Implementing Fusion-Equipped Parallel Skeletons by Expression Templates

Kiminori Matsuzaki\textsuperscript{1} and Kento Emoto\textsuperscript{2}

\textsuperscript{1} School of Information, Kochi University of Technology, Japan
matsuzaki.kiminori@kochi-tech.ac.jp
\textsuperscript{2} Graduate School of Information Science and Technology, University of Tokyo, Japan
emoto@ipl.t.u-tokyo.ac.jp

Abstract. Developing efficient parallel programs is more difficult and complicated than developing sequential ones. Skeletal parallelism is a promising methodology for easy parallel programming in which users develop parallel programs by composing ready-made components called parallel skeletons. We developed a parallel skeleton library SkeTo that provides parallel skeletons implemented in C++ and MPI for distributed-memory environments. In the new version of the library, the implementation of the parallel skeletons for lists is improved so that the skeletons equip themselves with fusion optimization. The optimization mechanism is implemented based on the programming technique called expression templates. In this paper, we illustrate the improved design and implementation of parallel skeletons for lists in the SkeTo library.

Keywords: Skeletal parallelism, fusion transformation, list skeletons, expression templates, template meta-programming.

1 Introduction

Hardware environments for parallel computing are now widely available. The popularization and growth of multicore CPUs call for more parallelism to utilize the potential of the hardware. Developing parallel programs, however, is more difficult and complex than developing sequential ones due to, for example, data distribution, communication, and load balancing.

Skeletal Parallelism \cite{1} is a promising methodology for this problem. In the skeletal parallelism, parallel programs are developed by composing ready-made components, called parallel skeletons, which are abstract computational patterns often used in parallel programs. Parallel skeletons conceal many details of parallelism in their implementation and, thus, allow for the development of parallel programs as if they were sequential programs. This paper considers parallel skeletons for data-parallel computations in which large amounts of data are processed in parallel.
Our group has intensively studied skeletal parallelism for data-parallel computations since the late 90’s. We have developed several methods for deriving skeletal parallel programs and for optimizing skeletal programs using fusion transformation [2] based on the constructive algorithmic theory [3]. To make these results easily available, we have developed a parallel skeleton library named SkeTo [4]: the name is from the abbreviation of Skeleton Library in Tokyo and it also means helper or supporter in Japanese. Three important features of the SkeTo library are:

- The library is implemented in standard C++ and MPI (Message Passing Interface), and we can widely use the library on distributed-memory environments as well as shared-memory ones. Users who know C++ can use the library without learning another language or library for parallel programming.
- The library provides parallel skeletons for data-parallel computation. Supported data structures are lists (one-dimensional arrays), matrices (two-dimensional arrays), and trees. Parallel skeletons over these data structures have similar interfaces.
- The library provides a mechanism of optimizing skeletal programs based on fusion transformation [5].

The SkeTo library version 0.3beta was released in January 2007. After this release, some problems were found that needed be resolved and the new version 1.0 of the SkeTo library was developed. Two important improvements of the library are:

- With the old version, users had to select the proper skeleton for their specific situation (e.g., a skeleton to overwrite lists). In the new version, selections are automatically done by the library.
- In the old version, the fusion optimization was implemented in OpenC++ [6]—a meta-programming language for C++. OpenC++ is now obsolete. In the new version, the fusion optimization mechanism is implemented using standard C++ in conjunction with the meta-programming technique called expression templates [7]. In addition, more powerful fusion rules than those of the old version are implemented.

The SkeTo library is available for several environments. In terms of the OS, it is available for Linux, Mac OS X, and Windows with cygwin; in terms of the Compiler, it is available for GCC versions 3.4 and 4.3, and Intel Compilers 9.1 and 11.1; in terms of the MPI library, it is available for mpich and OpenMPI.

This paper discusses the design and the implementation of the parallel list skeletons in the SkeTo library. The focus is on the self-optimization mechanism implemented with expression templates. The rest of the paper is organized as follows. Section 2 presents the sequential and the parallel definitions of the list skeletons provided in the SkeTo library and discusses how to optimize skeletal programs using fusion transformation. The implementation of the parallel list skeletons is discussed in Section 3. Section 4 evaluates the performance of the SkeTo library using two examples. Related work is reviewed in Section 5 and concluding remarks are presented in Section 6.