氏名	成富（なりとみ） 敬（たかし）
授与学位	博士（工学）
学位授与年月日	平成5年3月25日
学位授与の根拠法規	学位規則第5条第1項
研究科，専攻の名称	東北大学大学院工学研究科 （博士課程）情報工学専攻
学位論文題目	Parallel Algorithms for Large Scale Computation in Science and Technology （大規模科学技術計算の並列アルゴリズムに関する研究）
指導教官	東北大学教授 阿曽 弘具
論文審査委員	東北大学教授 阿曽 弘具　東北大学教授 丸岡 章 東北大学教授 西関 隆夫

論文内容要旨

Chapter 1 Introduction

Remarkable development of semiconductor device technology made it possible to simulate natural phenomena in numerical way as well as theoretical analyses and experiments. Because of the cost effectiveness in time and money and the capabilities of observing an unobservable phenomenon, the importance of the numerical simulation is becoming more and more in many areas of researches in science and technology such as device simulation, weather forecast, plasma physics, astrophysics and so on.

In the numerrical simulation problems, large-scale tridiagonal or block tridiagonal linear systems are solved and most of the computation time is spent on solving them. Hence, the key to realize a high-speed numerical simulation is shortening the time to find the solutions. To this end, there are two important factors to be considered. One is an algorithm to solve the large-scale systems and the other is an architecture to execute the algorithm. From the algorithmic viewpoint, there is a problem such that almost of the algorithms presently used are serial ones or modified ones of originally serial algorithms. On the other hand, from the architectural viewpoint, there is a problem such that the clock cycle that dominates the computing

capabilities of vector supercomputers are coming to their limits because of the limitation of the size reduction of device geometry. Hence, from the algorithmic viewpoint, finding parallel algorithms which are suited for several types of parallel processing is strongly desired. On the other hand, from the architectural viewpoint, a parallel processing with high parallelism is essential to realize a high speed numerrical simulation.

In this dissertation, parallel algorithms for the solution of large-scale tridiagonal and block tridiagonal linear systems are proposed. Furthermore, a VLSI-oriented parallel architecture : SALTS which executes the parallel algorithms in systolic way is described.

Chapter 2 Parallel Tridiagonal Solvers

One-dimensional simulation problems often result in solving tridiagonal linear systems. A tridiagonal linear system for an unknown vector $\boldsymbol{x}$ is represented in a matrix form as $\boldsymbol{Ax}=\boldsymbol{h}$, where $\boldsymbol{A}$ is an $N\times N$ nonsingular tridiagonal matrix, which is diagonally dominant. While, $\boldsymbol{x}$ and $\boldsymbol{h}$ are column vectors of length N.

In this chapter, firstly, the bi-recurrence algorithm is proposed as an efficient tridiagonal solver which has inherent parallelism based on the parallel first order recurrences. The algorithm is composed of three stages and each stage is computable in parallel with two processors. The time complexity as a serial algorithm is $8N-4$ for a tridiagonal linear system of order N which is comparative with the Gaussian elimination algorithm whose complexity is $8N-7$. For parallel implementation with two processors, the time complexity of the bi-recurrence algorithm is $4N-1$. Moreover, the bi-recurrence algorithm is applicable to several numerical computations such as solving band or block tridiagonal linear systems, inverting tridiagonal matrices and so on.

Secondly, a highly parallel tridiagonal solver based on the divide-and-conqure strategy is proposed. The solver is highly parallelizable and suited for highly parallel comuters, although complexity as a serial algorithm is approximately twice as much as the Gaussian elimination algorithm.

Chapter 3 Parallel Block Tridiagonal Solver

If we discretize a two- or three-dimensional domain over rectangular grids and number all of the grid points in natural ordering, then we have a large-scale block tridiagonal linear system of equations.

In order to solve such systems, iterative methods are mainly used. For example, ICCG (Incomplete Cholesky Conjugate Gradient) method and ACCR (Incomplete Cholesky Conjugate Residual) method are efficient and widely used block tridiagonal solvers, which are characterized by low memory consumtion and high convergence. The efficiency of these methods comes

from the preconditioning using the incomplete decomposition of A as $A=LU+R$. The preconditioning, however, is difficult to be parallelized fully because of the computational dependencies included in them. Hence, in order to solve block tridiagonal linear systems faster, parallelizing the preconditioning is essential.

In this chapter, a highly parallel iterative method named parallel preconditioned ICCG method is explained. It is an efficient parallel solver combining the overlapping preconditioner with the ICCG method. The preconditioner of the proposed method is the divided one of a large scale preconditioner into smaller scaled ones which have overlapping parts with two neighboring preconditioners. The effect of the parallel preconditioning on the convergence of the method is estimated experimentally for many size of boundary value problems of Poisson's equation. The results show that the parallel preconditioned ICCG method is computable in high parallelism, though the parallel preconditioners slightly increase the number of iterations in comparison with the original ICCG method. For example, when a two-dimensional domain of simulation is discretized over 250×250 rectangular grids, one large preconditioner is divided into 249 parallel preconditioners and they can be operated in fully parallel. Moreover, by substituting the experimental results into formula of complexity, the speedup is evaluated for the parallel preconditioned ICCG method. For example, for a model problem discretized over the 250×250 grid, if it is solved by 249 parallel preconditioners using 249 processors in parallel, the speedup will be approximately 146. From the experiments, it is find that the choice of the discretization grids depending on the number of divisions are important.

