Computer and Information Science

www.ccsenet.org/journal.html

Vol. 1, No. 3

August 2008

Summarization of Program Development in Multi-core Environment

Hongyu Li

College of Computer and Automatization

Tianjin Polytechnic University

Tianjin 300160, China

E-mail: milanbaggio18@sina.com

Abstract

It is not transparent from single core time to multi-core time for programmer not like the enhancement of clock frequency of processor, and if we don't design the programs what we compile aiming at multi-core characters, we can not obtain the enhancement of performance from multi cores. In this alternative time of old and new concepts, we should fully use former development experiences for references.

Keywords: Multi-core programming, Multithreading, Parallel computer, OpenMP

1. Introduction

The computer system adopting the multi-core processor is a parallel computer system with multi processors in the chip. This sort of parallel computer system only occurred in the super computer center and the cluster computer system in the past. At now, with that the processor chip has entered into the multi-core time, millions computer systems in our homes have been the parallel computer system with multi processors. This sort of parallel computer system is the system with parallel course and parallel thread, and many courses and threads can really run at parallel, and it is not like the single processor system that the course and thread can only alternatively run in CPU. But in the multi-core system, the running speed of the former single program can not be enhanced. To enhance the running performance of single program, we need redesign the former program, decompose the single computation task into many parallel subtasks, and let these subtasks respectively run in different processor cores. Thus, we need adopt different programming methods. With the advent of multi-core time, the parallel programming will be the necessary knowledge and skill in the software industry.

2. The advent of multi-core time revolution

In 2001, IBM pushed the Power4 processor based on dual-core, after that, both Sun and HP successively pushed the UltraSPARC 4 processor and PA-RISC8800 processor based on dual-core frame. But these RISC processors facing high-end application were too high to be popular, and they didn't obtain attentions by people. Until the second quarter in 2005, Intel issued the desktop dual-core processor based on X86, so the multi-core processor begun to go into common families.

Today, the multi-core processor has occupied more and more market shares, and the programming personnel also must face the collision brought by the revolution of multi-core processor. The multi-core programming is not only the opportunity but also the challenge, and it is the urgent task for us that how to grasp the direction and advance with time in the industry revolution. Because if we don't design the programs what we compile aiming at multi-core characters, we can not obtain the enhancement of performance from multi cores. In this alternative time of old and new concepts, what we can choose and whether we can use former development experiences for references? We should use former experiences for references and actively study the skill of parallel programming. The multi-core, especially the dual-core is very similar with the frame of dual-route SMP (symmetric multiprocessing).

From Figure 1, we can see that though there are differences in the dual-core technology between Intel and AMD, but the so-called dual-core processor is to centralize two computation cores in one processor. That is quite similar with the dual-route SMP system which centralizes two processors on one mainboard, and the difference is only in that the data exchange between two computation cores of the dual-core computation system doesn't need the front system bus (FSB), but the two processors of the dual-route processor system is to exchange data through FSB, which is one tiny detail that we should notice when we compile programs.

Similar with SMP programming, we must use the form of multithread or multi-course to compile the application program to obtain the performance improvement brought by multi-core processor aiming at the multi-core processor programming. So most experiences what we accumulate in the SMP parallel programming can be applied in the multi-core programming.

3. The revolution of programming

The advent of multi-core time brings large collision to our programming thinking. In order to fully utilize the multi-core property, we must learn to design the program with the partial thinking and compile the program with form of multi-course or multithread. Whether we should use the multi-course form or the multithread form to compile the program is one of confused problems for programmers, the author thought we should decide it according to concrete application, but in usual conditions, the multi-core programming using multithread has more advantages than multi-course.

Except for programming form, the motivation that we use multithread programming has been changed. In the past, for the Windows programmer, one of main reasons using multithread was to enhance the user experience, for example, to enhancing the response speeds of UI, I/O or network in the long-time computation. But in the multi-core time, the reason that we compile the application program is to fully utilize many computation cores, reduce the computation time or compute more tasks in the same period. For example, in the game programming, the form of multithread can detract the computation of collision inspection into many CPU cores to largely reduce the computation time, and it also can use the multi-core to make more meticulous inspection computation and can simulate more real collision.

