 CMC, vol.57, no.2, pp.307-319, 2018

A Memory-efficient Simulation Method of Grover's Search Algorithm

Xuwei Tang[1] , Juan Xu[1, 2, 3, *] and Bojia Duan[1]

Abstract: Grover's search algorithm is one of the most significant quantum algorithms, which can obtain quadratic speedup of the extensive search problems. Since Grover's search algorithm cannot be implemented on a real quantum computer at present, its quantum simulation is regarded as an effective method to study the search performance. When simulating the Grover's algorithm, the storage space required is exponential, which makes it difficult to simulate the high-qubit Grover's algorithm. To this end, we deeply study the storage problem of probability amplitude, which is the core of the Grover simulation algorithm. We propose a novel memory-efficient method via amplitudes compression, and validate the effectiveness of the method by theoretical analysis and simulation experimentation. The results demonstrate that our compressed simulation search algorithm can help to save nearly 87.5% of the storage space than the uncompressed one. Thus under the same hardware conditions, our method can dramatically reduce the required computing nodes, and at the same time, it can simulate at least 3 qubits more than the uncompressed one. Particularly, our memory-efficient simulation method can also be used to simulate other quantum algorithms to effectively reduce the storage costs required in simulation.

Keywords: Grover's search algorithm, probability amplitude, quantum simulation, memory compression.

1 Introduction

Quantum computation, which can accelerate a wide range of applications than their classical counterparts, has attracted much attention for the last three decades [Feynman (1982); Nielsen and Chuang (2000)]. Quantum computation has been applied to various fields, including quantum key agreement (QKA) [Liu, Chen, Ji et al. (2017); Liu, Xu, Yang et al. (2017)], quantum sealed-bid auction (QSA) [Liu, Wang and Yuan (2016); Liu, Wang, Ji et al. (2014)] and quantum machine learning [Lloyd, Mohseni and Rebentrost (2013)]. Notable among the quantum algorithms, Grover's search algorithm can achieve quadratic acceleration on search applications over unstructured data [Grover (1999)].

[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China.

[2] Institute for Quantum Computing, University of Waterloo, Canada.

[3] Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China.

[*] Corresponding Author: Juan Xu. Email: juanxu@nuaa.edu.cn.

There have been a wide range of generalization and applications of the algorithm, solving problems like pattern classifications and weight decision problem [Singh (2014); Uyanik and Turgut (2013); Yoder, Low and Chuang (2014)]. For the sake of proof of principle, implementation of the algorithm can be achieved via nuclear magnetic resonance, trapped-ion system, cavity-QED and so on [Araujo-Ferreira, Brasil, Soares-Pinto et al. (2012); Ivanov, Ivanov, Linington et al. (2010); Waseem, Ahmed, Irfan et al. (2013)]. But those methods cannot be scalable for large qubits now.

While realistic quantum computers have not been available yet, simulation on classical computers using traditional quantum circuit model can be utilized for studying and developing quantum algorithms [Karafyllidis (2005); Gutierrez, Romero, Trenas et al. (2010)]. On the supercomputer JUGENE with almost 300,000 processors, up to 42 qubits of the quantum system have been achieved [Henkel (2010)]. However, simulation of quantum circuits desires exponential memory resources, it is imperative that efficient methods should be implemented for simulating more qubits with less storage resources.

To reduce the memory resources of the algorithm's simulation, several compression methods on the storage of qubits and unitary operations were proposed. Sparse vector and matrix using a mathematica package were presented for quantum circuit simulation [Gerdt, Kragler and Prokopenya (2009)]. However, the memory resources and the calculation time have been reduced simply dramatically in terms of the sparsity of the qubit vector and unitary matrix. A saving of basis states' number was proposed by storing the qubit states whose amplitudes are non-zeros, which has shortened the qubit state lists during the simulation to some extent [Lu, Yuan and Zhang (2013)]. Whereas the length of the state list which includes both basis states and amplitudes has still been 2^{n+1}. We have proposed a high-performance Grover's algorithm simulation in Tang et al. [Tang, Xu and Zhou (2017)] combining the characteristics of Grover's algorithm and the parallelism of cloud computing, which dramatically improves the performance of the load balancing among multi-core, the utilization of memory space and the efficiency of simulation.

In this paper we propose a high compressed method in the simulation of Grover's algorithm. Draw the lessons from the compressed methods recommended in Plattner et al. [Plattner and Zeier (2012)], we analyzed the duplication of the probability amplitudes along the process of the computation in the algorithm, and calculated the number of no duplicate probability amplitudes (NDPA) at each computational step, with the result showing that the maximal number is 7. Applying this result, the amplitudes vector length in the state list can be sharply reduced to 7 rather than 2^{n+1}. And the efficiency of the Grover simulation algorithm has been improved significantly.

