社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2008－HPC－116 (23)
2008／8／6

量子計算の並列シミュレーションにおける通信量削減手法

柴田 章博[†1] 中田 尚[†1] 中西 正樹[†1]
山下 茂[†1] 中島 康彦[†1]

量子アルゴリズムの性能評価の方法として，量子計算シミュレーションがある．我々は，分散メモリ型並列計算機向けの高速なシミュレーションアルゴリズムの開発を行った．我々の手法では，ボトルネックとなっているノード間の通信量を削減することにより，シミュレーションの高速化を図っている．この手法は，量子回路に意図的に交換ゲートを挿入することで，よりノード間の通信量の少ない等価な量子回路に変換することを特徴としている．我々は，ランダムに作成した量子回路に提案手法を用いた時のノード間の通信量の変化を調べた．その結果，提案した手法は，分散メモリ型並列計算機のノード間の通信量を減らすことが可能であることが分かった．

A Method of Reducing Communication Costs for a Quantum Computer Simulator on a Distributed Memory Parallel Computer

Akihiro SHIBATA ,[†1] Takashi NAKADA,[†1]
Masaki NAKANISHI,[†1] Shigeru YAMASHITA[†1]
and Yasuhiko NAKASHIMA[†1]

Quantum computer simulators are widely used for evaluating performance of quantum algorithms. In this paper, we propose a fast simulation algorithm for distributed memory parallel computers. Our method can reduce communication costs between nodes by inserting swap gates to the original quantum circuit. We evaluated the performance of the proposed method by experiments. As a result, we show that the proposed method is able to reduce communication costs.

1. はじめに

量子計算機とは，量子的重ね合わせ状態と量子干渉効果を用いた超並列計算を実現する計算機である．将来，この量子計算機が実現すれば，古典計算機よりも高速にデータ検索や素因数分解等が行えることが知られている[1),2)]．量子アルゴリズムの性能は，量子回路を構成する最小単位である量子ゲートの数や雑音の影響からの強さ等の要因によって決定される．その各要因の評価において，量子回路の雑音の影響からの強さの評価等といった解析的な評価が困難な評価要因がある．

この問題を解決する手段として，古典計算機を用いた量子計算シミュレーションが存在する．n 量子ビットを古典計算機で表現するには，サイズが $O(2^n)$ のメモリ領域が必要である．そのため，メモリ領域と計算時間は量子ビット数に対して指数関数的に増加する．単一のプロセッサを用いてシミュレーションを行う場合，シミュレーションの高速化手法として，古典計算機内で量子状態を表す情報（以下，量子情報とする）を圧縮する手法が存在する．この手法の一つとして Quantum InformationDecision Diagram[6),7)]（以下，QuIDD とする）がある．QuIDD[6),7)] を用いると，特定の量子アルゴリズムに対しては，量子情報を表すために必要なメモリ量を削減することが可能である．しかし現在，並列計算機上に，QuIDD[6),7)] を用いることは効率が良くないと考えられている．そのため，1 つのノードのメモリに量子情報が収まるようにする必要がある．しかしながら，QuIDD[6),7)] を用いても量子情報を圧縮できない場合は多くあり，そのようなケースでは，量子情報の圧縮を行わずに並列計算機を用いたシミュレーションを行う方が適していると考えられる．したがって，並列計算機を用いた高速なシミュレーション手法の開発が必要である．並列計算機を用いた量子計算シミュレーションについては，文献[8),9)] が挙げられる．これらの研究は量子アルゴリズムの振舞いの解析に重点を置いており，シミュレーションの高速化には触れていない．図 1 に分散メモリ型並列計算機の概略図を示す．分散メモリ型並列計算機で

†1 奈良先端科学技術科学大学院大学 情報科学研究科
Graduate School of Information Science,Nara Institute of Science and Technology

シミュレーションを行う場合，そのメモリは各ノードに分散される．そして，必要に応じてノード間で通信を行いながら計算する．通信量はシミュレーションで扱う量子ビット数に対して指数関数的に増加する．そのため，ノード間の通信量は計算速度に影響を与える大きな要因となっている．ゆえに，ノード間の通信量を削減することが計算速度の高速化につながると考えられる．

