

The application of SpMT WaveCache in performance development of dynamic data flow computer

Zhou Ning¹, Zhang Jing²

¹Hebei University, Baoding, Hebei, China, 071002

²Agricultural University of Hebei, Baoding, Hebei, China, 071001

*Corresponding author's e-mail: e-mail: zhouning@hbu.edu.cn

Received 6 October 2013, www.cmmt.lv

Abstract

Along with the gradually rising of the frequency of single core processor, it also brought communication overhead, design difficulty, power consumption and other problems. These problems all come from memory wall, power wall, and etc. However, dynamic data flow computer well solved the problems such as memory wall, and catered to the development trend of current multi-core system architecture. Based on this, this paper made a study on the performance development of a typical dynamic data flow computer system structure - WaveScalar system structure, introduced in detail the application of SpMT WaveCache in dynamic data flow computer performance development, dug the speculative multithreading parallelism of data flow computer, and finally proved with practice that the performance of WaveScalar system structure was greatly improved, and thus is a method of dynamic data flow computer performance development deserving to be vigorously popularized and applied.

Keywords: data flow computer, WaveScalar, WaveCache, SpMT

1 Introduction

Along with the rising of single core processor frequency, it also brought many negative effects, such as communication overhead, design complexity, power consumption, etc. Moreover, most researchers and manufacturers thought that multi-core processor will certainly replace single core processor. It is also just because of this, the validity of Moore's law is continued. But after decades of development, the processor still uses the traditional von Neumann system structure [1-2]. The thread level parallel technology and the instruction level parallel technology get more and more constraints [3-5]. Moreover, some literature points out: the current complex microprocessor architecture is capacity-constrained on improving instruction level parallelism, while the data flow computer can well solve the problem of memory wall and adapt to the development trend of multi-core system architecture.

Superscalar technology and instruction level parallel technology encounter more and more challenges in the development. Thread level parallel technology begins to become a better way to improve the execution performance of serial program. Speculative multithreading technology (SpMT) gets rapid development as a thread parallel technology in recent years [6-8]. It uses hardware-software co-design method to realize the parallel execution of serial program. Because ScalarWave system structure designed an order memory mechanism, it can ensure that the mandatory language executes in order in memory system. So we should put the emphasis of the research on memory management module and control module, studying the speculative execution of multithreading in memory system.

2 Overview of SpMT WaveCache

2.1 SPMT EXECUTION MODEL

In speculative multithreading system structure, serial program is assigned to be executed in different speculative multithreading in accordance with a classification rule. Different speculative multithreading executes different parts, and is submitted in strict accordance with the order, thus to ensure the accuracy of the submitting result. When the program executes the firing command, if the resources at this time allow the firing, then a new speculative thread will be fired, and each thread can bring out multiple threads and submit the result in strict accordance with the sequential mode.

In SpMT execution model, the compiler divides the control flow graph of the program into the sequence of multiple threads. The first thread is assigned to be executed in a processing unit according to certain rules. At the same time, it can also create a new subsequent thread, and the subsequent thread can also create its own subsequent thread in other processing units. In specific operation, after the first thread finishing the execution, its subsequent thread will become the first thread to continue the execution. Such execution model can support the thread level speculative execution, because there is always only one non-speculative thread during the whole process of execution, namely the first thread.

In the control of this model, focus, it mainly focuses on the creation, termination and submission of the thread as well as the maintainer of the speculative thread. This is mainly in order to guarantee the equilibrium of the multi-core processor computing resources load [9-10]. After the completion of thread creation, the model will conduct automatic maintenance according to the specific circumstance, as shown in figure 1 - the thread state diagram of the model.

