Noname manuscript No.
(will be inserted by the editor)

Astrophysical Particle Simulations with Large Custom GPU Clusters on Three Continents

R. Spurzem · P. Berczik · T. Hamada · K. Nitadori · G. Marcus · A. Kugel · R. Männer · I. Berentzen · J. Fiestas · R. Klessen · R. Banerjee

Received: date / Accepted: date

Abstract We present direct astrophysical N-body simulations with up to six million bodies using our parallel MPI-CUDA code on large GPU clusters in Beijing, Berkeley, and Heidelberg, with different kinds of GPU hardware. The clusters are linked in the cooperation of ICCS (International Center for Computational Science). We reach about 1/3 of the peak performance for this code, in a real application scenario with hierarchically blocked timesteps and a core-halo density structure of the stellar system. The code and hardware is used to simulate dense star clusters with many binaries and galactic nuclei with supermassive black holes, in which correlations between distant particles cannot be neglected.

Keywords N-Body Simulations · Computational Astrophysics · GPU Clusters

R. Spurzem, P. Berczik, J. Fiestas
National Astronomical Observatories of China, Chinese Academy of Sciences, 20A Datun Rd., Chaoyang District, Beijing 100012, China E-mail: berczik,spurzem@bao.ac.cn

R. Spurzem, P. Berczik, I. Berentzen, J. Fiestas
University of Heidelberg, Astronomisches Rechen-Institut (ZAH), Mönchhofstr. 12-14, 69120 Heidelberg, Germany

K. Nitadori
RIKEN Institute, Tokyo, Japan

T. Hamada
Nagasaki Advanced Computing Center, Univ. of Nagasaki, Japan

G. Marcus, A. Kugel, R. Männer
University of Heidelberg, Dept. of Computer Science V, Central Inst. of Computer Engineering, located in Mannheim

I. Berentzen, R. Klessen, R. Banerjee
University of Heidelberg, Inst. für Theor. Astrophysik (ZAH), Germany

1 Introduction

Competitive astronomical and astrophysical research requires access to competitive computing facilities. Theoretical numerical modelling of astrophysical objects, their composition, radiation, and dynamical evolution has become a third basic method of astrophysical research, besides observation and pure theory. Numerical modelling allows one to compare theory with observational data in unprecedented detail, and it also provides theoretical insight into physical processes at work in complex systems. Similarly, data processing of astrophysical observations comprises the use of complex software pipeline to bring raw data into a form digestible for observational astronomers and ready for exchange and publication; these are, e.g., mathematical transformations like Fourier analyses of time series or spatial structures, complex template analyses or huge matrix-vector operations. Here fast access to and transmission of data, too, require supercomputing capacities.

We are undergoing a new revolution on parallel processor technologies, especially with regard to the Graphic Processing Units. GPUs have become widely used nowadays to accelerate a broad range of applications, including computational physics and astrophysics, image/video processing, engineering simulations, quantum chemistry, just to name a few (Egri 2007, Yasuda 2007, Yang, Wang & Chen 2007, Akeley et al. 2007, Hwu 2011). Graphics processing unit (GPUs) are rapidly emerging as a powerful and cost-effective platform for high performance parallel computing. The GPU Technology Conference 2010 held by NVIDIA in San Jose in autumn 2010 (http://www.nvidia.com/gtc) gave one snapshot of the breadth and depth of present day GPU (super)computing applications. Recent GPUs, such as the NVIDIA Fermi C2050 Computing Processor, offer

414 processor cores and extremely fast on-chip-memory chip, as compared to only 8 cores on a standard Intel or AMD CPU. Groups of cores have access to very fast shared memory pieces; a single Fermi C2050 device supports double precision operations fully with a peak speed of 515 Gflop/s; some GPU clusters in production still use the older Tesla C1060 card, which only has intrinsic single precision support. It can be circumvented by emulation of double precision operations (e.g. Nitadori & Makino 2008 for an example). In this paper we present first benchmarks of our applications on a Fermi based GPU cluster, kindly provided by LBNL/NERSC Berkeley (dirac cluster).