本稿では，並列計算機を用いた量子計算シミュレーションにおいて，ノード間の通信量を減らすことでシミュレーションの計算速度を向上させる手法を提案する．また，実験により提案手法を用いることで高速化が行えることを示す．本研究ではシミュレーション環境として，分散メモリ型並列計算機の1つである BioServer[3] を対象としている．BioServer[3] は1つの筐体に，CPU として組み込み用プロセッサの1つである FR550[4] が 128 個搭載され，各ノードに 256MByte のメモリが搭載されている．そのため，メモリ量は合計 32,768MByte となる．また，各ノードの Flash ROM には，OS として axLinux[5] が内蔵されている．

2. 量子計算シミュレーションアルゴリズム

2.1 基本設計

本研究における量子計算シミュレーションの方法について説明する．以降では，分散メモリ型並列計算機のノード数を k とし，n 量子ビットの計算をシミュレートするものとする．ただし，説明の簡単化のため k は2のべき乗とする．n 量子ビットは，古典計算機内でサイズが 2^n の配列として表現される．つまり，各基底に対する確率振幅が配列の各要素に対応する．よって，各配列の要素は複素数である．その複素数は，倍精度の浮動小数点の組として表現される．また，ノード数を k としているため，各ノードが保持する配列の要素数は $2^n/k$ となる．図2に各ノードが保持する配列についての概略図を示す．

本研究のシミュレーションでは，量子ゲートを1つずつ適用した際の量子状態の変化を追う．その計算は，各ノードが保持する配列を状態ベクトルに見立て，それにユニタリ行列を繰り返し掛けることによって量子状態を時間的に変化させる．また，必要に応じてノード間で配列の要素の通信を行う．以降本稿では，ノード間で通信する配列の要素数を通信量とする．

2.2 量子ゲートの箇所と通信量との関係

量子回路は，1ビットユニタリゲート，1ビット制御ユニタリゲートの2種類の量子ゲートによって構成することが可能である．これら2つの量子ゲートは，配置する箇所によってシミュレーション時の通信量が異なる．また，本研究では，交換ゲートを意図的に挿入することによって，ノード間の通信量の少ない

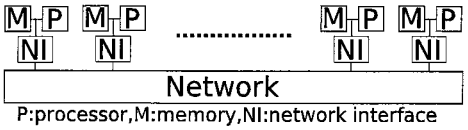

図 1 分散メモリ型並列計算機の概略図[9]．

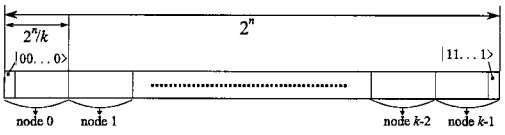

図 2 各ノードが保持する配列についての概略図．

等価な量子回路に変換することを考えている．そこで以降では，3種類の量子ゲートの配置箇所と通信量との関係を説明する．なお，基底状態の表記法として，$|00\ldots0\rangle$，$|11\ldots1\rangle$ のような2進表現と $|0\rangle$，$|2^n-1\rangle$ のような10進表現を混在して使用するが，特に断らない限り，これらは同一のものとする．n 量子ビットは，

$$\alpha_0|0_{n-1},0_{n-2},\ldots,0_1,0_0\rangle+\ldots +\alpha_{2^n-1}|1_{n-1},1_{n-2},\ldots,1_1,1_0\rangle \quad (1)$$

と表わされる．ここで，$0(1)_i$ の添え字 i は便宜上そのビットが何ビット目であるかを表すものとする．確率振幅 $\alpha_0,\ldots,\alpha_{2^n-1}$ を k 個のノードに分割して保持する．つまり，ノード $m(0 \le m \le k-1)$ は $|m\cdot 2^n/k\rangle,\cdots,|(m+1)\cdot 2^n/k-1\rangle$ に対する確率振幅 $\alpha_{m2^n/k},\cdots,\alpha_{(m+1)2^n/k-1}$ を保持することになる．

