A-026

インターネット上の遊休 PC を利用した量子探索シミュレータの開発

Grover's Algorithm over Internet

内藤 昌彦[†a)]　　築地 立家[††]

Masahiko Naito　　Tatsuie Tsukiji

1. まえがき

ブロードバンド環境でのインターネット利用者の人口普及率は 75.3%に達しており，ホームページの閲覧・メールの送受信等に利用されている．これらのコンシューマ PC の遊休時間を活用すれば，コストをかけずに，スーパーコンピュータ並の計算能力を獲得できることが，知られている．実際，SETI@HOME，タンパク質配列探索，分子構造探索，暗号鍵探索，メルセンヌ数探索等のパラメータ探索問題は，探索領域を分割して各 PC に振り分けるという，オープンループ型の分散計算処理によって，大規模な問題が解かれている．しかしながら，例えばフーリエ変換に見られるような，分割不可能な問題をクローズドループ型で処理する場合，通信速度オーバーヘッドが大きく，各 PC の計算速度・通信速度が不揃いであり，かつ各 PC がシステムから任意に入退出するような環境においては，システム構築の際に特別な工夫が要求される．

そこで，本実験では，システム構築の事例として，インターネット上のコンシューマ PC を活用して，Grover 量子探索アルゴリズムをシミュレーションする．Grover アルゴリズムは，N 次元空間上に一定のユニタリ変換を繰り返し施す．この繰り返し演算を k 台のクライアントでシミュレーションするために，次元分割 $N=N_1+N_2+\cdots+N_k$ を行って，各クライアントには N_i 次元分の処理を分担させて，その結果をサーバに報告する．サーバは，それらの報告結果を集計して次のループ処理のためのパラメータを決定し，各クライアントに通知する．幸い，Grover アルゴリズムにおいては，クライアント-サーバ間の通信量が小さく済むため，原理的には，クライアントの台数を増やせば，より大きなサイズ N の処理を行うことができる．その際，通信オーバーヘッドがボトルネックとならない程度に，各クライアントの分担次元数 N_i を大きくとることが望まれる．さらに，任意のクライアントが任意のタイミングでシステムに参入・離脱する状況に対処するため，次元の一部が突然消滅・生成するような空間上での量子探索シミュレーション手法を新たに提案して，本実験システムに導入する必要がある．

量子状態は不安定なので，量子演算の途中で量子エラーが発生すると考えられる．Grover アルゴリズムに対する解析の多くは，N 次元空間をマーク基底が張る空間とその他の基底が張る空間に直和分解した上で，便宜上，それぞれの空間上で均一の量子エラーが発生するものと仮定する[2,3,4,6]．一方，実際は，N 個の各基底上で不均一な量子エラーが発生するモデルも考えられる[1,5]．ただし，このモデルは解析が困難なため，数値実験を行う必要が生じる．しかし，実験に必要なメモリーサイズは N に比例するため，1 台の PC で可能な実験規模には限度がある．一般に，量子計算がもたらす高速化は次元数 N が大きいほどに威力を発揮するので，より大きな N について実験を実施するために，インターネット上のコンシューマ PC が持つメモリ資源を活用したい．

今回は，異なるブロードバンド環境に配置された 5 台のコンシューマ PC を用意して，システムを構築して，実験を実施した．その際に使用したネットワーク通信 API は，Globus を利用したオープンソースのミドルウェアである Ninf-G が提供するクライアント-サーバサービスである．ところで，Ninf-G は，クライアントには負担をかけないようなサービスを想定しているため，1 クラアント多サーバでの使用となっている．一方，本実験システムでは，クライアントがス

† 東京電機大学先端科学技術研究科，東京都
Graduate School of Advanced Science and Technology, Tokyo Denki University, 2-2 Kandanishiki-chou, Tiyoda-ku, Tokyo, 101-8457 Japan
†† 東京電機大学理工学研究科
Graduate School of Science and Engineering, Tokyo Denki University, Isizaka, Hatoyma-chou, Hiki-gun,Saitama, 350-0394 Japan
a) E-mail: 09udj02@ms.dendai.ac.jp

クリーンセーバー等の起動をきっかけにサーバにアクセスして，サーバから依頼された処理を分担処理する．そのため，多クライアント 1 サーバ型のネットワーク構成とした．Globus は，暗号化・複合化を伴う安全な通信機能を持つ反面，通信処理速度が遅い．そこで，各クライアント依頼する次元数 N_i を自動調整して比較的大きな次元数の問題を解かせることにより，通信遅延がボトルネックとなる事態を避けることができた．

この次元数調整システムは，各クライアントのパフォーマンスを計測しながら，適切な次元数を割り当てるため，パフォーマンスが異なるブロードバンド環境が混在するシステムにおいて，負荷の均一化を自動調整することもできた．

