Pythia: runtime decisions based on prediction (duration: 36 months)

Pythia– Common 2018 ANR call - JCJC (Jeunes chercheuses/Jeunes chercheurs)
Défi 7 – Axe 2: Sciences et technologies logicielles

Context, positioning and objectives

Context. High performance computing has become the cornerstone of many scientific fields ranging from medical research to climatology. Efficiently exploiting a supercomputer is difficult. In addition to expressing the parallelism of an application, it is necessary to manage the hardware resources of the machine. The way an application uses the hardware resources may have a significant impact on the performance: for instance, allocating memory correctly may improve an application execution time by 3.6x relative the default allocation strategy [1].

Runtime systems take the responsibility of exploiting the hardware in place of the program developer by providing an interface to the application. The task of defining efficient strategies for a particular hardware setting thus falls to the runtime, and the developer can focus on algorithmic. When selecting a strategy, runtime systems analyze how the application behaved in the past in order to predict how it will behave in the future. For example, some I/O libraries analyze the data read requests of an application in order to read the next blocks of data in advance and to improve the performance of I/O operations.

However, the runtime only reacts to the application instead of anticipating its behavior. This leads to multiple problems as some early decisions may conflict with later ones. For instance, the location where memory is initially allocated has a significant impact on further memory operations and the overhead of moving a memory page closer to a thread is significant for the application performance. Reacting to the application is also a problem for irregular applications where predicting the future behavior based on past event is difficult.

Positioning.

When deciding which execution strategy to apply, runtime systems try to predict the future behavior of an application. StarPU estimates the duration of a task to execute on a computing resource by analyzing the past execution of similar tasks [2].

The affinity between threads and their data has been extensively studied. Most work focus on detecting affinity problems in order to react to them [1,3]. Some other work proactively place data and threads in order to improve the locality of memory access. For this, predicting the affinity between threads is mandatory. This can be predicted for some programming models like OpenMP [4]. Thus, these work either react to bad locality at runtime, or statically place threads and memory at the application startup. None of these work can proactively adapt the thread placement depending on the application phase.

Prefetching disk I/O has been extensively studied for decades. The basic idea of prefetching is to predict the blocks of data that are likely to be read in the future in order to load them in advance and to overlap the disk access with computation [5–7]. While analyzing past I/O access permits to predict the future access for regular access patterns, other techniques have to be applied for irregular access patterns. Early works have focused on means for the developer to provide hints on future I/O operations to the I/O library [8]. This allows irregular access patterns to be predicted, but at the cost of a manual modification of the application. Some work try to predict the future access by speculatively executing the application in order to detect future I/O requests [9].

All the existing work either react to the application behavior by analyzing the events that happened during a temporal window, or use heuristics that apply a strategy once and for all. Instead, we propose to proactively take runtime decisions based on predictions of the future behavior of the application.

Objectives.

The main goal of the **Pythia** project is to design a framework that proactively takes decisions using an oracle that provides the runtime with predictions. By knowing the future, the runtime system can take the decisions that will minimize the application execution time.

The general idea of this project is to take advantage of the determinism of most HPC applications to predict their behavior: most applications have similar behavior from one execution to another (even when the input data changes). Thus, we propose to design a framework that will capture the application execution, analyze it, and serve as an oracle for the runtime system for future executions of the same application. When running the application again, the runtime dynamically builds a state automaton and compares it to the one provided by the oracle in order to predict the future behavior of the program.

The runtime decisions that can be taken by knowing the future are various and range from HPC-specific optimisations to more general decisions that could improve the performance of web services. Oracle-based decisions could allow runtime systems to schedule threads close to the resources they are about to use, to prefetch data from disks, to reduce the CPU frequency of a task that is not on the critical path, etc.

