The study of A Multithreaded Processor Architecture for Parallel Logical Languages

Masayuki Hosoi (910102)

School of Information Science,
Japan Advanced Institute of Science and Technology

February 15, 2002

Keywords: parallel,GHC,JTD,context,thread,multithread.

Abstract

This paper proposes architectural support for parallel logical programming languages and parallel execution mechanism suitable for the paradigms involving parallelism.Although the paradigms including functional,logical,and object oriented languages can be simulated in cost effectively on the sequential architecture, parallel execution potentialities involved in those paradigms has been ignored. And those have been little concerned with parallel execution mechanisms which can be commonly used those paradigms in spite of many studies about parallel execution dedicated to each paradigm. For example,a multithreaded processor architecture for functional programgs has been proposed.In this study,the authors intend to provide parallel execution mechanism commonly.First,the authors intend to use a multithreaded pipeline processor architecture by centering on parallel logical programming languages,especially,guarded Horn clauses,called GHC.And then,this architecture shows good performance in other paradigm involving parallelism.

1 Introduction

This paper proposes architectural support for parallel logical programming languages and parallel execution mechanisim suitable for the paradigms involving parallelism.Formerly,architectural support for programming languages has traditionally been popular as an in indirect way of application specific support. But "High-Level Language Computer" has been forgotten for at least 20 years. The present day,the Microprocessor focused our attention on RISC is vigorous. People tried that problems of software was solved

by hardware on "High-Level Language Computer".Nevertheless, value of computer systems for users is not the achievement of "High-Level Language Computer" but overall cost effectiveness. There are two major subjects in software development. One approach is attempts in engineering to grasp efficient methods of software development.The other is paradigm shifts from the paradigm of imperative languages which is based on von Neumann Architecture to others.One of the other paradigms is logical programming language which takes interest in parallel processing or parallel machine architecture.However,this is not agreed with overall cost effectiveness. ¿From the cost effectiveness view point,a machine as a hardware made by simpler structure and from less resources is priority even if it causes a little perfomance fall.Accordingly,"High-Level Language Computer" is not worth anymore if exists.

Generally speaking,programming languages processing is divided into three phases.First is language translation phase. Second is operation sequence control phase.Third is execution phase. In translation phase, we can see that it is cost effectiveness to assign the software compiler translation jobs. On the other hand,in order to speed up execution phase, using hardware leads execution phase themselves to highest effectiveness. Speed-up by hardware is effective in parallel processing. Remaining opinion that we have already discussed in "High-Level Language Computer" is operation control phase. So far, this is discussed in the problems of instruction issue or instruction scheduling for a sequential architecture having instruction pipeline.Although the paradigms including functional,logical,and object oriented languages can be simulated effectively in cost on the sequential architecture,those paradigms involving parallel execution potentialities are ignored. And in spite of many studies about parallel execution dedicated to each paradigm,it is little concerned with parallel execution mechanisms commonly used those paradigms.

2 Logical Programming Languages and the manner of process

This paper preliminarily concentrate to one paradigm. Because we study parallel executable mechanism commonly. This mechanism can be used for

each paradigm including logical,functional,and object oriented languages. And the authors will explore the idea of parallel executable potentialities hiding in the paradigm implying logical,functional,and object oriented languages. Focusing one paradigm is logical language. The authors center on parallel logical programming languages,especially guarded Horn clauses.We use the term "GHC" to describe language which is guarded Horn clauses. An ordinary sequential architecture execute GHC uneffectively. Mainly,there are two causes.One is a repetition of pattern matching called unification. The other is switching of running contexts which is resulted frequently. In order to execute unification efficently,data types must be determined as soon as possible.In this measures,it has been reported that a tag dispatch instruction which can jump to multi branches including from four-way to sixteen-way by A instruction.In other words,the instruction can issue from four to sixteen data types dependent instruction streams. We use the term "JTD" to describe a tag dispatch instruction. In order to execute switching of running contexts,switching of running contexts must be hidden effectively. According to studies of machines for parallel logical programming languages as before,the report says as follows: PE(Processing Elements),or uniprocessor was not treated importantly. What is treated importantly is parallel machines. Those parallel machines mean network structure, methods of communication among uniprocessors. GHC causes context switching. Context switching causes high frequency. This is heavyweight processes. Heaveweight processes casues low performance. (ICOT Technical Report:TR-670[July,1991].) Context switching is casued by synchronization among fine-grain processes(threads) for data dependencies. In order to avoid heavyweight processes,we need mechanism which can switch lightweight contexts by hardware.

3 Thread Scheduling

The author perform an experiment on simulation to prove our proposal. Simulation program is hand-compiled code of GHC.Consequently,the author must determine threading policy of hand-compiled code for multithreaded program. Thus,in this chapter,we remark Java as a typical language which can be design multithreaded programmings and test that Java

Virtual machine behavior which show how to determine thread which must be execute in time.