1ビットユニタリゲートは次の式の様に，2×2 の行列で表わされる．

$$U=\begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix} \quad (2)$$

この U を $i(0 \le i \le n-1)$ ビット目に適用することを考える．これを図にすると，図3の様に表わされる．このとき，ユニタリ行列 $X(2^n\times 2^n)$ はテンソル積を用いて，

$$X=I^{\otimes n-i}\otimes U\otimes I^{\otimes i-1} \quad (3)$$

と表わされる．ただし I は，

$$I=\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad (4)$$

である．図4にユニタリ行列 X の全体図を示す．図4よりユニタリ行列 X は，一辺の大きさが 2^i の小行列 X_{ele} が対角方向に 2^{n-i} 個並んだ規則的な素行列となることが分かる．この小行列 X_{ele} は，各行各列の非ゼロ要素が高々2の規則的な素行列である．このユニタリ行列 X を各ノードが保持する配列に掛けることによってその配列の状態を変化させる．よって，小行列 X_{ele} の一辺の大きさが $2^i \le 2^n/k$ のときはノード間の通信を行わずに量子ゲートのシミュレーションを行

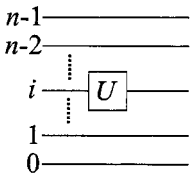

図 3　1 ビットユニタリゲートの概略図.

える．一方，小行列 X_{ele} の一辺の大きさが $2^i > 2^n/k$ のときは量子ゲートのシミュレーションを行うのにノード間で通信が発生する．このときの小行列 X_{ele} の一辺の大きさ 2^i と各ノードの通信相手について説明する．図 5 にその概略図を示す．まず，配列全体を 2^i の大きさごとのグループ $p(0 \leq p \leq 2^n/2^i - 1)$ に分割する．あるグループ p を取り出したとき，その拡大図が図 5 の下部となる．小行列 X_{ele} の各行において，要素 u_{11} と u_{12} に対応する 2 つのノードが通信を行い，要素 u_{21} と u_{22} に対応する 2 つのノードが通信を行う．小行列 X_{ele} の各行において，要素 u_{11} と u_{12} の間隔と要素 u_{21} と u_{22} の間隔は，常に $2^i/2$ である．よって，通信を行う 2 つのノードの間隔は常に，$2^i/2 \div 2^n/k = k2^{i-n-1}$ となる．故に図 5 より，ノード m が $pk2^{-i} \leq m < pk2^{-i} + k2^{i-n-1}$ のとき，ノード m はノード $m + k2^{i-n-1}$ と通信を行う．このとき，ユニタリゲート U の適用後のノード m が保持する配列の状態は，ノード m が保持する配列全てに u_{11} を掛けた値と，ノード $m + k2^{i-n-1}$ から送られてきた配列全体に u_{12} を掛けた値の和となる．ここで，$m + k2^{i-n-1}$ を m' と置く．よって，$|m \cdot 2^n/k\rangle, \cdots, |(m+1) \cdot 2^n/k - 1\rangle$ の確率振幅 $\alpha'_{m2^n/k}, \cdots, \alpha'_{(m+1)2^n/k-1}$ は，確率振幅 $\alpha_{m2^n/k}, \cdots, \alpha_{(m+1)2^n/k-1}$ と $\alpha_{m'2^n/k}, \cdots, \alpha_{(m'+1)2^n/k-1}$ を用いて，$u_{11}\alpha_{m2^n/k} + u_{12}\alpha_{m'2^n/k}, \cdots, u_{11}\alpha_{(m+1)2^n/k-1} + u_{12}\alpha_{(m'+1)2^n/k-1}$ と変化する．また，ノード m が $pk2^{-i} + k2^{i-n-1} \leq m \leq (p+1)k2^{-1} - 1$ のとき，ノード m はノード $m - k2^{i-n-1}$ と通信を行う．このとき，ユニタリゲート U の適用後のノード m が保持する配列の状態は，ノード m が保持する配列全てに u_{21} を掛けた値と，ノード $m - k2^{i-n-1}$ から送られてきた配列全体に u_{22} を掛けた値の和となる．ここで，$m - k2^{i-n-1}$ を m'' と置く．よって，$|m \cdot 2^n/k\rangle, \cdots, |(m+1) \cdot 2^n/k - 1\rangle$ の確率振幅 $\alpha'_{m2^n/k}, \cdots, \alpha'_{(m+1)2^n/k-1}$ は，確率振幅 $\alpha_{m2^n/k}, \cdots, \alpha_{(m+1)2^n/k-1}$ と $\alpha_{m''2^n/k}, \cdots, \alpha_{(m''+1)2^n/k-1}$ を用いて，$u_{11}\alpha_{m2^n/k} + u_{12}\alpha_{m''2^n/k}, \cdots, u_{11}\alpha_{(m+1)2^n/k-1} + u_{12}\alpha_{(m''+1)2^n/k-1}$ と変化する．ノード間で通信が発生しない場合の小行列 X_{ele} の一辺の大きさの最大値は $2^i = 2^n/k$ である．これは，ユニタリゲート U を量子ビット $n - logk - 1$ に適用する場合に当たる．ここで，$n - logk - 1$ を Non_Comm_bits と置く．よって，i が Non_Comm_bits よりも大きいとき，2 つの