Chapter 4 SALTS

In this chapter, a specialized systolic architecture SALTS (Systolic Architecture for Large Tridiagonal Systems) is proposed with several kinds of systolic algorithms. The architecture is used to solve large-scale tridiagonal and block tridiagonal linear systems for equations. SALTS is designed to reduce the whole number of processing cells, utilizing the regularity and the sparseness of the systems. For example, using an array of linearly connected three of processing cells in SALTS, the systolized bi-recurrence algorithm solves a tridiagonal linear system of order N in $3N+6$ time units. If the tridiagonal system is pertitioned into p subsystems, the highly parallel systolic tridiagonal solver finds the solution in $10(N/p)+6p+23$ time units with $3p$ cells.

On the other hand, to solve block tridiagonal linear systems efficiently, SALTS executes systolic algorithms for the operations that are commonly included in several kinds of iterative methods such as the ICCG method and the ICCR method as well as the parallel preconditioned ICCG method. In specific, systolic algorithms for matrix vector multiplication (MVM), vector summation or vector subtraction (VSU), inner product (IP), incomplete matrix decomposi-

tion (IMD) and forward-and-backward substitution (FBS) are proposed. For example, only sixty-four cells are necessary for IMD, only seven for MVM and for FBS. For IP and VSU, only one cell are necessary owing to the pipeline technique. Moreover, the algorithms are revised for faster parallel computations. By the acceleration for MVM and FBS, required cells are $N^{1/3}$ times more than the non-accelerated case, but the processing time is proportional to $N^{2/3}$.

SALTS consist of several kinds of processing cells, however their operations are similar. Hence, consideering the hardwere implementation, those cells are realized as only one kind of cell such that the cell involves all of the cell functions, and includes a switching mechanism that switches the function to corresponding one as required. Specificly, considering that cells are realized on VLSI, we design only one mask pattern. Consequently, we have systolic arrays with high reliability and high performance. Using SALTS for a large-scale computationin science and technology, a high-speed and high cost-performance simulation is achieved.

Chapter 5 Conclusions and Suggestions

Chapter 5 concludes with a summary of the preceding chapters, and indicate areas in need of further research.

審 査 結 果 の 要 旨

半導体デバイスの設計，プラズマ物理学，流体力学などの分野で対象とするシミュレーションが益々大規模化することに伴い，その計算の効率化を図ることが重要な課題となっている。著者は，自然現象の数値シミュレーションの計算の大部分が，三重対角やブロック三重対角連立方程式を解くことに当てられることに着目し，それらを解く効率のよい並列アルゴリズムや並列アーキテクチャを開発した。本論文はその成果をまとめたもので，全編5章からなる。

第1章は序論である。

第2章では，1次元空間を対象としたシミュレーションで使われる三重対角連立方程式を解く双方向再帰法を提案している。これは並列実行向きのアルゴリズムで，Nを格子点の数とするときその計算時間が，1プロセッサのときは8N－4，2プロセッサのときは4N－1となることを導いている。この方法は，並列実行が難しいガウスの消去法に代わる優れた方法である。また，分割統治の考え方に基づく高並列化法において双方向再帰法を用いることにより更に高速化できることを示している。

第3章では，2次元や3次元空間を対象とするとき使われるブロック三重対角連立方程式を解く並列前処理付き共役勾配法を提案している。これは係数行列の重なり分解により前処理を並列実行するアルゴリズムである。ポアソン方程式の境界値問題を実際に解くことにより，この方法の解の収束までの繰返し回数を評価し，並列実行により計算時間が大幅に短縮できることを示している。2次元の長方形領域を250×250の格子点でモデル化し，249個のプロセッサを用いると，従来の前処理付き共役勾配法に比べて約146倍の速度向上が達成できることを確かめている。これは実用上重要な成果である。

第4章では，シストリックアーキテクチャSALTSを提案し，行列ベクトル積や不完全行列分解などの基本演算をその上で実行する方法を与えている。また，p個に分解した係数行列に基づく高並列双方向再帰法を3p個のプロセッサからなるSALTSで実行する方法を与え，計算時間が10N／p＋6p＋23となることを示している。これは高速化の実現法として高く評価できる。更に，ブロック三重対角行列の構造を利用した高並列化手法を提案し，計算時間の大幅な短縮を図っている。

第5章は結論である。

以上要するに本論文は，大規模数値シミュレーションで重要な三重対角やブロック三重対角連立方程式を効率よく解く並列アルゴリズムを与えるとともに，それを実行する並列アーキテクチャを提案したもので，情報工学の発展に寄与するところが少なくない。

よって，本論文は博士（工学）の学位論文として合格と認める。