4 Design of A Proposed Architecture

A multithreaded processor architecture for functional programgs has been proposed. This processor has multiple hardware resources for every thread,and a thread select unit which picks out the next thread to be executed.Therefore,data hazards,branch hazards,and structure hazards are avoided.The processor makes no stalling the pipeline by a control unit dealing with cache miss.The processor is composed of superpipeline and can treat the same number of threads associated with a program counter,register file,all kinds of control register as the number of pipeline stage.This switching policy is switch on every cycle.This policy allows switching on every cycle,independent of whether it is a load or not. In other words,it switches contexts from different threads on cycle-by-cycle basis.Successive instructions become independent,which will benefit pipelined execution.The context switching can hide pipeline dependencies and reduce the context switch cost. Therefore,it can provides a performance advantage over switching. The main features of this processor is four.First,the processor is being fully utilized,as long as there are enough threads in the processor so that the number of pipeline stage are filled with the number of threads. Thus,it is only necessary to have enough threads to fill in stages of superpipeline.Once stages of superpipeline are filled,the processor is running at peak performance and additional threads do not speed the result. Second,the processor supports multiple instruction threads in a uniprocessor environment.Third,the architecture model consists of pipelined,dedicated units supporting multiple instruction issuing in every clock cycle. Fourth,the processor can hide pipeline hazards with issuing the same number of threads as the number of pipeline stages.While it has seventeen-stage instruction pipeline,it didn't conduct simulations on the processor to prove practical effect executing functional programgs.

Now,as we can see above,in order to overcome heavyweight contexts of GHC caused by synchronization among fine-grain processes(threads) for data dependencies and make the best use of parallel execution potentialities, the authors will aim at a multithreaded processor which can switch

lightweight contexts and propose that the Multithreaded Pipeline Processor for GHC Execution.This architecture can execute GHC efficiently , is of much benefit to each language implying parallelism ,and make the best of parallel execution potentialities.There are at least four things which the authors try in sequence in order to prove the proposal. First, the authors make multithreaded pipeline processor of simple structure. Second, the authors simulate execution of GHC in practice. Third,the authors have it improve performance by making the best of parallel execution potentialities in GHC. Fourth, the authors suggest common parallel execution mechanism which can be applied to each language implying parallelism.we use high-level logic synthesis sysytem PARTHENON which is Parallel Architecture Refiner THEorized by Ntt Original coNcept in order to design the processor and simulate it. The CPU executes each instruction in a series of small steps. This sequence of steps is frequently referred to as the fetch-decode-execute cycle.It is central to the operation of all computers. Thus,the execution cycle of a most fundamental instruction is three phases: fetch,decode,execution.These instruction phases can be executed by an instruction pipeline.Because of designing simpler processor and simulating more efficiently,we apply three-stage instruction pipeline as a most fundamental pipeline structure.Moreover,in order to process unification of GHC efficiently, JTD is needed as we can see in above.Consequently,the author designs two kinds of processors and carries out basic evaluations of the proposed mechanisms in order to show that the proposed architecture is efficient.One is pipeline processor including three-stage instruction pipeline and JTD.The other is multithreaded pipeline processor including three-stage instruction pipeline and JTD.

5 Simulation

The authors conduct four simulation whose programs is all hand-compiled code on those processors to prove the proposal whose architecture is efficent for execution in GHC and each languge implying parallelism. First,the authors test whether JTD is efficient or not.Second,the authors perform thread saturation tests in order to confirm that the processor is being fully utilized,as long as there are enough threads in the processor so that the

number of pipeline stages are filled with the number of threads. Third,the authors cause test program implying a necessity of a synchronization run in order to show hiding effect of processing of synchronization. Fourth,the authors carried out test program implying parallelism which does not depend the characteristic code of the parallel paradigm.As this program, we use matrix multiplication.

6 Consideration

Those perfomance evaluations indicate that JTD is capable of reducing clock cycles by about seventy percent, Multithreaded architecture can provide that high performance pipeline for execution of GHC,execution of GHC including JTD, and execution of program implying parallelism and make the values of CPI about 1.0. As a result,the authors consider two things.First,multithreaded pipeline processor including JTD can greatly improve GHC execution performance by making the best of parallel execution potentialities in GHC.Seconds, multithreaded pipeline processor including JTD can provide parallel execution mechanism commonly which is of much benefit to each language implying parallelism.

7 Conclusion

The authors conclude two things.First,multithreaded pipeline processor including JTD can greatly improve GHC execution performance by making the best of parallel execution potentialities in GHC.Seconds, multithreaded pipeline processor including JTD can provide parallel execution mechanism commonly which is of much benefit to each language implying parallelism.