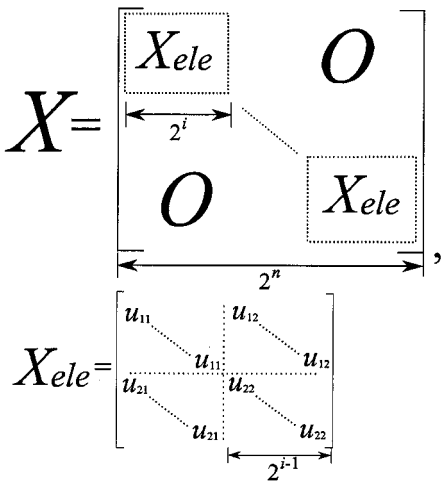

図 4　ユニタリ行列 X の全体図.

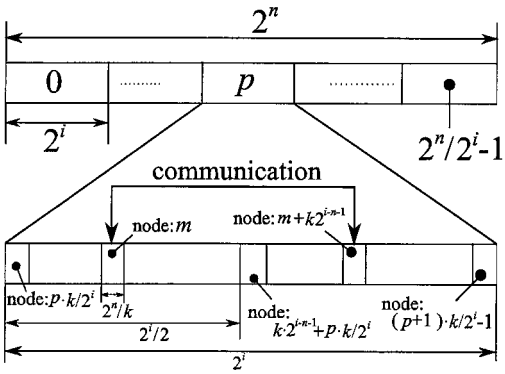

図 5　小行列 X_{ele} の一辺の大きさと各ノードの通信相手の関係.

ノードの組全てについて，それぞれが持つ配列の要素を全て交換することになる．故に，全てのノード間で通信される配列の要素数は，$2^n/k \times k = 2^n$ となる．

1 ビット制御ユニタリゲートは，ターゲットビットとコントロールビットによって構成される．$i, j(1 \leq i(j) \leq n(i \neq j))$ をそれぞれターゲットビット，コントロールビットの配置箇所とする．これを図にすると図 6 の様に表わされる．1 ビット制御ユニタリゲートは，コントロールビットの値が 1 の時にのみ，ユニタリ変換 U を i 番目の量子ビットに適用する変換のことである．1 ビットユニタリゲートのときと同様に，i が Non_Comm_bits 以下の場合は，各ノードで計算されるためノード間の通信は発生しない．しかし i が Non_Comm_bits よりも大きい場合は，各ノードが確率振幅の値を計算するのに他のノードが保持する配列の要素を必要とするため，ノード間で通信が発生する．コントロールビットビットが 1 である基底の個数は全体の半分であるため，通信発生時にノード間で通信さ

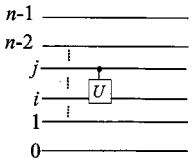

図 6　1 ビット制御ユニタリゲートの概略図.

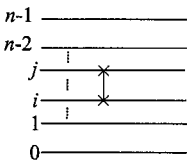

図 7　交換ゲートの概略図.

れる配列の要素数は 2^{n-1} となる.