2. Grover アルゴリズムシミュレーション

Grover アルゴリズムは，N 次元の複素空間に作用する位相変換行列 F と平均値を中心とした振幅の折り返し C の積 CF を繰り返し施す操作である．具体的には，N 個の基底 $|i\rangle$, $i=0,1,\ldots,N-1$ の中の $|s\rangle$ を唯一つのマーク基底とし，さらに $|t\rangle=\frac{1}{\sqrt{N}}\sum|i\rangle$, ただし $\sum|i\rangle$ は全基底の和，とすると，$F=-2|s\rangle\langle s|+\sum|i\rangle$, $C=-2|t\rangle\langle t|+\sum|i\rangle$ である．

また，ガウスノイズの与え方は，現在の量子状態に対して $\sum\delta|i\rangle\langle i|$, ただし $\delta=A\cos(2\pi x_i)$, を加算してから，正規化すればよい．ただし，A は 0 以上 1 以下の実数であり，ノイズ量と呼ぶことにする．

本実験システムでは，積 CF を施す操作を多クライアント 1 サーバ型の分散計算システム上で図 1.1～図 1.5 のように実現する．ただし，図中では，*N=16, s=0001*(2 進数)，クライアントは 5 台 (IP=192.168.1.3～7)である．

ステップ 1.　各クライアントは，現在の振幅にガウスノイズを与える．各基底の初期振幅値は同一とする．

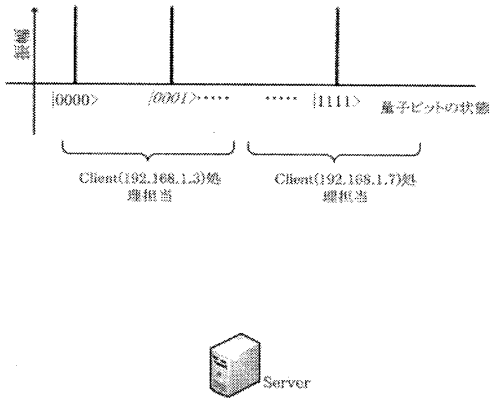

図 1.1 初期化

Figure 1.1　Initialize

ステップ 2.　マーク基底の位相だけを反転する．

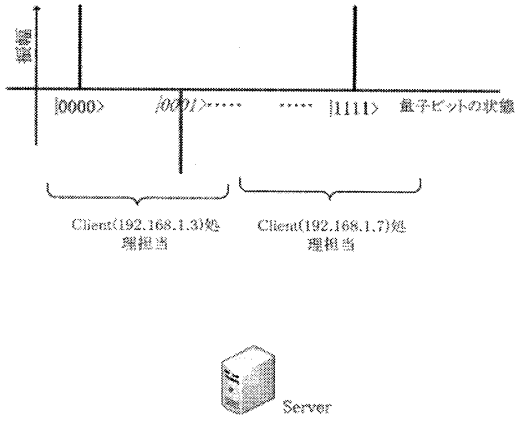

図 1.2 反転

Figure 1.2　Reverse

ステップ 3.　各クライアントは，担当する次元基底の振幅の小計を求めて，RPC(Remote Procedure Call)関数呼び出しによりサーバに送付する．

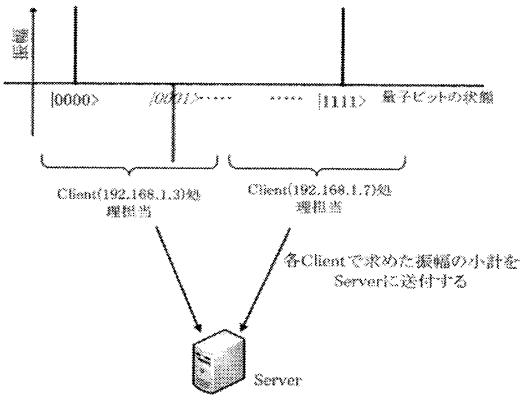

図 1.3 関数呼び出し

Figure 1.3　Call Function

ステップ 4. 各クライアントからの小計を全て受信した後に，振幅の平均値を求めて，RPC 関数を復帰させることにより，各クライアントに送付する．

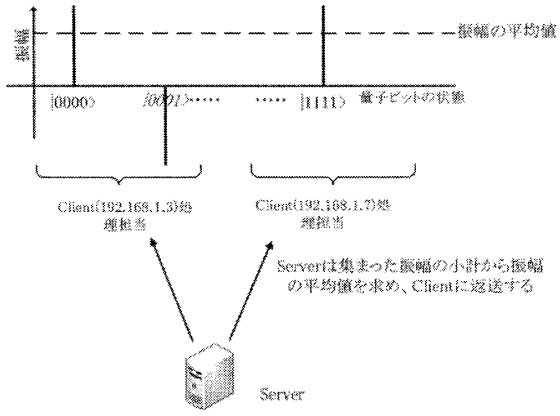

図 1.4 関数復帰

Figure 1.4 Return Main

ステップ 5. 各クライアントは，平均値を中心にして，振幅値を折り返す．そして，ステップ 1 に戻る．

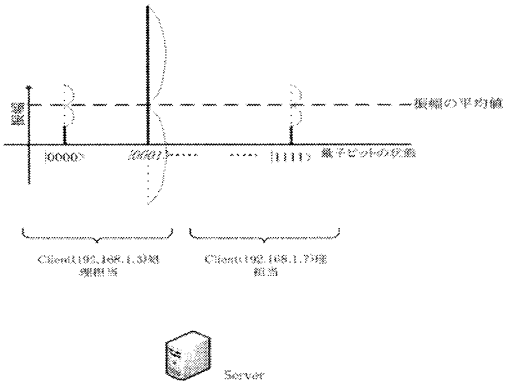

図 1.5 振幅値折り返し

Figure 1.5 Amplitude Back

3. 1 クライアント多サーバ VS 多クライアント 1 サーバ

