data volumes. An example of such a simulation is the integration of a star cluster using a direct method (see Chapter 2). The round-trip time of wide area networks is constrained by the speed of light, and can only be reduced to some extent by improving the network. For example, a connection between Amsterdam and Tokyo, which are 9300 kilometers apart, will always have a round-trip time of at least 0.062 s.

The lower bandwidth capacity of wide area networks mainly increases the communication overhead of simulations that exchange large buffers of data, and limits the scalability of such runs. However, the difference in bandwidth capacity between local and wide area networks is sufficiently small to allow efficient execution of production simulations across a global infrastructure consisting of multiple supercomputers (see Chapters 4-6). Also, the bandwidth of wide area networks can be increased by optimizing the network configuration, or using multiple network paths concurrently.

We have run cosmological simulations across up to 4 supercomputers, and predict that simulations can be performed efficiently across up to ~ 16 sites in a global grid if a wide area throughput rate of 400 MB/s is obtained.

7.2.2 Recommendations for planet-wide N-body simulations

In this work we have used a wide range of N-body applications, planet-wide computational infrastructures, network interconnections and middleware. The optimal choice of interconnections and middleware is crucial for the performance and deployability of a planet-wide simulation, and depends strongly on the chosen computational resources and N-body application. Here we provide recommendations for four N-body applications, three of which run in parallel across multiple sites. These are direct-method N-body integration, tree integration and tree/particle-mesh integration. The fourth application is a hybrid N-body simulation which runs on a single site at a given time, but migrates between sites and can switch to a different integration method at runtime (as described in Chapter 3). The N-body applications are linked to three types of planet-wide infrastructures, including a network of supercomputers, a network of Beowulf PC clusters and a network of sites equipped with specialized hardware. We provide a list of recommended network interconnections for these applications and resources in Table 7.1.

Supercomputers are best connected using lightpaths, as a connection to the internet provides sub-optimal performance and is considered undesirable by many supercomputer centers for security reasons. The use of a shared light path is sufficient in most cases, though the large amount of data exchanged in large TreePM simulations justifies the use of a dedicated path. Simulations run over a planet-wide network of Beowulf clusters or specialized hardware sites are more limited in size, and the security policies of these sites tends to be less strict. The communication performance of direct and hybrid simulations is often not bound by network bandwidth, but rather by network latency. Since the network latency of light paths is almost identical to that of regular internet connections, it is more convenient to use the publicly available internet.

We provide a list of recommended middleware for the aforementioned applications and resources in Table 7.2. We recommend MPWide or an other customized socket library when running simulations across sites connected by light paths, as these connections require manual tuning of each communication channel to achieve optimal performance. Since the throughput rate is of lesser importance for direct N-body simulations, the ease of deployment of a cross-site MPI implementation often outweighs the performance benefit provided by MPWide. Using a cross-site MPI implementation is also an option for Tree or TreePM simulations across a network of Beowulf clusters, if the MPI implementation has been sufficiently optimized for this network.

7.2.3 Creating a planet-wide distributed supercomputer

The creation of a globally distributed computer requires the connection of computational sites over long distances. Connecting these sites is straightforward when the sites are accessible from regular internet and the user has administrative rights on each site. Administrative rights are easily obtained for small clusters of PCs, GPUs or GRAPEs and the bandwidth of regular internet may be sufficient for distributed runs using these smaller sites. However, the overall efficiency of a distributed simulation improves

Resource type	Application type			
	Direct	Tree	TreePM	Hybrid
supercomputers	shared path	shared path	ded. path	shared path
clusters of PCs	internet	shared path	shared path	internet
specialized sites	internet	shared path	shared path	internet

Table 7.1: List of recommended interconnections for four application and three types of computational resources. The recommended interconnections are given for direct N-body integration (second column), shared time-step tree N-body integration (third column), tree/particle-mesh N-body integration (fourth column) and hybrid N-body simulations which switch between multiple solvers (fifth column). The computational resources considered include a planet-wide network of supercomputers (second row), a network of Beowulf PC clusters (third row) and a network of clusters equipped with specialized hardware (e.g. GPUs or GRAPEs, fourth row). The recommended types of interconnection include regular internet connection (given by "internet"), shared light path (given by "shared path") and dedicated light paths (given by "ded. path").

Resource type	Application type			
	Direct	Tree	TreePM	Hybrid
supercomputers	MPW	MPW	MPW	LA + MPW
clusters of PCs	cMPI	cMPI or MPW	cMPI or MPW	LA
specialized sites	cMPI	MPW	MPW	LA

Table 7.2: List of recommended middleware for four application and three types of computational resources. An explanation of the rows and columns is given in Tab. 7.1. The recommended types of middleware include cross-site MPI implementations (given by "cMPI") such as MPICH-G2 [70] or OpenMPI [37], local-site vendor MPI combined with MPWide (given by "MPW") and Living Application middleware (given by "LA").

for larger problem sizes. To run distributed simulations more efficiently we therefore require sites with more computational capacity, such as supercomputers.