Scientists are using GPU since about five years already for scientific simulations, but only the invention of CUDA (Compute Unified Device Architecture, Akeley et al. 2007) as a high-level programming language for GPUs made their computing power available to any student or researcher with normal scientific programming skills. The number of scientific papers in the Harvard Astrophysics Database with GPU in title or abstract is about 250 between 2005 and 2009, and since 2010 alone again some 250 entries, which illustrates the strong increase in last years. CUDA is limited to GPU devices of NVIDIA, but the new open source language OpenCL will provide access to any type of many-core accelerator through an abstract programming language. Computational astrophysics has been a pioneer to use GPUs for high performance general purpose computing (see for example the AstroGPU workshop in Princeton 2007 http://www.astrogpu.org). It started with the GRAPE (Gravity Pipe) accelerator boards from Japan 10 years ago (Makino et al. 2003, Fukushige et al. 2005). Recently, clusters using GRAPE and GPU were used for direct N-body codes to model the dynamics of supermassive black holes in galactic nuclei (Berczik et al. 2006, Berentzen et al. 2009), the dynamics of dense star clusters (Belleman et al. 2008, Portegies Zwart et al. 2007), in gravitational lensing ray shooting problems (Thomson et al. 2010), in numerical hydrodynamics with adaptive mesh refinement (Schive et al. 2010, Wang et al. 2009, Wang, Abel & Kaehler 2009) and magnetohydrodynamics (Wong et al. 2009), or Fast Fourier transformation (Chen at al. 2010, Cui et al. 2009). While it is relatively simple to obtain good performance with one or few GPU relative to CPU, a new taxonomy of parallel algorithms is needed for parallel clusters with many GPUs (Barsdell et al. 2010). Only "embarrassingly" parallel codes scale well even for large number of GPUs, while in other cases like hydrodynamics or FFT on GPU the speed-up is somewhat limited to 10-50 for the whole application, and this number needs to be carefully checked whether it compares the GPU performance with single or multi-core CPUs. A careful study of the algorithms and their data flow and data patterns, is useful and has led to significant improvements, for example for particle based simulations using smoothed particle hydrodynamics (Berczik et al. 2007, Spurzem et al. 2009) or for FFT (Chen at al. 2010, Cui et al. 2009). Recently new GPU implementations of Fast-Multipole Methods (FMM) have been presented and compared with Tree Codes (Yokota & Barba 2010, Yokota et al. 2010). FMM codes have first been presented by Greengard & Rokhlin (1987). It is expected that on the path to Exascale applications further - possibly dramatic - changes in algorithms are required; at present it is unclear whether the current paradigm of heterogeneous computing with one CPU and an accelerator device GPU will remain dominant.

2 Astrophysical Application

Dynamical modelling of dense star clusters with and without massive black holes poses extraordinary physical and numerical challenges; one of them is that gravity cannot be shielded such as electromagnetic forces in plasmas, therefore long-range interactions go across the entire system and couple non-linearly with small scales; high-order integration schemes and direct force computations for large numbers of particles have to be used to properly resolve all physical processes in the system. On small sclaes inevitably correlations form already early during the process of star formation in a molecular cloud. Such systems are dynamically extremely rich, they exhibit a strong sensitivity to initial conditions and regions of phase space with deterministic chaos. Typically in a globular star cluster time scales can vary between a million years (for an orbit time in the cluster) to hours (orbital time of the most compact binaries). The dynamics of dense stellar systems so far has been treated as a classical Newtonian problem until recently for only a few 10^5 particles, much less than necessary. Only with the advent of accelerated hardware (GRAPE and GPU) realistic particle numbers can be approached.

Direct N-Body Codes in astrophysical applications for galactic nuclei, galactic dynamics and star cluster dynamics usually have a kernel in which direct particle-particle forces are evaluated. Gravity as a monopole force cannot be shielded on large distances, so astrophysical structures develop high density contrasts. High-Density regions created by gravitational collapse coexist with low-density fields, as is known from structure formation in the universe or the turbulent structure of the interstellar medium. A high-order time integrator in connection with individual, hierarchically blocked time

steps for particles in a direct NBody simulation provides the best compromise between accuracy, efficiency and scalability (Makino & Hut 1988, Aarseth 1999a,b, 2003, Spurzem 1999, Harfst et al. 2007). With GPU hardware up to a few million bodies could be reached for such models (Berczik et al. 2005, 2006, Gualandris & Merritt 2008). Note that while Greengard & Rokhlin (1987) already mention that their algorithm can be used to compute gravitational forces between particles to high accuracy, Makino & Hut (1988) find that the self-adaptive hierarchical time-step structure inherited from Aarseth's codes improves the performance for spatially structured systems by $\mathcal{O}(\mathcal{N})$ - it means that at least for astrophysical applications with high density contrast FMM is not a priori more efficient than direct N-body (which sometimes is called "brute force", but that should only be used if a shared time step is used, which is not the case in our codes). One could explain this result by comparing the efficient spatial decomposition of forces (in FMM, using a simple shared time step) with the equally efficient temporal decomposition (in direct N-body, using a simple spatial force calculation).