The Pythia tool-chain will be composed of three key components:

- a trace collection tool for capturing the application execution. The tracing tool intercepts the application calls to a set of functions (MPI functions, OpenMP parallel constructs, etc.) and record events in trace files. This tool will be based on EZTrace [10], a framework for performance analysis developed at Télécom SudParis.
- a post-mortem analysis tool that will analyze the traces, detects recurrent patterns of events, and generate a summary. The output of this tool is a state automaton that describes the successions of interactions between the application and the runtime system. We will extend a pattern discovery tool we developed in the past [11] in order to interact with the other parts of the tool-chain.
- a runtime system that uses the results of the post-mortem analysis tool to decide which strategy is the best-suited. During the execution of the application, a state automaton describing the sequence of events is generated and is compared to the automaton generated offline in order to predict the events that will happen in the near future. The runtime system can thus base its decisions on the events that are likely to happen next.

To demonstrate the applicability of the Pythia tool-chain, we will implement a set of runtime heuristics for various subsystems. While the tool-chain offers many possibilities of optimization, we will focus on two runtime decisions in this project: thread binding based on the used resources, and I/O prefetching.

Thread binding: The affinity between a thread and the resources it uses (network controllers, disks, memory banks, etc.) is critical for performance [12]. As an application runs, its threads use multiple resources that may vary from one phase to another. Thanks to its knowledge of the future behavior of a thread, the runtime system could migrate the thread close to the resource it is about to use extensively in order to improve the locality of the thread and thus the performance.

I/O prefetching Since fetching data from disk is orders of magnitude slower than accessing data from the memory, disk I/O may have a huge impact on the performance of some applications. As a result, predicting which data will be accessed in the future is critical for performance. While regular I/O access patterns can be predicted [5], irregular patterns cannot be predicted from past access only. The sequences of events from the past executions of an application could be used to predict even irregular I/O access that will be issued by a program.

Project organisation

The Principal Investigator of the Pythia project is François Trahay. He will devote eighty percent of his research time to the project. François Trahay is an associate professor at Télécom SudParis (TSP) since 2011. He is a member of the HP2 team of the computer science department, which investigates high-performance systems. He received his Ph.D. degree in computer science from the University of Bordeaux in 2009. He has been working on runtime systems for high performance computing since 2006. His research interests now mostly focus on performance analysis for HPC and distributed systems. He is the project leader and main developper of the EZTrace framework for performance analysis. He is the co-author of 19 research papers in top international conferences, workshops and journals such as IEEE TPDS, IPDPS, ICPP, IEEE Cluster, ACM TACO, or IJPP.

In addition to the principal investigator, the project will involve the HP2 team at Télécom SudParis. HP2 is composed of experts in runtime systems for high-performance computing (Dr. Élisabeth Brunet, and Dr. Amina Guermouche), and for operating systems (Prof. Gaël Thomas).

Funding For this project, we ask for 168k€. Included in this budget are: a PhD student ($\sim$ 110k€) , a large (100+ cores) NUMA machine for running experiments ($\sim$ 30k€), two annual trips to conferences for two researchers ($\sim$ 20k€), and the funding for organizing two workshops ($\sim$ 8k€).

Impact

In the short term, we expect that the result of the Pythia project will impact most HPC applications as the problematics of locality and I/O overhead is commonplace. Furthermore, this project addresses several challenges (scheduling, I/O, auto-tuning, etc.) from the Challenges of Exascale Computing [13].

In the long-term, we expect that the Pythia project will change the way runtime systems are designed, and the tool-chain will be used for other runtime decisions.

This project mainly addresses the "Défi 7 > Axe 2 > Science et Technologies logicielles" challenge. Indeed, Pythia will design mechanisms that allow runtime systems to take better decisions. While the two example runtimes that will be designed will focus on performance of HPC applications, we expect that the mechanisms provided by Pythia could be used to other domains (eg. big data) and to other means (eg. reducing energy consumption). Pythia will also address several *Technical Research Priorities* from the ETP4HPC Strategic Research Agenda [14] including priority #2 System software and management, and priority #3 programming environment.