This paper is organized as follows: Notations and a brief overview of Grover's algorithm is given in Section 2. In Section 3, the NDPA theorem in the context of single solution of Grover's algorithm is proposed with the proof process followed. Then we illustrate the high compressed simulation method using NDPA theorem in Section 4 and state the experiments on quantum simulation and analyze the results and performance of simulation in Section 5. Finally, the conclusion is drawn in Section 6.

2 Background

The related notations will be firstly discussed. The superposition of n qubits, i.e. $|\psi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$, is labeled as a probability amplitude vector: $amp = (\alpha_0, \alpha_1, ..., \alpha_{N-1})^T$, $N = 2^n$ where $|i\rangle$ and α_i represent the basis state and the relevant probability amplitude respectively. Furthermore, two functions are involved here. The function *unique(amp)* returns a vector containing no duplicate probability amplitudes (written as NDPA vector) in *amp*. And the function *unique_num(amp)* returns the number of elements in *unique(amp)* (written as NDPA number).

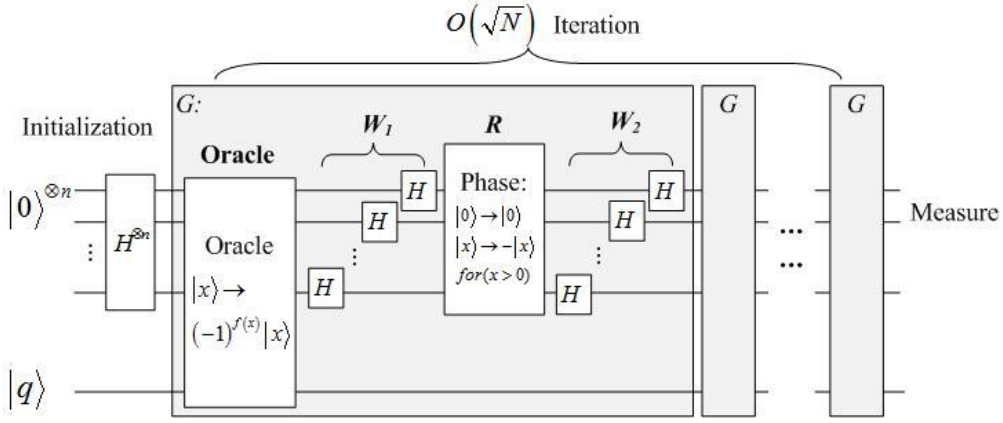

Figure 1: Circuit frame of Grover algorithm

Let us simply remind the readers of the process of Grover search algorithm [Grover (1999)]. This algorithm employs pure states of n qubits which is initialized to the superposition of all computational basis states $|\psi\rangle = 1/\sqrt{2^n} \times \sum_{i=0}^{2^n-1} |i\rangle$. Then the Grover iteration can be divided into four stages which are labeled as Oracle, W_1, R and W_2, and the quantum circuits are illustrated in Fig. 1. Finally, the final measurement of the external system is followed.

Oracle: Applying Oracle to recognizing the solution, and it can be simple expressed as $|x\rangle(-1)^{f(x)}|x\rangle$.

W_1: Applying the Hadamard transform $H^{\otimes n}$.

R: Performing a conditional phase shift $2|0\rangle\langle 0| - I$, and make the probability amplitude of every computational basis state except $|0\rangle$ receive a phase shift of -1.

W_2: Applying the Hadamard transform $H^{\otimes n}$ again.

Repeating the Grover iteration about $R = O(\sqrt{N/M})$ times we can obtained solutions with a high probability when the first n quantum bits are measured. For a more intuitive

and clearer understanding of Grover algorithm in geometry, we assume that X_0 is the set of non-solutions, and X_1 is the set of solutions. Then assume $|\alpha\rangle = \frac{1}{\sqrt{N-M}} \sum_{x \in X_0} |x\rangle$, and $|\beta\rangle = \frac{1}{\sqrt{M}} \sum_{x \in X_1} |x\rangle$.

