The Catwalk Project – A quick Development Path for Performance Models

According to an old legend, the inventor of chess was asked by his king, who was thrilled by the board game, to name any reward he wanted. The inventor requested that one grain of wheat should be placed on the first square of a chessboard, two grains of wheat on the second, and so forth, doubling the number of grains for every new square. Initially, the king laughed at the inventor for asking such a low price, but later made the surprising discovery that he would not even be close to be able to pay the full reward.

Today, many HPC application developers find themselves in the situation of the king when trying to scale their code to larger numbers of processors. All of a sudden, a part of the program starts consuming an excessive amount of time. Of course, in contrast to the king in our legend, computational scientists usually possess the mathematical skills to recognize a simple geometric series. On the other hand, the laws according to which the resources needed by the code change as the number of processors increases are often much more laborious to infer and also may vary significantly across individual parts of complex modular programs. This is why analytical performance modeling is rarely attempted to predict the scaling behavior before problems manifest themselves and why this technique is still confined to a small community of experts. Unfortunately, discovering latent scalability bottlenecks through experience is painful and expensive. Removing them requires not only potentially numerous large-scale experiments to track them down, prolonged by the scalability issue at hand, but often also major code surgery in the aftermath. Not infrequently, this happens at a moment when the manpower is needed elsewhere, which is especially true for applications on the path to Exascale, which have to address numerous technical challenges simultaneously, ranging from heterogeneous computing to resilience. Since such problems usually emerge at a later stage of the development process, dependencies between their source and the rest of the code that have grown over time can make remediation even harder.

If today developers decide to model the scalability of their code, and many shy away from the effort, they first apply both intuition and tests at smaller scales to identify so-called kernels, which are those parts of the program that are expected to dominate its performance at larger scales. This step is essential because modeling a full application with hundreds of modules manually is not feasible. Then they apply reasoning in a time-consuming process to create analytical models that describe the scaling behavior of their kernels more precisely. In a way, they have to solve a chicken-and-egg problem: to find the right kernels, they require a pre-existing notion of which parts of the program will dominate its behavior at scale – basically a model of their performance. However, they do not have enough time to develop models for more than a few pre-selected candidate kernels, inevitably exposing themselves to the danger of overlooking non-scalable code.

The objective of the Catwalk project is therefore to provide a flexible set of tools to support key activities of the performance modeling process, making this powerful methodology accessible to a wider audience of HPC application developers. The tool suite will be used to study and help improve the scalability of applications from life sciences, fluid dynamics, and particle physics.

The project is coordinated by Prof. Dr. Felix Wolf of the Laboratory for Parallel Programming, German

