Simulating N-Body Systems on the Grid Using Dedicated Hardware

Derek Groen[1,2], Simon Portegies Zwart[1,2], Steve McMillan[3], and Jun Makino[4]

[1] Section Computational Science, University of Amsterdam, Amsterdam, The Netherlands
djgroen@science.uva.nl
[2] Astronomical Institute "Anton Pannekoek", University of Amsterdam, Amsterdam, The Netherlands
[3] Drexel University, Philadelphia, United States
[4] National Astronomical Observatory, Mitaka, Japan

Abstract. We present performance measurements of direct gravitational N-body simulation on the grid, with and without specialized (GRAPE-6) hardware. Our intercontinental virtual organization consists of three sites, one in Tokyo, one in Philadelphia and one in Amsterdam. We run simulations with up to 196608 particles for a variety of topologies. In many cases, high performance simulations over the entire planet are dominated by network bandwidth rather than latency. Using a global grid of GRAPEs our calculation time remains dominated by communication over the entire range of N, which was limited due to the use of three sites. Increasing the number of particles will result in a more efficient execution and for $N \gtrsim 2 \cdot 10^6$ we expect the computation time to overtake the communication time. We compare our results with a performance model and find that our results are in accordance with the predicted values.

1 Introduction

The simulation of a star cluster is commonly performed by direct-method N-body integrators [1]. The gravitational force on individual stars in such simulations is calculated by aggregating the force contributions from all other particles in the system. This is a compute-intensive operation that requires the use of parallel algorithms or dedicated hardware (such as GRAPEs [2], or GPUs [3]) for simulating more than a few thousand stars. Several parallel algorithms have been developed for N-body simulations, including a copy algorithm [4], where updated particles are exchanged between all processes, and a ring algorithm [5].

Parallelization of GRAPEs appears to be an efficient way to reduce the wall-clock time for individual simulations [4,5,6]. The gravitational N-body problem has calculation time complexity $\mathcal{O}(N^2)$, whereas the communication scales only with $\mathcal{O}(N)$. For sufficiently large N, the force calculation time will therefore overtake the communication time. For a local cluster of GRAPEs with low-latency

© Springer-Verlag Berlin Heidelberg 2008

and high bandwidth network, break-even between calculation and communication is reached at $N \sim 10^4$ [6].

Generally, GRAPE clusters are not cheap and few institutions can afford such dedicated hardware solutions. An alternative to purchasing a large cluster is provided by a computational grid [7]. Grid technology can be applied to combine several clusters (with or without GRAPEs) into one collective infrastructure. Doing so is beneficial, as the purchase and maintenance costs are shared among institutes. The grid provides security mechanisms to allow uniform registration and authentication, monitoring tools to detect idle GRAPEs and meta-schedulers to divide the workload over the participating sites. These features make grids both more scalable and flexible than cluster setups that connect using `ssh` keys [8].

The real challenge for the grid is to develop new applications for astronomical problems that have yet to be solved. For example, the simulation of an entire galaxy requires at least several PFLOP/s of computational power and the development of a hybrid simulation environment [9]. Such an environment performs several astrophysical simulations on vastly different temporal and spatial scales, simulating gravity interactions as well as stellar evolution and treatment of close encounters.

To facilitate these tightly-coupled multi-physics simulations on the PFLOP/s scale, we require an extensive and heterogeneous grid infrastructure consisting of several powerful compute clusters. Although grid technology has often been applied to facilitate high-throughput computing [10] or loosely-coupled simulations, little research has been done on investigating how the grid can be efficiently used to solve tightly-coupled HPC problems. By using grid technology for this specific set of problems, we can potentially fullfill the computational requirements for performing petascale multi-physics simulations.

Using a grid infrastructure for HPC has another drawback, as networks between grid sites have completely different characteristics compared to local area networks. Earlier experiments indicate that a grid of regular PCs across Europe improves overall performance for relatively small N [5], but these results do not necessarily extend to a global grid of GRAPEs, which has intercontinental network lines connecting special purpose nodes. We address the question for which problem size a world-wide grid has practical usage, and how to optimize our simulations to take advantage of the grid. In this, we focus on grids equipped with GRAPE hardware.