The initial state can be written as

$$|\varphi\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle = \sqrt{\frac{N-M}{N}} \left(\frac{1}{\sqrt{N-M}} \sum_{x \in X_0} |x\rangle \right) + \sqrt{\frac{M}{N}} \left(\frac{1}{\sqrt{M}} \sum_{x \in X_1} |x\rangle \right) \quad (1)$$
$$= \sqrt{\frac{N-M}{N}} |\alpha\rangle + \sqrt{\frac{M}{N}} |\beta\rangle = \cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|\beta\rangle$$

After one iteration,

$$G|\varphi\rangle = H^{\otimes n}(2|0\rangle\langle 0| - I)H^{\otimes n} O|\varphi\rangle = (2|\varphi\rangle\langle\varphi| - I)(I - 2|\beta\rangle\langle\beta|)|\varphi\rangle$$
$$= \left(2 \begin{bmatrix} \cos^2\frac{\theta}{2} & \cos\frac{\theta}{2}.\sin\frac{\theta}{2} \\ \cos\frac{\theta}{2}.\sin\frac{\theta}{2} & \sin^2\frac{\theta}{2} \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) . \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} . \begin{bmatrix} \cos\frac{\theta}{2} \\ \sin\frac{\theta}{2} \end{bmatrix} \quad (2)$$
$$= \begin{bmatrix} \cos\frac{3\theta}{2} \\ \sin\frac{3\theta}{2} \end{bmatrix} = \cos\frac{3\theta}{2}|\alpha\rangle + \sin\frac{3\theta}{2}|\beta\rangle$$

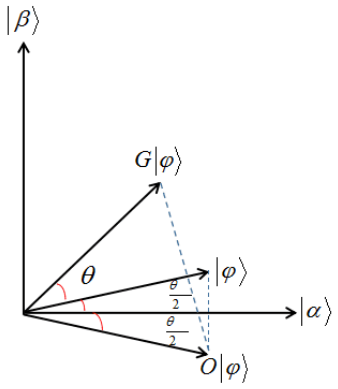

Figure 2: The action of a single Grover iteration *G* (In the figure all states are unit vectors. In order to show the action of *G* clear, $|\alpha\rangle$ and $|\beta\rangle$ are lengthened a little)

It can be seen in Fig. 2 that the vector $|\varphi\rangle$ is rotated by θ towards the superposition $|\beta\rangle$ of all. Thus by performing about *R* Grover iterations, rotate $|\varphi\rangle$ within an angle $\frac{\theta}{2} \le \frac{\pi}{4}$ of $|\beta\rangle$. And then obtaining the solutions is with very high probability through measuring. During the iteration, the Oracle qubit $|q\rangle$ [Trieu (2009)] initially prepared in the state $1/\sqrt{2} \times (|0\rangle - |1\rangle)$ is used for marking the target basis state $|k\rangle$. Assume that $|q\rangle$ is on the highest order of all qubits, making each probability amplitude satisfy the following equation:

$$\alpha_i = -\alpha_{N+i} (i = 0,1,\cdots,N-1). \tag{3}$$

To simplify the process of the proof, we can make an appointment of omitting Oracle qubit $|q\rangle$ in stages W_1 and W_2 of the proof, i.e. we can just consider the condition for the first 2^n basis states, where the latter 2^n basis states are exactly the opposite of the first ones.

3 Probability amplitudes analysis of Grover's algorithm

Quantum simulation has become the main method of current quantum theory research, but insufficient memory is the primary factor limiting the simulation to continue. In order to solve this problem, we present a memory-efficient simulation method. The theorem which is named as NDPA theorem is the soul of the memory-efficient simulation method. We explore the statistics of the probability amplitudes of Grover's algorithm, and propose the NDPA theorem in the context of single solution.

Theorem. *The maximal NDPA number is seven after each step of the computation in Grover search algorithm, i.e.*

$$unique\_num(amp) \le 7. \tag{4}$$

3.1 Proof

The proof includes two major steps. Step I is on the analysis of the amp vector after Oracle operation of any Grover iteration, and Step II is divided into four stages according to the process of one Grover iteration which calculates the NDPA numbers. The coefficients are integer-valued by eliminating the scaling factor in the proof.

Step I: The qubit $|q\rangle$ is omitted, with the length of the *amp* vector being 2^n. According to [Hanson, Ortiz, Sabry et al. (2014)], the *amp* vector after Oracle operation of any r-th Grover iteration including only two different elements can be described as follows:

$$amp_{Oracle_r}^{out} = \left(\underbrace{a_r,\cdots,a_r}_{k}, b_r, \underbrace{a_r,\cdots,a_r}_{2^n-k-1} \right)^T \triangleq \left(\underbrace{a,\cdots,a}_{k}, b, \underbrace{a,\cdots,a}_{2^n-k-1} \right)^T, \tag{5}$$

where b_r is the probability amplitude of the target basis state, a_r is the probability amplitude of the other basis states which are all the same, and the subscript r represents the r-th Grover iteration.

