Quantum Rough Counting and Its Application to Grover's Search Algorithm

Naphan Benchasattabuse, Prabhas Chongstitvatana
Department of Computer Engineering
Faculty of Engineering, Chulalongkorn University
Bangkok, Thailand
e-mail: naphanben@gmail.com, prabhas.c@chula.ac.th

Chatchawit Aporntewan
Department of Mathematics and Computer Science
Faculty of Science, Chulalongkorn University
Bangkok, Thailand
e-mail: chatchawit.a@chula.ac.th

***Abstract*—We propose in this paper a rough counting algorithm and modified version of Grover's search algorithm. The proposed counting algorithm uses Deutsch-Jozsa's algorithm to roughly estimate the number of items satisfying some search conditions by reconstructing the output distribution. The modified Grover's search combines the counting into one part of the algorithm and uses the counting result to determine the number of Grover's iterations taken to get such items. We provide an analysis of expected probability of the proposed algorithm taking into account the problem size, the number of satisfying items and the query complexity to show that the proposed modified Grover's search is optimal.**

Keywords-component; Grover's algorithm; Deutsch-Jozsa's algorithm; quantum algorithm; quantum counting; quantum searching

I. INTRODUCTION

In the field of computer science, almost every problem can be solved by searching for solutions. Searching for solutions is to define the domains of possible solutions, the satisfying conditions and go through all or some of the possibilities and find those which satisfy the conditions. Using the problem's structure or some heuristics may improve searching performance significantly by not going through all possibilities but select only a small portion to test under those conditions. Despite knowing the structures of the problem, it does not however derive at solutions any quicker than a brute force approach or does not provide the optimal solution which is ideally required. Although the exact searching or optimal searching algorithm can deliver the best and optimal answer, but its speed might not be quick enough, and speed is almost always the most important factor in any algorithm. Many problems have an algorithm that can efficiently solve them while those that do not still have to rely on searching, such as those that lie in NP-Completeness.

A quantum computer is shown to be more powerful than a classical one [1]. This is because some specialized problems which can be efficiently solved by a quantum computer cannot be solved efficiently by a classical computer. Examples of those algorithms are Shor's integer factorization and discrete logarithm [2] which could break the RSA encryption, Deutsch-Jozsa's determining whether a binary function is constant or balanced [3], and Simon's period finding [4].

Quantum algorithm for searching an unstructured database or determining input to black-box function has been proposed by L.K. Grover in 1996 [5, 6]. With this method, an item can be found in the database that satisfies the search query or the desired output from the black-box function using $O(\sqrt{N})$ function evaluations over the classical one that requires $O(N)$ function evaluations when N is equal to the number of database entries and all possible input of the function respectively. Grover's algorithm is a specific version of a general amplitude amplification technique which is discussed in detail in [7] and the concept of quantum query complexity of a black-box function is also discussed in [8] by Nayak and Wu. A collection of black-box or oracular search in quantum computing which exploits the super position of states in quantum computation is summarized in [9]. One may wonder why Grover's algorithm is significant since there are usually no black-box functions which can be accessed in reality, and most databases are structured and can usually retrieve items that satisfy the search queries with better query complexity than $O(\sqrt{N})$. Another query example is entertained here. We want to find an input to a hash function which outputs a word "output". Even with full knowledge of how the function does the hashing and what valid inputs are, inverting the process is problematic or might be impossible to do so. This is where Grover's algorithm comes in and can be used to solve such a problem.

Grover's algorithm has one drawback however. In order to use the first version of the algorithm, the number of items that satisfy the search query must be known beforehand. Many computer scientists have modified the Grover's algorithm so that it can be used when multiple items could satisfy a query even in cases when the quantity is not revealed but the counting can be derived. Boyer, Brassard, Høyer, and Tapp [10] have proposed their algorithm with an in-depth analysis over the Grover's original algorithm and their version. Brassard, Høyer, and Tapp also discovered quantum counting algorithm [11] which is to count the number of elements fulfilling the search query rather than finding one with small probability of error within $\Theta(\sqrt{Nt})$ function evaluations when t is the count acquired.

In this paper, we will propose another counting algorithm using the Deutsch-Jozsa's algorithm to roughly estimate the number of solutions which satisfies the search query then calculate the number of Grover's iterations to be applied and measure the result. Although the proposed counting can only achieve rough t within error of $\sqrt{N}$, when using this approximation with Grover's algorithm, the expected query

978-1-5386-6350-9/18/$31.00 ©2018 IEEE

complexity is still $O(\sqrt{N})$ which is the same as [10] and additional rough knowledge about the size of t can be obtained as well.