2 Technical Setup and Performance Model

We have constructed a heterogeneous grid of GRAPEs, which we call the Global GRAPE Grid (or G3). The G3 consists of five nodes across three sites, where each of the nodes is equipped with a GRAPE-6Af special purpose computer. We provide the technical details of our grid setup in 2.1, while we briefly describe the applied performance model in 2.2.

2.1 Technical Setup

Of the five nodes in the G3, two nodes are located at Tokyo University (Tokyo, Japan), two are located at the University of Amsterdam (Amsterdam, the Netherlands) and one is at Drexel University (Philadelphia, United States). Local nodes in the G3 are connected by Gigabit Ethernet, whereas different sites are connected with regular internet. In Table 1 we present the specifications of the G3. Each of the computers in the G3 is set up with Globus Toolkit middleware[1] and MPICH-G2[2].

Table 1. Specifications for the nodes in G3. The first column gives the name of the computer followed by its country of residence (NL for the Netherlands, JP for Japan and US for the United States). The subsequent columns give the type of processor in the node, the amount of RAM, followed by the operating system, the kernel version and the version of Globus installed on the PC. Each of the nodes is equipped with a 1 Gbit/s Ethernet card and GRAPE-6Af hardware. Local nodes are interconnected with Gigabit Ethernet.

name	location	CPU type	RAM [MB]	OS	kernel version	Globus
vader	NL	Intel P4 2.4GHz	1280	Ubuntu 5.10	2.6.5	4.0.3
palpatine	NL	Intel P4 2.67GHz	256	RHEL 3	2.4.21	4.0.3
yoda	JP	Athlon 64 3500+	1024	FC 2	2.6.10	3.2.1
skywalker	JP	Athlon 64 3500+	1024	FC 2	2.6.10	3.2.1
obi-wan	US	2x Xeon 3.6GHz	2048	Gentoo 06.1	2.6.13	4.0.4

In Table 2 we present the network characteristics, latency and bandwidth, of the connections within G3. We tested local area network (LAN) and wide area network (WAN) connections using the UNIX `ping` command to measure latency. We use `scp` for measuring the network bandwidth, transferring a 75 MB file, rather than referring to theoretical limits because the majority of bandwidth on non-dedicated WANs is used by external users. For our performance measurements, we used a standard implementation of MPICH-G2 without specific optimizations for long-distance networking. As a result, the MPI communication makes use of only 40%-50% of the available bandwidth[3]. If we were to optimize MPICH-G2, or add support for grid security to already optimized MPI libraries, such as Makino's tcplib[4] or OpenMPI, our bandwidth use would be close to the bandwidth use of a regular file transfer.

The N-body integrator we have chosen for our experiments uses block time-steps [11] with a 4th order Hermite integration scheme [12]. The time steps with which the particles are integrated are blocked in powers of two between a

[1] `http://www.globus.org`

[2] `http://www3.niu.edu/mpi/`, in the future: `http://dev.globus.org/wiki/MPICH-G2`

[3] For more information we refer to a research report from INRIA: `http://hal.inria.fr/inria-00149411/en/`

[4] See: `http://grape.mtk.nao.ac.jp/∼makino/softwares`

Table 2. Characteristics of local and wide network connections. Latency indicates the required time for sending 1 byte through the network connection. The bandwidth indicates the transfer capacity of the network connection. The bandwidth was measured with a 75MB `scp` file transfer.

connection	latency [ms]	bandwidth (theory) [MB/s]	bandwidth (real) [MB/s]
Amsterdam LAN	0.17	125.0	11.0
Tokyo LAN	0.04	125.0	33.0
Amsterdam - Tokyo WAN	266.0	57.0	0.22
Amsterdam - Phil. WAN	104.0	312.5	0.56
Philadelphia - Tokyo WAN	188.0	57.0	0.32

minimum of 2^{-22} and a maximum of 2^{-3} N-body time unit [13]. During each time step, the codes perform particle predictions, calculate forces between particles and correct particles on a block of active particles (see [6] for a detailed description of the integration scheme). Particle corrections include updates of positions and velocities, and computation of new block time steps of particles. For our experiments we use two implementations of an N-body integrator. One of these codes runs on a single PC with and without GRAPE, whereas the other is parallelized with MPI using the ring algorithm (see [5] for more details).