Step II: By splitting any one Grover iteration into four stages, which is illustrated in Fig.1, we will calculate the NDPA number respectively and proof that the Eq. (4) will be satisfied after each step of the computation.

(1) Referring to Grover [Grover (1999)] and Eq. (5), omitting the oracle qubit, there is

$$amp_{Oracle_r}^{in} = \left(\underbrace{a,\cdots,a}_{k}, -b, \underbrace{a,\cdots,a}_{2^n-k-1} \right)^T . \tag{6}$$

Oracle can be carried out by the following three elementary gates: Phase-X gates, control-X gates and Toffoli gates which just involve the exchange of the amplitudes of the related basis states after each step [Lu, Yuan and Zhang (2013)]. Taking on the oracle qubit, given any i, after the i-th step of operation, all elements in the vector amp_i^{out} belong to the following set:

$$\{a, b, -a, -b\}. \tag{7}$$

(2) The second stage involves n continuous Hadamard gates. Consider the process of the form $W_1 = H^{\bigotimes_n} = A_1 A_2 A_3 \cdots A_n$ with A_i denoting the Hadamard gate performed on the i-th qubit. The non-zero elements in matrix A_i are as follows:

$$\begin{aligned} &A_{t\times 2^i+m,t\times 2^i+m} = A_{t\times 2^i+m,t\times 2^i+m+2^{i-1}} \\ &= A_{t\times 2^i+m+2^{i-1},t\times 2^i+m} = A_{t\times 2^i+m+2^{i-1},t\times 2^i+m+2^{i-1}} = -1, \end{aligned} \tag{8}$$

where n denotes the number of qubits, $1 \le i \le n$, $t = 0,1,2,\cdots,2^{n-i}-1$, $m = 0,1,2,\cdots,2^{i-1}-1$.

Mathematical induction can be used to prove that the probability amplitude vector satisfies the following equation after each step of W_1,

$$\begin{aligned} &\alpha_{t\times 2^i} = 2^i \times a, \quad if \quad t \ne t_1, m = 0, \\ &\alpha_{t\times 2^i+m} = 0, \quad if \quad t \ne t_1, m \ne 0, \\ &\alpha_{t_1\times 2^i} = \left(2^i - 1\right)\times a + b, \quad if \quad t = t_1, m = 0, \\ &\alpha_{t_1\times 2^i+m} = (-1)^{\overline{\bar{m}\cdot k \bmod 2^i}} \times (b-a), \quad if \quad t = t_1, m \ne 0, \end{aligned} \tag{9}$$

where $t = 0,1,2,\cdots,2^{n-i}-1, m = 0,1,2,\cdots,2^i-1$.

By Eq. (9), the elements in the vector $amp_{A_i}^{out}$ belong to the following set:

$$\{2^i a, 0, \left(2^i - 1\right) \times a + b, b - a, a - b\}. \tag{10}$$

(3) The output of the second stage is

$$\begin{aligned} amp_{W_1}^{out} &= \Big(2^n \times a + b - a, \left(-1\right)^{\overline{1} \cdot \overline{k}} \times \left(b - a\right), \\ &\quad \left(-1\right)^{\overline{2} \cdot \overline{k}} \times \left(b - a\right), \cdots, \left(-1\right)^{\overline{2^n - 1} \cdot \overline{k}} \times \left(b - a\right)\Big)^T \\ &\triangleq \left(-d, \left(-1\right)^{\overline{1} \cdot \overline{k}} \times c, \left(-1\right)^{\overline{2} \cdot \overline{k}} \times c, \cdots, \left(-1\right)^{\overline{2^n - 1} \cdot \overline{k}} \times c\right)^T, \end{aligned} \tag{11}$$

where $c = b - a, d = -(2^n \times a + b - a)$.

The third stage can be treated as one particular case of Oracle operation, i.e. the element $|0\rangle$ can be regarded as the target element (up to an irrelevant global phase of -1), thus given any i, after the i-th step of operation, all elements in the vector amp_i^{out} belong to the following set:

$$\{c, -c, d, -d\}. \tag{12}$$

(4) Similarly to the second stage, mathematical induction can be used to prove that the probability amplitude vector satisfies the following equation after each step of W_2,

$$\begin{aligned} &\alpha_{k_1} = d + \left(2^i - 1\right) \times c, \quad if \quad t = 0, m = k_1 \\ &\alpha_m = d - c, \quad if \quad t = 0, m \neq k_1 \\ &\alpha_{t \times 2^i + k_1} = \left(-1\right)^{\overline{2^i t} \cdot \overline{k}} \times 2^i \times c, \quad if \quad t \neq 0, m = k_1 \\ &\alpha_{t \times 2^i + m} = 0, \quad if \quad t \neq 0, m \neq k_1 \\ &\left(t = 0, 1, \cdots, 2^{n-i} - 1, \quad m = 0, 1, \cdots, 2^i - 1\right), \end{aligned} \tag{13}$$

where n denotes the number of qubits, and $1 \le i \le n$.