On the other hand, cosmological N-body simulations use thousand times more particles (billions, order 109), at the price of allowing less accuracy for the gravitational force evaluations, either through the use of a hierarchical decomposition of particle forces in time (so-called neighbour scheme codes, Ahmad & Cohen 1973, Makino & Aarseth 1992, Aarseth 2003), or in space (tree codes, Barnes & Hut 1986, Makino 2004, Springel 2005). Another possibility is the use of fast-multipole algorithms (Greengard & Rokhlin 1997, Dehnen 2000 , 2002, Yokota & Barba 2010, Yokota et al. 2010) or particle-mesh schemes (PM, Hockney & Eastwood 1988, Fellhauer et al. 2001) which use FFT for their Poisson solver. PM schemes are the fastest for large systems, but their resolution is limited to the grid cell size. Adaptive codes use direct particle-particle forces for close interactions below grid resolution (AP3M, Pearce & Couchman 1997, Couchman et al. 1995). But for astrophysical systems with high density contrasts tree codes are more efficient. Recent codes for massively parallel supercomputers try to provide adaptive schemes using both tree and PM, such as the well-known GADGET and treePM codes (Springel 2005, Xu 1995, Yoshikawa & Fukushige 2005, Ishiyama et al. 2010).

3 Hardware

In this article we report on new results obtained from our recently installed GPU clusters using NVIDIA Tesla C1060 cards in Beijing, China (laohu cluster with 85 Dual Intel Xeon nodes and 170 GPU's) and Heidelberg, Germany (kolob cluster with 40 and titan cluster with 32 nodes, both featuring Dual Intel Xeon nodes and Tesla GPU's of the pre-Fermi single precision only generation), and on a recent cluster with Fermi C2050 cards in Berkeley. In Germany, at Heidelberg University, our teams have operated many-core accelerated clusters using GRAPE hardware for many years (Harfst et al. 2007, Spurzem et al. 2004, 2007, 2008, 2009) Part of our team is now based at the National Astronomical Observatories of China (NAOC) of Chinese Academy of Sciences (CAS), in Beijing. NAOC is part of a GPU cluster network covering ten institutions of CAS, aiming for high performance scientific applications in a cross-disciplinary way. The top level cluster in this network is the recently installed Mole-8.5 cluster at Institute of Process Engineering (IPE) of CAS in Beijing (2 Pflop/s single precision peak from order 2000 Fermi C2050 devices). The total capacity of the CAS GPU cluster network is nearly 5 Pflop/s single precision peak. Here we report on the part of the system which is running at NAOC for mostly astrophysical simulations. The laohu GPU cluster features 170 NVIDIA Tesla C1070 GPUs running on 85 nodes. In China GPU computing is blooming, the top spot in the list of 500 fastest supercomputers in the world (http://www.top500.org) plus a couple of further entries in the top 20 are now occupied by China. The top system in our CAS network is currently number 19. Research and Teaching in CAS institutions is focused on broadening the computational science base to use the clusters for supercomputing in basic and applied sciences.

4 Software

The test code which we use for benchmarking on our clusters is a direct N-body simulation code for astrophysics, using a high order Hermite integration scheme and individual block time steps (the code supports time integration of particle orbits with 4th, 6th, and 8th order schemes). The code is called φGPU, it has been developed from our earlier published versions φGRAPE (using GRAPE hardware instead of GPU, Harfst et al. 2007). It is parallelisd using MPI, and on each node using many cores of the special hardware. The code was mainly developed and tested by three of us (Peter Berczik, Tsuyoshi Hamada, Keigo Nitadori) and is based on an earlier version for GRAPE clusters (Harfst et al. 2007). The code is written in C++ and based on Nitadori & Makino (2008) earlier CPU serial code (yebisu). The present version of φGPU code we used and tested only with the recent GNU compilers (ver. 4.1 and 4.2).