In the multi-core time, we should be more careful to choose the programming language. Relative to compiled languages such as C/C++/Fortran, the script languages such as C#/java/Python may be the better choice. The reason is that the script language is super, and it usually offer the original support to the multithread, such as the System Threading.Thread of C#, the java.lang. Thread of java, and the Threading.Thread of Python. Comparably, the compiled language usually supports the multithread through the relative base of the platform, such as Win32 SDK and POSIX threads. There is not uniform standard, so more details are should consider when we compile multithread program through C/C++, which increase the project costs. Now, through the users of C/C++ are numerous, but the script language in the multi-core time will be more popular, because the script language usually has not ISO standard, so it can be changes conveniently, and the explainer and compiler will be occur quickly aiming at multi-core. However, the script languages such as PHP, Ruby and Lua are difficult to obtain programmers' favors, because they don't offer the interior core thread support, and their multithreads are for users, and they even can not support the thread, so the multithread program complied by them can not still fully utilize the multi-core advantages.

Through C/C++ losses part advantages in the multithread programming because of the lack of support of language, but because most present main operation systems offer API to create the thread by the form of C language interface, so C/C++ has quite abundant program base, which can compensate the deficiency of language to some extent. We can not only use the Win32 SDK, but can use POSIX threads, MFC and boost.thread to compile the multithread program. Through these bases offer certain encapsulations and reduce programmers' multithread burdens to certain extent, but for the multi-core programmers whose aims are to enhance the performance of condensed program, these methods are still very complex, because the use of these bases almost needs increase double key codes, and the corresponding transfer and testing costs will be increased largely. The better choice is to use the compiler such as OpenMP to support and strengthen the fundamental base of multithread. The OpenMP appoints the parallel code segment through using #pragma compiler order, and it changes the program little, and it can appoint the serial edition to compile for debugging conveniently, and it can also coexist with the compilers that can not support OpenMP.

It is obvious that through the script language offers the original support of the multithread programming on the language layer, but it doesn't lead the C/C++ far. The essential reason is the base of the script language, the fundamental base and CRT/STL of data structure and algorithm. The fundamental base of C/C++ is designed and developed by the serial form. To modify the fundamental base aiming at the multi-core programming, which is the extremely urgency faced by all compiled languages, is the war of life and death to separate the leading advantage of two camps, but the property is centralized in the script languages such as C#/java/Python in certain company or organization, which will decide the key battle. That is the reason why we commend choosing the script language to compile program.

4. Multi-core programming

With the advance of time, we will finally face the multi-core system to design program. For the multi-core programming, the author thought it is equal to the parallel programming sharing memory, and the multi-core programming can use former experiences of past parallel programming for references, such as the partial design thinking, the parallel design method and multifold parallel support methods.

First, we discuss the partial design thinking. Because the thread is the minimum unit to distribute CPU resource for the operation system, so if we want to design the multi-core parallel program, we must form the designing thinking to

divide the program into parts. The Mr. Hua Luogeng's overall planning method can be referred to discuss how to divide the program into parts.

For example, we want to steep the teal to drink. The situation is that there is not boiled water, the water jug, the teakettle and the teacups should be washed, the fire burns, and there is tea leaf. How to do?

Method A: wash the water jug, fill the cool water, put the teakettle on the fire, when we wait the boiled water, we can wash teakettle and teacups, take the tea leaf, and when the water is boiled, and we can steep the tea to drink.

Method B: first do some preparation works, wash water jug, teakettle and teacups, and take the tea leaf, and when every thing is ready, we fill and boil the water, and when the water is boiled, we steep the tea to drink.

