A-011

既存のプログラミング言語のための量子探索機構

A Quantum Search Mechanism for Existing Programming Languages

山下　義陽*　　桑原　寛明**　　國枝　義敏**
Yoshiharu Yamashita*　Hiroaki Kuwabara **　Yoshitoshi Kunieda**

1. はじめに

量子コンピュータは，従来のコンピュータで指数関数時間を必要とする特定の問題に，指数関数時間以下のアルゴリズムを適用できるハードウェアである．量子コンピュータを利用したアルゴリズムとして，ランダムなデータベースの探索アルゴリズムである Grover のアルゴリズム [1] が存在する．

Grover のアルゴリズムは，要素数 N の未整序なデータベースの探索を $O(\sqrt{N})$ で行う．プログラム中の線形探索を Grover のアルゴリズムを利用した探索 (以下，Grover の量子探索) に置き換えることで線形探索より少ないオーダーで探索を行うことができる．量子コンピュータは，量子論理ゲートを組み合わせた量子回路に従って演算を行う．量子論理ゲートは量子ビットに対する操作であり，プログラマがその操作を組み合わせ，量子回路を記述する．そのため，プログラマが量子コンピュータで演算を行うには，量子計算とアルゴリズムを理解する必要がある．プログラマにとって，量子計算とアルゴリズムの理解は大きな負担となるため，プログラマが量子コンピュータの仕組みや性質を意識せずとも，Grover の量子探索を行える仕組みが必要である．

本研究では，従来のプログラミング言語によるプログラムから Grover のアルゴリズムを利用した配列探索 (以下，Grover の量子配列探索) を実行するための機構を提案する．プログラマは，提案した機構に基づいて実装されたインタフェースを呼び出すことで $O(\sqrt{tN})$ の配列探索 (t は解の個数) をブラックボックスとして利用する．ブラックボックスの内部では，Grover の配列探索に必要な量子回路の生成，量子コンピュータへの実行要求，その実行結果の解析を行う．

2. 量子コンピュータ

量子コンピュータは，量子力学の法則に従って演算処理を行う非ノイマン型のコンピュータである．現在，ハードウェアの研究が行われており，従来のコンピュータのコプロセッサとして実用化されると予想されている [2]．従来のプロセッサでは処理に多くの時間を必要とする問題を，量子コプロセッサで短時間に処理が可能な場合に，量子コプロセッサがその問題を処理する．

量子コンピュータは，情報素子として量子ビットを使用し，計算モデルとして量子回路を利用する．量子ビットは，不確定な状態である重ね合わせ状態をとりうる．量子回路は，ユニタリー変換である量子論理ゲートから構成される．

2.1 Grover のアルゴリズム

Grover のアルゴリズムは L.K.Grover によって考案された未整序のデータベースに対する量子探索のアルゴリズムである [1]．このアルゴリズムは，要素数 N 個内に t 個の解が存在する場合，$O(\sqrt{N/t})$ の計算量と $O(logN)$ の記憶領域を消費して探索を行う．

2.2 量子数え上げ

あらかじめ解の個数を特定できている場合，Grover のアルゴリズムの適用回数である $\frac{\pi}{4}\sqrt{N/t}$ を求めることができる．量子数え上げ [3] は，どれが解か特定できていない状況であっても，N 個の要素の中で大まかな解の個数 t を $O(\sqrt{tN})$ の計算量で確率的に得ることができる [4]．

*立命館大学大学院理工学研究科
**立命館大学情報理工学部

```
1  int a[N];
2  int b[N];
3
4  for(i=0;i<N;i++)
5      if(a[i]==b[i]) pushStack(i,stack);
6
7  return stack;
```

図 1: 配列の線形探索の擬似コード

2.3 QACM

QACM(Quantum Addressable Classical Memory) は，データベースとしてデータを蓄える量子コンピュータ上のメモリである [5]．QACM を利用するには，あらかじめデータを格納する必要がある．QACM では，入力としてアドレスを与えることで，アドレスに対応するデータが出力される．指定するアドレスに重ね合わせ状態を用いることができる．アドレスを重ね合わせ状態で入力すると，指定されたアドレスに格納されたデータが重ね合わせ状態で出力される．