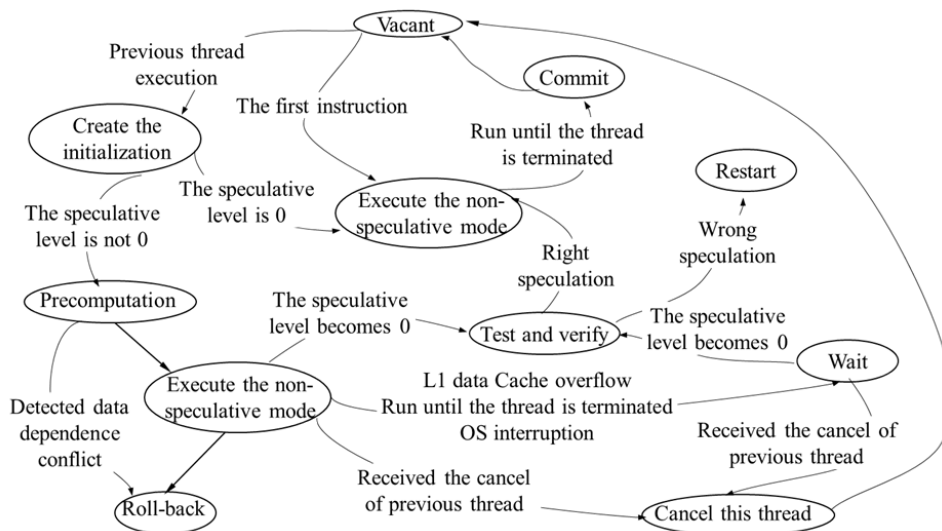


FIGURE 1 the thread state diagram of SpMT

2.2 INTRODUCTION OF WAVESCALAR SYSTEM STRUCTURE

WaveScalar system (ScalarWave system) is actually a dynamic data flow instruction level system. It has very strong scalability. Swanson et al. built a simulation platform WaveScalar. Its biggest difference with the general data flow computer is that it can support the programming and execution model of mandatory language. For program developers, they don't have to learn new programming language in this platform. They can finish the programming work by using its original mandatory language. This can effectively reduce the learning pressure of program developers, and make they quickly adapt to the new system. It is just this characteristic promoted the development of WaveScalar system, and thus promoted the development of the dynamic data flow computer.

Combining with primary WaveScalar system structure, Marzulo et al constructed a new transactional memory system, putting forward wave transactional WaveCache. It is actually realized by increasing hardware unit wave context table and memory operating history table in original primary WaveScalar system structure. In this wave transactional WaveCache, it abstracts each transactional WaveCache into concrete affair, to set up a transactional memory management system. When the wave completed all memory operations of this affair, the wave is a transaction commit; if it appears the situation when data conflict is detected in the process of transaction execution, immediately cancel all operations, and restore all transactional contexts of the wave through the context table, to execute it anew.

3 Application analysis of SpMT WaveCache in dynamic data flow computer performance development

With the development of science and technology as well as the development of data flow computer, people's requirements on data flow computer are also higher and higher, more and more new technologies are applied to the development of dynamic data flow computer. The application of SpMT WaveCache in dynamic data flow computer performance development is mainly to develop the hidden speculative multi-threading parallel performance of data flow computer.

3.1 TRANSACTIONAL MULTITHREADING

Transactional WaveCache can only support instruction level speculative execution of single wave, but cannot solve thread level speculative execution. So we designed a multi-threaded programming model based on it, and added thread synchronization unit, thread context table and thread history table based on its original hardware system structure, attempting to dig the performance of dynamic data flow computer by such transactional multithreading.

Relevant research shows that the threads in dynamic data flow computer system are different in size, the instruction execution therein is unpredictable, the thread's context switching frequency is always changing, and the switching frequency is related to specific application. So we designed a transactional multithreading based on the size of thread granularity. We designed the size of thread granularity into the size of 30-150 waves. Here we ignored the separate memory characteristic of WaveCache system, to improve the performance of the system by increasing the granularity of the thread. This is because: when the granularity of the thread is too thin, the context switching of the thread will be very frequently, so it is difficult to realize the parallel of thread granularity, and thus lowered the performance of the system.

In order to realize the multithreading communication of ScalarWave system, we determine a serial number for each thread, add a thread ID on the basis of the original wave number, and use <t,w,d> to represent a data token in WaveCache, among which, t is the thread number, w is the wave number, and d is the data value. Use multiple instructions to realize the switching among thread number, wave number, and data value, and realize the correct transmission of parameters between threads. In addition, in order to effectively strengthen the communication between threads, and reduce the overhead generated in the communication between threads, we build critical data index for multithreading in SpMT WaveCache, so as to promote the mutual communication between threads. Synchronization between threads in traditional ScalarWave system structure often needs to transmit the data between two threads by instructions and complete the mutually exclusive operation by combined command. However, the ScalarWave system structure designed in this paper is a thread synchronization design based on the lock. The thread synchronization unit in memory buffer can effectively guarantee the synchronization between threads.