Ninf-G パッケージが提供するサンプルプログラムは 1 クライアント多サーバ型のネットワーク構成となっている．一方，本実験システムは，多クライアント 1 サーバ型を採用した．その妥当性を確かめるために，両ネットワーク構成の処理性能を比較する実験を実施した．1 クライアント多サーバ型のネットワークでは，ステップ 4 においてクライアントが各サーバからの小計を集めて集計を行う．その際，各サーバからの RPC 復帰開始時刻がまちまちならば，待ち時間が発生すると思われる．実際，閉じた LAN 環境に 5 台のサーバを接続して，各サーバの復帰開始時刻から全ての復帰が完了するまでの時刻の差分を 5 回計測した結果が，表 1 である．この結果から，各サーバの復帰開始時刻には 80ms 程度の差分の積み上げが生じている．これは，シングルタスクで逐次的に RPC を各サーバに対して発したために，サーバ 1 台あたり 80ms 程度の RPC 処理時間が積み上がったと考えられる．従って，サーバの台数が無制限に増えるとすると，RPC 処理時間も無制限に増えてしまうことが予想される．

表 1 各サーバの復帰開始から全復帰が完了するまでの時間

Table 1 Waiting time of each server's return from RPC.

回数 / サーバ名	1	2	3	4	5	平均
192.168.1.3	502.00	425.00	408.00	410.00	741.00	497.20
192.168.1.4	314.00	341.00	322.00	326.00	323.00	325.20
192.168.1.5	230.00	249.00	237.00	239.00	235.00	238.00
192.168.1.6	160.00	171.00	164.00	166.00	160.00	164.20
192.168.1.7	78.00	79.00	80.00	83.00	76.00	79.20

(単位:ms)

次に，同一の閉じた LAN 環境において，1 サーバ 10 クライアント（新方式）と 10 サーバ 1 クライアント型（旧方式）の両者を構築して，Grover シミュレータを走らせた時の処理時間を比較したのが，表 2 である．台数に比例して，旧/新の処理時間比も増大している．これは，新方式では，旧方式のような台数に比例した RPC 逐次処理の積み上げによる通信遅延が発生しないためと考えられる．この結果，新方式の旧方式に対する優位性が確かめられた．

表 2 1 サーバ 10 クライアント VS 10 クライアント 1 サーバ

Table 2 Processing times of 1Server-10Client and 10Client-1Server

処理方式 / 桁数	新処理方式 (Server=1台)	旧処理方式 (Server=10台)	旧／新 (倍)
N=18 (2^{18})	36517.3	195999.7	5.367311
N=22 (2^{22})	163934.1	777855.2	4.744926
N=25 (2^{25})	731547.2	2330637.6	3.185902

（単位：ms）

4. ソフトウェア構成

本実験システムは，クライアントをワーカ，サーバをマスタとする，マスタワーカ型の分散処理システムといえる[7]．これにより，RPC は，クライアント側からのサーバ関数起動として，実装される．ただし，この RPC が復帰してしまうと，呼び出されたサーバ関数が初期化されて処理結果を保持できないため，クローズドループ型の処理が行えない．そこで， RPC 呼び出しで起動する通信処理関数の外に，集計処理プログラムをサーバに常駐させて，両者をプロセス間通信で接続することにより，演算の途中経過を保持する

