Chapter 6
Practical Examples of Automated Development of Efficient Parallel Programs

ABSTRACT

In this chapter, some examples of application of the developed software tools for design, generation, transformation, and optimization of programs for multicore processors and graphics processing units are considered. In particular, the algebra-algorithmic-integrated toolkit for design and synthesis of programs (IDS) and the rewriting rules system TermWare.NET are applied for design and parallelization of programs for multicore central processing units. The developed algebra-dynamic models and the rewriting rules toolkit are used for parallelization and optimization of programs for NVIDIA GPUs supporting the CUDA technology. The TuningGenie framework is applied for parallel program auto-tuning: optimization of sorting, Brownian motion simulation, and meteorological forecasting programs to a target platform. The parallelization of Fortran programs using the rewriting rules technique on sample problems in the field of quantum chemistry is examined.

INTRODUCTION

In this chapter, the developed software tools described in Chapter 5 (the algebra-algorithmic integrated toolkit for design and synthesis of programs (IDS), the rewriting rules system TermWare and the auto-tuning framework

DOI: 10.4018/978-1-5225-9384-3.ch006

TuningGenie) are applied for design, generation and transformation of sample programs for multicore processors and graphics processing units. IDS toolkit (Andon, Doroshenko, Tseytlin, & Yatsenko, 2007; Doroshenko, Ivanenko, Ovdii, & Yatsenko, 2016; Doroshenko, Zhereb, & Yatsenko, 2013) uses algebraic specifications based on Glushkov's system of algorithmic algebra (SAA) (see Chapter 1), which are represented in a natural linguistic form, namely, the algorithmic language SAA/1 considered in Chapter 2. IDS is based on the method of dialogue design of syntactically correct algorithm schemes, which eliminates syntax errors during construction of algorithm specifications . To automate parallelizing and optimizing transformations of programs being designed, the rewriting rules system TermWare (Doroshenko & Shevchenko, 2006) is used. TuningGenie framework (Ivanenko, Doroshenko, Zhereb, 2014) is applied to automate the adjustment of programs to a target computing environment. The application of the mentioned software tools is illustrated by the development of programs in various subject domains (sorting, meteorological forecasting, quantum chemistry and other) written in C#, Java and Fortran languages.

It should be noted that despite being one of the first programming languages, Fortran is still widely used, in particular, for solving scientific and engineering computation-intensive problems. Its popularity is due to its relative simplicity and lack of complex facilities (e.g., pointers), closeness to mathematical description of a problem and efficiency of generated binary code. Another reason for continued use of Fortran is that in more than 50 years of its existence, a vast repository of programs, libraries and routines for solving different scientific problems has been developed. Algorithms implemented in such programs are still valuable, however, there is a need to adapt this legacy code to new parallel computing platforms. Furthermore, due to a size and a complexity of existing code, manual adaptation is not a practical option: there is a need for automated tools to facilitate conversion of legacy code to modern parallel platforms (Buttari et al., 2007).

There has been an extensive research in the area of parallelizing existing sequential code, in particular, for multicore architectures. Some approaches require manual code modification and provide facilities that help a developer to express the parallelism. Such approaches include parallel libraries (Leijen, Schulte, & Burckhardt, 2009), parallel extensions to existing languages ("OpenMP Application Programming Interface", 2015) and new parallel languages (Saraswat, Sarkar, & von Praun, 2007). Another research direction