3.2 EXECUTION MODEL

We add thread context in transactional speculative multi-threaded execution model. Its main function is to quickly recover its context table when thread execution is failed, and then re-execute the thread. This context table is stored in the memory buffer. It preserves the ID, attribute and input operation data in the process of thread execution. The added history table records all memory operation instructions of the thread, can realize instruction execution trace. Moreover, this history table also contains the thread ID and the speculative attribute of the thread. This increased the function of the history table to a certain extent. The content of this history table is shown in table 1.

TABLE 1 The content of thread memory history table

Thread ID	Wave Number	Current Number of Store	Sp.Field	Store/Load	BackUp Field for Store Operations
-----------	-------------	-------------------------	----------	------------	-----------------------------------

3.3 TWO-LEVEL TRANSACTION COMMITTING MECHANISM

In order to ensure the smoothness and accuracy of communication between threads, we designed a two-level transaction committing mechanism in this system. The first level is to conduct transactional submission in wave-level granularity, and the second level is to conduct transactional submission in thread-level granularity. This two-level transaction committing mechanism, on the one hand, can satisfy the parallelism of coarse-grained threads, and on the other hand, can also execute whole process tracking of fine-grained threads, to realize the tracking debugging processing of thread execution, so as to effectively improve the reliability of the software system. In order to realize the two-level transaction committing mechanism, we designed TValidate and TCommitted attribute value in the system. The former denotes when all waves are submitted, set the attribute as true and commit the thread-level transaction. Then set the attribute of the latter as true, until the submission of this thread-level transaction is completed, and then write it into the shared memory. The committing process of thread-level transaction is shown in the figure below.

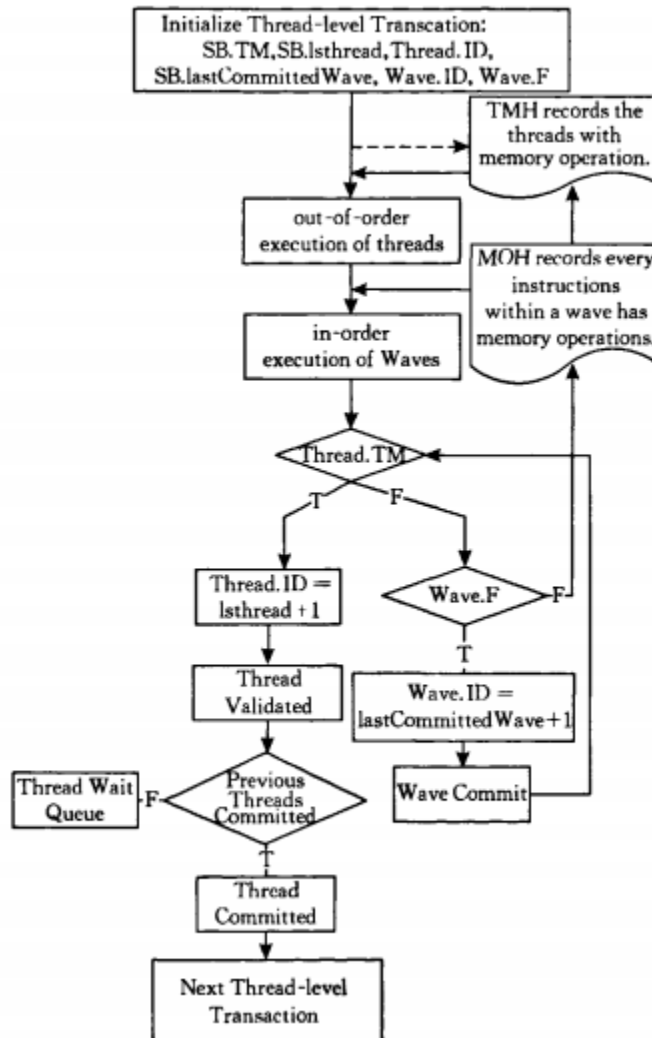


FIGURE 2 the committing process of thread-level transaction

TABLE 2 The micro-system structure parameters of SpMT WaveCache

Main memory delay	200 clock periods	Processing unit book/domain	8 (4pods)	Number/cluster of domain	4
Network delay	Within Pod: 1 clock period Within Domain: 5 clock periods Within Cluster: 9 clock periods Inter-Cluster: 9 clock periods + propagation delay of clustering between clusters	Input queue of processing unit	128 inlets, 4 blocks	The assembly line depth of processing unit	5 level
WaveScalar	4K static instruction (128 pieces static instruction with each processing unit)				
		Output queue of processing unit	4 inlets, 2 ports (1 reading and 1 writing)	Network switching	2 ports, two-way