ことにした．集計処理プログラムは，すべてのクライアントからデータを受け取った後に集計処理を開始し，次のループのためのパラメータを算出してから，各クライアントからの RPC を一斉に復帰させるので，この通信処理機構により，遠隔地に離散配置されたクライアントの処理プロセスを同期させることもできた．

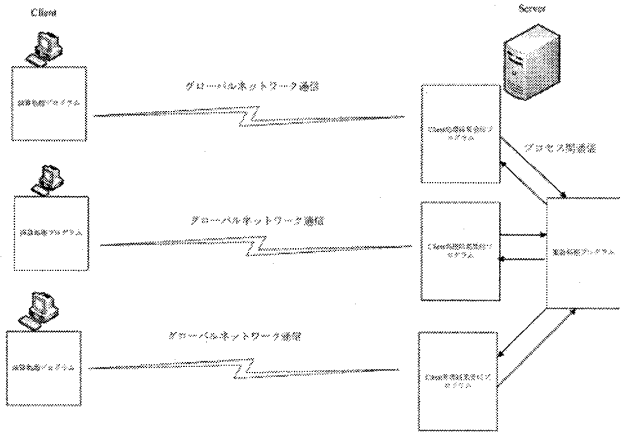

図 2　クライアント・サーバ構成図

Figure 2　Client-Server System

サーバーは演算処理のため参入してきたクライアントを管理するため，各クライアントに設定した個別の数値（クライアント番号）をメモリに登録する．クライアント番号を受け取ったサーバーは演算処理実行に必要な初期値を戻り値としてクライアントに戻す．

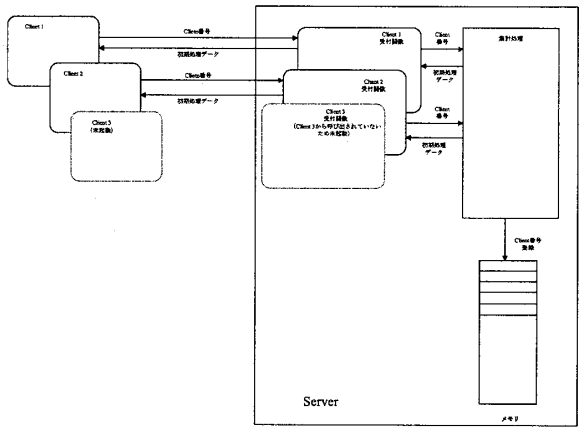

図 3 処理開始時のクライアン登録

Figure 3 Client Registration at process start

演算実行中は処理結果と共にクライアント番号をパラメータとして引き渡し，処理開始時に登録した番号と一致するかチェックする事で途中演算処理参加者の管理を行う．

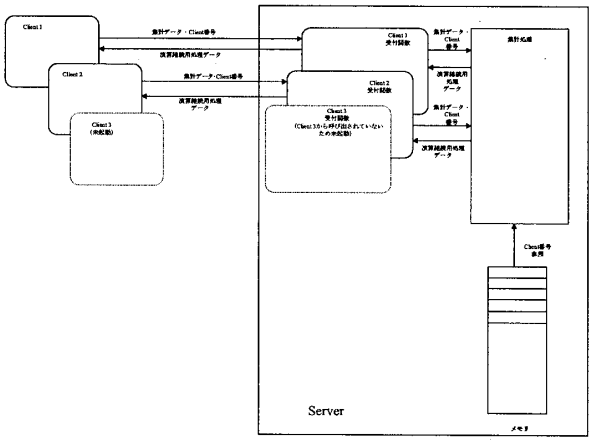

図 4 演算中のクライアント管理

Figure 4 Process Client Check

途中演算処理参加者は他のクライアントと同様にクライアント番号を呼び出しパラメータとしてサーバーを呼び出す．サーバーは参加時点での演算処理実行に必要な初期値を戻り値として途中参加クライアントに戻す．

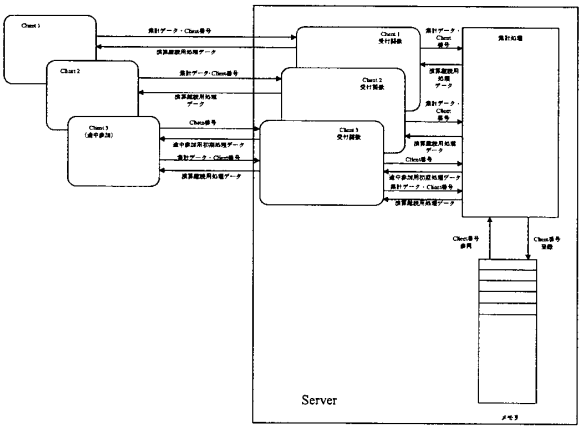

図 5　途中参加クライアント発生

Figure 5　Client experience while attending

サーバーは集計データ受信時タイマで監視を行い，通信の異常やハングアップ等の発生により無応答になってしまったクライアントの発生をチェックする．

タイムアウトが発生し演算を継続出来なくなったクライアントが発生した場合は無反応になったクライアント番号をメモリから削除する．メモリから削除されたクライアントが復活し，集計データを送付してきた場合は終了指示を送る．以上の方法により継続演算中若しくは新規参入の方法で参加してきたクライアント以外を Reject することで演算環境を保護する．

