Comparison of Data Dependence Analysis Tests

Miia Viitanen and Timo D. Hämäläinen

Tampere University of Technology, Institute of Digital and Computer Systems,
P.O. Box 553, FIN-33101, Tampere, Finland
timo.d.hamalainen@tut.fi

Abstract. Comparison of six data dependence analysis algorithms is presented. The algorithms are purposed for a parallel compiler that is being developed for a configurable multi-DSP system PARNEU. The algorithms are implemented in SUIF compiler framework and benchmarked with Perfect Club, Audio Signal Processing, and Media Bench test problems. Proprietary PARNEU programs that have been manually parallelised are also included. Performance in terms of accuracy and execution time of the data dependence algorithms has been measured and compared. The results show that the Omega test is the most accurate but also takes most execution time for benchmarks with for-loop parallelism.

1 Introduction

Parallel processing has traditionally been used in scientific computations in large-scale computers but now increasingly in embedded systems. This is reflected on inherently parallel, embedded processors as well as System-on-Chip implementations. However, the utilisation of parallel architectures has several difficulties, of which programming is among the worst one. This is further emphasized in configurable architectures [15].

In an ideal case, a parallel compiler takes care of the application algorithm mapping to the architecture. Such compilers have been studied intensively for parallel computers that have much resources and homogeneous architecture [2][14]. Unfortunately, those are not directly suitable for typical embedded parallel architectures [6].

Our research focuses on development of parallel compiler for a DSP based system called PARNEU [9]. It was developed for signal-processing applications, and it consists of a master and a scalable number of slave processors as well as configurable communication nodes also capable of performing simple operations. After several manual mappings of algorithms, the goal is to make the process automated.

As an essential part of this work, this paper presents results for implementation and comparison of several data dependence analysis tests. The goal is to find out the accuracy differences between the analysis algorithms with many different problems. Data dependence analysis is not only essential for automatic detection of parallelism, but also necessary for many other important compiler transformations that improve memory locality, load balancing, and reduce the overhead due to task initiation and synchronization.
In the next two sections, we briefly introduce the PARNEU compiler and data dependence tests chosen for the comparison. After that, their implementations are discussed. The last sections present used benchmark problems and results with analysis.

2 Parneu Compiler

SUIF (Stanford University Intermediate Format) [8] framework has been chosen because of its ready made front-end for C and flexibility to freely modify and add new passes. An overview of the PARNEU compilation flow is depicted in Fig. 1. The sequential source code is first compiled into SUIF. Then the analysis and optimisation passes are applied. After that, the parallel code generator maps operations to the Processing Units (PU) of PARNEU. Information on the capabilities of PUs (memory, performance) and communication network is required at this phase. As a result, there are SUIF codes for each processor, which are further converted in C and compiled with the processor compiler. Communication primitives are available for the code generation. The data dependence analysis pass is performed before inserting the communication functions and parallelising the program.

![Fig. 1. Parallel compilation flow for PARNEU](image)

The original SUIF paralleliser is targeted to shared-memory multiprocessors and is thus being re-written because PARNEU is a distributed memory system like many embedded systems. The standard data dependence analysis has been further improved with the implementation of other tests compared in this paper.

3 Dependence Tests

Data intensive signal processing and scientific algorithms often contain loops performing a set of operations on data arrays. Thus, a common way to increase performance is to exploit parallelism among iterations of a loop [13]. For this reason all of the data dependence tests chosen were limited to the analysis of this loop-parallelism. In addition, only static analysis was carried out in each test problem.