By Eq. (13), the elements in the vector $amp_{A_i}^{out}$ belong to the following set:

$$\left\{d + \left(2^i - 1\right)c, d - c, 2^i c, -2^i c, 0\right\}. \tag{14}$$

After the calculation of the NDPA vectors, we should also analyze the situation in the present of the Oracle qubit in stage II and IV. As is illustrated in Tab. 1, the first column expresses the operation stage of Grover iteration. The second column lists the amplitudes sets with no duplicate elements of the first 2^n basis states, while the corresponding equations are showed in the sixth. Then Eq. (3) and the second column give the amplitudes sets with no duplicate elements of the last 2^n basis states, which are shown in the third column. By the union operation of the front two columns, we get the sets in the fourth column and the relevant numbers in the fifth column. Obviously, the Eq. (4) is

satisfied according to the numbers in the fifth column. Now the proof of the NDPA theorem is completed.

Table 1: Unique numbers of amplitudes analysis in Grover iteration

Opera-tion	NDPA set (first 2^n states)	NDPA set (last 2^n states)	NDPA set (all 2^{n+1} states)	NN [a]	Eq [b]
Oracle	-	-	$\{\pm a, \pm b\}$	4	(7)
$W_1(A_i)$ $(1 \le i \le n)$	$\{2^i \times a,$ $(2^i - 1) \times a + b,$ $\pm(a-b), 0\}$	$\{-2^i \times a,$ $-(2^i - 1) \times a - b,$ $\pm(a-b), 0\}$	$\{\pm 2^i \times a,$ $\pm((2^i - 1) \times a + b),$ $\pm(a-b), 0\}$	7	(10)
R	-	-	$\{\pm c, \pm d\}$	4	(12)
$W_2(A_i)$ $(1 \le i \le n)$	$\{d + (2^i - 1) \times c,$ $d - c, \pm 2^i \times c, 0\}$	$\{-(d + (2^i - 1) \times c),$ $-(d-c), \pm 2^i \times c, 0\}$	$\{\pm(d + (2^i - 1) \times c),$ $\pm(d-c), \pm 2^i \times c, 0\}$	7	(14)

[a] NN: NDPA Number

[b] Eq: Equation

3.2 Verification

We verified the theorem on MATLAB. In the program, a float array *amp[]* is used to store the probability amplitudes in one Grover iteration; the function *unique()* returns an array representing NDPA vector and the integer array *uni_num[]* denotes the NDPA numbers along the whole computational process in which each data stores the length of *unique(amp)*. The major corresponding pseudo code is as follows.

```
UNIQUE_NUMBER(AMP)
1. amp() = initialize();
2. for j ← 1 1 to (pi / 4) * sqrt(N)
3.    do amp(:,1) ← Oracle();
4.     for i ← 2 to steps
5.        do amp(:,i) ← operation * amp(:,i−1);
6.     for i ← 1 to steps
7.        do uni_num(j,i) ← length(unique(amp(:,i)));
8.     Oracle();
```

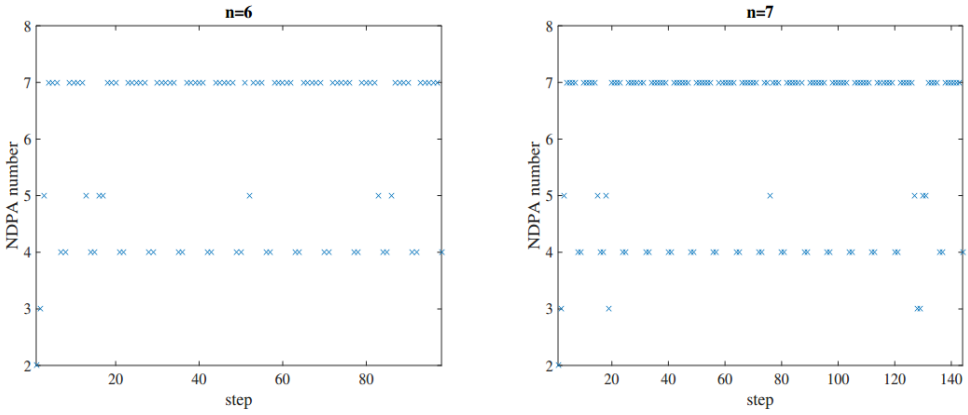

Figure 3: NDPA numbers along the whole computational process in Grover's algorithm

Fig. 3 shows the behavior for $n=6$ and $n=7$ where we can easily observe that the maximal NDPA number is seven, i.e. the theorem is valid.