We initialize the simulations using Plummer spheres that were in virial equilibrium and performed our simulations using a softening parameter of 2^{-8}. Since our simulations are performed over one N-body time unit , the realization of the N-body system is not critical to the timing results.

2.2 Performance Model

To further understand our experiments we adopted the parallel performance model described by [14], which is based on the work of [15], [6] and [5], but then applied to the grid.

To apply these models, we need to measure several parameters. These include τ_{pred}, which is the time to predict a single particle, τ_{force}, which is the time to calculate the forces between two particles, and τ_{corr}, which is the time spent to correct a single particle. We measure the values for τ_{pred}, τ_{force} and τ_{corr} by using a sample N-body simulation with 32768 particles, and provide them in table 3 for the various nodes in the G3.

We have applied the performance model to the results presented in Sect.3. In Fig. 1 we compare the measured wall-clock time for the ring algorithm on the grid with the performance model. To guide the eye, the results for a single GRAPE are also presented in both figures. The performance model tracks the real measurements quite satisfactorily, giving a slightly lower computation time for a single GRAPE while giving a slightly higher computation time for a simulation across grid sites.

The communication overhead of a distributed computer often renders high performance computing on a grid inefficient. However, for sufficiently large N,

Table 3. Machine performance specification and machine-specific constants. The first two columns show the name of the machine, followed by the country of residence. The last three columns give the time required for to perform one particle prediction (τ_{pred}), the time required for one force calculation between two particles (τ_{force}) and the time required for correcting one particle (τ_{corr}) respectively, all in microseconds.

name	location	τ_{pred} [μs]	τ_{force} [μs]	τ_{corr} [μs]
vader	NL	0.247	0.216	4.81
palpatine	NL	0.273	0.193	2.39
yoda	JP	0.131	0.110	1.29
skywalker	JP	0.131	0.110	1.29
obi-wan	US	0.098	0.148	1.14

there will eventually be a point where relatively little time is lost communicating, and the compute resources are efficiently used. In Fig. 1 we can see that, for GRAPE-enabled simulations, the model predicts break-even between calculation and communication around $N \simeq 2 \cdot 10^6$. For large N, a grid of two GRAPEs will therefore outperform a single GRAPE.

3 Results

We have performed a number of simulations on local machines and on the G3, which consists of timing simulations lasting one N-body time unit and short simulations that have been profiled. We measured the full wall-clock execution time for the longer simulations and we profiled the shorter simulations.

3.1 Timing Results of N-Body Calculations

We run the N-body codes, discussed in Sect.2.1, on a single PC and across the network in parallel using $N = 1024$ to $N = 65536$ (a few additional calculations were performed with $N > 65536$). The runs were performed with and without GRAPE. We present our results in Fig. 1. If a simulation is run multiple times with the same problem set, the execution time may be slightly different per run. This variation is within the margin of 1.07 over 4 runs, and can be primarily attributed to fluctuations in the network bandwidth.

Single PC. The performance on a single PC (represented by the thick dashed line with bullets in Fig.1) is entirely dominated by force calculations, which scales as $\mathcal{O}(N^2)$. As the number of steps per N-body time unit increases with N, the execution time scales slightly worse than N^2.

Grid of PCs. The performance on the G3 using all three sites, without using GRAPE, is given by the thin dashed line with triangles. For $N < 24576$, the performance is dominated by network communication. Given that p indicates the number of processes, the network communication scales as $\mathcal{O}(N \log p)$ [6].

