早稲田大学大学院理工学研究科

博 士 論 文 概 要

論 文 題 目

大規模問題に対するマルチプロセッサシステム上での
統一的並列処理手法の研究

A Unifined Approach for Parallel Processing of Large-Scale Applications on Multiprocessor Systems

申 請 者

プレムチャイサワディ ウィチェン

PREMCHAISWADI WICHIAN

電気工学専攻計算機制御研究

1991 平成 3 年 12 月

Fast and efficient computers are in high demand in many areas of science and engineering. Large-scale computations are often performed in these application areas and there has always been a demand for faster computers for new applications that require more intensive computation. To satisfy the need, two methods have been used to achieve higher computing speed, namely; increasing the speed of the circuitry and increasing the number of operations that can take place concurrently, through either pipeline or parallelism. However, the speed of light puts a ceiling on the speed on which electronic components of a certain size can operate. Hence, the use of parallelism is the only approach to improve the computing speed. VLSI technology made replication of hardware units affordable. Multiprocessor architectures using several identical processors in parallel have been introduced. There is a consensus among computer designers today that the multiprocessor approach is the only direction one can take in hope for unlimited processing power, if the exchange of information between processors and memory does not become a major bottleneck and a source of slowdown.

The multiprocessor approach introduces also three new requirements not encountered before. 1) Each problem must be partitioned into tasks. 2) The partitioned tasks must be scheduled to processors for execution. 3) Synchronization of control and data flow must be performed during execution. Thus, there is a high demand for the techniques to resolve the above mentioned problems to perform parallel processing on multiprocessor systems efficiently.

Parallel processing of an application program on a multiprocessor system can be achieved by two approaches. The first one is by using some parallel programming languages to specify the parallelism or by using a serial-to-parallel compiler to exploit the parallelism within the program. The second one is an application oriented approach. This approach is performed by taking advantages of the specific characteristics and inherent parallelism within the application to be processed. Then, a special parallel compiler, taking into account both applications to be solved and architecture of hardware to be implemented, is developed and used to extract parallelism in the application program.

In using parallel programming language, the user needs to specify the parallelism within the program, even for problems for which some parallel algorithms exist, the design can be extremely tedious and the resulting program becomes difficult to debug. For the conversion from serial to parallel by a compiler, the user can more concentrate on the problem to be solved. However, the parallelism obtained is limited depending on the efficiency of the compiler. At present, the compilers usually can perform parallel processing only of the DO-loop. There is also parallelism in other levels that must be exploited to increase the overall performance of parallel processing. For many applications, there is a need for higher computation speed to finish the computation within the required time. Thus, all types and levels of parallelism within the program must be exploited for parallel processing. In this situation, the application oriented approach may be the solution.

This dissertation mainly concerns the application oriented approach. The objectives of the dissertation are twofold:

1. To develop a unified approach for parallel processing of large-scale applications on multiprocessor systems using fine grain tasks to exploit as much parallelism as possible with minimum overhead.

2. To implement the proposed scheme on an actual multiprocessor system and evaluate its performance on the real applications that the parallel processing is difficult to realize such as the solution of nonlinear differential equations, the solution of sparse linear equations, circuit simulations and the dynamic system simulations.

The dissertation consists of seven chapters.

Chapter 1 describes the overview, objective and outline of the dissertation.

Chapter 2 presents a unified approach for parallel processing of large-scale applications. The proposed scheme is composed of four steps: 1) Tasks generation with the consideration given to the task size (granularity). 2) Task graph representation to analyze data dependency which consists of output-dependence and flow-dependence and anti-dependence. 3) Tasks scheduling to allocate the generated tasks onto processors with the minimum parallel execution time. 4) Parallel machine code generation to generate the optimal machine code by using the information from the scheduling process such as the optimal use of registers to pass data on the same processor.

Then, the implementation method of the proposed scheme is presented. The architecture of a multiprocessor system OSCAR is described as an example of multiprocessor system on which the scheme is implemented and evaluated.

Chapter 3 is concerned with the parallel processing scheme of the solution of nonlinear differential algebraic equations using backward differential formula (BDF). The scheme can be applied for both stiff and non stiff systems without any knowledge to the solution of the system to be solved. The BDF method is an variable order and step size integration method which composes of loops and conditional branch structure. Hence, we can not apply DOALL and DOACROSS techniques, which are popular parallel processing schemes for DO loops on a multiprocessor system. Thus, no efficient parallel processing scheme for the solution has been so far proposed. The experimental results show that the parallel processing of the solution can be realized efficiently using the proposed scheme.

Chapter 4 presents a parallel processing scheme of sparse linear equations. The symbolic code generation method used by electronics circuit simulators which has been known as the fastest solution method for the solution of sparse linear equations is employed to generate loop-free code, with the use of the Markowitz reordering method to minimize fill-ins. This application is concerned with the parallel processing of fine grain tasks of scalar computation

which has been known to be difficult to process on multiprocessor systems due to communication overhead. The special purpose compiler is used to generate the loop-free code and partition it into fine grain tasks automatically. The experimental results show that the proposed scheme can resolve the difficulties for parallel processing of fine grain task of the solution of sparse linear equations on an actual multiprocessor system.

Chapter 5 describes a parallel processing of circuit simulations using the direct method which is the most accurate solution method for circuit simulations. The equation formula is performed by using modified nodal equations which has been used in the conventional circuit simulations. The circuit simulators consist of two main parts which consume most of the simulation time, namely: linearization of nonlinear devices and solution of nonlinear differential equations including a solution of sparse linear equations. Although many researches have been conducted to increase the solution speed of circuit simulation, the parallel processing is performed only partially. The parallelism is exploited in only high level such as the parallel processing of subcircuits in which one processor is used to solve a subcircuit. There is substantial parallelism inside subcircuit computation, which is difficult to exploit. To increase the overall speed of circuit simulation, all of the computation within subcircuit is parallelized using the proposed scheme. The test results show that the speed up of each part of subcircuit computation is obtained and the overall performance of the circuit simulator is improved.

Chapter 6 is concerned with the real time dynamic system simulations. The simulation of this kind of applications needs high computation speed to finish within required time. The design of these systems are usually supported by computer aided design tools such as a graphical input engineering block like in $Matrix_x$. Then, a multiprocessor system is used as the back-end processors to execute the code generated from the front end processor. The parallel processing of such a system is now performed by generating the Fortran code from the graphical input engineering blocks like the Hyper-Build in $Matrix_x$ and the compiler is used to generate the parallel machine code. However, the user must assign tasks to processors themselves, which is not efficient and needs much effort of the user. To resolve these difficulties, the proposed parallel processing scheme is employed. The experimental results using the application oriented approach and the use of parallel Fortran language compiler on an actual multiprocessor system are shown to be efficient and more useful in practices.

Chapter 7 is the conclusion of the unified approach for parallel processing for various applications and the suggestion of the future works.