Method C: clean the water jug, fill the cool water, put the water jug on the fire, wait that the water is boiled, and when the water is boiled, we find the tea leaf in hurry, wash the teakettle and teacups, and steep the tea to drink.

Which method can save time? Of course, the first method is the best one, the others waste time. Suppose Mr. Hua Luogeng has two robots to steep tea, the best method is to distribute the work according to the "Method A", i.e. robot A go to boil the water, and robot B wash the tea set, and when the water is boiled, the tea can be steeped. Unconsciously, we use the partial thinking, i.e. distributing irrelevant works to different processor to implement.

The above partial thinking is simple and direct, but for complex task, it is not so easy to find the partial plan, so we need the method of parallel design to guide us. Through tens years' researches of parallel program, former researchers have summarized many effective methods of parallel design, here, we introduce one classic method, i.e. the data correlation graph. Still taking the classic steeping tea as the example, we can draw the following data correlation graph seen in Figure 2.

From Figure 2, we can see that the data correlation graph is a directional graph, and every acme in the figure represents a task which should be completed, and the arrowhead denotes the task pointed by the arrowhead depends on the task educed by the arrowhead, and if there is not the route from one task to another task, so two tasks are not irrelevant, so we can implement parallel treatment. If Mr. Hua want to steep the tea to drink himself, so the smaller oblong in the data correlation graph of Method A is can be parallel, and if Mr. hua has two robots to help him steep tea and there are two water faucets at least for robots, so the bigger oblong area can be parallel and obtain higher efficiency. So the resources that can be reasonably used are more, the parallel acceleration ratio is higher.

Now that the partial thinking and method of parallel program design can be referred, we only lack in how to develop the parallel program. There are three popular ideas.

(1) Extended compiler. The parallel compiler can find and express the parallel character in the present serial language program, for example, the Intel C++ Compiler has the functions to automatic parallel circulation and vector data operation. This method leaving parallel works to compiler reduces the costs to compile the program, but because of the complex combination of circulation and branch languages, the compiler can not identify quite more parallel codes and falsely compile serial edition.

(2) Extended serial compiled language. It is the most popular method, and it can denote lower layer language to obtain parallel program through increasing function transfer or compiling orders. The users can create and end the parallel course or thread, and offer function of synchronization and communication. The outstanding bases include MPI and OpenMP, and in the explained script camp, the ParallelPython also win partial market.

(3) Creating a parallel language. Through it is a crazy idea, but in fact, in tens of years, there are many people to do the work all along, for example, HPF (High Performance Fortran) is the extension of Fortran90, and it supports the parallel data program through various modes.

In the future multi-core programming, we will find that there is more knowledge in the computation domain to deserve us to study, and there is more extensive space to realize out ideas.

References

Akett Roberts, interpreted by Li, Baofeng et al. (2007). *Multi-core Programming Technology.* Beijing: Electronic Industry Press. Mar. 2007.

Chen, Guoliang. (2002). *Parallel Computer System Structure.* Beijing: Higher Education Press. Sep. 2002.

David R. Butenhof, interpreted by Yulei et al. (2003). *POSIX Multithread Programming.* Beijing: China Power Press. Apr. 2003.

Huangkai. (2000). *Extended Parallel Computation Technology, Structure and Programming.* Beijing: Machine Industry Press. May 2002.

Jim Beveridge, interpreted by Houjie et al. (2002). *Win32 Multithread Programming.* Wuhan: Huazhong University of Science and Technology Press. Jan. 2002.

Kalla R. (2004). IBM Power 5 Chip: A Dual-core Multithreaded Processor. *IEEE Micro.*

Kai Hwang, interpreted by Zheng, Weimin et al. (1995). *Higher Computer System Structure.* Beijing: Tsinghua University Press.

Micheal J. Quinn, interpreted by Chen, Wenguang et al. (2004). *MPI and OpenMP Parallel Programming for C Language.* Beijing: Tsinghua University Press. Oct. 2004.

