No. 555 January 2017

Fundamentals of a numerical cloud computing for applied sciences- Preparing cloud computing for "Simulation-as-Service"

M. Geveler, S. Turek

ISSN: 2190-1767

Fundamentals of a numerical cloud computing for applied sciences – Preparing cloud computing for `Simulation-as-a-Service´

Markus Geveler*, Stefan Turek°

* markus.geveler@math.tu-dortmund.de
° ture@featflow.de

Institute for Applied Mathematics
TU Dortmund

Abstract
This paper is supposed to be used as a contribution for the `Consultation on Cloud Computing Research Innovation Challenges for WP 2018-2020´ as called for by the European Commission (DG CONNECT, unit `Cloud and software´). We propose to encourage and support fundamental interdisciplinary research for making the benefits generated by cloud computing accessible to the applied science community.

Introduction: Why cloud computing and high performance computing are contradicting

The basic idea of cloud computing (CC) is to abstract from an IT infrastructure including compute-, memory-, networking- and software resources by virtualization. These resources are made accessible to the user in a dynamic and adaptive way. The major resulting advantages compared to a specially tailored `in-house solution´ can be found in a transparent and simple usage, enhanced flexibility due to scalability and adaptivity to a specific need and finally in the increased efficiency due to savings in energy and money spent. The latter is due to scaling effects, operational efficiency, consolidation of resources and reduction of risks. The application is literally independent from any (local) data and compute resources as these can be concentrated effectively. All together, these advantages may some day supersede the traditional local / regional data center approach which can be found on the level of modern universities and research centers. From the point of view of data center management and operations, CC leads to a higher occupancy and therefore efficiency: The inevitable granularity effects that occur with medium or large workloads can be tackled with a backfilling of many small jobs. In addition, due to the fact that a specific application run's need for resources may vary from time to time, left-over capacities can be provided in a profitable `pay per use´ style.

In High Performance Computing (HPC) on the other hand, virtualization and abstraction concepts contradict the usual approaches especially in the simulation of technical processes: Here, the focus is put on enhancing the performance of an application by explicitly optimize for a certain type of hardware. This requires an a priori knowledge of the hardware which usually is given by the fact, that universities and regional research facilities have their own local or regional compute centers with comparatively static hardware components. This point of view can in some cases generate several orders of magnitude of performance gains and we call this concept hardware-oriented numerics. This paradigm comprises the simultaneous optimization for hardware, numerical and energy efficiency on all levels of application development [1,2,3,4]. One effort in hardware-oriented numerics is to optimize code and develop or choose numerical methods with respect to a heterogeneous hardware ecosystem: Multicore CPUs are as straight-forward as hardware accelerators like GPUs, FPGAs, Xeon Phi processors and system on a chip designs such as ARM-based CPUs with integrated GPUs. In addition, there are non-uniform memory architectures on the device level as well as heterogeneous communication infrastructures on the cluster level. The usual design pattern however is to optimize code for a (single) given hardware configuration where the simulation code is then optimized in a comparatively expensive way due to this proximity to hardware details. This development process is therefore the complete opposite of relying on a virtualization approach.

Today's scientific cloud computing is not feasible for numerical simulation

Up to today all efforts to make use of CC techniques in the science community can be characterized by what we call scientific cloud computing (SCC), which basically has been very successful for a specific type of application: In the scope of Big Data often a direct projection of a problem to a bag of tasks programming model can be found. Also other problems that are constituted by smaller independent tasks, where the coupling and therefore communication is minimal or zero can be coped with easily in a cloud environment. In numerical simulation on the other hand a strong coupling of the very computationally intense subproblems is the standard case. This induces a comparatively high synchronization need, requiring low communication latencies. The execution models of CC are literally blind for this type of strong coupling because the virtualization shuts down any attempt to optimize inter process communication. We believe, that the development of numerical simulation software should be characterized by the synthesis of hardware, numerical, and energy efficiency. Hence for this type of application a CC concept which takes into account the heterogeneity of compute hardware would be most feasible: According to our vision in future scenarios the user of such codes might want to choose for run time optimization in different metrics: Flexibility in the selection in which way a specific run should be allocated to a certain type(s) of compute node(s) are required. This flexibility has not been accounted for in the development of numerical code frameworks yet. A direct result of the service providers internalizing the concept of hardware-oriented numerics would be that the user of the service would be able to make an a priori choice for the core requirements for the run. For instance it could be decided whether an allocation of hardware should be made in order to minimize wall clock time or minimize energy to solution. Other hardware specifics could be made allocateable such as the type and properties of the communication links between nodes. The service would then return a number of allocations based upon available hardware. After selection, a complex optimization problem then has to be solved: The simulation software has to be able to select numerical algorithmic components that fit to this allocation and finally, a load balancing has to be performed for the individual problem to be solved.