7.2.3.1 Configuration of a global supercomputer

Interconnecting larger computing sites to create a globally distributed supercomputer takes considerable effort. The communication performance of regular internet is insufficient to efficiently exchange large data volumes between sites, and optical paths with a capacity of at least 1 Gbps need to be arranged to connect the supercomputers.

A considerable part of the work performed for this thesis involved arranging and testing wide area networks. In the most extreme case it took over a year to arrange a connection between two supercomputers. This overhead for reserving temporary dedicated light paths over long distances does not necessarily decrease for later reservations of the same network. Arranging a shared and permanent optical network also takes a long time, but shared networks require little effort to use once established and

properly configured. A drawback of shared networking is the presence of background communication traffic, which causes the performance of data exchanges to become less predictable. Analyzing and predicting the communication performance of long-lasting runs can be made easier if the communications performed over shared optical paths by each user would be tracked by a public monitoring system.

Optimizing communication performance over long distance paths requires the cooperation of all organizations involved. For intercontinental paths that span across several organizations, configuring the network for optimal performance becomes hard to accomplish for administrators and virtually impossible for users. In part, the connection can be optimized in user space by using a communication library that allows for path-specific tuning (see Chapter 6). Centralizing the management of intercontinental paths reduces the need for user-space tuning, but may be difficult to accomplish politically.

7.2.3.2 Scheduling and running on a global supercomputer

A simulation that runs across multiple supercomputers not only requires the sites to be interconnected, but also requires its processes on all sites to be run simultaneously. If one site takes longer to start its part of the simulation than others, the application will waste CPU hours awaiting the availability of the missing site. As supercomputers are heavily used, they normally have a queue of jobs waiting to be executed and few or no idle nodes at a given time. This load is lower during nighttime and during the weekends, upon which test runs of limited scale become possible without prior reservation.

Long-lasting production runs are currently difficult to perform on a globally distributed supercomputer, as the scheduling and reservation policies are different for each site. For example, the Cray machine in Tokyo enforces a maximum runtime of 8 hours per job but allows users to chain jobs together in a way that follow-up jobs do not need to be requeued. In contrast, the Cray machine in Edinburgh enforces a maximum runtime of 12 hours, but when jobs are chained together on this machine the follow-up jobs are treated as newly submitted jobs and requeued accordingly. Any time spent in the queue by a job on one site causes jobs on other sites to wait, and therefore waste CPU hours. Scheduling long-lasting production runs across supercomputers is therefore quite inefficient unless all sites adopt a uniform scheduling policy that does not requeue follow-up jobs or a reservation system is put in place.

At the time of writing, many supercomputers do not yet have a system for advance reservation. This makes it difficult to reserve resources on multiple supercomputers for a distributed run. The most convenient way to reserve resources for distributed runs would be through the use of a system that centrally coordinates reservations on multiple sites (e.g., HARC [79], GridARS [125] or GRMS [73]). However, such a system may impose modifications on local reservation policies or require direct access to supercomputers, which can be a security risk. Alternatively, supercomputer centers can implement local reservation systems, allowing users to manually reserve nodes on each site for a given period. This increases the overhead for users to arrange a job across supercomputers compared to a centralized system, but eliminates the waste of CPU hours that is incurred when running across sites without reservations.

We have shown that a globally distributed supercomputer can be used to efficiently run large scale N-body simulations. The technology to run long-lasting production simulations across the planet is available, but some additional political and organizational effort is required to establish a planet-wide distributed supercomputer suitable for production runs.

7.2.4 Future perspectives and applications

Although the focus of this thesis is N-body simulations on planet-wide infrastructures, the results and conclusions presented in this work may also apply to other classes of applications. Using a globally distributed computer is especially attractive for research fields where new insights can be obtained by running parallel applications on an unprecedented scale (e.g., cosmology).

This thesis provides methods and examples to map applications to planet-wide infrastructures, and obtain a sustained boost in compute performance. The living simulation method presented in Chapter 3 and the MPWide communication library presented in Chapter 6 can be applied to allow the execution of other application classes on a planet-wide infrastructure. We have shown that N-body simulations can be run efficiently over planet-wide infrastructures, but mostly considered the ease-of-use, security and fault-tolerance aspects to be outside the scope of this work. These factors are of limited importance for the application performance, but are essential to consider when making a planet-wide distributed supercomputer suitable for production runs. The recent work performed in the fields of grid and cloud computing provides a good starting point to investigate these aspects of distributed supercomputer environments.

Using planet-wide infrastructures for high performance computing also introduces new parallelization challenges. The network topologies between supercomputers are sparse and have high response times. These factors should be taken into account when parallel algorithms are ported to a global computer environment. Good scalability can be achieved both within, and between supercomputer sites by adopting hierarchical parallelization schemes such as the one presented in Chapter 5. By developing hierarchical schemes for other application classes, and enabling them to run efficiently in a distributed environment, we will be able to execute a much wider range of scientific applications on powerful planet-wide distributed supercomputers.