3. Grover の量子配列探索

本稿で扱う配列探索とは，配列 $a[N]$，$b[N]$ に対し $a[i] = b[i]$ となるすべての添え字 i を得ることである．逐次的に配列探索を行う擬似コードを図 1 に示す．

解となる添え字が t 個存在する場合，Grover の量子探索と量子数え上げを用いて $a[i] = b[i]$ を満たす t 個の添え字 i を $O(\sqrt{tN})$ で求める．

量子配列探索は，Grover の量子探索と量子数え上げを利用することで探索を行う．Grover の量子探索と量子数え上げを行う量子回路は，$a[i]$ と $b[i]$ の値を比較した結果を格納した量子ビットをもとに作用する．$a[i]$ と $b[i]$ の値を比較した結果を量子ビットに格納するために，QACM を組み込んだ量子回路を利用する [6]．

量子コンピュータ上で Grover の量子配列探索を行うためのアルゴリズムを以下に示す．

アルゴリズム

Step1. 量子数え上げによる解の個数の予測を行う．同じ予測値が 2 回得られるまで量子数え上げを繰り返す

Step2. 同じ解の個数の予測値を得られたなら，その予測値を t とする ($t = 0$ の場合，探索終了)

Step3. Grover の量子探索を行い，配列の添え字 i を得る

Step4. 得られた結果が正しい結果か判定するために a[i] と b[i] を比較する

Step5. 得られた結果が正しい場合は Step6 へ，間違っている場合は Step8 へ

Step6. 得られた結果をスタックに格納し，解を探索候補から除外する

Step7. t の値を 1 つ減らし，$t \leq 0$ なら Step1 へ，$t > 0$ なら Step3 へ

Step8. 5 回連続で Grover の量子探索を失敗した場合は Step1 へ，Grover の量子探索の連続失敗回数が 5 回より少ない場合は Step3 へ

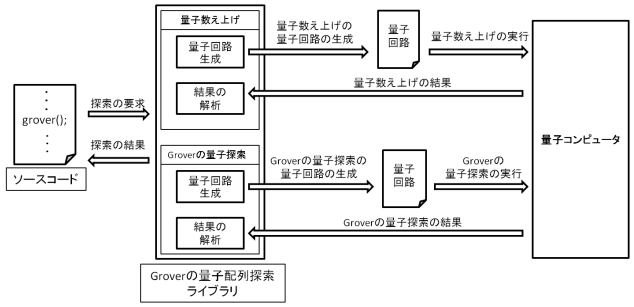

図 2: 提案機構の全体図

```
1  %/**
2  % * @param:a データ配列,array_size_A 要素数,
3  % *         b データ配列,array_size_B 要素数
4  % * @return:探索結果を格納した
5  % *          int型の配列の先頭アドレス
6  % **/
7  %int* grover(int *a, unsigned int array_size_A,
8  %            int *b, unsigned int array_size_B);
```

図 3: 関数 grover の仕様

4. 提案機構の実装

4.1 提案機構の構成

Grover の量子配列探索を行う提案機構の全体図を図 2 に示す．既存の言語で実装した Grover の量子配列探索のインタフェースが呼び出されると，Grover の量子配列探索ライブラリを通して量子数え上げと Grover の量子探索を量子コンピュータ上で実行する．量子数え上げと Grover の量子探索を実行するために，ライブラリ内部で量子回路を生成し，量子コンピュータは生成された量子回路に従って演算を行う．その演算結果をライブラリが受け取り，演算結果を解析する．最終的にライブラリがインタフェースの呼び出し元へ探索結果を返す．

4.2 関数 grover

量子配列探索を行う C 言語向けのインタフェースである関数 grover の仕様を図 3 に示す．関数 grover の引数は，探索対象となる配列 a と配列 b の先頭アドレスと，それぞれの配列のサイズである．戻り値は，$a[i] = b[i]$ を満たす添え字 i の値が入った int 型の配列の先頭アドレスを格納したポインタである．

