Mathematical and Computer Modelling 51 (2010) 974–984

Contents lists available at ScienceDirect

Mathematical and Computer Modelling

journal homepage: www.elsevier.com/locate/mcm

Efficient heuristic algorithms for path-based hardware/software partitioning

Wu Jigang [a,b,*], Thambipillai Srikanthan [b], Ting Lei [b]

[a] School of Computer Science and Software, Tianjin Polytechnic University, 300160, China

[b] School of Computer Engineering, Nanyang Technological University, 639798, Singapore

ARTICLE INFO

Keywords:
Algorithm
Heuristic
Tabu search
Dynamic programming
Hardware/Software partitioning

ABSTRACT

Hardware/software (HW/SW) partitioning is one of the crucial steps of co-design systems. It determines which components of the systems are implemented in hardware and which ones are in software. In this paper the computing model is extended to cater for the path-based HW/SW partitioning with the fine granularity in which communication penalties between system components must be considered. On the new computing model an efficient heuristic algorithm is developed, in which both speedup in hardware and communication penalty are taken into account. In addition, an efficient tabu search algorithm is also customized in this paper to refine the approximate solutions produced by the heuristic algorithm. Simulation results show that the heuristic algorithm runs fast and is able to produce high-quality approximate solutions. Moreover, the tabu search algorithm can further refine them to nearly optimal solutions within an acceptable runtime. The difference between the approximate solutions and the optimal ones is bounded by 0.5%, and it hardly increases with the increase of the problem size.

1. Introduction

Hardware/software (HW/SW) co-design has become one of the primary applications of electronic system level tools and methodologies. It provides new opportunities for the development of high speed, low power electronic products such as embedded, communication, multimedia, and intelligent transport systems. Traditional approaches for the design of simple systems were carried out manually. However, as the systems to design have become more and more complex, manual approaches have become infeasible. Thus, it is now imperative to involve design automation at the highest possible level, in order to deal with the high complexity, increased time-to-market pressures and a set of possibly conflicting constraints.

In hardware/software (HW/SW) co-design systems, application-specific hardware is usually much faster than software, but it is significantly more expensive. Software on the other hand is cheaper to create and to maintain, but slow. Hence, performance-critical components of the system should be realized in hardware and non-critical components in software. HW/SW partitioning is to decide which components of the system should be implemented in hardware and which ones in software. It has been shown that efficient techniques for HW/SW partitioning can achieve results in performance, power or energy superior to software-only solution.

There are many different academic approaches to solve the HW/SW partitioning. The traditional approaches include hardware-oriented and software-oriented. The former starts with a complete hardware solution and iteratively moves parts of the system to the software as long as the performance constraints are fulfilled [1–3], while the latter starts with a software program moving pieces to hardware to improve speed until the performance of the final system meets the given constraint [4–6]. It has been shown that the HW/SW partitioning is NP-hard for most cases. Thus, in algorithmic aspect, simulated annealing algorithms [4,7,8], dynamic programming algorithm [9,10], integer programming approaches [11,12]

* Corresponding author.
E-mail addresses: asjgwu@gmail.com, asjgwu@ntu.edu.sg (J. Wu).

0895-7177/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.mcm.2009.08.029

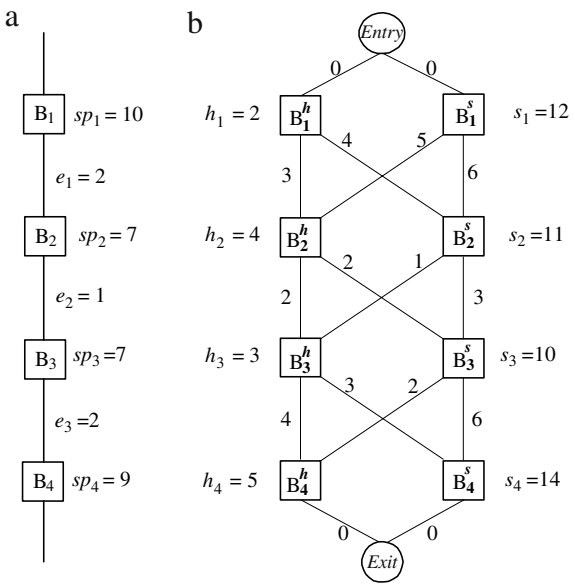

Fig. 1. Computing model comparison for 4 blocks. (a): Old model indicated by inherent speedups and extra speedups [9,10]. (b): New model indicated by source execution times and all communication times [16].

and genetic algorithms [13,14] are generally utilized to perform the system partitioning and hardware exploration. All these approaches can work perfectly within their own co-design environments, but it is impossible to compare them, because of the large differences in their co-design environments and the lack of benchmarks [15].

Advanced profiling techniques provide us a 'hot path', such as the body of a loop, that consists of the executed components with high frequency in a given application. The HW/SW partitioning for the whole application thus can be approximately solved by efficiently partitioning the selected hot path. This paper focuses on the efficient heuristic algorithms for the HW/SW partitioning on the selected hot path. Initially, we extend the computing model such that all possible communications between the neighboring components are taken into account, in order to better reflect real applications. Then we propose a fast heuristic algorithm on the new computing model to generate an approximate solution with good quality. In order to get nearly optimal solutions, we also customize a tabu search algorithm for the HW/SW partitioning to further refine the approximate solutions. Simulation results show that the proposed tabu search algorithm can refine the approximate solutions, so that the solution error to the optimal ones is no more than 0.5% for the cases considered in this paper, even if the partitioning problem is of fine granularity.

This paper is organized as follows. In Section 2, we present the computing models and formal description of the HW/SW partitioning problem. In Section 3, we describe the proposed heuristic algorithm followed by the tabu search algorithm. In Section 4, we provide the simulation results to highlight the solution quality of the proposed algorithms, by comparing them with the exact solutions. Finally, we conclude our work in Section 5.

2. Computing models and formulations

In [9,10], HW/SW partitioning was described below: the hot path of a given application consists of a sequence of n blocks, denoted as $\mathcal{B} = \{B_1, B_2, \ldots, B_n\}$, that may be moved between hardware and software. Each B_i is followed by B_{i+1} for $i = 1, 2, \ldots, n-1$. Hardware blocks and software blocks cannot execute in parallel. The adjacent hardware blocks are assumed to be able to communicate the read/write variable they have in common directly between them without involving the software side. $\mathcal{H}$ denotes the set of blocks assigned to hardware; $\mathcal{S}$ denotes the set of blocks assigned to software. The objective is finding a partitioning for $\mathcal{B}$ such that $\mathcal{B} = \mathcal{H} \cup \mathcal{S}$ and $\mathcal{H} \cap \mathcal{S} = \emptyset$, which yields the best speedup while having a total area penalty no more than the available hardware area.

The corresponding computing model utilized in [9,10] is shown in Fig. 1(a), where sp_i denotes the inherent speedup of moving block B_i to hardware, and e_i denotes the extra speedup which is incurred because of blocks being able to communicate directly with each other when they are both placed in hardware. It is noteworthy to point out that, communication time becomes more important especially for the fine granularity HW/SW partitioning. However, the computing model shown in Fig. 1(a) does not completely take the communication time into consideration, *e.g.*, when either of the two neighbors is placed in hardware.

In this paper we work on a new model as shown in Fig. 1(b) for the HW/SW partitioning, in which all types of the communication time are taken into account, no matter how the blocks implement. We employ the following notations throughout this paper.

- s_i denotes the execution time of B_i in software, $1 \leq i \leq n$.