Inter-process trace compression

Keywords : performance analysis, HPC, exascale

Context

High performance computing has become the cornerstone of many scientific fields ranging from medical research to climatology. Efficiently exploiting a supercomputer is difficult. Recent developments in the software and hardware stacks of supercomputers have made them difficult to operate. Due to the complexity of hardware architectures, numerous effects have a strong influence on performance (NUMA effects, cache effects, vectorization, etc.). In addition, supercomputers are now heterogeneous and applications now have to exploit several types of processing units (CPUs, GPUs, ...), increasing the number of component that may degrade performance.

Developpers thus rely on performance analysis tools to assist them in optimizing their application [1, 2, 3]. These tools collect performance data during the execution of the application and generate trace files that can be analyzed post-mortem. However, when running an application at a large scale (typically thousands of CPUs), each thread will generate performance data, and the resulting trace will be so large that it cannot be processed.

A prototype [4] trace format exploits the repetitive nature of most parallel applications [5] to perform on the fly data compression. It detects at runtime sequences of events that repeat and replaces them with meta-events. As a result, each thread of the application is described by a *grammar* (see Fig. 1) that depicts the sequences of events that occured. Moreover, the timestamps of each event, and the durations of each sequence are stored in separate files in order to analyze the performance of the application. This prototype can summarize the behavior of an application. It scales as the number of events per thread increases.

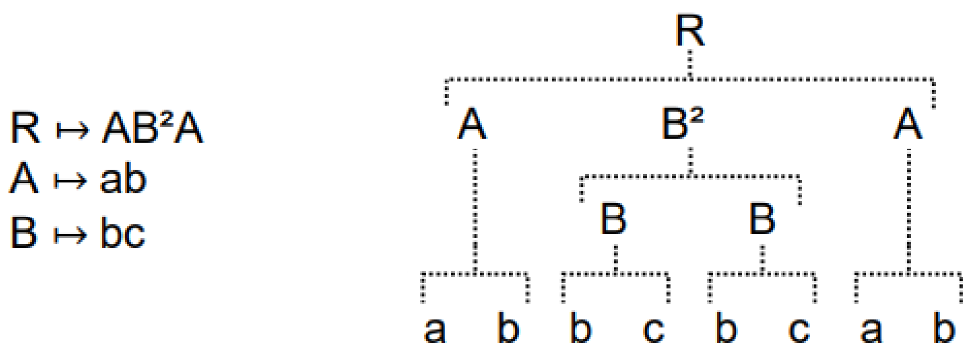

Fig. 1: Unfolding of a grammar representing the trace "$abbcbcab$".

Research project

As the number of thread of an application increases, the traces generated by our prototype will grow linearly. This is a scaling problem for large scale systems that may consist of thousands of threads.

The goal of the project is improve the scalability of a tracing tool when the number of threads increase. To do that, we plan to study inter-trace similarity detection. Indeed, most HPC applications are "regular" [6]. The MPI processes run the same program, and the generated trace of threads are mostly the same. As a result, the detected grammars (see Fig. 1) contain similar rules, with some events that may slightly differ.

The main steps of this project are:

- Study the HTF tracing library
- Compare and detect similar events by computing a distance metric (eg. Hamming distance)
- Perform inter-trace compression
- Analyze the scalability of inter-trace compression on various real life applications

Contact

François Trahay francois.trahay@telecom-sudparis.eu
Benagil group
Télécom SudParis

References

[1] Geimer, M., et al. "The Scalasca performance toolset architecture." Concurrency and computation: Practice and experience, 22(6), 702-719, 2010.
[2] F. Trahay, et al. "EZTrace: a generic framework for performance analysis" CCGrid 2011.
[3] Shende, S. S., & Malony, A. D. "The TAU parallel performance system." The International Journal of High Performance Computing Applications, 20(2), 287-311, 2006.
[4] https://github.com/trahay/Hierarchical-Trace-Format
[5] F. Trahay, et al. "Selecting points of interest in traces using patterns of events." PDP 2015.
[6] Colin, A., et al. "PYTHIA: an oracle to guide runtime system decisions." CLUSTER 2022.