

EVALUATION AND MAPPING OF APPLICATIONS ON HETEROGENEOUS MULTIPROCESSOR SYSTEMS

ARYABARTTA SAHU



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY DELHI

JULY 2009

©Indian Institute of Technology Delhi (IITD), New Delhi, 2009

EVALUATION AND MAPPING OF APPLICATIONS ON HETEROGENEOUS MULTIPROCESSOR SYSTEMS

by

ARYABARTTA SAHU

Department of Computer Science and Engineering

Submitted in fulfillment of the requirements of the degree of

Doctor of Philosophy

to the



Indian Institute of Technology Delhi
July 2009

Certificate

This is to certify that the thesis titled **Evaluation and Mapping of Applications on Heterogeneous Multiprocessor Systems** being submitted by **Aryabartta Sahu** for the award of **Doctor of Philosophy in Computer Science & Engg.** is a record of bona fide work carried out by him under our guidance and supervision at the **Deptt. of Computer Science & Engg., Indian Institute of Technology Delhi**. The work presented in this thesis has not been submitted elsewhere, either in part or full, for the award of any other degree or diploma.

M. Balakrishnan

Professor

Deptt. of Computer Science & Engg.
Indian Institute of Technology Delhi

Preeti Ranjan Panda

Associate Professor

Deptt. of Computer Science & Engg.
Indian Institute of Technology Delhi

Acknowledgments

I am greatly indebted to my supervisors Prof. M Balakrishnan and Dr. Preeti Ranjan Panda for their valuable technical guidance and moral support. I would like to thank Prof. Anshul Kumar, Dr. Kolin Paul, Prof. Subhashis Benerjee and Prof. G.S. Visweswaran for their valuable feedback, suggestions and help in all respect. I would like to thank Anant Vishnoi, Nagaraju Pothineni, Neeraj Goel and Vikram Goyal for their feedback during our informal discussions and my other colleagues specially V.N. Muralidhara, Uma mudengudi, Smruti Padhy, Ayesha Choudhury, Lava Bhargava, Sonali Chouhan, Kameshwar Rao, G. Krishnaih and B.V.N. Silpa for their cooperation and support. I would also like to thank the staff of Philips, FPGA and Intel laboratories, IIT Delhi for their help. I am very thankful to Naval Research Board, Govt. of India for being enabler of my work.

My family members showed immense patience and provided me great moral support during the course of my work.

Aryabartta Sahu

Abstract

In recent years, embedded system applications mostly pertain to image processing, vision and wireless networking domains. These applications exhibit high degree of parallelism apart from being very compute intensive in nature. The implication is that multiprocessor solutions are likely to be implementation platforms for embedded applications that are software dominated. Multiprocessor solutions imply a huge increase in the design space from which an optimal configuration is to be selected at design time. This is because even for a fixed processor, the number of possible configurations due to the large number of configuration parameters like memory size, cache, interconnection switches, synchronization techniques, etc. contribute to the design space. Efficient performance estimation based exploration is essential to be integrated into the design framework to effectively explore this vast design space.

In this work, we describe a methodology for retargetable (i.e., user specified target processor and architecture) software estimation to enable design space exploration of heterogeneous multiprocessor systems.

The work reported in the thesis is divided into four parts:

- Software estimation of C programs on a single processor,
- Software estimation of multi-threaded programs on multiprocessors,
- Software estimation of parallel applications, expressed in Cilk on multiprocessors, and
- A framework for multiprocessor design space exploration targeted for Cilk programs.

For implementing an embedded application, there are varieties of commercially available processors, each one offering a different design point based on factors such as performance and power consumption. Though this thesis deals with multiprocessor design space exploration, selection of a suitable processor remains a crucial activity to design a multiprocessor.

Retargetable software estimation on uni-processor starting from “C” programs takes two inputs: processor description in HMDES (High-level Machine DEscription) and “C”

description of the task. Our estimator generates a set of outputs associated with the task behavior on the processor including the execution time. Execution time estimation of a task mapped onto the target processor is carried out by correlating task requirements with the processor capabilities. The methodology proposed includes the effect of basic compiler optimizations and integrates light weight memory simulation and instruction mapping to improve the accuracy of estimates. In this process, we have used SUIF (Stanford University Intermediate Format) compiler framework to develop a uni-processor performance estimator. Results of various benchmarks on a range of processors using this technique shows that it is possible to predict the performance with a maximum error of around 14%. We have used this framework and methodology to develop software estimation for multiprocessors.

In the second part of our work, we describe software estimation of multi-threaded programs on a multiprocessor. Our methodology takes three inputs; fork-join representation of application, high level description of the multiprocessor target architecture and binding (showing mapping of application components onto architecture resource elements). Output is the application performance on a target multiprocessor architecture. Our framework assumes all application components like computation tasks as well as communication and synchronization protocols are specified in “C” language. Further, it is assumed that all these tasks and constructs execute on an instruction set processor (in contrast to Application Specific ICs (ASICs)). Uni-processor software estimation has been used to estimate the task execution time for each individual processor. To these estimates, delays due to communication and synchronization protocol have been separately estimated and added. To estimate performance degradation due to shared resources like memory and bus, “appropriate” access traces from multiple sources are generated using the uni-processor framework. For the shared resources, interval analysis technique is employed on this “synthetic” traces to predict delays due to contention. Results using different application programs show that using such an approach, it is possible to predict performance with average errors of around 11%. Further, we conclusively show that performance not only depends on the architecture but also depends heavily on the binding of application components onto multiprocessor resources.