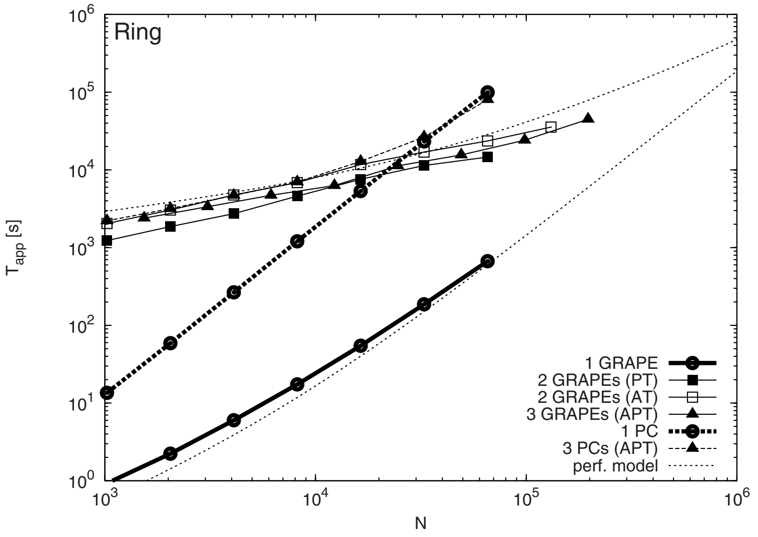

Fig. 1. The time for running the application for 1 N-body time unit (T_{app}) as a function of the number of stars (N) using the ring algorithm. The two thick lines give the results for a single CPU with GRAPE (lower solid curve) and without (top dashed curve). We make the distinction between solid curves to present the results for simulations run with GRAPE, and dashed curves to give the results without GRAPE. The results on the grid are presented with four different lines, based on the three included locations. Each of these runs is performed with one node per site. The results for the WAN connection Philadelphia–Tokyo (given in the legend by PT), Amsterdam–Tokyo (AT) and Amsterdam–Philadelphia–Tokyo (APT) are indicated with the solid curves with filled squares, open squares and filled triangles, respectively. The dashed curve with filled triangles gives the results for the Amsterdam–Philadelphia–Tokyo connection but without using GRAPE. Dotted lines indicate the performance of runs with GRAPE according to the performance model.

For our grid-based simulation experiments without GRAPE, break-even between communications and force calculations is achieved around $N \sim 4 \cdot 10^4$. For larger N, the execution time is dominated by force calculations, rather than network communication.

Single PC with GRAPE. The performance on a single PC with GRAPE, given by the thick solid line with bullets, is dominated by force calculations, although communication between host and GRAPE, and operations on the host PC have an impact on performance for $N < 16384$. For such small N, the GRAPE performs less efficiently, because many blocks are too small to fill the GRAPE pipelines. For larger N, force calculations become the performance bottleneck, and the scaling of the execution time becomes that of a single PC without GRAPE.

Grid of PCs with GRAPE. The performance on two sites in the G3 (with GRAPEs) is given by the thin solid line with solid squares for calculations

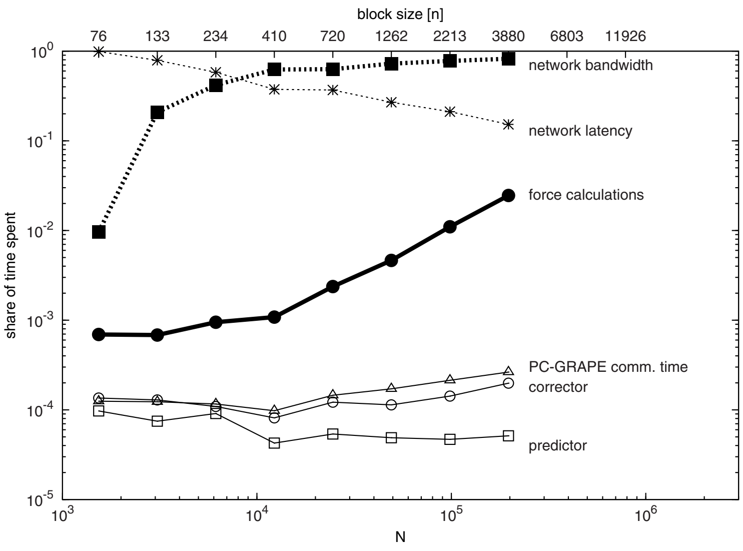

Fig. 2. Share of wall-clock time spent on individual tasks during a single time step, using 3 nodes on 3 sites. Solid lines indicate tasks performed locally. The thick solid line with filled circles represents time spent on force calculations, and the thin solid lines give the result for time spent on communication between PC and GRAPE (open triangles), particle corrections (open circles) and particle predictions (open squares) respectively. Dotted lines indicate time spent on communication between nodes. The thin dotted line with asterisks indicates time spent on communication latency between nodes and the thick dotted line with solid squares indicates time spent on using the network bandwidth.