Fig. 1 Left: NAOC GPU cluster in Beijing; 85 nodes with 170 NVIDIA Tesla C1070 GPUs, 170 Tflop/s hardware peak speed, installed 2010; Right: Frontier kolob cluster at ZITI Mannnheim, 40 nodes with 40 NVIDIA Tesla C870 GPU accelerators, 17 Tflop/s hardware peak speed; installed 2008. Each line corresponds to a different problem size (particle number), which is given in the key.

The MPI parallelization was done in the same "j" particle parallelization mode as in the earlier φGRAPE code (Harfst et al. 2007). The particles are divided equally between the working nodes and in each node we calculate only the fractional forces for the active "i" particles at the current timestep. Due to the hierarchical time step scheme the number $N_{\rm act}$ of active particles (due for a new force computation at a given time level) is usually small compared to the total particle number N, but its actual value can vary from $1 \ldots N$. The full forces from all the particles acting on the active particles we get after using the global MPI_SUM communication routines.

We use native GPU support and direct code access to the GPU with only CUDA. Recently we use CUDA 2.2. Multi GPU support is achieved through MPI parallelization; each MPI process uses only a single GPU, but we can start two MPI processes per node (to use effectively the dual CPU's and GPU's in the NAOC cluster) and in this case each MPI process uses its own GPU inside the node. Communication always (even for the processes inside one node) works via MPI. We do not use any of the possible OMP (multi-thread) features of recent gcc 4.x compilers inside one node.

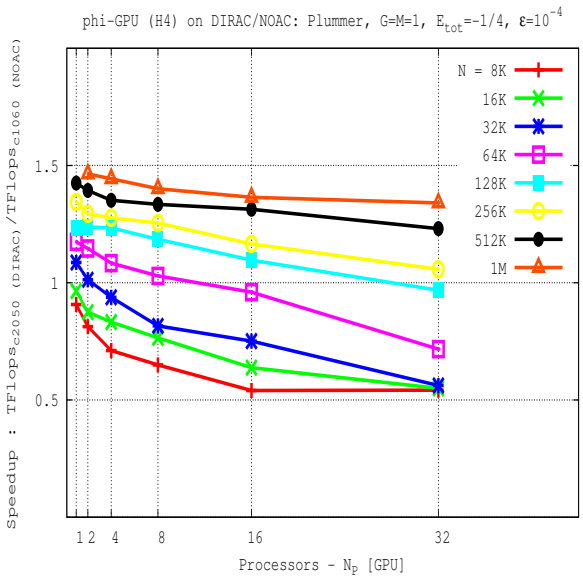

Fig. 3 Top: Comparing the performance of φGPU on equal numbers of Tesla C1060 (laohu cluster in Beijing) and Fermi C2050 (dirac cluster in Berkeley) accelerated nodes; speed in Teraflop/s reached as a function of number of processes, each process with one GPU; limit to 32 GPU's as this is the maximum on the dirac cluster. Each line corresponds to a different problem size (particle number), which is given in the key.