4.3 実装環境

本研究では，プログラミング言語は C 言語と QCL(Quantum Computation Language)[7]，量子コンピュータとして QCL のシミュレータを使用し，提案機構を実装した．QCL は，アーキテクチャに依存しない量子計算向けプログラミング言語である．QCL の現在のシミュレータでは 32 量子ビットが使用できる量子ビットの上限となっている．

4.4 関数 grover の実行例

配列添え字 i={2,7} を解とした関数 grover の実行例を図 4 に示す．実行の際，途中経過を出力する．途中経過では，量子数え上げの結果 (Quantum Counting)，量子配列探索の結果 (GroverSearch) を出力する．

5. 考察

本研究で提案した機構では，既存の言語で実装したインタフェースを呼び出すことで，Grover の量子配列探索を実行する．量子コンピュータでの配列探索をブラックボックスとして利用することで，プログラマの労力を減らすことができる．

Grover のアルゴリズムを用いて t 個の解を求めた場合の Grover のアルゴリズムの適用回数は $\sum_{k=1}^{t} \frac{\pi}{4}\sqrt{N/k} \approx \frac{\pi}{2}\sqrt{tN}$ となる．量子数え上げのオーダーは $O(\sqrt{tN})$ である．量子数え上げを行った後に，Grover のアルゴリズムを用いて t 個の解を求めた場合のオーダーは $O(\sqrt{tN})$ となる．したがって，Grover の量子配列探索は解の個数が t 個の場合に $O(\sqrt{tN})$ となる．Grover の量子配列探索は線形探索と比較して優れていることがわかる．

```
1  $ ./sample
2  Quantum Counting:2
3  Quantum Counting:2
4  GroverSearch:7
5  GroverSearch is success.
6  GroverSearch:2
7  GroverSearch is success.
8  Quantum Counting:0
9  Quantum Counting:0
10 Search finished.
11 retult[0]:7
12 retult[1]:2
```

図 4: 実行例

Grover のアルゴリズムと量子数え上げは，確率的アルゴリズムである．確率的アルゴリズムを用いる場合は，最大計算量を減らすことが非常に重要となる．そのため，Grover の量子配列探索において際限なく探索が失敗する場合を想定しなければならない．際限なく探索が失敗する場合の対応策として，Grover の量子配列探索と並行して線形探索を行うことが考えられる．

6. おわりに

本稿では，既存のプログラミング言語での量子探索を利用した配列検索を量子コンピュータの仕組みや性質を意識せずともブラックボックスとして利用できる仕組みを提案した．

今後の課題として，際限なく探索を失敗する場合への対応と量子数え上げの失敗による探索終了への対応が挙げられる．

謝辞

本研究を進めるにあたり有益なコメントを頂いた立命館大学情報理工学部の山下茂教授へ深く感謝いたします．

参考文献

[1] L. K. Grover. A fast quantum mechanical algorithm for database search. *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96*, pages 212–219, 1996.

[2] M. Udrescu, L. Prodan, and M. Vlădutiu. Using HDLs for describing quantum circuits: a framework for efficient quantum algorithm simulation. In *Proceedings of the 1st conference on Computing frontiers*, pages 96–110. ACM, 2004.

[3] G. Brassard, P. Høyer, and A. Tapp. Quantum counting. *Automata, Languages and Programming*, 20244(20244):820–831, 1998.

[4] 徳永裕己, 今井浩, 小林弘忠. Grover の量子探索アルゴリズムの応用. 情報処理学会研究報告.AL, アルゴリズム研究会報告, 70(5):33–40, 1999.

[5] M.A.Nelsen and I.L.Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[6] S. Yamashita. How to Utilize Grover Search in General Programming. *Laser physics*, 16(4):1–5, 2006.

[7] B. Omer. Structured quantum programming. *PhD thesis,Techinical University of Vienna*, (May 2003), 2009.