Extended Software Development Workshop on Load Balancing for Particle Simulations

Location: Forschungszentrum Jülich, Jülich Supercomputing Centre
Webpage: https://www.cecam.org/workshop-details/206
Dates: 6-7 September 2018 (1st part) and 3-7 June 2019 (2nd part)
Organizers: Godehard Sutmann, Burkhard Dünweg and Ignacio Pagonabarraga

1 State of the art

Parallel computing offers a large potential for scientific simulations in terms of both increasing time/length scales and reducing computational time. If the number of processors P is increasing, the computational time should ideally decrease inversely proportional to P. However, this is often not the case because the administrative overhead, which is related to communication between processors or data access from memory hierarchies is increasing, which reduces the computational efficiency. Another source of non-ideal behaviour is often found in the load imbalance of the work distribution, i.e. not all the processors have the same amount of work to do and some finish earlier than others, leading to waiting or idle times. This imbalance might have different sources, which may be due to heterogeneities in the underlying architecture, non-ideal distribution of initial work onto the processors or the dynamical evolution of the system due to inherent system dynamics. Depending on the numerical problem and the underlying parallel algorithm, the load can be recalibrated during the runtime. It is, however, crucial to minimize the overhead which is associated to the recalibrating in order to achieve any benefit in reducing the runtime, which has a direct impact onto both the computational and the energy efficiency of the code. Depending on the nature of the underlying numerical method, different types of algorithms have been proposed and implemented in selected codes. Problems related to long range interactions have been considered in some cases with graph partitioning algorithms or methods based on space filling curves. These methods come with a relatively large overhead and can be efficiently applied if the underlying problem is expensive, so that the overhead is hidden. For particle based simulations, as they often appear in soft matter and statistical physics and which evaluate short range interactions, fast solutions have to be considered which efficiently redistribute load imbalances. For this class of simulations, orthogonal recursive bisection have been considered, which come into different flavours and which can be implemented either as full correction scheme or, for some sub-classes, as iterative refinement schemes. The latter ones have the advantage that they can be applied more frequent without allowing the system to deviate too strongly from the perfect partition. At Jülich, the load balancing library ALL has been developed which implements different variants of iterative methods, which can be applied to a large number of different application scenarios. As a matter of fact, a larger part of the simulation codes, which are relevant for E-CAM have no or not sufficient abilities to redistribute dynamically the work load during the simulation and therefore, the ESDW aimed (i) in discussing technical pre-requisites and possible obstacles for integrating the library and (ii) to link the library to existing codes and extend their functionality and increase their efficiency.

2 Training provided

The ESDW has been taken place for two times so far. The first meeting in September 2018 was restricted to 2 days and was intended to get an overview over the simulation code portfolio, related to E-CAM. The codes, which were represented, were: ESPResSo (University Stuttgart, Germany), ESPResSo++(Max-Planck Institute for Polymer Research, Mainz, Germany), GC-AdResS (FU Berlin, Germany), DL_Poly (STFC, Daresbury, UK), DL_Meso (STFC, Daresbury, UK), MP2C (Forschungszentrum Jülich, Germany), IMD (University of Stuttgart), HemeLB (University College London, UK), OpenPhase (Ruhr-University Bochum). During the first meeting it turned out that more or less all codes could potentially benefit from the library. The technical requirements could be classified into those, which would need small adjustments to the codes or some more intense rewriting of conceptional design considerations. All the codes were based on orthogonal domain decomposition approaches, which also are realised in ALL. One major obstacle was the generalisation to non-static computational domain boundaries and potentially to more involved communication patterns between processors. Fortunately the library also contains implementations for the tensor product decomposition method, which can be considered as a minimal modification of a static orthogonal decomposition method. It was agreed that there is a realistic possibility for all the codes to benefit from the load balancing library. Further discussions were devoted to possible modifications of the functionality of the library and to adjustments of the interface to specific codes. The second part of the workshop was then more devoted to hands-on and practical experiences with code+ALL. The first half- and part of the second day was reserved for reports on experiences and progress towards implementations. In addition to most of the codes of the first event, another MD code (ls1-mardyn, Technical University Munich, Germany) and a Particle-in-Cell code (Ruhr-University Bochum, Germany) were represented. In addition, as guest speaker, Marta Garcia from Barcelona Supercomputing Centre gave an overview over their own approach of load balancing via a node based work redistribution on a multiprocessor node. Furthermore, Wednesday morning was reserved for a tutorial of the POP Center-of-Excellence, where participants obtained an introduction to performance analysis, which they could apply to their own codes. During the meeting, 4 codes were able to integrate parts of the library, e.g. HemeLB, ls1-mardyn and the PIC code could benefit from the iterative refinement technique of staggered mesh technique, while ESPResSo++ could integrate the tensor product decomposition method. The other codes could, at least, get a clear picture of what has to be modified, in order to get the functionality to work. This implementation, testing and benchmarking could be a topical part for a third workshop. Current limitations or obstacles are related to a too rigid initial design of the simulation codes, which have to be made more flexible towards changing computational domain geometries.

3 List of software development projects

Modules will be prepared for the following software: (1) description of the library design; (2) tensor product method; (3) staggered mesh method; (4) topological mesh method; (5) histogram based method for static load balancing; (6) interface description for connecting to external codes. All the modules are related to HPC. Industries were not represented during the meeting, but it is expected that they will profit directly, since codes like DL_Meso are used by industry.