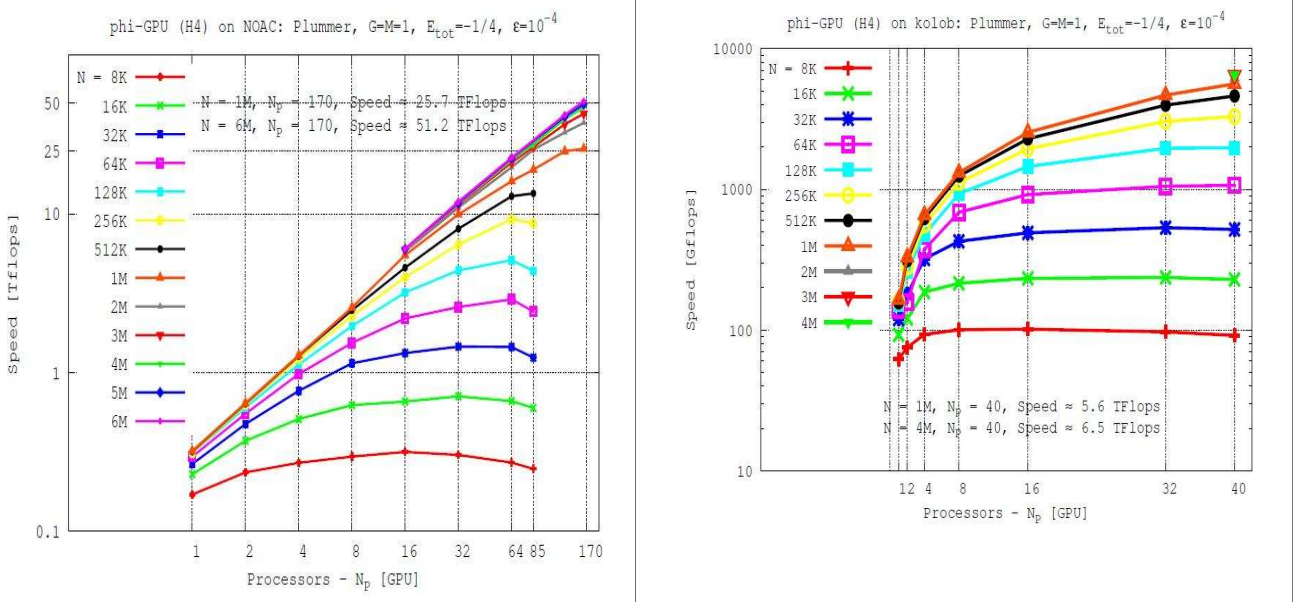

Fig. 2 Strong scaling for different problem sizes; top: NAOC GPU cluster in Beijing; speed in Teraflop/s reached as a function of number of processes, each process with one GPU; 51.2 Tflop/s sustained were reached with 164 GPUs (3 nodes with 6 GPUs were down at the time of testing). Bottom: Same benchmark simulations for the Frontier kolob cluster at ZITI Mannnheim, 6.5 Tflops/s reached for four million particles on 40 GPU's. Each line corresponds to a different problem size (particle number), which is given in the key. Note that the linear curve corresponds to ideal scaling.

5 Results of Benchmarks

The figures show results of our benchmarks, with a maximum of 164 GPU cards used (3 nodes i.e. 6 cards were down during the test period). The largest performance was reached for 6 million particles, with 51.2 Tflop/s in total sustained speed for our application code, in a astrophysical run of a Plummer star cluster model, simulating one physical time unit (about one third of the orbital time at the half-mass radius). Based on these results we see that we get a sustained speed for 1 NVIDIA Tesla C1070 GPU card of 312 Gflop/s (i.e. about one third of the theoretical hardware peak speed of 1 Tflop/s). Equivalently, for the smaller kolob cluster with 40 NVIDIA Tesla C870 GPU's in Germany, we obtain 6.5 Tflop/s with 4 million particles. This is 162.5 Gflop/s per card.

Finally an interesting result can be seen on the new dirac cluster at NERSC/LBNL Berkeley, where we compare in Fig. 3 the performance on an equal number of Tesla C1060 and Fermi C2050 accelerators, using the emulated double precision for some operations on C1060 and the full double precision support on C2050. As expected the gain is of the order of 50% on the part of the GPU computation, which can only be seen for large particle numbers and smaller number of GPUs.

6 Conclusions

We have presented implementations of force computations between particles for astrophysical simulations, using our GPU clusters with MPI parallel codes in China and Germany. The overall parallelization efficiency of our codes is very good as one can see from the near ideal speedup in Fig. 2, and in accord with our earlier results on GRAPE clusters (Harfst et al. 2007). The larger simulations (several million particles) show nearly ideal strong scaling (linear relation between speed and number of GPU's) up to our present maximum number of nearly 170 GPU's - no strong sign of a turnover yet due to communication or other latencies. Therefore we are currently testing the code implementation on much larger GPU clusters, such as the Mole-8.5 of IPE/CAS.

The wall clock time T needed for our particle based algorithm to advance the simulation by a certain physical time integration interval scales as

$$T = T_{\rm host} + T_{\rm GPU} + T_{\rm comm} + T_{\rm MPI} \quad (1)$$

where the components of T are (from left to right) the computing time spent on the host, on the GPU, the communication time to send data between host and GPU, and the communication time for MPI data exchange between the nodes. In our present implementation all components are blocking, so there is no hiding of

communication. This will be improved in further code versions, but for now it eases profiling. The dominant term is in the linearly rising part of the curves in Fig. 2 just $T_{\rm GPU}$, while the turnover to flat is dominated by MPI communication. The interested reader may refer to the previous paper of Harfst et al. (2007) for further details about our definitions and measurements (for the case of GRAPE instead of GPU, but analogous) and to our paper in preparation (Berczik et al. 2011) for new data with the new GPU hardware.

