Dynamic Load Balancing for Large Scale Particle Simulations

Sebastian Eibl[1], Florian Schornbaum[1] and Ulrich Rüde[1,2]

[1]*Lehrstuhl für Informatik 10 (Systemsimulation), Friedrich-Alexander-Universität Erlangen-Nürnberg, Cauerstr. 11, 91052 Erlangen, Germany*
[2]*CERFACS, 42 Avenue Gaspard Coriolis, 31057 Toulouse, Cedex 01, France*

Current and future applications of rigid body dynamics involve an ever growing number of particles. Example applications are in the pharmaceutical industries, chemical industries or additive manufacturing. Simulations of billions of particles can only be performed by using specialized software. This software must use today's largest supercomputers efficiently. In these highly parallel environments special attention must be paid to the load balance between the processes. Since the overall performance is dependent on the slowest process overloaded processes will lead to a decrease of the performance. Typically rigid body dynamics algorithms are parallelized by partitioning the domain spatially into subdomains. These are subsequently assigned to processes. Each process evolves the particles contained in its subdomain(s) in time. The domain partitioning must be chosen carefully. The workload associated with each subdomain is related to the particles within. Deciding on the domain partitioning at the beginning of a simulation and using it throughout the whole simulation works well for homogeneous setups where the particle density does not fluctuate much. But typical rigid body dynamics simulations involve a huge number of moving particles. This may change the workload of the subdomains constantly throughout the simulation. Therefore a static domain partitioning that achieved a good load balance at the beginning of the simulation might not be appropriate later on. To guarantee good performance over the whole simulation the software framework needs to dynamically rebalance the workload. It has to update its domain partitioning and redistribute it at regular intervals. This allows to avoid unbalanced workloads. Many popular software packages like LIGGGHTS, DEMOOP, `ls1 mardyn` already incorporate various load balancing strategies [1, 2, 3]. For fluid dynamics simulations good scaling behaviour has already been reported for diffusion based load balancing algorithms [4, 5].

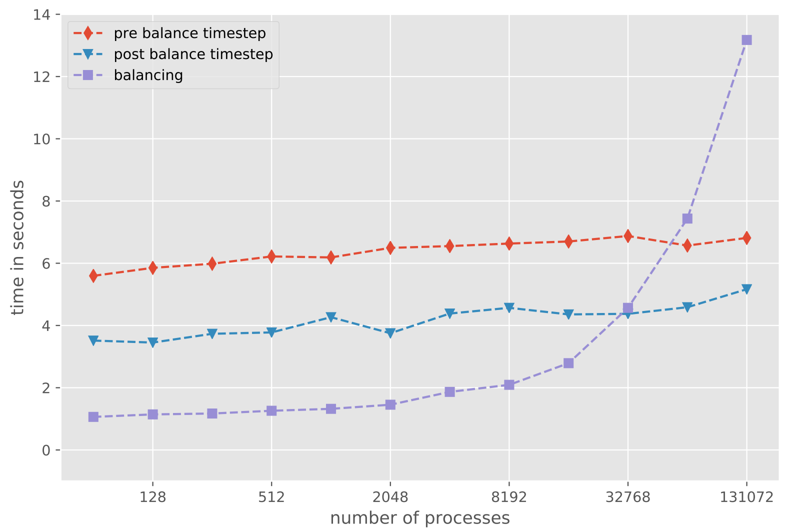

Fig. 1: Comparison of the time spent in one simulation time step before and after the load balancing step with the time needed for actual load balancing. For the load balancing a algorithm based on Hilbert space filling curves is used. The simulation setup is a box half filled with spherical particles. The number of particles together with the simulation domain size is scaled with the number of processes involved.

In this work, we study the applicability of load balancing algorithms based on space filling curves [6] and diffusion based algorithms [7] for rigid body dynamics simulations. We investigate the scaling behaviour of these algorithms on the state-of-the-art supercomputer Juqueen (TOP500 rank: 21) located at the Jlich Supercomputing Centre. This supercomputer allows simulations with more than one million processes which we use to determine the usability of these algorithms for future large scale simulations. Preliminary results for Hilbert space filling curves presented in Fig. 1 show suboptimal scaling behaviour. The performance of the load balancing starts to drop significantly above 8192 processes which prohibits scaling up to the full machine (1.8 million processes). Due to inherent global gather operations needed for space filling curves performance decreases for a large number of processes. We will also try to reproduce the results obtained for diffusive load balancing algorithms in fluid dynamics [5] in rigid body dynamics.

References

[1] R. Berger, C. Kloss, A. Kohlmeyer, and S. Pirker, "Hybrid parallelization of the LIGGGHTS open-source DEM code," *Powder Technology*, vol. 278, pp. 234–247, July 2015.

[2] D. T. Cintra, R. B. Willmersdorf, P. R. M. Lyra, and W. W. M. Lira, "A hybrid parallel DEM approach with workload balancing based on HSFC," *Engineering Computations*, vol. 33, pp. 2264–2287, Nov. 2016.

[3] A. Heinecke, W. Eckhardt, M. Horsch, and H.-J. Bungartz, *Supercomputing for Molecular Dynamics Simulations: Handling Multi-Trillion Particles in Nanofluidics*. Springer, 2015.

[4] F. Schornbaum and U. Rüde, "Massively parallel algorithms for the lattice boltzmann method on NonUniform grids," *SIAM Journal on Scientific Computing*, vol. 38, pp. C96–C126, Jan. 2016.

[5] F. Schornbaum and U. Rüde, "Extreme-scale block-structured adaptive mesh refinement," *arXiv preprint arXiv:1704.06829*, 2017.

[6] P. M. Campbell, K. D. Devine, J. E. Flaherty, L. G. Gervasio, and J. D. Teresco, "Dynamic octree load balancing using space-filling curves," *Williams College Department of Computer Science, Tech. Rep. CS-03-01*, 2003.

[7] G. Cybenko, "Dynamic load balancing for distributed memory multiprocessors," *Journal of Parallel and Distributed Computing*, vol. 7, pp. 279–301, Oct. 1989.