between Philadelphia and Tokyo, and by the thin solid line with open squares for calculations between Amsterdam and Tokyo. The performance on the G3 using all three sites is given by the thin solid line with triangles. For all problem sizes N we have measured, the grid speedup Γ [16] is less than 0.15, indicating that the performance is dominated by network communication. The network communication time scales better than the force calculation time, therefore, if N is sufficiently large, force calculation time will overtake the network communication time. However, this break-even point lies at much higher N than for a grid of PCs, because the use of GRAPE greatly decreases the time spent on force calculations.

3.2 Profiling of the N-Body Simulations

We have chosen one parallel algorithm (ring) and one resource topology (3 nodes on 3 sites) to profile the simulation during one integration time step. The block size n for every measurement was fixed using a formula for calculating average block size $\left(n = 0.20N^{0.81}\right)$, which has been used for the same initial conditions in [3]. During execution, we measured the time spent on individual tasks, such as force calculations or communication latency between processes. We have profiled our simulations for $N = 1024$ up to $N = 196608$, using the timings measured on the process running in Tokyo. The results of these measurements are given in Fig. 2.

We find that for larger N, low bandwidth of our wide area network affects the outcome of the performance measurements, and that MPI calls are only able to use about a quarter of the available bandwidth for passing message content. For $N \gtrsim 5 \cdot 10^5$ we expect the force calculation to take more time than network latency. If we were to use the network bandwidth more efficiently for such a large number of particles, the execution time would be dominated by force calculations. The usable bandwidth can be increased either by using a more efficient MPI implementation (see Sect.2.1) or by using a dedicated network. Using our current networking and MPI implementation, we expect that for $N \gtrsim 2 \cdot 10^6$ particles the force calculation time overtakes the bandwidth time.

4 Conclusions

We studied the potential use of a virtual organization in which GRAPEs are used in a wide area grid. We tested the performance model with an actual grid across three sites, each of which is located on a different continent. We used GRAPE hardware in Japan, the Netherlands and the USA in parallel for calculations of 1024 up to 196608 particles.

With these particle numbers we were unable to reach superior speed compared to a single GRAPE. However, we were both able to run simulations consisting of 3 times as many particles and to outperform a single computer without GRAPE. We estimate that a small intercontinental grid of GRAPEs will reach superior performance compared to single GRAPE for $N \gtrsim 2 \cdot 10^6$ particles. If we were to increase the bandwidth by two orders of magnitude, e.g. by using dedicated light paths, we expect the grid of GRAPEs to outperform a single GRAPE for $N \gtrsim 4 \cdot 10^5$.

We have mainly discussed the use of GRAPEs in a virtual organization, but new developments in using graphical processing units appear to achieve similar speeds as GRAPEs [3,17,18]. In addition, GPUs are equipped with a larger amount of memory, which allows us to exploit more memory-intensive, but also faster, parallel algorithms.

Although our proof-of-concept infrastructure was of limited size, we have shown that it is possible to use dedicated hardware components located across clusters for high performance computing. As we have profiled and modelled a single-physics N-body simulation on the G3, we can proceed by bringing a multi-physics simulation environment (such as MUSE [5]) to the grid. Scheduling the complex communications between stellar dynamics, evolution and collision simulations on a grid infrastructure provides a previously unexplored direction for future work.

Alternatively, a much larger grid can be used to simulate a very large N-body system. Although break even between communication and computation

[5] `http://muse.li`

occurs at relatively large N for block time-step N-body simulations over regular internet, shared time-step simulations perform much more favorably, especially when using dedicated light paths.

Acknowledgements. We are grateful to Mary Inaba and Cees de Laat for discussion and support in realizing this grid setup, and to Alessia Gualandris for providing some of the codes and feedback on this work. We are also grateful to Alfons Hoekstra, Marian Bubak and Stefan Harfst for fruitful discussions on the contents of this paper. This research is supported by the Netherlands organization for Scientific research (NWO) grant #643.200.503 and by the European Commission grant for the QosCosGrid project (grant number: FP6-2005-IST-5 033883), and we thank SARA computing and networking services, Amsterdam for technical support.