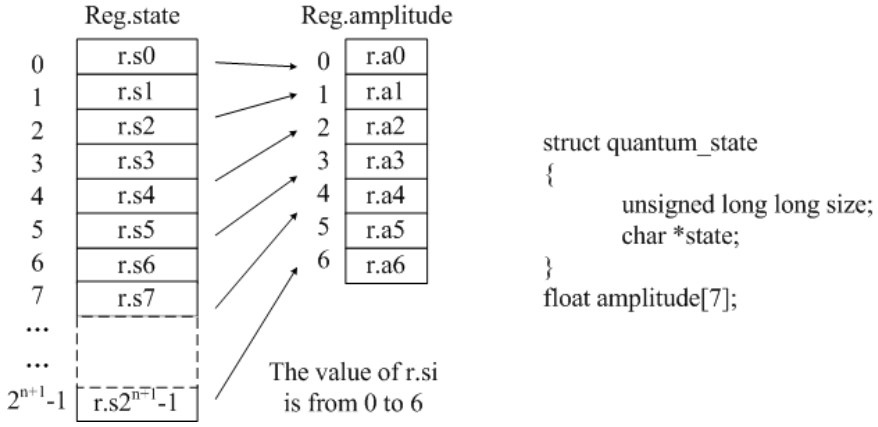

Figure 4: Structure of the node type including state and amplitude

4 The memory-efficient simulation method

In this section we apply the theorem to quantum simulation of Grover algorithm for the purpose of compressing the memory size by developing a C program. As in the original method, the state list and amplitude list are both stored exponential whose maximal length is 2^{n+1}. Applying the NDPA theorem, the size of amplitude list can shrink to a constant. The relevant data structures in our program are shown in Fig. 4, where we use a seven-length list to store the unique amplitude at a single step along the algorithm. The key procedures of our program are depicted in Fig. 5(a), which serve to detail the compressed simulation method. The program begins with initialization of the amplitude list and the state list. Along the whole process of simulation algorithm, the quantum gates are performed one after another until the final iteration is over. Only the Hadamard gate involves amplitude updating procedure. Then the state list storing the index of the

amplitude refreshes and a round of the quantum register updating after quantum gate operation is finished.

Furthermore, we detail the amplitude updating procedure, depicted in Fig. 5(b). An amp_temp list is initialized for storing the amplitudes after gate operation. If any computed amplitude data do not exist in the amp_temp list, the new data will be added to the amp_temp list. Then the program updates the state list with the new index of the amplitude. All basis states' index numbers will be refreshed before the procedure is over with the help of an auxiliary list marking the basis state calculated.

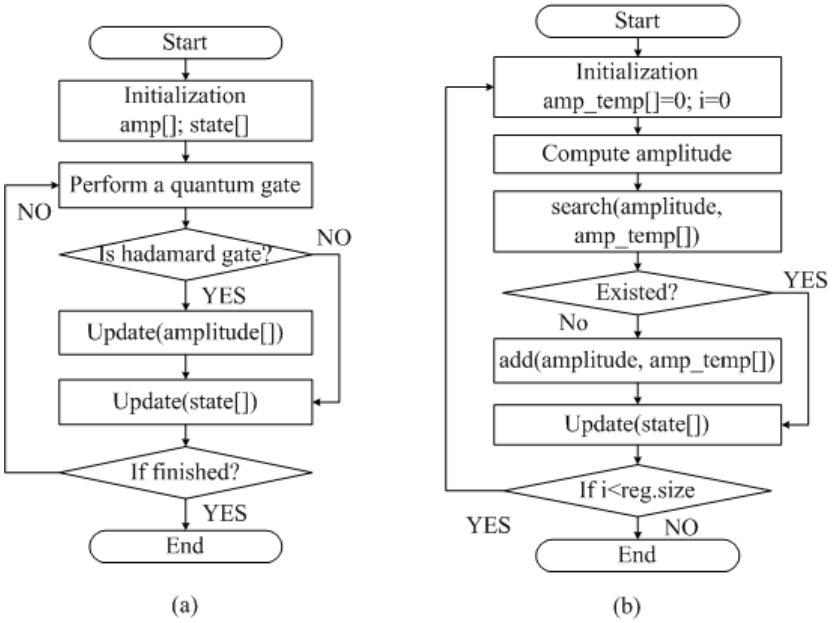

Figure 5: (a) Flowchart overview of the main procedure. (b) Flowchart overview of the amplitude updating procedure

