Parallel Computing and I/O

Message passing safety and correctness checks at compile time using Rust

- Author: Michael Blesel
- Type: Master's Thesis
- Date: 2021-02-15
- Reviewers: Jun.-Prof. Dr. Michael Kuhn, Prof. Dr. Thomas Ludwig
- Supervisors: Jun.-Prof. Dr. Michael Kuhn, Jannek Squar
- Download: PDF

Message passing is the de facto standard for large scale parallelization of applications in the domain of high performance computing. With it comes a lot of complexity and possible parallelization errors. Traditional programming languages like C and Fortran and the message passing interface MPI do not provide many compile time correctness checks for such parallel applications. This thesis shows the design and implementation of a new message passing library using the Rust programming language. Rust focuses heavily on memory safety features and rigorous compile time correctness checks. The thesis explores whether a message passing library can be designed in a way that utilizes these features to provide an easier experience for developers of HPC software by conducting better compile time correctness checks. Many possible errors in MPI software are related to unsafe memory usage. Rust's memory ownership concept can be applied to message passing operations to provide better error protection. Furthermore modern language features like generic programming can be used to achieve a more convenient and safer interface for message passing functions. The thesis demonstrates that some common erroneous code patterns of message passing with MPI can be avoided with this approach. It however also shows that particular error classes regarding the correctness of the communication schemes of message passing applications require further correctness checking tools, such as static analysis or compiler modifications.