In this paper, we presume that our readers are familiar with basic notions of quantum computing [12].

II. The Algorithm

A. Abstract problem

First, formalizing the problem to be an abstract problem must be done such that it can be followed with ease.

Let $f(x)$ be a binary function which outputs only 0 or 1 and can be evaluated in constant time. Let $N = 2^n$ be the possible inputs to $f(x)$. Let $A = \{x_1, x_2, x_3, \ldots, x_t\}$ be a subset of possible input such that $f(x_i) = 1$, for all $x_i \in A$. Let t be the number of solutions, the number of inputs which correspond to specific search conditions. The problem is to find any single x_i that is in A.

B. Algorithm Overview

As briefly discussed in the Introduction, the proposed search algorithm can be split into two parts: -

1. Roughly estimate t by the counting algorithm and call it t_0.
2. Use t_0 to determines how many Grover's iteration needed to apply to achieve the highest probability of getting the desired solution.

In the proposed counting algorithm, some aspects of the Deutsch-Jozsa's algorithm are used to estimate t to get t_0 and the calculation from [10] is used to calculate how many Grover's iterations should be performed to obtain an answer.

C. Deutsch-Jozsa's Algorithm Review

Deutsch-Jozsa's algorithm is an algorithm that showcases the power of quantum computer over the classical one. The algorithm solves the problem by determining whether a given binary function $f(x)$ is constant or balanced over an even number of possible input space. The $f(x)$ is considered constant if $f(x) = 0$ or $f(x) = 1$ for all input x and balanced when half the input makes $f(x) = 0$ and the other half makes $f(x) = 1$.

The improved version of Deutsch-Jozsa's algorithm [13] is as followed:

1. Prepare n qubits register, and a single qubit register and initialize qubits to $|\phi_1\rangle = |0\rangle^{\otimes n}|1\rangle$.
2. Apply Hadamard transform on $|\phi_1\rangle$ to get $|\phi_2\rangle$.
3. Apply U_f to $|\phi_2\rangle$ and get $|\phi_3\rangle$.
4. Apply Hadamard transform to the first n qubits of $|\phi_3\rangle$ to get $|\phi_4\rangle$.
5. Measure the query register from $|\phi_4\rangle$ (the first n qubits).
6. If the result of measurement is $|0\rangle^{\otimes n}$ then $f(x)$ is constant otherwise balanced.

Where U_f is the quantum accessible version of $f(x)$ which is reversible and $U_f|\phi\rangle|0\rangle = |\phi\rangle|0 \oplus f(x)\rangle$.

D. Rough Counting Algorithm

In the fourth step of Deutsch-Jozsa's algorithm, the amplitude of $|0\rangle^{\otimes n}$ in the first register (the first n qubits) is in the form of

$$\sum_i \frac{(-1)^{f(x)}}{2^n} \tag{1}$$

Using this information, by sampling this state enough times. The ratio of t/N can be estimated by (1) and the probability of getting $|0\rangle^{\otimes n}$ is

$$P\left(|0\rangle^{\otimes n}\right) = \left(\frac{N-2t}{N}\right)^2 \tag{2}$$

E. Probability of Getting the Solution from Grover's Algorithm

Using equations from [10], let $\sin^2\theta = t/N$ while j is number of iterations applying Grover's circuit and P_j be the probability of obtaining a solution after applying Grover's iteration j times.

$$P_j = \sin^2((2j+1)\theta) \tag{3}$$

In cases where t is known. The number of iterations that should be taken to get highest probability of arriving at a solution can be calculated using (3). Let m be the optimal number of iterations taken, m can be described by

$$m = \lfloor (\pi - 2\theta)/4\theta \rfloor. \tag{4}$$

F. Modified Grover's Algorithm

Now to describe the proposed algorithm. First, assume that $1 \le t \le N/2$.

1. Initialize qubits to $|\phi\rangle = \sum_{i=0}^{N} \frac{1}{\sqrt{2^n}}|i\rangle|1\rangle$.
2. Apply U_f (constructed like Deutsch-Jozsa's algorithm).
3. Apply Hadamard transform to first n qubits.
4. Measure the first register.
5. Repeat step 1-4 for $\sqrt{N}$ times and let z be number of measurements resulting $|0\rangle^{\otimes n}$.
6. Calculate the number of iterations using (2) and (3) by setting $P(|0\rangle^{\otimes n}) = z/\sqrt{(N)}$ and let the result be m_0
7. Apply Grover's iteration for m_0 times.
8. Measure the register and compare the result with desired output.