Now the compression ratio can be effortlessly figured out by theoretical analysis. For the uncompressed method, the state list stores 8 *bytes* which type is unsigned long long int for 2^{n+1} times, the amplitude list stores as float_complex type needing 8 *bytes* for 2^{n+1} times. Thus, the complete storage space for the uncompressed list is now:

$$(8bytes+8bytes)\times 2^{n+1}=2^{n+5}bytes. \tag{15}$$

In comparison, for the compressed method, the state list (*Reg:state*) stores the location substituted for the numerical value of the corresponding amplitude for 2^{n+1} times, i.e. the value range of the elements in *Reg:state* list is from 0 to 6, so 1 byte is sufficient for storing one location of the amplitude; and the amplitude list stores as float_complex type for only 7 times. Here we also need another byte as auxiliary to label the calculated state for 2^{n+1} times and a temporary amplitude list for calculation. Thus the complete storage space for the compressed list is now:

$$(1byte \times 2^{n+1} + 8bytes \times 7) \times 2 = (2^{n+2} + 112)bytes. \quad (16)$$

Compared to 2^{n+5} bytes for the uncompressed one, a saving of nearly 87.5% can be achieved when n is infinity.

Table 2: Memory of LQ and CG programs with different qubits

Program	Qubits								
	23	24	25	26	27	28	29	30	31
LQ	256[a]	512	1024	2048	4096	8192	-[b]	-	-
CG	32	64	128	256	512	1024	2048	4096	8192

[a] Unit:(MB)

[b] '-' indicates that the hardware platform could not achieve the result.

5 Experiments and analysis

The experiments reported in this paper were performed on an Intel Xeon E5506 CPU with 4 cores at 2.13 GHz clock rate, 47.2 GiB of memory. The hardware platform may restrict the performance of the experiments, however, the trends in time and memory size can be seen from the results reported below. We have also run the libquantum1.1.1 program for comparison [Weimer (2013)]. The two programs we implemented are itemized as follows:

- Libquantum (LQ): The serial code of the Grover algorithm executed is from the libquantum library, according to the release of libquantum1.1.1 [Weimer (2013)].
- Compressed_Grover (CG): The algorithm is simulated according to the method described in Section 4.

We compare the time consumption of LQ and CG and analyze the complexity of simulation process. We set the network delay as δ and the communication consumption between processes is $k\omega$ (where k is the number of nodes). As the number of qubits is large enough the extra-cost $\Delta_{extra} = k\omega + \delta$ can be approximated by a constant, so there is a little difference between the time efficiency of LQ and CG.

For verifying the result on the basis of the experiment data, Tab. 2 compares the memory cost of the two programs. Interestingly, although CG program compress the length of amplitude list from 2^{n+1} to 7, the trends of the total memory cost of the two programs both grow exponentially with the number of qubits n, which is for the reason that the storage of the basis states list increases exponentially with the qubits. Moreover, it is clear that 3 more qubits can be simulated in CG program than in LQ program base on the same storage memory. Taking 256 MB memory for example, 26 qubits can be simulated in CG program, while in LQ program, only 23 qubits can be implemented. Furthermore, 8 or more computing nodes will be needed to simulate 31 qubits in LQ program, but only one computing node is needed in CG program. The result shows that the compression method is effective, which helps saving nearly 87.5% memory when n is infinity.

6 Conclusion

In summary, we have proposed an efficient simulation method of Grover's search algorithm in the context of a single solution. In particular, we proposed a theorem showing that the storage of the probability amplitudes can be compressed to a constant which is independent of the number n of qubits for large n. We have also applied the theorem to the simulation of Grover's algorithm based on the classical computers, and the results show when n is infinite 87.5% memory can be saved and three more qubits can be simulated in the same storage space. In the future work, this work can be used in the simulation on General Purpose GPU and more qubits can be simulated. In addition, the memory-efficient simulation method of Grover's search algorithm we proposed will be used to optimize the training process of support vector machine and relevance vector machine.

Acknowledgement: This work was supported by Funding of National Natural Science Foundation of China (Grant No. 61571226, Grant No. 61701229).

References

Araujo-Ferreira, A.; Brasil, C. A.; Soares-Pinto, D.; Deazevedo, E.; Bonagamba, T. J. et al. (2012): Quantum state tomography and quantum logical operations in a three qubits nmr quadrupolar system. *International Journal of Quantum Information*, vol. 10, no. 2, pp. 4899-4924.