To our knowledge the direct N-body simulation with six million bodies in the framework of a so-called Aarseth style code (Hermite scheme 4th order, hierarchical timestep, integrating an astrophysically relevant Plummer model with core-halo structure in density for a certain physical time) is the largest such simulation which exists so far. However, the presently used parallel MPI-CUDA GPU code φGPU is on the algorithmic level of NBODY1 (Aarseth 1999) - though it is already strongly used in production, useful features such as regularisation of few-body encounters and an Ahmad-Cohen neighbour scheme (Ahmad & Cohen 1973), which would bring the code on the level of NBODY6, are not yet implemented. There is an existing NBODY6 code for acceleration on a single node with one or two GPU's (work by Aarseth & Nitadori, see nbody6 at http://www.ast.cam.ac.uk/šverre/web/pages/nbody.htm) and there is NBODY6++ (Spurzem 1999), a massively parallel code for general purpose parallel computers. An NBODY6++ variant using many GPU's in a cluster is work in progress. Such a code could potentially reach the same physical integration time (with same accuracy) using only one order of magnitude less floating operations. The NBODY6 codes are algorithmically more efficient than φGPU or NBODY1, because they use an Ahmad-Cohen neighbour scheme (Ahmad & Cohen 1973), which reduces the total number of full force calculations needed again (in addition to the individual hierarchic time step scheme), i.e. the proportionality factor in front of the asymptotic complexity N^2 is further reduced.

The φGPU code is already now useful for astrophysical production runs to model the dynamics of supermassive black holes in dense stellar systems in galactic nuclei (cf. e.g. Khalisi et al. 2007, Berentzen et al. 2009, Amaro-Seoane et al. 2010a, 2010b, Pasetto et al. 2010, Just et al. 2010, Berczik et al. 2011).

We have shown that our GPU clusters for the very favourable direct N-body application reach about one third of the theoretical peak speed sustained for a real application code with individual time steps. In the future we will use larger Fermi based GPU clusters such as the Mole-8.5 cluster at the Institute of Process Engineering of Chinese Academy of Sciences in Beijing (IPE/CAS) and more efficient variants of our direct N-body algorithms; details of benchmarks and science results, and the requirements to reach Exascale performance, will be published elsewhere.

Acknowledgments

We cordially thank Institute of Process Engineering (IPE) of Chinese Academy of Sciences (CAS), Ge Wei, Wang Xiaowei and their colleagues for continuous support and cooperation; we gratefully acknowledge computing time on the dirac cluster of NERSC/LBNL in Berkeley and thank Hemant Shukla, John Shalf, Horst Simon for providing the access to this cluster and for cooperation in the International Center of Computational Science (ICCS, http://iccs.lbl.gov). Chinese Academy of Sciences has supported this work by a Visiting Professorship for Senior International Scientists, Grant Number 2009S1-5 (RS), and National Astronomical Observatory of China (NAOC) of CAS by the Silk Road Project (RS,PB, JF partly). The special supercomputer laohu at the High Performance Computing Center at National Astronomical Observatories of China, funded by Ministry of Finance under the grant ZDYZ2008-2, has been used. Simulations were also performed on the GRACE supercomputer (grants I/80 041-043 and I/81 396 of the Volkswagen Foundation and 823.219-439/30 and /36 of the Ministry of Science, Research and the Arts of Baden-Württemberg). P.B. acknowledges the special support by the NAS Ukraine under the Main Astronomical Observatory GRAPE/GRID computing cluster project. P.B.'s studies are also partially supported by the program Cosmomicrophysics of NAS Ukraine. The kolob cluster is funded by the excellence funds of the University of Heidelberg in the Frontier scheme.

References

Aarseth, S. J., Gravitational N-Body Simulations, 2003, Gravitational N-Body Simulations, by Sverre J. Aarseth, pp. 430. ISBN 0521432723. Cambridge, UK: Cambridge University Press, November 2003.

Aarseth, S. J., From NBODY1 to NBODY6: The Growth of an Industry, 1999a, Publications of the Astronomical Society of the Pacific 111, 1333

Aarseth, S. J., Star Cluster Simulations: the State of the Art, 1999b, Celestial Mechanics and Dynamical Astronomy 73, 127

Akeley, Kurt and Nguyen, Hubert and Nvidia, GPU Gems 3, Programming Techniques for High-Performance Graphics and General-Purpose Computation, 2007, Addison-Wesley Professional

Amaro-Seoane, P., Sesana, A., Hoffman, L., Benacquista, M., Eichhorn, C., Makino, J., Spurzem, R., Triplets of supermassive black holes: astrophysics, gravitational waves and detection, 2010, Monthly Notices of the Royal Astronomical Society 402, 2308

