Towards Cross-stack Dynamic Resource Affinity Management

Balazs Gerofi
RIKEN Center for Computational Science
JAPAN
bgerofi@riken.jp

1 INTRODUCTION AND MOTIVATION

Recent years have brought an explosion in workload diversity deployed in high performance computing (HPC) environments. As opposed to the classical HPC simulations, which stay rather static throughout their lifetimes, these emerging applications utilize various different runtime systems (e.g., implicitly via libraries), rely on auxiliary helper threads, and often dynamically adjusting their resource requirements. Furthermore, different application components are increasingly composed into workflow type of execution.

The primary issue is that most components assume full control over compute resources, leading to oversubscription and unbalanced resource distribution across the participating runtime pieces. There is a need for the software stack to provide standard facilities so that dynamic runtime components can synchronize their resource usage. This poster addresses this issue and lays out our vision for a cross-layer resource affinity management system. We describe some of our motivating application use-cases and introduce the design of the mapping coordinator.

2 EXAMPLE USE-CASES

We have collected a number of real-world application use-cases and organized them along their common properties.

2.1 Dynamic Tasks Performed by a Single Application

A number of applications spawn part of their runtime components on demand depending on the given execution phase. One example is Tensorflow using Intel's DNN library. While internal thread pools of the Tensorflow engine use native pthreads the DNN library is parallelized using OpenMP. No standard interfaces allow control and co-ordination among multiple thread groups of an application.

2.2 Utility Threads performed by Auxiliary Libraries

Another important scenario is auxiliary or utility threads spawned by runtime engines (e.g., the asynchronous communication threads of the MPI library). Unless one ensures that utility threads are placed to adequate compute resources, they may interfere with compute threads of the application adversely impacting overall performance. One example is the GeoFEM application from The University of Tokyo which utilized asynchronous collective operations in MPI.

2.3 Rebalancing MPI and OpenMP workers

An Arbitrary Lagrangian-Eulerian (ALE) hydrodynamics simulation from the French Alternative Energies and Atomic Energy Commission (CEA) relies on multi-phase computation. While one phase can be efficiently parallelized using MPI, the other one is more efficient using all node resources for an OpenMP region in a single process address space. In order to achieve this node resources need to be reserved and released between phases so that the application can rearrange its CPU usage.

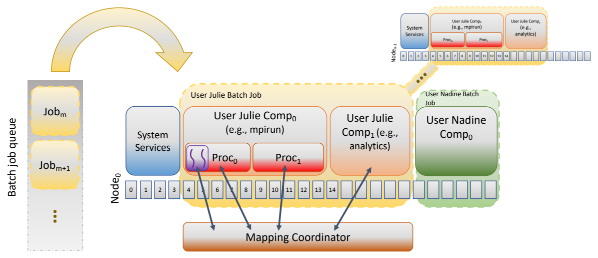

Figure 1: Overview of the Mapping Coordinator and its relation to the typical HPC software track.

3 DESIGN

Non of the above mentioned use-cases can be easily deployed on the currently available HPC software stack. In order to address these issues we are designing a software component that keeps track and orchestrates resource usage of various runtime components and enables interaction among components to dynamically reconfigure their allocation. We call this the *Mapping Coordinator*, shown in Figure 1. The poster will provide further details of its functionality as well as an overview of the proposed APIs.

4 RELATED WORK

Various tools provide extended functionalities to the basic Linux resource orchestration mechanisms utilized by parallel applications [1–4]. Our effort embraces most of these with the aim of a more general solution.

REFERENCES

[1] François Broquedis, Jérôme Clet-Ortega, Stéphanie Moreaud, Nathalie Furmento, Brice Goglin, Guillaume Mercier, Samuel Thibault, and Raymond Namyst. 2010. Hwloc: a Generic Framework for Managing Hardware Affinities in HPC Applications. In *Proceedings of the 18th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP2010).* IEEE Computer Society Press, Pisa, Italia.
[2] Balazs Gerofi, Rolf Riesen, and Yutaka Ishikawa. 2018. Making the Case for Portable MPI Process Pinning. https://eurompi2018.bsc.es/sites/default/files/uploaded/EuroMPI2018_paper_40.pdf Poster presented at the 25th European MPI Users' Group Meeting, EuroMPI 2018, Barcelona.
[3] Edgar A. León. 2017. mpibind: A Memory-Centric Affinity Algorithm for Hybrid Applications. In *International Symposium on Memory Systems (MEMSYS'17).* ACM, Washington, DC.
[4] J. Treibig, G. Hager, and G. Wellein. 2010. LIKWID: A lightweight performance-oriented tool suite for x86 multicore environments. In *Proceedings of PSTI2010, the First International Workshop on Parallel Software Tools and Tool Infrastructures.* San Diego CA.