交換ゲートとは，2 つの量子ビットの交換の操作を行う量子ゲートである．これを図にすると，図 7 の様に表わされる．量子ビット i,j を交換させたとすると，$|a_{n-1},a_{n-2},\ldots,a_i,\ldots,a_j,\ldots,a_1,a_0\rangle$ の確率振幅 $\alpha'_{a_{n-1}\ldots a_i\ldots a_j\ldots a_0}$ は

$$\alpha'_{a_{n-1}\ldots a_i\ldots a_j\ldots a_0} = \alpha_{a_{n-1}\ldots a_j\ldots a_i\ldots a_0} \qquad (5)$$

と変化する．同様に，$|a_{n-1},a_{n-2},\ldots,a_j,\ldots,a_i,\ldots,a_1,a_0\rangle$ の確率振幅 $\alpha'_{a_{n-1}\ldots a_j\ldots a_i\ldots a_0}$ は

$$\alpha'_{a_{n-1}\ldots a_j\ldots a_i\ldots a_0} = \alpha_{a_{n-1}\ldots a_i\ldots a_j\ldots a_0} \qquad (6)$$

と変化する．よって，交換ゲートによって交換される量子ビット i,j がどちらも Non_Comm_bits 以下の場合は，各ノード内で計算されるためノード間で通信が発生しない．しかし，交換ゲートによって交換される量子ビット i,j のどちらかが Non_Comm_bits よりも大きい場合は，他のノードが保持する配列の要素を必要とするため，ノード間で通信が発生する．i,j ビットがどちらも 0 若しくは 1 である基底の個数は $2^{n-2}\times 2 = 2^{n-1}$ である．よって，$(i,j)=(1,0)$ か $(0,1)$ である基底の個数は $2^n - 2^{n-1} = 2^{n-1}$ となるため，ノード間で通信される配列の要素数は 2^{n-1} となる．シミュレーション対象の量子回路にあらかじめ交換ゲートが存在する場合，シミュレーションを行う前に交換ゲートによって交換される 2 つの各量子ビット上に存在する量子ゲートの箇所を互いの移動先の量子ビット上に移動しておくことで，交換ゲートを適用することによるノード間の通信は発生しない．以上より，ノード間の通信量は，量子ゲートの個所によって決定されることが分かる．シミュレーションの計算速度を向上させるためには，ノード間の通信量を減少させることが必要である．交換ゲートを量子回路に意図的に挿入することで量子ゲートのユニタリ変換 U が適用される量子ビットを下位の量子ビットに移動させることが可能である．その結果，等価でかつ，ノード間の通信量を減少させた量子回路を作成することが可能になる．このことについて簡単な例を示す．図 8 に簡単な量子回路を示す．量子ビット 0 上に存在する量子ゲートを計算するときにのみノード間で通信が発生しないとする．このとき，量子ビット 1 上の量子ゲートを計算するのに必要な通信量は，$2^2/2\times 1$（1 ビット制御ユニタリゲート）$+2^2\times 1$（1 ビットユニタリゲート）=6 となる．この量子回路に対して交換ゲートを挿入すると図 9 の様になる．このとき，図 8,9 の

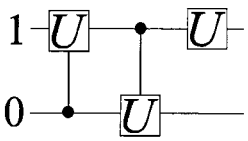

図 8　交換ゲートの挿入前.

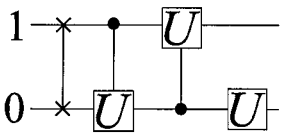

図 9　交換ゲートの挿入後.