Amaro-Seoane, P., Eichhorn, C., Porter, E. K., Spurzem, R., Binaries of massive black holes in rotating clusters: dynamics, gravitational waves, detection and the role of eccentricity, 2010, Monthly Notices of the Royal Astronomical Society 401, 2268

Barnes, J., Hut, P., A hierarchical O(N log N) force-calculation algorithm, 1986, Nature 324, 446

Barsdell, B. R., Barnes, D. G., Fluke, C. J., Advanced Architectures for Astrophysical Supercomputing, 2010, ArXiv e-prints arXiv:1001.2048, to appear in the proceedings of ADASS XIX, Oct 4-8 2009, Sapporo, Japan (ASP Conf. Series)

Belleman, R. G., Bedorf, J., Portegies Zwart, S. F., High performance direct gravitational N-body simulations on graphics processing units II: An implementation in CUDA, 2008, New Astronomy 13, 103

Berczik, P., Nakasato, N., Berentzen, I., Spurzem, R., Marcus, G., Lienhart, G., Kugel, A., Männer, R., Burkert, A., Wetzstein, M., Naab, T., Vasquez, H., Vinogradov, S. B., Special, hardware accelerated, parallel SPH code for galaxy evolution., 2007, SPHERIC - Smoothed Particle Hydrodynamics European Research Interest Community. 5

Couchman, H. M. P., Thomas, P. A., Pearce, F. R., Hydra: an Adaptive-Mesh Implementation of P 3M-SPH, 1995, The Astrophysical Journal 452, 797

Dehnen, W., A Hierarchical O(N) Force Calculation Algorithm, 2002, Journal of Computational Physics 179, 27

Dehnen, W., A Very Fast and Momentum-conserving Tree Code, 2000, The Astrophysical Journal 536, L39

Egri, G., et al., K., Comp. Phys. Comm. 177, 631 (2007)

Fellhauer, M., Kroupa, P., Baumgardt, H., Bien, R., Boily, C. M., Spurzem, R., Wassmer, N., SUPERBOX - an efficient code for collisionless galactic dynamics, 2000, New Astronomy 5, 305

Fukushige, T., Makino, J., Kawai, A., 2005, GRAPE-6A: A Single-Card GRAPE-6 for Parallel PC-GRAPE Cluster Systems, Publications of the Astronomical Society of Japan 57, 1009

reengard, L., Rokhlin, V., A fast algorithm for particle simulations, 1987, Journal of Computational Physics 73, 325

Greengard, L., Rokhlin, V., A Fast Algorithm for Particle Simulations, 1997, Journal of Computational Physics 135, 280

Harfst, S., Gualandris, A., Merritt, D., Spurzem, R., Portegies Zwart, S., Berczik, P., Performance analysis of direct N-body algorithms on special-purpose supercomputers, 2007, New Astronomy 12, 357

Hockney, R. W., Eastwood, J. W., Computer simulation using particles, 1988, Bristol: Hilger

Hwu, W.-M. W., Gpu Computing Gems, 2011, Morgan Kaufman Publ. Inc.

Ishiyama, T., Fukushige, T., Makino, J., GreeM: Massively Parallel TreePM Code for Large Cosmological N -body Simulations, 2009, Publications of the Astronomical Society of Japan 61, 1319

Just, A., Khan, F. M., Berczik, P., Ernst, A., Spurzem, R., Dynamical friction of massive objects in galactic centres, 2010, Monthly Notices of the Royal Astronomical Society 411, 653

Khalisi, E., Amaro-Seoane, P., Spurzem, R., A comprehensive NBODY study of mass segregation in star clusters: energy equipartition and escape, 2007, Monthly Notices of the Royal Astronomical Society 374, 703

Makino, J., Hut, P., Performance analysis of direct N-body calculations, 1988, The Astrophysical Journal Supplement Series 68, 833

Makino, J., Aarseth, S. J., On a Hermite integrator with Ahmad-Cohen scheme for gravitational many-body problems, 1992, Publications of the Astronomical Society of Japan 44, 141

Makino, J., Fukushige, T., Koga, M., Namura, K., 2003, Publications of the Astronomical Society of Japan 55, 1163

Makino, J., A Fast Parallel Treecode with GRAPE, 2004, Publications of the Astronomical Society of Japan 56, 521