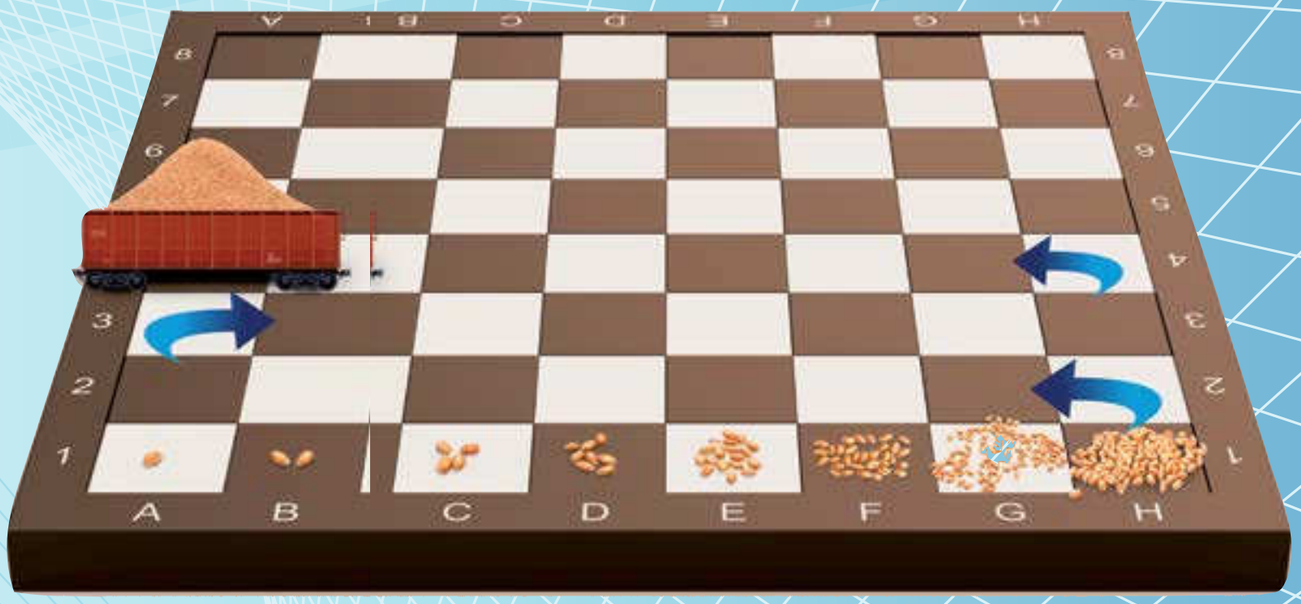

Figure 1: Many scalability bottlenecks are almost unnoticeable at lower scales but become prohibitive once the number of processes is increased beyond a certain point.

Projects

Autumn 2013 • Vol. 11 No. 2 • inSiDE

Research School for Simulation Sciences at Aachen. Further partners are the Institute for Scientific Computing of the Technische Universität Darmstadt (Prof. Dr. Christian Bischof), the Institute of Computer Systems of the Swiss Federal Institute of Technology Zurich (Prof. Dr. Torsten Hoefler), Jülich Supercomputing Centre, Forschungszentrum Jülich (Dr.-Ing. Bernd Mohr), and the Goethe Center for Scientific Computing of Goethe University Frankfurt (Prof. Dr. Gabriel Wittum).

A first result of the Catwalk project, after running for less than one year, is a novel tool that instead of modeling only a small subset of an application program manually, generates an empirical performance model for each part of the target program automatically, significantly increasing not only the coverage of the scalability check but also its speed [1]. All it takes to search for scalability issues even in full-blown codes is to run a manageable number of small-scale performance experiments, launch the tool, and compare the extrapolated performance of the worst instances to expectations. To make this possible, we exploit several assumptions:

We take advantage of the observation that the space of the function classes underlying these models is usually small enough to be searched by a computer program. An iterative refinement process maximizes both the efficiency of the search and the accuracy of our models.

We abandon model accuracy as the primary success metric and rather focus on the binary notion of scalability bugs. Similar to a thread checker, every scalability problem we identify is a success as long as false positives that send us in a wrong direction are rare. False negatives are, of course, undesirable but acceptable as long as the number of scalability bugs we find justifies the effort.

We create requirements models alongside execution-time models. A comparison between the two can illuminate the nature of a scalability problem. Also, the generation of requirements models is less affected by performance variations.

Given that our tool relies on the standard performance-measurement infrastructures Scalasaca [2] and Score-P [3], the extra software that we developed is so lightweight that it is economically feasible to provide it in production-level quality. Finally, we generate not only a list of potential bugs but also human-readable models that can be further elaborated to conduct a variety of deeper analyses such as investigating the possibility of cache spills.

This project is part of the DFG Priority Programme 1648 Software for Exascale Computing (SPPEXA). More information can be found at http://www.vi-hps.org/projects/catwalk/

References

[1] **Calotoiu, A., Hoefler, T., Poke, M., Wolf, F.**
Using Automated Performance Modeling to Find Scalability Bugs in Complex Codes, Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis (SC 2013), Denver, USA, 2013

[2] **Geimer, M., Wolf, F., Wylie, B.J.N., Ábrahám, E., Becker, D., Mohr, B.**
The Scalasca performance toolset architecture, Concurrency and Computation, Practice and Experience, 22(6):702–719, April 2010

[3] **an Mey, D., Biersdorff, S., Bischof, C., Diethelm, K., Eschweiler, D., Gerndt, M., Knüpfer, A., Lorenz, D., Malony, A.D., Nagel, W.E., Oleynik, Y., Rössel, C., Saviankou, P., Schmidl, D., Shende, S.S., Wagner, M., Wesarg, B., Wolf, F.**
Score-P: A Unified Performance Measurement System for Petascale Applications, Proceedings of the CiHPC: Competence in High Performance Computing, HPC Status Konferenz der Gauß-Allianz e.V., Schwetzingen, Germany, June 2010, pages 85–97 Gauß-Allianz, Springer, 2012

• Bernd Mohr [1]
• Felix Wolf [2]
• Alexandru Calotoiu [2]
• Torsten Hoefler [3]

[1] Jülich Supercomputing Centre

[2] German Research School for Simulation Sciences

[3] Swiss Federal Institute of Technology Zurich

Projects

Autumn 2013 • Vol. 11 No. 2 • inSiDE