量子回路はどちらも等価である．図 9 を計算するのに必要な通信量は，$2^2/2\times 1$（交換ゲート）$+2^2/2\times 1$（1 ビット制御ユニタリゲート）=4 となる．よって，図 9 を計算するのに必要な通信量は図 8 のそれよりも小さくなる．ただし，闇雲に交換ゲートを挿入してしまうと通信量は増加してしまう．次の節では，この様に通信量を効果的に減少させるための交換ゲートの挿入個所を自動的に探索するアルゴリズムを説明する．

2.3　交換ゲートの挿入箇所のアルゴリズム

本研究における交換ゲートの挿入箇所のアルゴリズムについて説明する．図 10 に本アルゴリズムの説明で用いる各変数の説明のための量子回路を示す．量子状態の時間発展方向へと量子ゲートを順番に注目していくとする．量子ビットを表す変数を $j(0\le j\le n-1)$ とする．量子ゲート数 N からなる量子回路に対して，各時間ステップ $i(0\le i\le N-1)$ で適用する量子ゲートを G_i とする．また，量子ゲート G_i のユニタリ変換 U が適用される量子ビットを $G_i\_bit_U$ とする．量子ビット $G_i\_bit_U$ 上において，量子ゲート G_i から時間発展方向を見たときに，時間ステップの最も早い量子ゲートを G_p とする．量子ゲート G_{i+1}, …, G_{p-1} のうち，量子ビット $G_i\_bit_U$ が Non_Comm_bits 以下である量子ゲートに着目する．それを時間ステップの早い量子ゲートから順に $G_{q(j)_0}, G_{q(j)_1}, \cdots$ とする．量子ゲート $G_{p+1},\cdots,G_{n-1}$ のうち量子ビット $G_i\_bit_U$ が Non_Comm_bits 以下である量子ゲートに着目する．それらの量子ゲートのうち，各量子ビットにおいて時間ステップが最も早い量子ゲートを $G_{r(j)}$ とする．量

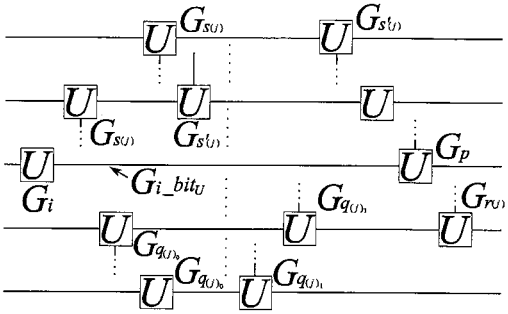

図 10 量子ゲートの箇所と変数の関係.

子ゲート $G_{i+1}, \cdots, G_{p-1}$ のうち量子ビット $G_i\_bit_U$ が Non_Comm_bits よりも大きくなる量子ゲートに注目する．それらの量子ゲートのうち，各量子ビットにおいて時間ステップが最も早い量子ゲートを $G_{s(j)}$ とする．さらに，量子ゲート $G_{s(j)}$ から時間発展方向を見たときに，時間ステップの最も早い量子ゲートを $G_{s'(j)}$ とする．

各量子ゲートの種類によって重み H を与える．前節より，$G_i\_bit_U$ が Non_Comm_bits よりも大きい場合，G_i が 1 ビットユニタリゲートならば 2^n の通信量が，また G_i が 1 ビット制御ユニタリゲートならば $2^n/2$ の通信量が必要である．そこで，1 ビットユニタリゲートに対しては 2 の重みを，また 1 ビット制御ユニタリゲートに対しては 1 の重みを与える．

仮に，交換ゲートによって，量子ゲート G_i, G_p を Non_Comm_bits 以下のある量子ビット j' へと移動したとする．その後，量子ビット j' への交換ゲートの再挿入は注意する必要がある．前節で，交換ゲートを挿入するとその交換ゲートを適用するのにノード間で通信が発生することを説明した．そのため，交換ゲートの挿入の効果が得られるのは，時間ステップ p 以降となる．故に，量子ビット j' への交換ゲートの再挿入は，量子ゲート G_p の後にする必要がある．これは，量子ゲート G_i, G_p を Non_Comm_bits 以下のある量子ビット j' へと移動した後，時間ステップ i から p までの間は量子ビット j' の操作を禁止すると言い換えることができる．

