Hessisches Kompetenzzentrum
für Hochleistungsrechnen

Tracking Type Information to Dynamically Find Type Related Bugs in MPI Applications

Project Manager
Dr. Alexander Hück

Principal Investigator
Prof. Dr. Christian Bischof

Project Term
2018 - 2019

Clusters
Lichtenberg Cluster Darmstadt

Additional Software
MUST, Clang/LLVM, CORAL, Spec

Institute
Department of Computer Science

University
Technische Universität Darmstadt

Introduction

MUST, a dynamic MPI correctness checker, is extended with a type and memory allocation tracking sanitizer called TypeART for C/C++ codes based on the LLVM compiler framework. The sanitizer extracts type information and inserts instrumentation to track memory allocations and deallocations relevant to MPI communication. This allows MUST to check for type compatibility between the type-less communication buffer and the declared MPI datatype at all phases of the MPI communication, i.e., (1) message assembly, (2) message transfer, and (3) message disassemble into the buffer of the receiving process. Previously, MUST was only able to check phase 2, as it works by overloading the MPI calls in target program. The results show that our approach typically exhibits acceptable runtime and memory overheads.

Methods

We evaluate our approach on benchmarks taken from SPEC MPI 2007 (104.milc and 122.tachyon) and two Coral mini applications (LULESH 2.0 and amg2013). We measured the memory and runtime impact of our sanitizer tool and compared the results with the unmodified benchmark codes to assess the overall impact of our approach.

Results

Our empirical study of the TypeART extension focused on

1. coverage and correctness of the memory instrumentation pass,

2. standalone runtime and memory overheads induced by the required type tracing, and,

3. the overall impact w.r.t. runtime and memory of TypeART when fully integrated into MUST, including MPI type checks.

Our results, overall, demonstrated the feasibility of our approach — all relevant memory allocations were instrumented in the target applications, and the induced overheads are reasonable. However, the impact is dependent on the target programs behaviour. Namely, how many memory allocation calls are executed during normal program execution. In particular, amg2013 had over 25 million heap allocations that were tracked by TypeART compared to less than a million for the other applications. Hence, the induced runtime overhead was approximately at a factor of 3x. However, for all other tested applications, the runtime overhead was less than 1.5x. Memory overhead, on the other hand, was less than 1.2x for all tested applications.

Discussion

We presented TypeART, a sanitizer tool, based on the LLVM compiler framework. It tracks runtime type information for memory allocations relevant in MPI communication as an extension to the MPI correctness tool MUST. The observed overhead of our implementation is reasonable and extends the MUST tool by providing type-related information of the type-less void buffer passed to the MPI library.

Publications

A. Hück et al., ”Compiler-aided Type Tracking for Correctness Checking of MPI Applications, ”2018 IEEE/ACM 2nd International Workshop on Software Correctness for HPC Applications (Correctness), Dallas, TX, USA, 2018, pp. 51-58. https://doi.org/10.1109/Correctness.2018.00011

Last Update: 2021-07-27 19:49