Slurm PMIx Extension for Dynamic MPI Processes in a Plan-Based Environment

René Pascal Becker
becker29@zedat.fu-berlin.de

I. Exposé

According to SchedMD, a wide variety of High Performance Computing (HPC) systems uses their Simple Linux Utility for Resource Management (SLURM) [1]. Companies worldwide rely on the workload management of SLURM for their HPC systems. However, SLURM itself cannot guarantee the completion time of computations.

Burchard et al. [2] proposed an architecture for resource management that is aware of Service Level Agreements (SLAs). This architecture can enable plan-based scheduling on HPC systems to ensure completion times. Ongoing research tries to integrate SLURM into the SLA-aware architecture. In the following, we will refer to the architecture as a Virtual Resource Manager (VRM) because it acts as a meta-scheduler for multiple sites.

A. Applications in an HPC System

Users that want to run a parallel program need to use a Message Passing Interface (MPI) implementation to run the program across multiple compute nodes in an HPC system. These nodes are individual computers designated for computations in an HPC system. The MPI library is responsible for the communication between individual processes over different network types and partially for network fault handling [3]. To communicate with the system management stack, MPI libraries may make use of a Program Management Interface (PMI) library such as PMIx [4].

Different implementations of MPI and PMI exist. The PMIx library aims to be suitable for exascale and to eliminate orchestration issues in PMI2 [4]. The associated PMIx standard is actively worked on as the latest PMIx 5.0 Specification release shows [5]. OpenMPI is an actively worked on, open-source MPI implementation adaptable by its modular design [3]. It is capable of transparently dealing with network failovers. Users may choose between different supported implementations [6].

OpenMPI supports the MPI2 specification. For this reason, a user can spawn dynamic processes within an MPI process [7]. Dynamic processes are helpful for any application with dynamic behaviour.

B. Communication with the VRM

The VRM requires control, information and modification capabilities for SLURM to enforce plan-based scheduling. [8] and [9] target these aspects and form the basis of the thesis. A previous software project combined the work and proved that a mocked VRM could influence the MPI rank of individual processes. The work uses a modified version of OpenMPI that modifies the process ranks. The test case uses the PMIx support of OpenMPI.

C. Subject of the Thesis

SLURM has to implement PMI support itself because PMI requires the system management stack to support the interface. Currently, SLURM does not support dynamic process spawning via PMIx [6]. This thesis aims to enable support for dynamic processes in SLURM with PMIx in a plan-based environment. A possible implementation must consider the enforced restrictions by the VRM and act accordingly.

D. Organization

Regarding the above, the thesis will provide background information on the current state of the work on the plan-based environment, the PMIx communication schema, SLURM’s PMIx module and MPI2 dynamic process spawning. We will analyze the requirements we must consider for the implementation in the context of the plan-based environment, PMIx specification and SLURM. The thesis will then present the implementation and its embedding in the SLURM ecosystem. The implementation will then undergo testing against the previously introduced requirements. We will present the results and discuss them if necessary. Afterwards, the thesis proposes possible points of interest for future work.

REFERENCES