III. RESULTS

In order to analyze a query complexity of this algorithm, let us look at each part separately. The second part is the normal Grover's algorithm which requires $O(\sqrt{N})$ function evaluations or more precisely $O(\sqrt{N/t_0})$. The first part, the counting, can be considered the same as running Deutsch-Jozsa's algorithm for $\sqrt{N}$ times, thus requiring $O(\sqrt{N})$ function evaluations. We can conclude that this proposed algorithm query complexity combining the two parts is in $O(\sqrt{N})$ which is optimal.

Let us analyze the probability of obtaining the correct solution from the proposed algorithm. From the first part, estimating the ratio of t/N, a total of $\sqrt{N}$ times is done sampling the register. Using (2), it can be inferred that

$$t = \frac{N - N\sqrt{P\left(|0\rangle^{\otimes n}\right)}}{2}. \tag{5}$$

Subsequently, out of $\sqrt{N}$ sampling, if $|0\rangle^{\otimes n}$ is sampled z times, estimation of t can be made and called it t_0 using $P_0 = P(|0\rangle^{\otimes n}) = z/\sqrt{N}$ and (5). Since it is known that $1 \le t \le N/2$, using (2) and (5) will work without having to consider the sign inside the square root. Now, let us look at the expected probability of this proposed algorithm.

Let $P_{success}$ be the probability of obtaining a correct solution,

$$\frac{i}{\lfloor\sqrt{N}\rfloor} = \left(\frac{N - 2t_i}{N}\right)^2 \tag{6}$$

$$\sin^2\theta_i = t_i / N \tag{7}$$

$$m_i = \left\lfloor (\pi - 2\theta_i)/4\theta_i \right\rfloor \tag{8}$$

$$P_{success} = \sum_{i=0}^{\lfloor\sqrt{N}\rfloor} \binom{\lfloor\sqrt{N}\rfloor}{i} P_0^i (1-P_0)^{\lfloor\sqrt{N}\rfloor - i} \sin^2((2m_i+1)\theta) \tag{9}$$

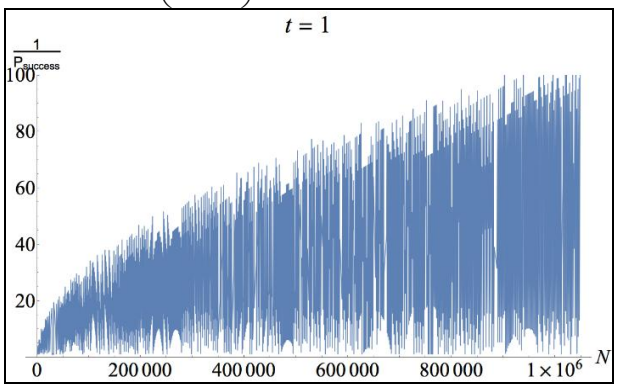

Figure 1. Expected total runs $(1/P_{success})$ while N grows with $t = 1$.

Fig. 1, 2, and 3 below depict the relations of $1/P_{success}$ and N when t = 1, 50, and 100 while N grows respectively. The y-axis $1/P_{success}$ is the expected number of runs needed to get a correct solution.

IV. DISCUSSIONS

Normally when N is not a power of two, the binary $f(x)$ can be modified by adding input space to be equal to a power of two and have output of those added input corresponds to zero. For simplicity sake, we plotted Fig. 1, 2, and 3 with continuous N which reveal the trend of expected number of runs required for obtaining a correct solution with the proposed search algorithm with upper bound linear to N. Further to the case of $1 \le t \le N/2$, we can modify the algorithm to works in the case of $t > N/2$ by doubling the input space and have the added input evaluated to zero or just switch to use majority finding algorithm.

The proposed algorithm achieves optimality in query complexity of $O(\sqrt{N})$, the same as [10] but also provides an estimation of number of solutions in addition to providing the solution. On the other hand, counting algorithm in [11] which outputs approximate relative count within an error of $\sqrt{N/t}$ that requires expected number of $\Theta(\sqrt{tN})$ function evaluations with success probability of at least 3/4 comparing to the proposed rough counting algorithm which output estimate t_0 of within an error of $\sqrt{N}$ in average cases.