量子ゲートを $G_{s(j)}$ が存在する量子ビット数に 1 を足した値を A とする．1 を足しているのは，$G_i\_bit_U$ を考慮しているためである．計算を行うのにノード間で通信が発生しないビット数 $n - \log k$ から操作を禁止している量子ビット数を引いた値を B とする．値 B よりも値 A が大きい場合，値 B に対応する量子ビットに対して優先順位を与えて例外処理を行う．まずは，$A \leq B$ の場合の処理について説明する．量子ゲート G_i, G_p の重みの合計値を H_1 とする．次に，Non_Comm_bits 以下でかつ，操作が禁止されていない各量子ビットにおける量子ゲート $G_{q(j)_0}, G_{q(j)_1}, \cdots$ の重みの合計値を $H_2(j)$ とする．交換ゲートを適用するのにノード間で $2^n/2$ 通信が発生することを考慮すると，重みの合計値の差 $H_3 = H_1 - H_2(j)$ が 2 以上となる量子ビットに対して交換ゲートを挿入すると，量子回路を計算するのに必要な通信量は減少する．よって，その量子ビットは交換ゲートの挿入の対象とする．そこで，重みの合計値の差 H_3 が 2 以上で，かつ量子ゲート $G_i, G_{q(j)_0}$ との時間ステップ間隔 $q(j)_0 - i$ が最も大きくなる量子ビットと量子ビット $G_i\_bit_U$ との間に交換ゲートを挿入する．ただし，量子ゲート $G_{q(j)_x}$ が存在しない場合は，量子ゲート $G_{r(j)}$ を量子ゲート $G_{q(j)_0}$ と置き換えて，時間ステップ間隔を $r(j) - i$ として計算する．また，量子ゲート $G_{q(j)_0}$，$G_r(j)$ がどちらも存在しない場合は，量子ゲート G_i との時間ステップ間隔を無限大とみなす．つまり，量子ゲート $G_{q(j)_0}$，$G_r(j)$ がどちらも存在しない量子ビットを優先的に選ぶ．交換ゲートの挿入後，先ほど述べた通り時間ステップ i から p まで，量子ゲート G_p が存在する量子ビットの操作を禁止する．

次に，$A > B$ の場合の処理である例外処理について説明する．まず，各量子ビットにおける量子ゲート $G_{s(j)}, G_{s'(j)}$ の重みの合計値 $H(j)$ を計算する．仮に，交換ゲートによって量子ゲート $G_{s(j)}, G_{s'(j)}$ を Non_Comm_bits 以下のいづれかの量子ビットに移動させたとする．その後，量子ゲート $G_{s'(j)}$ の後にのみ，その量子ビットへの交換ゲートの再挿入を許していた．量子ゲート G_i から見たとき，量子ゲート $G_i, G_{s'(j)}$ の時間ステップ間隔が小さいほど，この交換ゲートの再挿入が許されるまでに必要な時間ステップ数は小さくなる．そこで，重みの合計 H が 2 以上でかつ，量子ゲート $G_i, G_{s'(j)}$ の時間ステップ間隔 $s'(j) - i$ が最も小さくなる量子ビットを優先的に交換ゲートの挿入の対象とする．この量子ビットと Non_Comm_bits 以下のどの量子ビットとの間に交換ゲートを挿入するかについての処理は，$A \leq B$ の場合の処理と同じである．

3. 評　　価

本研究における交換ゲートの挿入のアルゴリズムの性能を実験によって調べた．その実験では，量子ゲートをランダムに配置することによって作成した量子回路に提案手法を適用させたときのノード間の通信量の減少率を調べた．BioServer[3] に実装することを想定し，ノード数は 128 と，量子ビット数は 28 とした．量子ゲート数を 1000〜10000 の間で 1000 単位で変化させ，量子ゲートをランダムに配置することによって量子回路を作成した．また，各量子ゲート数において，量子回路を 100 回作成した．量子回路の作成後，提案手法の適用前と後のノード間の通信量と量子ゲート数の平均値を計算した．表 1，2 に実験結果を示す．表

表 1　実験結果（通信量の変化）.

量子ゲート数	提案手法未適用時 通信量 [Gbyte]	提案手法適用時 通信量 [Gbyte]	減少率 [%]
1000	813.1	296	63.6
2000	1593.7	634.3	60.2
3000	2410.7	920.9	61.8
4000	3235.8	1226.4	62.1
5000	4019.7	1535.5	61.8
6000	4840.5	1844.2	61.9
7000	5650.7	2147.3	62
8000	6459.6	2448.2	62.1
9000	7250	2740.5	62.2
10000	8049.9	3067	61.9

表 2　実験結果（交換ゲートの挿入後の量子ゲート数の変化）.