References

1. Aarseth, S.J.: Direct n-body calculations. In: Goodman, J., Hut, P. (eds.) Dynamics of Star Clusters. IAU Symposium, vol. 113, pp. 251–258 (1985)
2. Fukushige, T., Makino, J., Kawai, A.: GRAPE-6A: A Single-Card GRAPE-6 for Parallel PC-GRAPE Cluster Systems. Publications of the Astronomical Society of Japan 57, 1009–1021 (2005)
3. Portegies Zwart, S.F., Belleman, R.G., Geldof, P.M.: High-performance direct gravitational N-body simulations on graphics processing units. New Astronomy 12, 641–650 (2007)
4. Makino, J., Kokubo, E., Fukushige, T.: Performance evaluation and tuning of grape-6 - towards 40 "real" tflops. sc 00, 2 (2003)
5. Gualandris, A., Portegies Zwart, S., Tirado-Ramos, A.: Performance analysis of direct n-body algorithms for astrophysical simulations on distributed systems. Parallel Computing 33(3), 159–173 (2007)
6. Harfst, S., Gualandris, A., Merritt, D., Spurzem, R., Portegies Zwart, S., Berczik, P.: Performance analysis of direct N-body algorithms on special-purpose supercomputers. New Astronomy 12, 357–377 (2007)
7. Foster, I., Kesselman, C., Tuecke, S.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of High Performance Computing Applications 15(3), 200–222 (2001)
8. Boku, T., Onuma, K., Sato, M., Nakajima, Y., Takahashi, D.: Grid environment for computational astrophysics driven by grape-6 with hmcs-g and omnirpc. ipdps 05, 176a (2005)
9. Hoekstra, A.G., Portegies Zwart, S.F., Bubak, M., Sloot, P.M.A.: Petascale Computing: Algorithms and Applications, 1st edn., pp. 147–159. Chapman and Hall/CRC (2008)
10. Abramson, D., Giddy, J., Kotler, L.: High performance parametric modeling with nimrod/g: Killer application for the global grid? ipdps 00, 520 (2000)
11. McMillan, S.L.W.: The use of supercomputers in stellar dynamics; proceedings of the workshop, institute for advanced study, Princeton, NJ, June 2-4 (1986). In: Hut, P., McMillan, S.L.W. (eds.) The Use of Supercomputers in Stellar Dynamics. Lecture Notes in Physics, vol. 267, p. 156. Springer, Berlin (1986)

12. Makino, J., Aarseth, S.J.: On a Hermite integrator with Ahmad-Cohen scheme for gravitational many-body problems. Publications of the Astronomical Society of Japan 44, 141–151 (1992)
13. Heggie, D.C., Mathieu, R.D.: Standardised Units and Time Scales. In: Hut, P., McMillan, S.L.W. (eds.) The Use of Supercomputers in Stellar Dynamics. Lecture Notes in Physics, vol. 267, p. 233. Springer, Berlin (1986)
14. Groen, D., Portegies Zwart, S., McMillan, S., Makino, J.: Distributed N-body simulation on the grid using dedicated hardware. New Astronomy 13, 348–358 (2008)
15. Makino, J.: An efficient parallel algorithm for O(N^2) direct summation method and its variations on distributed-memory parallel machines. New Astronomy 7, 373–384 (2002)
16. Hoekstra, A.G., Sloot, P.M.A.: Introducing grid speedup g: A scalability metric for parallel applications on the grid. In: Sloot, P.M.A., Hoekstra, A.G., Priol, T., Reinefeld, A., Bubak, M. (eds.) EGC 2005. LNCS, vol. 3470, pp. 245–254. Springer, Heidelberg (2005)
17. Hamada, T., Iitaka, T.: The Chamomile Scheme: An Optimized Algorithm for N-body simulations on Programmable Graphics Processing Units. ArXiv Astrophysics e-prints, astro-ph/0703100 (March 2007)
18. Belleman, R.G., Bédorf, J., Portegies Zwart, S.F.: High performance direct gravitational N-body simulations on graphics processing units II: An implementation in CUDA. New Astronomy 13, 103–112 (2008)