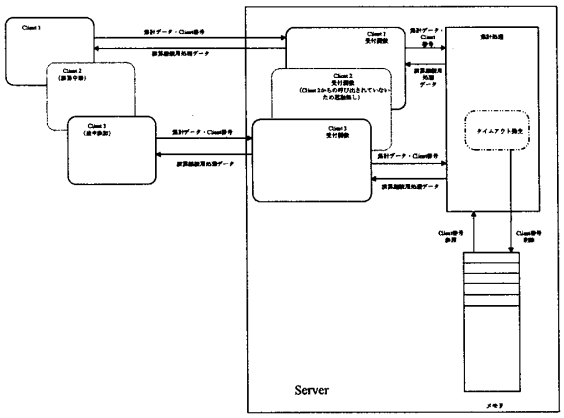

図 6　クライアント Timeout

Figure 6　Client Time-out

5. 実験システムの構成

表 3 のように，1 台のサーバと4台のクラアントを，異なるブロードバンド環境に配置して，Grover アルゴリズムシミュレーションの分散計算実験を行った．

表 3　各 PC の性能とネットワーク環境

Table 3　Specifications of PCs and their networks

機種名	@Sycon	@Sycon	Dell GX620	Dell GX520	Dell GX520
CPU	Pentium4	Pentium4	Pentium4	Pentium4	Pentium4
Clock	2.8GHz	2.8GHz	2.8GHz	2.8GHz	2.8GHz
回線種別	CATV	CATV	B-Flet's	ADSL	ADSL
回線速度	160Mbps	160Mbps	100Mbps	26Mbps	24MBps
Domain	dendai.ddo.jp	informatics.ddo.jp	tatsuie.ddo.jp	searoman.ddo.jp	yoshinori.ddo.jp
IP	116.64.95.99	119.172.226.251	121.93.52.142	220.99.31.22	220.209.94.172
役割	Server	Client	Client	Client	Client

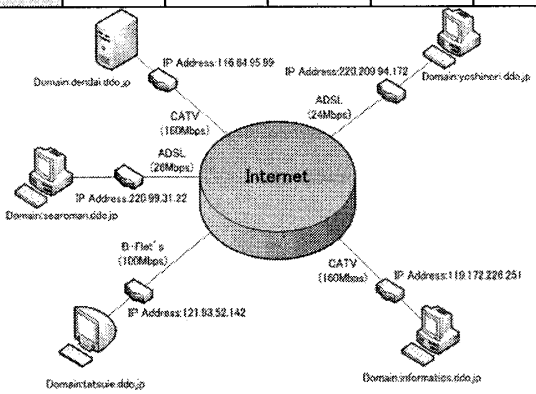

図 7　ネットワーク接続図

Figure 7　Network Connection

6. 通信ボトルネック

各クライアントに均等の次元数（2 進数桁で表示）を処理させたときの，Grover アルゴリズムのステップ 1～5 までの処理時間（全体）とその時に各クライアントが要した処理時間を ms 単位で計測したものが表 4 である．クライアントの処理時間は，サーバがクライアントからの RPC を復帰させてから次の RPC を受理するまでの差分時間を計測しているため，通信時間も含まれる．

いずれの次元数においても，yoshinori の処理時間と全体との差分は小さく，サーバの処理時間はそれ以下である．とくに次元数が 17 桁まではサーバの処理時間は 2ms 未満である．また，13 桁から 19 桁まで次元数が 64 倍ふえたにもかかわらず，全体処理時間が 1.34 倍しか増えていないのは，通信処理時間が次元数によらず一定の大きさをもつためであろう．また，19 桁から 21 桁にかけて各クライアントおよび全体の処理時間が大きく増大しているのは，計算処理時間が通信処理時間以上に大きくなったためと考えられる．

表 4　増大する次元数に対する各クライアントの処理時間

Table 4　Each client's processing time for increasing input size

次元数（2進数桁）	13	15	17	19	21
informatics	491.33	496.48	534.97	679.74	1183.11
tatsuie	612.36	627.19	666.94	816.4	1309.46
searoman	666.97	724.97	924.71	1002.01	1411.89
yoshinori	918.2	965.03	1158.13	1189.38	1530.84
全体	918.52	966.29	1160.17	1257.59	1602.42