Zheng, Weimin. (1998). *Computer System Structure.* Beijing: Tsinghua University Press.

Table 1. Comparison of various languages to support multithread

	Compiled languages such as C/C++	Scripts such as C#/java/Python	Scripts such as PHP/Ruby/Lua
Whether the language support the multithread	No	Yes	No
Whether the base support the multithread	Yes	Yes	No
Whether supporting interior core thread	Yes	Yes	No
Whether support user thread	Can be simulated	Can be simulated	Yes
The complexity of thread programming	Commonly/Easy	Easy	N/A
Recommendation degree	★★★☆	★★★★	★☆

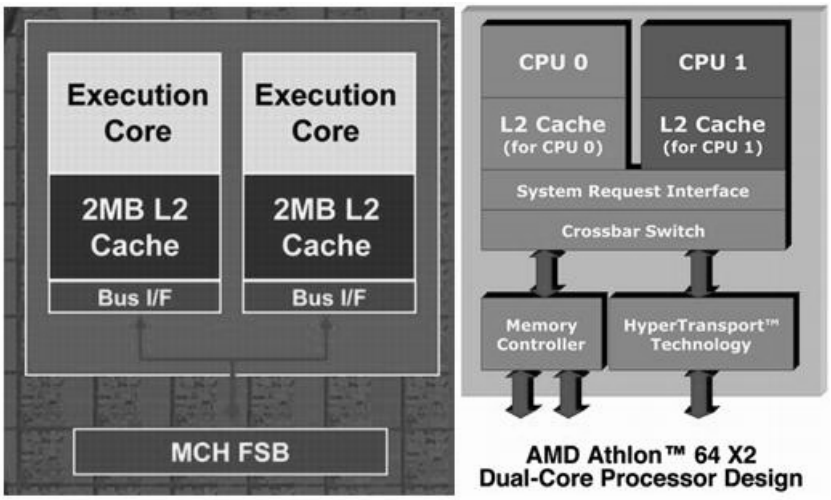

Figure 1. Dual-core CPU Structures of Intel and AMD

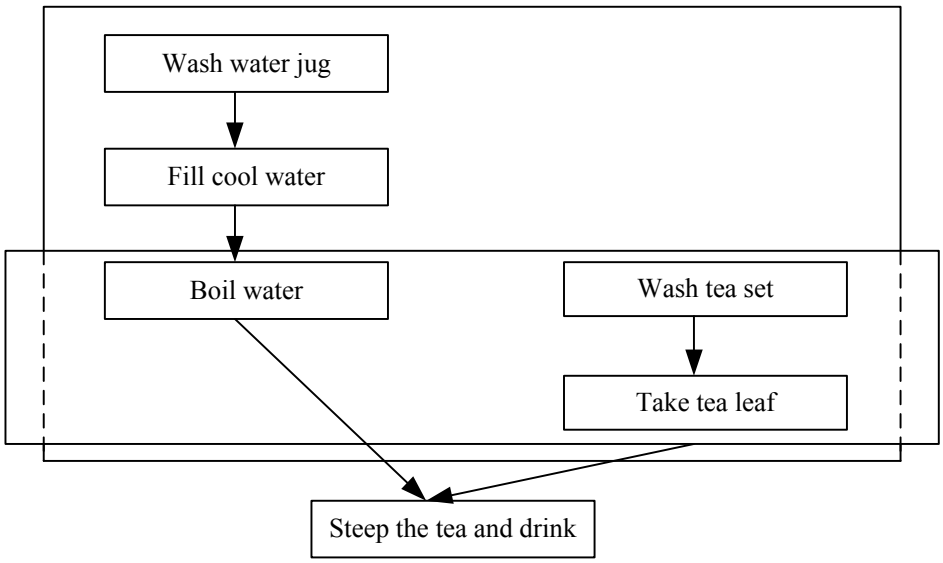

Figure 2. Data Correlation of Method A in “Overall Planning Method”