Multiprocessing and use of MPSoC are growing rapidly in the embedded systems space. On the other hand, writing and debugging of parallel multi-threaded code is

not only difficult and cumbersome, but also results in inefficient code. Many parallel programming languages have been developed that assist writing and debugging of multi-threaded code. Cilk is one such language developed at MIT in mid-90s. Cilk uses an optimal run time scheduler to map the application on to the multiprocessor architecture. In this work, we have developed a novel methodology for performance estimation of Cilk programs on multiprocessors that can be used in the early phase of design space exploration. The methodology incorporates retargetable (i.e. user specified target processor and architecture) estimation of Cilk multi-threaded applications on MPSoC. We start with generation of dynamic process spawn graph and timing annotation of that graph. We then perform retargetable estimation of task transfer time, task spawn time and task sync time by instrumenting the source code of Cilk run time scheduler and Pthread library. This enables our estimator to capture the behavior of the run time scheduler of Cilk. For validation purposes, we have ported Cilk generated multi-threaded code to a multiprocessor simulator. Results show that using estimation, it is possible to predict the performance with an average error of 19% but with a speed up of over 300x vis-a-vis simulation.

Finally, we use the above methodology to create a framework to explore the multiprocessor architectural space. This framework has been used to generate optimal architecture for a number of benchmarks. The benchmarks are specified in Cilk and multiprocessor architectures in HMDES. The framework uses a genetic algorithm in exploring the vast design space and identifying suitable candidates meeting the performance constraints for detailed simulations.

Contents

List of Figures	ix
List of Tables	xi
1 Introduction	1
1.1 Background	1
1.1.1 Design space of multiprocessors	2
1.1.2 Design space exploration	5
1.1.3 Design evaluation methods	6
1.1.4 Design evaluation using software estimation	9
1.2 Review of previous work	10
1.3 Our approach	13
1.4 Brief description of our work	14
1.4.1 Uni-processor software estimation	15
1.4.2 Multiprocessor software estimation	16
1.4.3 Software estimation of Cilk programs on multiprocessor SoCs . .	17
1.4.4 Multiprocessors design space exploration of applications specified in Cilk	17
1.5 Organization of our work	18
2 Performance Estimation of Uni-processor Systems	19
2.1 Introduction and motivation	19
2.2 Previous work	22
2.3 Performance estimation flow	25
2.3.1 Instruction mapping	27

2.3.2	Basic compiler optimizations	28
2.3.3	Variable memory access latencies	30
2.3.4	Effect of data initialization, type conversion and variable declaration	33
2.4	Uniprocessor estimation results	34
2.5	Generation of synthetic trace	35
2.6	Discussion	37
3	Performance Estimation of Heterogeneous Multiprocessor Systems	41
3.1	Introduction and motivation	41
3.2	Previous work	43
3.3	Performance estimation methodology	45
3.3.1	Application specification	45
3.3.2	Architecture specification	46
3.3.3	Partition description	46
3.3.4	Performance estimation strategies	46
3.4	Performance estimation of multi-threaded tasks	47
3.5	Resource contention analysis of fork-join	50
3.5.1	Interval analysis: An abstract interpretation	50
3.5.2	Delay calculation in interval analysis	51
3.5.3	Result of interval analysis	53
3.6	Multiprocessor performance estimation results	54
3.7	Discussion	57
4	Performance Estimation of Cilk Programs on Multiprocessors	59
4.1	Introduction	59
4.2	Previous work	60
4.3	Cilk: Multiprocessor programming language	61
4.4	Cilk application performance estimation on multiprocessor	63
4.5	Process spawn graph	65
4.5.1	Generation of process spawn graph	65
4.5.2	Annotation of process spawn graph	68
4.6	Estimation of scheduling parameters	69
4.6.1	Training benchmarks	70

4.6.2	Instrumentation of Cilk-RTS lib and Pthread Lib	71
4.7	Fast mimic of work stealing scheduler	72
4.8	Modeling cache and shared resource contention	74
4.8.1	Modeling methodology	74
4.8.2	Generation of traces	74
4.8.3	Delay calculation	76
4.9	Validation platform	79
4.10	Experimental results	80
4.11	Discussion	82
5	Design Space Exploration of Multiprocessor Systems	85
5.1	Introduction	85
5.2	Design space exploration	87
5.3	Genetic algorithm based exploration	88
5.3.1	Genetic algorithm	88
5.3.2	Design parameters	89
5.4	Experiment and results	91
6	Conclusions and Future Work	95
6.1	Conclusions and contributions	95
6.1.1	Software estimation of uni-processor	95
6.1.2	Software estimation for multiprocessors	96
6.1.3	Software estimation of Cilk program on multiprocessors	97
6.1.4	Multiprocessor design space exploration	98
6.2	Future work	98
	Bibliography	101
	A Cradle CT3400 Architecture	113
	B High level Machine Description (HMDES)	115
	C Mapping of thread APIs	119