（単位：ms）

7. 次元数の自動調整による負荷分散

通信ボトルネックにみあう大きさの次元数を確保し，かつクライアントの負荷分散を図るために，下の状態遷移図に従ってソフトウェアを再設計して，サーバが毎回の Return からその直後の Call までの時間計測によりクライアントの処理時間をモニタリングして，フェーズ 1 において次元数の自動調整を行った．

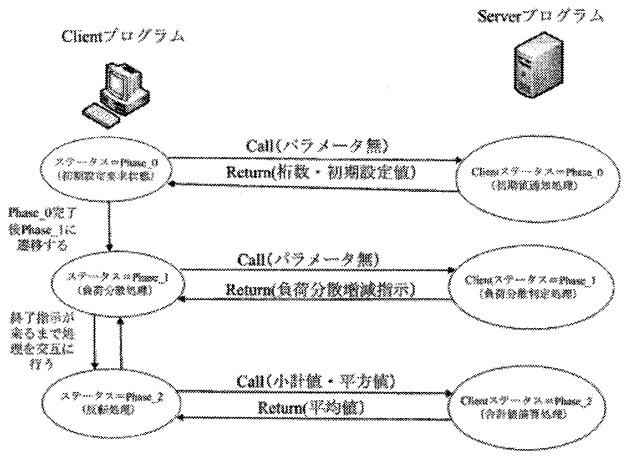

図 8　状態遷移図

Figure 8　State Transition Diagram

図 9 は，Grover アルゴリズムの Step1～5 を 400 回ループさせた時に，各クライアントの処理時間が *1000ms±100ms* に収まるように次元数調整をしたときのループ毎の各クライアント処理時間のグラフであり，図 10 はそのときの全体次元数のグラフ，図 11 はそのときのマーク基底観測確率である．ただし，ノイズ量は 0.05 とし，初期の各クライアントの次元数を 8192 次元（13 桁）から始めて，各ループで各クライアントの次元数を 8192 次元（13 桁）だけ増減させた．結果的に，130 回程度ループした後に，各クライアントは 900～1000ms 程度の処理時間に収束して，かつ全次元数は 2.4×10^6 次元程度（約 21 桁）に収束した．