The investigator of the project will continue his commitment to open-source as all the tools developed during the project will be freely available as open-source software. Besides, the software will be maintained beyond the end of the project and efforts will be made to bootstrap a visible and active community of users and contributors, through appropriate channels and organizations.

We expect that the outcome of the project will trigger collaborations or enforce existing ones. Pythia could be used as part of the new collaborations that we started with INSA Lyon and CEA. François Trahay is currently initiating a collaboration with DDN, the leading company on I/O architectures for HPC. We expect that Pythia could be part of a supplementary relationship with DDN. Finally, François Trahay already worked on parallel I/O [7] with researchers for the Southwest University in Chongqing (China). Pythia could be used for continuing such international collaborations.

The scientific results of this project will be published in major conferences and journals related to high-performance computing, operating systems, distributed and parallel systems, performance analysis such as IPDPS, ASPLOS, or SOSP.

References

[1] M. Dashti, A. Fedorova, J. Funston, F. Gaud, R. Lachaize, B. Lepers, V. Quema, and M. Roth, "Traffic management: a holistic approach to memory placement on numa systems," in *ACM SIGPLAN Notices*, vol. 48, no. 4. ACM, 2013, pp. 381–394.

[2] C. Augonnet, S. Thibault, and R. Namyst, "Automatic calibration of performance models on heterogeneous multicore architectures." in *Euro-Par Workshops*, 2009, pp. 56–65.

[3] M. Diener, E. H. Cruz, P. O. Navaux, A. Busse, and H.-U. Heiß, "kmaf: Automatic kernel-level management of thread and data affinity," in *PACT*, 2014.

[4] F. Broquedis, O. Aumage, B. Goglin, S. Thibault, P.-A. Wacrenier, and R. Namyst, "Structuring the execution of openmp applications for multicore architectures," in *IPDPS 2010*.

[5] P. Cao, E. W. Felten, A. R. Karlin, and K. Li, "Implementation and performance of integrated application-controlled file caching, prefetching, and disk scheduling," *ACM Transactions on Computer Systems (TOCS)*, vol. 14, no. 4, pp. 311–343, 1996.

[6] X. Ding, S. Jiang, F. Chen, K. Davis, and X. Zhang, "Diskseen: Exploiting disk layout and access history to enhance i/o prefetch." in *USENIX ATC*, 2007.

[7] J. Liao, F. Trahay, B. Gerofi, and Y. Ishikawa, "Prefetching on storage servers through mining access patterns on blocks," *IEEE Transactions on Parallel and Distributed Systems*, vol. PP, no. 99, 2015.

[8] R. H. Patterson, G. A. Gibson, E. Ginting, D. Stodolsky, and J. Zelenka, "Informed prefetching and caching," in *SOSP'95*, 1995, pp. 79–95.

[9] F. Chang and G. Gibson, "Automatic i/o hint generation through speculative execution." USENIX, 1999.

[10] F. Trahay, F. Rue, M. Faverge, Y. Ishikawa, R. Namyst, and J. Dongarra, "EZTrace: a generic framework for performance analysis," in *CCGrid*, 2011.

[11] F. Trahay, E. Brunet, M. Mosli Bouksiaa, and J. Liao, "Selecting points of interest in traces using patterns of events," in *PDP*, 2015.

[12] S. Moreaud and B. Goglin, "Impact of numa effects on high-speed networking with multi-opteron machines," in *PDCS*, 2007.

[13] S. Ashby, P. Beckman, J. Chen, P. Colella, B. Collins, D. Crawford, J. Dongarra, D. Kothe, R. Lusk, P. Messina *et al.*, "The opportunities and challenges of exascale computing," *Summary Report of the Advanced Scientific Computing Advisory Committee (ASCAC) Subcommittee*, pp. 1–77, 2010.

[14] ETP4HPC, "ETP4HPC strategic research agenda 2015 update."