量子ゲート数		
提案手法未適用時	提案手法適用時	増加率 [%]
1000	1062	6.2
2000	2128	6.4
3000	3194	6.5
4000	4256	6.4
5000	5323	6.5
6000	6391	6.5
7000	7453	6.5
8000	8516	6.5
9000	9581	6.5
10000	10643	6.4

1 より，作成した量子回路に提案手法を適用することによって，回路を計算するのに必要な通信量は，提案手法の適用前のそれに比べて半分以下に小さくなっていることが分かる．また表 2 より，量子ゲート数に比例して交換ゲート数が増加していることも分かる．そして表 2 より，交換ゲートを挿入した後の量子回路の量子ゲート数は，交換ゲートを挿入する前の量子回路の量子ゲート数のそれと比べて，約 6%の量子ゲート数が増加していることが分かる．しかし，シミュレーションを行ったときに，ノード間の通信時間が実行時間に最も影響を与えているため，この量子ゲート数の増加が実行時間に与える影響は小さいと考えられる．

量子ゲート数を増加させても，通信量の減少率が安定している理由は次のように考えられる．量子ビット数が 28 で，128 ノードの並列計算機を用いることを想定していた．つまり，各ノードは 7 つの基底に対応するため，最下位ビットから 21 ビットに対応する量子ゲートを適用するのにノード間で通信が発生しない．故に，ノード間で通信が発生する量子ビット数よりも，ノード間で通信が発生しない量子ビット数の方が十分に大きい．その結果，量子ゲート数が増加しても提案手法の効果が得られたと考えられる．

4. まとめと今後の予定

分散メモリ型並列計算機における量子計算シミュレーションの高速化のために，量子回路に交換ゲートを意図的に挿入することで，ノード間の通信量の少ない等価な量子回路に変換する手法を提案した．また，提案手法を汎用 PC で実験した．その結果，提案手法を適用することによって，量子回路を計算するのに必要な通信量が半分以下に減少することが分かった．また，量子ゲート数に比例して，挿入する交換ゲート数が増加しているが，実行時間に与える影響が非常に少ないことが分かった．今後は，BioServer[3] に提案手法に基づく量子計算シミュレータを実装し，その性能の評価を行っていく予定である．また，本提案手法を有効的に活用できる量子アルゴリズムの調査も並行に行っていく予定である．

謝辞　本研究は，本研究は科研費（19300012, 19700010, 18700011）の助成を受けたものである．また，富士通株式会社バイオ IT 事業開発本部イノベーション Shunsaku プロジェクトの協力によって行われたものである．

参 考 文 献

1) L.K.Grover, A Fast Quantum Mechanical Algorithm for Database Search,quant-ph/9605043, (November 1996).
2) Peter W. Shor, Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, quant-ph/9508027, (January 1996).
3) 富士通株式会社, FIND Vol.22 No.1, 24-26 http://edevice.fujitsu.com/jp/catalog/find/22-1j/, (2004).
4) 富士通株式会社, http://jp.fujitsu.com/microelectronics/products/micom/frv/hard/fr550/.
5) 株式会社アックス,http://www.axlinux.com/.
6) Igor L. Markov, John P. Hayes, Improving Gate -Level Simulation of Quantum Circuits, George F. Viamontes, quant-ph/0309060v2, (2003).
7) George F. Viamontes, Igor L. Markov, John P. Hayes, Graph-based simulation of quantum computation in the density matrix representation, quant-ph/0403114v2, (2004).
8) K. De Raedt, K. Michielsen, H. De Raedt, B. Trieu, G. Arnold, M. Richter, Th. Lippert, H. Watanabe, N. Ito, Massive Parallel Quantum Computer Simulator, quant-ph/0608239v1, (2006).
9) Jumpei Niwa, Keiji Matsumoto, Hiroshi Imai, General-Purpose Parallel Simulator for Quantum Computing, quant-ph/0201042v1, (2002).