Hessisches Kompetenzzentrum
für Hochleistungsrechnen

Fault Tolerance and Locality-Aware Work Stealing for Dynamically Generated Dependent Tasks on Clusters



Project Manager
Mia Reitz

Researchers
Rüdiger Nather

Principal Investigator
Prof. Dr. Claudia Fohry

Project Term
2023 - 2024

Clusters
Lichtenberg II Cluster Darmstadt

Additional Software
OpenMPI, Hazelcast

Institute
Research Group Programming Languages / Methodologies

University
University Kassel

JUSTUS-LIEBIG-UNIVERSITÄT GIESSEN
TECHNISCHE UNIVERSITÄT DARMSTADT
UNIKASSEL VERSITÄT
Philipps Universität Marburg

Introduction

Programming environments for today's supercomputers must support the design of efficient programs and handle issues such as: programmer productivity, i.e., the human effectiveness in writing programs; application irregularity, i.e., the limited planability of the computation; and fault tolerance, i.e., the ability to cope with hardware failures during program execution. While these issues are hard to achieve with traditional parallel programming environments, a promising paradigm to tackle all of them together is Asynchronous Many-Task (AMT) programming. Here the computation is coded into many small work packages (tasks), which may depend on each other. The tasks are processed by a limited number of workers (e.g. processes), which often balance their load via work stealing. AMT programming environments enjoy growing popularity on single multicore computers, where they provide powerful functionalities such as dynamic task generation at runtime to facilitate the expression of irregularity, and load balancing via work stealing to improve resource utilization.
On large supercomputers, i.e., clusters of such machines, in contrast, AMT functionalities are still limited. A major hurdle for AMT deployment there is the need to combine load balancing with low communication costs. In particular, tasks should run close to their data, which is denoted as locality. Similarly, fault tolerance takes on crucial importance in clusters, due to their larger number of hardware components. Although AMT is potentially well-suited to provide fault tolerance, concrete algorithms and techniques are still rare.

Methods

In this project, we have targeted the realization of fault tolerance. In a first step, we have proposed a novel fault tolerance scheme for an existing cluster AMT library that supports tasks that depend on their child task results. Further, we have proposed a formula for predicting running times of our fault tolerance scheme in case of multiple worker failures.
As a basis for our future studies, we have then developed a prototypical cluster AMT library that captures the primary features of current single-machine environments in a simple form: dynamic task generation at runtime, global work stealing, and dependent tasks (realized with the future construct).

Results

We have evaluated our fault tolerance scheme in experiments with four benchmarks on up to 1536 cores on the Lichtenberg cluster, and observed protection overheads of up to 43.98% and negligible recovery costs. Moreover, we have experimentally validated our running time predictions of our fault tolerance scheme. We have observed an error of up to 1.11% compared to estimated running times for up to 32 worker failures.
To evaluate our prototypical AMT library, we have experimentally compared it to a similar, but less flexible library, in four benchmarks and up to 1536 cores. Depending on the benchmark, our results have shown that our library outperforms

JUSTUS-LIEBIG-UNIVERSITÄT GIESSEN
TECHNISCHE UNIVERSITÄT DARMSTADT
UNIKASSEL VERSITÄT
Philipps Universität Marburg

the less flexible library by up to 5%.

Discussion

The protection overheads of our fault tolerance scheme are lower than overheads of a traditional and established approach. We expect that the overheads can be reduced in future work, e.g., through low-level optimizations and algorithmic improvements.
Our experimental comparison of our prototypical AMT library has shown that the running time differences compared to another similar library are small, but our experiments so far have been limited. That said, the results suggest that our AMT library can keep up to a less flexible one in performance. In future research, we will build on our fault tolerance scheme to provide fault tolerance to our flexible AMT library. This new fault tolerance scheme considers hard failures, i.e., the loss of one or multiple processes. Methodically, the next project includes the development of novel algorithms and other techniques to realize soft- and hard failure protection and locality optimization, their implementation in the prototypical AMT environment, and their experimental evaluation.

Publications

Reitz, M., Fohry, C. Task-Level Checkpointing and Localized Recovery to Tolerate Permanent Node Failures for Nested Fork–Join Programs in Clusters. SN COMPUT. SCI. 5, 320 (2024) https://doi.org/10.1007/s42979-024-02624-8

Nather, R.; Reitz, M; Fohry, C. Distributed, Resilient and In-Memory Storage of Key-Value Data for HPC, Paralel Data Systems Workshop at SC24, Atlanta, USA, November 17 – 22 (2024)

Last Update: 2024-12-10 15:55