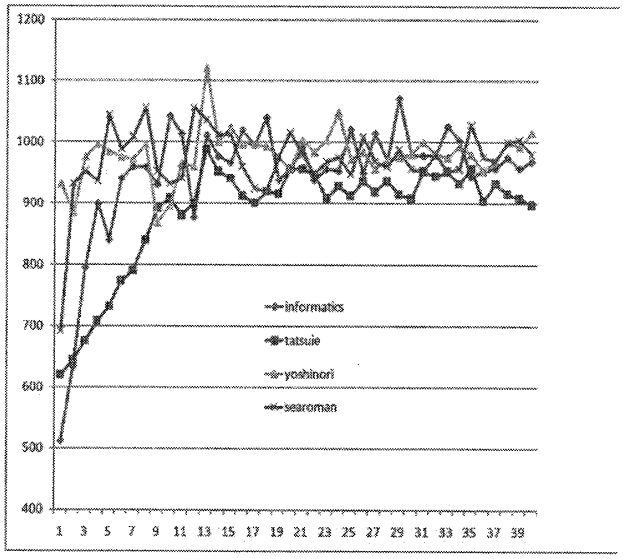

図 9　各クライントの処理時間（横軸：ループ数/10 縦軸：ms）

Figure 9　Each client's processing time per iteration

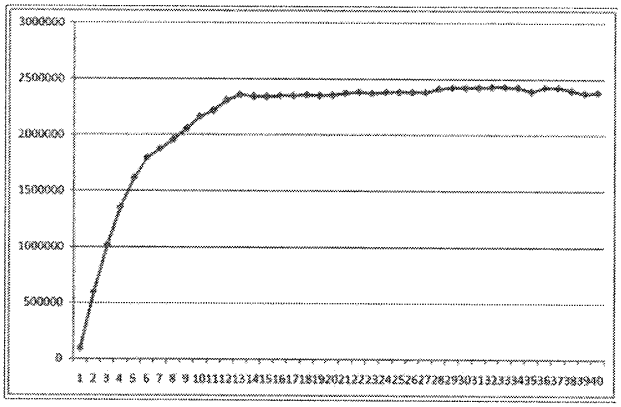

図 10　全体の次元数（横軸：ループ数/10 縦軸：次元数）

Figure 10　Problem size per iteration

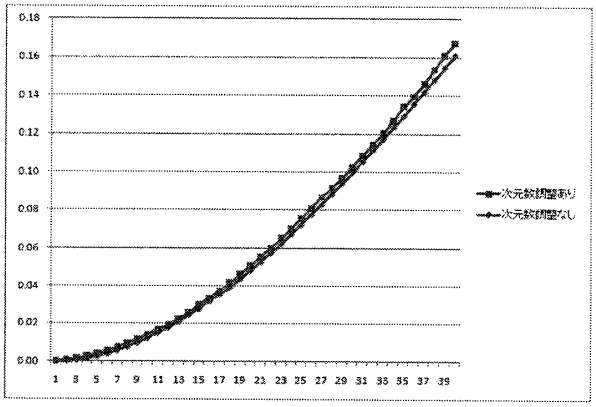

図 11　マーク基底観測確率（横軸：ループ数/10 縦軸：確率）

Figure 11　Probability of success per iteration

8. 次元の生成と消滅

一般家庭の PC 利用をするため，任意のクライアントが任意のタイミングでシステムに参入・離脱することが想定される．また，前章の次元数調整機能においても，クライアントの次元の一部が消滅・生成するような空間上での量子探索シミュレーション手法が必要である．本実験システムでは，マーク基底の振幅はサーバが保持し，非マーク基底の振幅はクライアントが保持するものとした．そして，サーバには，全基底の振幅の平均値と共に，非マーク基底の振幅の平均値も保持させた．クライアントの次元が新たに生成されるときに，その時点での非マーク基底の振幅の平均値振幅を，新次元の各基底振幅値に割り当てた．また，クライアントの次元が消滅するときには，その次元の基底振幅値は全て廃棄した．この設定を本実験システムに組み込んで，各ループでマーク基底の観測確率を計測した．図 11 は前章実験の際の観測確率（次元数調整あり），および各クライアントの次元数を 0.6×10^6 次元に固定したときの観測確率（次元数調整なし）である．図 12 はノイズ量を 0.3 にして，クラ

イアント次元数を 15 桁に固定して，Grover アルゴリズムを 900 回ループさせたときの観測確率（NoAbort），およびその途中の 200～600 回にかけて各クライアントを 1 度だけシステムから離脱・参入させたときの観測確率（Abort）である．図 13 は，ノイズ量を 0.05 に設定したときの同様のグラフである．いずれのグラフにおいても，Abort の場合の観測確率が NoAbort の場合よりも高くなっている．その理由は，クライアントが計算途中で離脱・参入することにより，その担当次元分の振幅値が非マーク基底平均値にリセットされるため，振幅値の分散が失われてしまった結果と思われる．

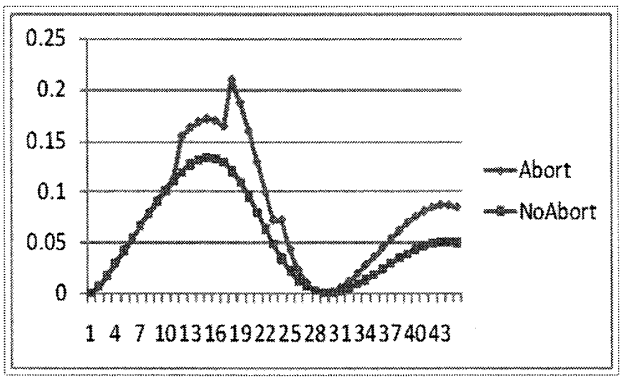

図 12　クライアント離脱・参入時の観測確率

（横軸：ループ数/20 縦軸：確率　ノイズ量：0.3）

Figure 12　Probability of success when each client aborts (A=0.3)

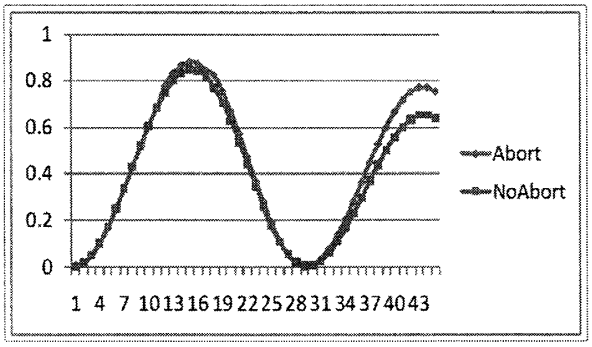

図 13　クライアント離脱・参入時の観測確率

（横軸：ループ数/20 縦軸：確率　ノイズ量：0.05）

Figure 13　Probability of success when each client aborts (A=0.05)

9. ガウスノイズと観測確率

各クライアントを常時接続し，その次元数を 13，17，21 桁に固定し，ノイズ量 A を各種設定して，Grover アルゴリズムを 900 回ループさせたときのマーク基底観測確率を下に示す．