Nakasato, N., Oct-tree Method on GPU, 2009, ArXiv e-prints arXiv:0909.0541

Nitadori, K., Makino, J., Sixth- and eighth-order Hermite integrator for N-body simulations, 2008, New Astronomy 13, 498

Oliker, L. Green flash: Designing an energy efficient climate supercomputer, ISDPS, pp.1, 2009 IEEE International Symposium on Parallel & Distributed Processing, 2009,

Pasetto, S., Grebel, E. K., Berczik, P., Chiosi, C., Spurzem, R., Orbital evolution of the Carina dwarf galaxy and self-consistent determination of star formation history, 2011, Astronomy and Astrophysics 525, A99

Pearce, F. R., Couchman, H. M. P., Hydra: a parallel adaptive grid code, 1997, New Astronomy 2, 411

Portegies Zwart, S. F., Belleman, R. G., Geldof, P. M., High-performance direct gravitational N-body simulations on graphics processing units, 2007, New Astronomy 12, 641

Schive, H.-Y., Tsai, Y.-C., Chiueh, T., GAMER: A Graphic Processing Unit Accelerated Adaptive-Mesh-Refinement Code for Astrophysics, 2010, Astrophysical Journal Supplement Series 186, 457

Springel, V., The cosmological simulation code GADGET-2, 2005, Monthly Notices of the Royal Astronomical Society 364, 1105

Spurzem, R., Direct N-body Simulations, 1999, Journal of Computational and Applied Mathematics 109, 407

Spurzem, R., Berczik, P., Hensler, G., Theis, C., Amaro-Seoane, P., Freitag, M., Just, A., Physical Processes in Star-Gas Systems, 2004, Publications of the Astronomical Society of Australia 21, 188

Spurzem, R., Berczik, P., Berentzen, I., Merritt, D., Nakasato, N., Adorf, H. M., Brüsemeister, T., Schwekendiek, P., Steinacker, J., Wambsganß, J., Martinez, G. M., Lienhart, G., Kugel, A., Männer, R., Burkert, A., Naab, T., Vasquez, H., Wetzstein, M., From Newton to Einstein N-body dynamics in galactic nuclei and SPH using new special hardware and astrogrid-D, 2007, Journal of Physics Conference Series 78, 012071

Spurzem, R., Berentzen, I., Berczik, P., Merritt, D., Amaro-Seoane, P., Harfst, S., Gualandris, A., Parallelization, Special Hardware and Post-Newtonian Dynamics in Direct N - Body Simulations, 2008, Lecture Notes in Physics, Berlin Springer Verlag 760, 377

Spurzem, R., Berczik, P., Marcus, G., Kugel, A., Lienhart, G., Berentzen, I., Männer, R., Klessen, R., Banerjee, R. Accelerating Astrophysical Particle Simulations with programmable hardware (FPGA and GPU), Computer Science - Research and Development (CSRD), 23, 231-239 (2009)

Thompson, A. C., Fluke, C. J., Barnes, D. G., Barsdell, B. R., Teraflop per second gravitational lensing ray-shooting using graphics processing units, 2010, New Astronomy 15, 16

Wang, P., Abel, T., Kaehler, R., Adaptive mesh fluid simulations on GPU, 2010, New Astronomy 15, 581

Wong, H.-C., Wong, U.-H., Feng, X., Tang, Z., Efficient magnetohydrodynamic simulations on graphics processing units with CUDA, 2009, ArXiv e-prints arXiv:0908.4362

Xu, G., A New Parallel N-Body Gravity Solver: TPM, 1995, The Astrophysical Journal Supplement Series 98, 355

Yasuda, Koji, Journ. Comp. Chem. 29, 334, (2007)

Yang, J., Wang, Y., Chen, Y., GPU accelerated simulation, J.Comp.Phys. 221 (2007), 799

Yokota, R., Barba, L., Treecode and fast multipole method for N-body simulation with CUDA, 2010, ArXiv e-prints arXiv:1010.1482

Yokota, R., Bardhan, J. P., Knepley, M. G., Barba, L. A., Hamada, T., Biomolecular electrostatics simulation with a parallel FMM-based BEM, using up to 512 GPUs, 2010, ArXiv e-prints arXiv:1007.4591

Yoshikawa, K., Fukushige, T., PPPM and TreePM Methods on GRAPE Systems for Cosmological N-Body Simulations, 2005, Publications of the Astronomical Society of Japan 57, 849