Feynman, R. P. (1982): Simulating physics with computers. *International Journal of Theoretical Physics*, vol. 21, pp. 467-488.

Gerdt, V. P.; Kragler, R.; Prokopenya, A. N. (2009): A mathematica package for simulation of quantum computation. *Computer Algebra in Scientific Computing*, vol. 5743, pp. 106-117.

Grover, L. K. (1999): Quantum mechanics helps in searching for a needle in a haystack. *Quantum Entanglement and Quantum Information-Proceedings of Ccast*, vol. 79, pp. 325-328.

Gutierrez, E.; Romero, S.; Trenas, M. A.; Zapata, E. L. (2010): Quantum computer simulation using the cuda programming model. *Computer Physics Communications*, vol. 181, no. 2, pp. 283-300.

Hanson, A. J.; Ortiz, G.; Sabry, A.; Tai, Y. T. (2014): Discrete quantum theories. *Journal of Physics A-Mathematical and Theoretical*, vol. 47, no. 11, pp. 115305.

Henkel, M. (2010): Quantum computer simulation: New world record on jugene, 2010.

Ivanov, S.; Ivanov, P.; Linington, I.; Vitanov, N. (2010): Scalable quantum search using trapped ions. *Physical Review A*, vol. 81, no. 4, pp. 82-87.

Uyanik, K.; Turgut, S. (2013): A new sure-success generalization of grover iteration and its application to weight decision problem of boolean functions. *Quantum Information Processing*, vol. 12, no. 11, pp. 3395-3409.

Karafyllidis, I. G. (2005): Quantum computer simulator based on the circuit model of quantum computation. *IEEE Transactions on Circuits and Systems I-Regular Papers*, vol. 52, no. 8, pp. 1590-1596.

Liu, W. J.; Chen, Z. Y.; Ji, S.; Wang, H. B.; Zhang, J. (2017): Multi-party semi-quantum key agreement with delegating quantum computation. *International Journal of Theoretical Physics*, vol. 56, no. 10, pp. 3164-3174.

Liu, W. J.; Wang, F.; Ji, S.; Qu, Z. G.; Wang, X. J. (2014): Attacks and improvement of quantum sealed-bid auction with epr pairs. *Communications in Theoretical Physics*, vol. 61, no. 6, pp. 686-690.

Liu, W. J.; Wang, H. B.; Yuan, G. L. (2016): Multiparty quantum sealed-bid auction using single photons as message carrier. *Quantum Information Processing*, vol. 15, no. 2, pp. 869-879.

Liu, W. J.; Xu, Y.; Yang, C. N.; Gao, P. P.; Yu, W. B. (2017): An efficient and secure arbitrary n-party quantum key agreement protocol using bell states. *International Journal of Theoretical Physics*, no. 6, pp. 1-13.

Lloyd, S.; Mohseni, M.; Rebentrost, P. (2013): Quantum algorithms for supervised and unsupervised machine learning. *Eprint Arxiv*.

Lu, X.; Yuan, J.; Zhang, W. (2013): Workflow of the grover algorithm simulation incorporating cuda and gpgpu. *Computer Physics Communications*, vol. 184, no. 9, pp. 2035-2041.

Singh, M. P.; Rajput, B. S. (2014): New maximally entangled states and pattern classification in two-qubit system. *International Journal of Theoretical Physics*, vol. 53, no. 9, pp. 3226-3238.

Nielsen, M. A.; Chuang, I. (2000): *Quantum Computation and Quantum Information*. Cambridge University Press, Britain.

Plattner, H.; Zeier, A. (2012): *In-memory Data Management: Technology and Applications*. Springer Berlin.

Tang, X.; Xu, J.; Zhou, Y. (2017): Optimization of grover's algorithm simulation based on cloud computing. *International Conference on Intelligent Data Engineering and Automated Learning*, pp. 83-93.

Trieu, D. (2009): Large-scale simulations of error prone quantum computation devices. http://juser.fz-juelich.de/record/7578/files/IAS_Series_02.pdf.

Waseem, M.; Ahmed, R.; Irfan, M.; Qamar, S. (2013): Three-qubit grover's algorithm using superconducting quantum interference devices in cavity-qed. *Quantum Information Processing*, vol. 12, no. 12, pp. 3649-3664.

Weimer, H. (2013): Libquantum-the c library for quantum computing and quantum simulation. http://www.libquantum.de/.

Yoder, T. J.; Low, G. H.; Chuang, I. L. (2014): Fixed-point quantum search with an optimal number of queries. *Physical Review Letters*, vol. 113, no. 21.