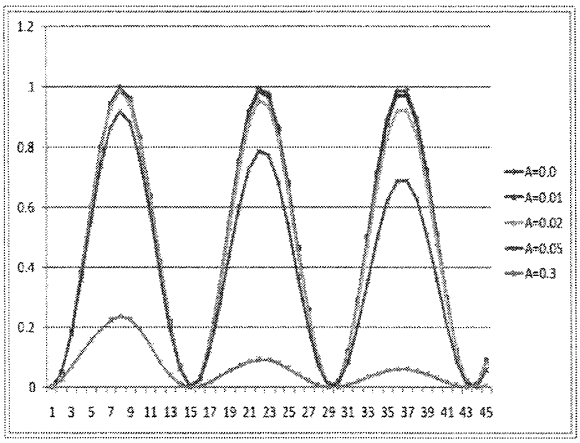

図 14　マーク基底観測確率

（横軸：ループ数/20　縦軸：確率　桁数：13）

Figure 14　Probability of success when each client has 13 bits of dimension

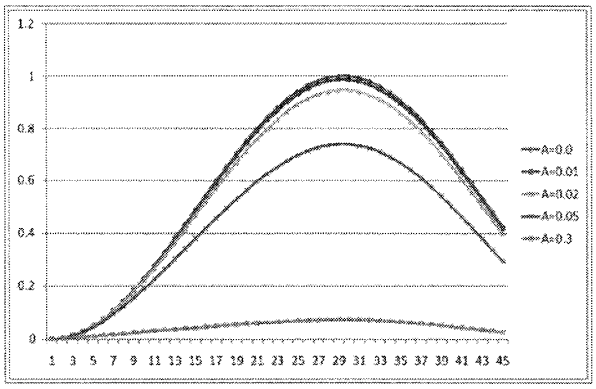

図 15　マーク基底観測確率

（横軸：ループ数/20　縦軸：確率桁数：17）

Figure 15　Probability of success when each client has 17 bits of dimension

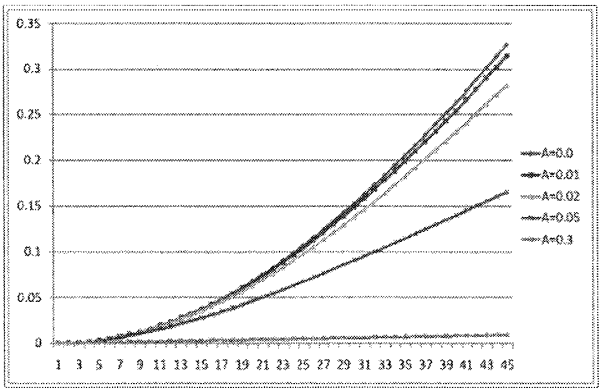

図 16　マーク基底観測確率

（横軸：ループ数/20 桁数：21　縦軸：確率）

Figure 16　Probability of success when each client has 21 bits of dimension

10. 今後の課題

今後，多くの一般利用者に参加してもらうには，クライアント側のワンクリックインストール，実験状況のモニタリング，探索解をランダムに配置してその発見者に報酬を与えるイベントの開催，等の各種サービスを充実させる必要がある．

また，クライアントの次元が新たに生成される際に，平均量だけでなく分散量も考慮した振幅の設定を行う必要があると考えられる．

文　　献

[1] M. Lei, D.J. Feng, L. Yun and L. Hui, "White Noise in Quantum Random Walk Search Algorithm," Chinese Physics Letters, vol. 23, no.4, pp. 779-782, 2006.

[2] Y. Li, L. Ma and J. Zhou, "Gate imperfection in the quantum random-walk search algorithm," J. Phys. A: Math. Gen., vol.39, pp. 9309-9319, 2006.

[3] G.L. Long, Y.S. Li, W.L. Zhang and C.C. Tu, "Dominant gate imperfection in Grover's quantum search algorithm," Phys. Rev. A, vol.61, pp. 042305--042309, 2000.

[4] A. Gábris, T. Kiss, and I. Jex, "Scattering quantum random-walk search with errors," Phys. Rev. A, vol.76, pp.062315-062324, 2007.

[5] J. Niwa, K. Matsumoto and H. Imai, "General-Purpose Parallel Simulator for Quantum Computing," Lecture Notes In Computer Science, vol.76, pp. 230 - 251, 2002.

[6] B.P. Norman and M.R. Altaba, "Noise in Grover's quantum search algorithm," Phys. Rev. A, vol.61, pp.012301-012305, 1999.

[7] 谷村 勇輔，廣安 知之，三木 光範，"グリッド計算環境でのマスターワーカシステムの構築，" 情報処理学会論文誌，コンピューティングシステム，vo.45, no.6, pp.197-207, 2004.