Towards a numerical cloud computing

In order to realize this vision, there are two fundamental problems to solve:

(1) Specially tailored numerics as well as load balancing strategies as well as
(2) mapping, scheduling and operation strategies for numerical simulation have to be developed.

In (1) numerical components in a code framework have to be revisited or developed from scratch with respect to (2) by adjusting them to the respective strategies. Such numerical alternatives range from preconditioners in linear solvers to whole discretization approaches on the model level. Different hardware specific implementations have to be provided and tuned in order to enable the optimizer in (2) to succeed, which is closely related to performance engineering. This has to be undergone with respect to all levels of parallelism in modern hardware architectures and on all levels of an application.

On the other hand, the systems / strategies developed in (2) have to be sensitive for the effects of specific numerics on specific hardware. This problem is often closely related to numerical scaling, convergence and complexity theories. These theories and related skills are usually not addressed as an integral part of the training in computer science or service providers / operators. Here an automatic tuning system has to be developed that is capable of deciding what type of numerics is to be used for a given hardware allocation and which parts of the data are distributed to which part of the hardware by a static or even dynamic load balancing. The latter is an even more complex problem keeping in mind the heterogeneity even within one specific allocation, where CPUs are for instance to be saturated alongside GPUs. This optimization problem is very similar to how compilers schedule instructions on the processor level. It is also multi-dimensional, as not only raw performance has to be optimized for but also energy to solution as stated in the previous section.

Hence we emphasize that these two components, (1) and (2), cannot be brought up

independently: Specialists from the domain of applied mathematics, performance engineers and application specialists are required for the former, whereas the latter is to be coped with by computer sciences and service providers / specialists.

Conclusion

For the reasons brought up in this consulting paper, we propose to extend the flexibility of scientific cloud computing towards a numerical cloud computing in the following way: Given a heterogeneous cloud hardware architecture on the node level, with a priori unknown specifics and amount of nodes in an allocation, software frameworks for numerical simulation have to be enabled to `react´ on the properties of this allocation as described in the previous section. This would make possible the continuation of hardware-oriented numerics and thus combine the advantages of a very hardware-centric approach with those of virtualization.
For the sake of such a `Simulation-as-a-Service´ interdisciplinary research is needed that fuses knowledge in classical cloud computing approaches with experiences from the applied sciences.

Acknowledgements

This work has been supported in part by the German Research Foundation (DFG) through the Priority Program 1648 ‘Software for Exascale Computing’ (grant TU 102/48) as well as grant TU 102/50-1.

References

[1] GEVELER, M., RIBBROCK, D., GÖDDEKE, D., ZAJAC, P., AND TUREK, S. 2013. Towards a complete FEM–based simulation toolkit on GPUs: Unstructured grid finite element geometric multigrid solvers with strong smoothers based on sparse approximate inverses. Computers and Fluids 80 (July), 327–332. doi: 10.1016/j.compfluid.2012.01.025.

[2] GEVELER, M., REUTER , B., AIZINGER, V., GÖDDEKE, D., AND TUREK, S. 2016. Energy efficiency of the simulation of three-dimensional coastal ocean circulation on modern commodity and mobile processors – a case study based on the Haswell and Cortex-a15 microarchitectures. Computer Science – Research and Development, 1-10 (June). doi: 10.1007/s00450-016-0324-5.

[3] TUREK, S., BECKER , C., AND K ILIAN , S. 2006. Hardware–oriented numerics and concepts for PDE software. Future Generation Computer Systems 22, 1–2, 217–238. doi:10.1016/j.future.2003.09.007.

[4] TUREK, S., GÖDDEKE, D., BECKER , C., BUIJSSEN, S., AND WOBKER, H. 2010. FEAST —- realisation of hardware–oriented numerics for HPC simulations with finite elements. Concurrency and Computation: Practice and Experience 6 (May), 2247–2265. Special Issue Proceedings of ISC 2008. doi:10.1002/cpe.1584.