INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 9, Number 2, Pages 193–207

GPU ACCELERATED PARALLEL BRANCH PREDICTION FOR MULTI/MANY-CORE PROCESSOR SIMULATION

LIQIANG HE GUANGYONG ZHANG AND JINGDONG JIANG

Abstract. Branch Prediction is a common function in nowadays microprocessors. Branch predictor is duplicated in each core of a multi/many-core processor and makes prediction for multiple concurrent running programs respectively. To evaluate the parallel branch prediction in a multi/many-core processor, existing schemes generally use a parallel simulator running on a CPU that does not have a real massive parallel running environment to support the simulation and thus has a bad simulating performance. In this paper, we use a real many-core platform, GPU, to perform a parallel simulation of branch prediction for the future general purpose multi/many-core processor design. We verify the correctness of the GPU based parallel branch predictor against the traditional CPU based branch predictor. Experiment result shows that the GPU based parallel simulation scheme obtains a two to ten times of speedup over the CPU platform when the issue rate ranging from one to four instructions per cycle, and it shows that the GPU based scheme is a promising way to improve the simulation speed for future multi/many-core processor research.

Key words. Branch Prediction, Parallel Simulator, GPU, and Multi/Many-core Processor.

1. Introduction

Branch prediction is a commonly used function in nowadays superscalar or multicore microprocessor. It uses the branch history (either local or global history or both) to predict whether a next branch instruction is taken or not. The accuracy of a branch predictor affects the control flow of a running program with more or less instructions executed along the wrong paths and then affects the final performance of the program. Lots of researches have been done related to branch prediction [1-3] in the past decades.

Branch prediction research generally needs a simulator. Existing schemes either use a cycle-by-cycle based simulator which runs a program in its simulating environment and uses a real executing flow to investigate the functionality of a branch predictor, or use a trace based simulator which is much simpler and faster than the former but loses some run-time accuracy.

In multicore and many-core processor, branch predictor is duplicated in each core of the processor. Each predictor records its own branch history from the running program in the host core and makes the particular prediction respectively. There is a big design space that can be explored for branch predictors in a multi/many-core system. For example, Branch predictors in different cores can (a) cooperate with each other to increase the prediction accuracies for multi-threaded program, or (b) be dynamically combined into more powerful predictors, or (c) switch off parts of them to save power if their behavior is the same. Investigating or exploring the design space of the parallel branch prediction for a multi/many-core processor needs a parallel branch predictor simulator. A general technique to build a parallel simulator in academic literature is to parallelize the traditional sequential simulator

Received by the editors December 9, 2009 and, in revised form, January 10, 2011.
2000 *Mathematics Subject Classification.* 68Uxx.
This work is supported by Inner Mongolia Natural Science Foundation Project No. 20080404MS0901, and the Ph.D Research Startup Foundation of Inner Mongolia University No. 208041.

using array structure[4-5] or Pthread programming scheme[6]. In a general simulating environment without a big memory support, this technique may be suitable for research on multicore processor with less than sixteen cores but it is absolutely not useful or possible for multicore with more than thirty-two cores or for many-core cases. Some other researches [7-9] rely on FPGA to do parallel simulation for multi/many-core processors. Although the simulation speed is fast, the ability and the scalability are limited by the hardware itself.

In this paper, we extend our previous work [10] and use a real many-core platform, GPU (Graphic Processing Unit), to help improve the simulation speed for massive parallel branch predicting research for future multi/many-core processor. It is well known that GPU is originally designed to target regular massive parallel computing such as matrix operations, FFT, and lineal algebra. But the processor simulation, including branch prediction, cache accessing, pipeline processing, has very irregular program behavior which GPU does not favor initially. In this work, we try to (a) map an irregular simulating program to a regular organized GPU structure and (b) use the existing massive parallel GPU platform to help the multi/many-core processor architecture research, especially parallel branch prediction. Although only the branch prediction is considered, it is a case study and start point of research on multi/many-core simulation using GPU platform for the future microarchitecture research.

We rewrite most of the code of the branch predictor component in a widely used superscalar processor simulator, SimpleScalar [11], and let it run in an NVIDIA GTX275 GPU processor [12]. We verify our result (including the control flow and branch predicting outputs of the simulated program) from GPU runs against the one from the original CPU based running. Experiment results show that (a) the GPU based code can perform exactly the same functionality as the compared CPU based code which verifies the correctness of our code and shows the ability of GPU to do irregular operations, and (b) the GPU code can potentially faster the simulation speed, up to ten times, for the branch prediction simulating with its many-core structure when compared with the serialized CPU code.

Comparing with our previous work, we make the following new contributions in this paper:

- Consider the specific GPU features, and optimize our implementation of the GPU based parallel branch prediction simulation.
- Verify the correctness of previous work [10], and show the speedup results on new hardware platform.
- Through experiment on three typical types of workloads, a trend of the simulating speedup on GPU platform is obtained, and the maximum speedup values at 8K simulated cores or threads are observed.
- Make sensitivity analysis of the instruction issue rates in the simulated cores and the number of instructions being simulated.

The rest of this paper is organized as follows. Section 2 presents the GPU architecture and programming model. Section 3 introduces the rationale of the branch predictor used in this paper and the organization of the parallel branch predictor in future many-core microprocessor. Section 4 describes the design and implementation of our GPU based parallel branch prediction simulator. Section 5 gives the experimental methodology and Section 6 presents and analyzes the results. Then, Section 7 discusses the related work, and finally Section 8 concludes this paper.