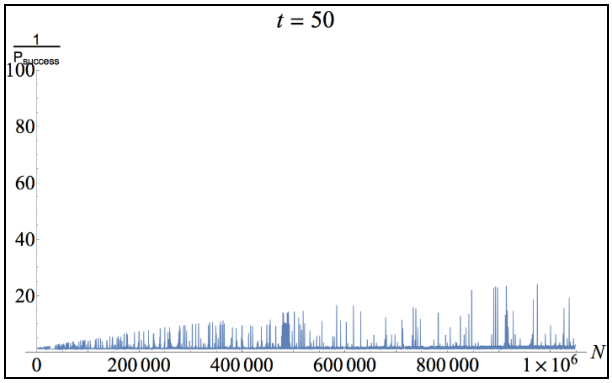

Figure 2. Expected total runs $(1/P_{success})$ while N grows with $t = 50$.

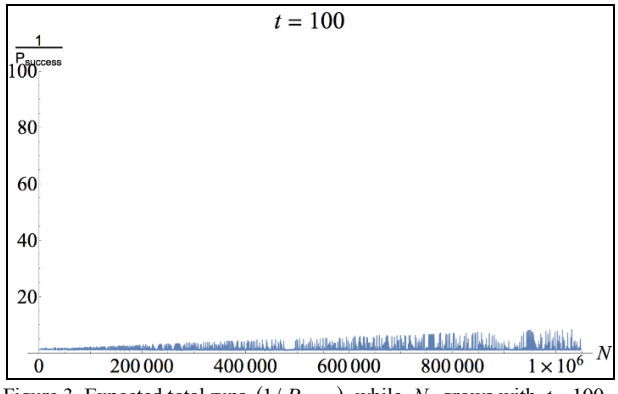

Figure 3. Expected total runs $(1/P_{success})$ while N grows with $t = 100$.

The disparity will be quite insignificant when N is large and t is small which is normally the case with most applications of Grover's algorithm.

The second part of the algorithm can also be replaced with search algorithm in [10] and use t_0 as a starting point and the bound of randomly choosing value of iteration to speed up the search process or adopt the randomized strategy in case t_0 deviates too much from the real t.

In conclusion, the proposed algorithm can perform roughly the same as [10] and [11] of which are optimal in both tasks.

ACKNOWLEDGMENT

This work was supported by Chula Computer Engineering Graduate Scholarship for CP Alumni.

REFERENCES

[1] D. Deutsch, "Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer", Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 400, no. 1818, pp. 97-117, 1985.

[2] P. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer", SIAM Review, vol. 41, no. 2, pp. 303-332, 1999.

[3] D. Deutsch and R. Jozsa, "Rapid Solution of Problems by Quantum Computation", Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 439, no. 1907, pp. 553-558, 1992.

[4] D. Simon, "On the Power of Quantum Computation", SIAM Journal on Computing, vol. 26, no. 5, pp. 1474-1483, 1997.

[5] L.K. Grover, "Quantum mechanics helps in searching for a needle in a haystack", Physical Review Letters, Vol. 79, no. 2, pp. 325–328, 1997.

[6] L.K. Grover, "A framework for fast quantum mechanical algorithms", In Proceedings of the 30th ACM Symposium on Theory of Computing: STOC'98, pp. 53–62, 1998.

[7] G. Brassard, P. Høyer, M. Mosca and A. Tapp, "Quantum amplitude amplification and estimation", Contemporary Mathematics, Vol. 305, pp. 53–74, 2002.

[8] A. Nayak and F. Wu, "The quantum query complexity of approximating the median and related statistics", In Proceedings of the 31st ACM Symposium on Theory of Computing: STOC'99, pp. 384–393, 1999.

[9] P. R. Giri and V. E. Korepin, "A Review on Quantum Search Algorithms," Quantum Information Processing, vol. 16, no. 12, Dec. 2017.

[10] M. Boyer, G. Brassard, P. Høyer and A. Tapp, "Tight Bounds on Quantum Searching", Fortschritte der Physik, vol. 46, no. 4-5, pp. 493-505, 1998.

[11] G. Brassard, P. Høyer and A. Tapp, "Quantum counting", In Proceedings of the International Conference on Automata, Languages and Programming: ICALP'98, pp. 820–831, 1998.

[12] M. A. Nielsen and I. L. Chuang, Quantum computation and quantum information, 10th anniversary ed. Cambridge ; New York: Cambridge University Press, 2010.

[13] R. Cleve, A. Ekert, C. Macchiavello and M. Mosca, "Quantum algorithms revisited", Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 454, no. 1969, pp. 339-354, 1998.