Test results show that the performance of SpMT WaveCache system has 2 to 3 times of ascension compared with superscalar system, and the execution speed is also improved. However, SpMT WaveCache system also has some problems. For example, thread execution failure causes the re-execution of the thread, and its direct consequence is overhead increase. This indicates that this SpMT WaveCache system remains to be further optimized and improved. In addition, on the test of acceleration ratio, we found that the linear acceleration ratio of SpMT WaveCache system does not improve. This shows that the size of the thread block affected the execution efficiency. When the big thread block only has fewer threads to complete the task, the execution efficiency of concurrent thread is not excavated very well; on the contrary, the small thread block has many threads to complete the task. Frequent switching between threads increases the overhead, and reduces the execution efficiency to a certain extent. So we can draw such a conclusion: on cyclic computational

task, speculative multithreading has higher linear acceleration ratio, while the task with high branch frequency only has lower linear acceleration ratio.

4 Conclusion

With the sustainable development of social economy, dynamic data flow computer will get rapid development and application. Although the research in this paper well dig out the potential performance of data flow computer, the speculative execution overhead of big thread block is large, and the thread-level transactions take up larger context space, it is very prone to have execution failure. Repeated execution adds overhead, and affects the parallel ability of the thread. Next, we still need to further enhance the development work of data flow computer, and make efforts to improve the efficiency of transactional execution, and thus promote the popularization and application of dynamic data flow computer in economic production.

References

[1] Filo Ján, Zaušková Anna 2010 2D Navier-Stokes Equations in a time dependent domain with neumann type boundary conditions *Journal of Mathematical Fluid Mechanics* 12(1) 1-46

[2] Cot L, Raymond J-P, Vancostenoble J 2009 Exact controllability of an aeroacoustic model with a neumann and a dirichlet boundary control *SIAM Journal on Control and Optimization* 48(3) 1489-518

[3] Kannan M, Srivatsa S K 2006 Hardware implementation of instruction level parallel architecture incorporating special functional units for image processing algorithms *Information Technology Journal* 5(3) 416-21

[4] Luo Jin-Ping, Zhou Xing-Ming, Chen Shu-Ming 2000 New instruction level parallel processing model based on optical interconnection *Tien Tzu Hsueh Pao/Acta Electronica Sinica* 28(11) 96-8

[5] Xie Xuejun, Yu mingyan, Ye Yizheng 2006 Improvement of instruction-level parallelism via speculative execution *WSEAS Transactions on Computers* 5(10) 2270-75



[6] Li Yuancheng, Zhao Yinliang, Yin Peipei, Han Bo 2010 A cost estimation based speculative path prediction method for speculative multithreading *Hsi-An Chiao Tung Ta Hsueh/Journal of Xi'an Jiaotong University* 44(12) 22-7

[7] Wei Yuanke, Zhao Yinliang, Song Shaolong, Wang Xuhao, Yin Peipei, Li Ting 2010 Zhao, Yinliang; Song, Shaolong; Wang, Xuhao; Yin, Peipei; Li, Ting *Hsi-An Chiao Tung Ta Hsueh/Journal of Xi'an Jiaotong University* 44(12) 10-5

[8] Nagpal Rahul, Bhowmik Anasua 2012 Criticality guided energy aware speculation for speculative multithreaded processors *Parallel Computing* 38(6-7) 329-41

[9] Shigeto Yusuke, Sakai Mikio 2011 Parallel computing of discrete element method on multi-core processors *Particuology* 9(4) 398-405

[10] Lee Jong-Bok 2012 A performance study of multi-core out-of-order superscalar processor architecture *Transactions of the Korean Institute of Electrical Engineers* 61(10) 1502-7

Authors	
	<p>Zhou Ning, 1980, Baoding, Hebei Province, P.R. China</p> <p>Current position, grades: Master degree, the lecturer of Hebei University, China. Scientific interest: His research interest fields include Machine learning and artificial intelligence. Publications: more than 10 papers published in various journals. Experience: He has teaching experience of 12 years, has completed three scientific research projects.</p>
	<p>Zhang Jing, 1981.11, Baoding, Hebei Province, P.R. China</p> <p>Current position, grades: the lecturer of Agricultural University of Hebei Province, China. University studies: received her Master of Laws from Hebei University in China. Scientific interest: Her research interest fields include ideological and political education major Publications: more than 12 papers published in various journals. Experience: She has teaching experience of 11 years.</p>