Parallelizing the LM OSEM Image Reconstruction on Multi-Core Clusters

Philipp CIECHANOWICZ, Philipp KEGEL, Maraike SCHELLMANN,
Sergei GORLATCH, and Herbert KUCHEN
University of Münster, Germany

Abstract. In this paper we present four different parallel implementations of the popular LM OSEM medical image reconstruction algorithm. While two of them use libraries such as MPI, OpenMP, or Threading Building Blocks (TBB) directly, the other two implementations use algorithmic skeletons of the Münster Skeleton Library *Muesli* to hide the parallelism. We compare the implementations w.r.t. runtime, efficiency, and programming style and show the resulting benchmarks which have been conducted on a multi-processor, multi-core cluster computer.

Keywords. algorithmic skeletons, Muesli, MPI, OpenMP, TBB, medical imaging

Introduction

Writing a parallel program for a multi-processor computer can be a complicated task. When using communication libraries such as MPI directly, the abstraction level is rather low. The programmer has to think about decomposing the problem, integrating the partial solutions, and bother with communication problems. More recently, multi-core processor architectures have evolved. While this provides additional parallelism, this also increases the complexity of writing efficient programs. Although it is comparatively clear how to make use of multi-processor architectures, this is not yet the case for multi- and especially many-core architectures. In this paper we evaluate four different parallel implementations of the popular LM OSEM image reconstruction algorithm and compare them w.r.t. to runtime, efficiency, and programming style. While two of our implementations use MPI, OpenMP, and TBB directly, the other two use the Münster Skeleton Library *Muesli* as a framework for hiding the parallelism.

The remainder of this paper is structured as follows: Section 1 introduces some fundamentals. The LM OSEM image reconstruction algorithm is presented in detail in Section 2. Then, Section 3 discusses our four parallel implementation alternatives. The resulting benchmarks are shown in Section 4. Finally, Section 5 concludes with related work and gives an outlook to future work.

1. Fundamentals

Hybrid MPI-OpenMP programming is a popular approach to exploit parallelism on clusters with multiple processors per node. MPI facilitates inter-node communication and