Decoherence in optimized quantum random-walk search algorithm*

Zhang Yu-Chao(张宇超)[a)b)], Bao Wan-Su(鲍皖苏)[a)b)†], Wang Xiang(汪 翔)[a)b)], and Fu Xiang-Qun(付向群)[a)b)]

[a)] Zhengzhou Information Science and Technology Institute, Zhengzhou 450004, China
[b)] Synergetic Innovation Center of Quantum Information and Quantum Physics, University of Science and Technology of China, Hefei 230026, China

(Received 6 January 2015; revised manuscript received 11 March 2015; published online 20 June 2015)

This paper investigates the effects of decoherence generated by broken-link-type noise in the hypercube on an optimized quantum random-walk search algorithm. When the hypercube occurs with random broken links, the optimized quantum random-walk search algorithm with decoherence is depicted through defining the shift operator which includes the possibility of broken links. For a given database size, we obtain the maximum success rate of the algorithm and the required number of iterations through numerical simulations and analysis when the algorithm is in the presence of decoherence. Then the computational complexity of the algorithm with decoherence is obtained. The results show that the ultimate effect of broken-link-type decoherence on the optimized quantum random-walk search algorithm is negative.

Keywords: quantum search algorithm, quantum random walk, decoherence

PACS: 03.67.Lx, 03.67.Ac **DOI:** 10.1088/1674-1056/24/8/080307

1. Introduction

Quantum computation shows a huge advantage in dealing with many classical problems because of its powerful parallel computing ability. So far, some remarkable quantum algorithms have been proposed, including Grover's algorithm[1] and Shor's algorithm.[2] Compared with the classical algorithm, Grover's algorithm provides a quadratic increase in speed and Shor's algorithm provides an exponential increase in speed. The random-walk is a strategy often used in the design of classical algorithms. As a random-walk on a general graph can be used to address hard problems such as k-SAT and graph connectivity.[3,4] So it is a natural question to ask whether the algorithm based on quantum random walks that can provide an increase in speed. With the in-depth study of quantum random walk, mixing time and hitting time on some graphs which are superior to the classic counterparts are constantly explored.[5–7] It shows a potential in the design of the quantum algorithm. Therefore, the research of quantum random walks[8–10] and the algorithms based on quantum random walks have become a hot topic.

In 2003, a quantum random-walk search algorithm (called as SKW algorithm)[11] was proposed by Shenvi. It iterates $(\pi/4)\sqrt{2^{n+1}}$ times with a success rate of 0.5 approximately, reaching quadratic increase in speed on an unstructured 2^n-sized database. In 2008, Potocek put forward the optimized quantum random-walk search algorithm (optimized SKW algorithm).[12] It iterates $(\pi/4)\sqrt{2^n}$ times with a success rate of 1 approximately. These two algorithms are not superior to Grover's algorithm in computational complexity, but they have certain advantages in physical implementation and resistance to errors.[13,14] Since then, many algorithms based on quantum random walks have been proposed.[15–17] On the other hand, when solving the spatial search problem, algorithms based on quantum random-walks are proved to be better than Grover's algorithm.[18,19]

In practice, errors in the implementation of quantum algorithms are inevitable. The error in each step may be very small; but the scale of quantum computation is usually exponential, small errors may accumulate into large errors and affect the algorithm results. Different errors have different effects on the algorithm. Studying the effects on the quantum algorithm can improve the reliability of it and be beneficial for the implementation of the algorithm. There are two main types of errors. One results from the gate imperfection, and errors in most of the existing research are generated by it. In 2000, Long found systematic errors in phase inversions and random errors in Hadmard–Walsh transformations are the dominant effects in Grover's algorithm.[20] In 2003, Shenvi studied the robustness of Grover's algorithm to a random phase error in the oracle and the error tolerability of the algorithm.[21] In 2006, Li studied the effects of the error in phase inversions on the SKW algorithm and found that in addition to reducing the probability of the marked state eventually, the error also reduces the maximum probability of the unmarked state, which is different from Grover's algorithm.[22] The other is generated by decoherence. Decoherence is due to the quantum system disturbed by the influence of random events, such as random measurements. It will result in the entanglement of quantum states and superposition properties being destroyed, weakening the coherence. As the particularity and randomness of this

*Project supported by the National Basic Research Program of China (Grant No. 2013CB338002).
†Corresponding author. E-mail: 2010thzz@sina.com
© 2015 Chinese Physical Society and IOP Publishing Ltd

kind of error are generated, it is difficult to make a relevant theoretical analysis. Investigations on this kind of error in the algorithm are rare. In 2009, Abal *et al.* investigated different errors in the SKW algorithm including the error generated by decoherence in the shift operator.[23]

In addition, the other significance of studying decoherence in the quantum algorithm based on quantum random walks is that it may have positive effects on the algorithm.[24] Because quantum random walk is highly sensitive to the effects of decoherence, the right amount of decoherence can result in a particular probability distribution and quantum speed up. This is also very encouraging for the prospects of using quantum random walks as the basis of powerful quantum algorithms. Currently, the effects of decoherence on quantum random walks on various graphs have been studied fully.[25–27] In 2003, Kendon and Tregenna found a small amount of decoherence in a walk on the line or the cycle is beneficial, producing more uniform distributions (line) and faster mixing to a uniform distribution (cycle).[24] In 2008, Marquezino found a controlled amount of decoherence help in obtaining a shorter mixing time to a uniform probability distribution over the hypercube.[28] Thus, we notice that the optimized SKW algorithm is also based on a walk on the hypercube. When it is in the presence of decoherence, it will make the probability distribution uniform over the hypercube, which concerns how the success rate of the algorithm can be achieved. On the other hand, the mixing time concerns the required number of iterations. So, the final effects of decoherence on the optimized SKW algorithm remain to be studied.

This paper investigates the effects of decoherence generated by random broken links in the hypercube on the optimized SKW algorithm. First defining the shift operator with the possibility of broken links, the optimized SKW algorithm with decoherence is depicted. Then through numerical simulations and analysis, it is found that decoherence will reduce both the maximum success rate of the algorithm and the required number of iterations. The expression of the maximum success rate and the expression of the required number of iterations are obtained respectively. Based on the above relationships, we get the algorithmic computational complexity when it is in the presence of decoherence. We obtain the critical values of the decoherence rate $(\ln a)/(3\cdot 2^{n-2})$ and $(\ln\sqrt{2^n})/(3\cdot 2^{n-2})$. When the decoherence rate varies in different intervals of them, the algorithmic complexity will vary, taking $O(\sqrt{2^n})$ and $O(2^n)$ as the critical values.

The rest of this paper is organized as follows. In Section 2, we introduce the optimized SKW algorithm. In Section 3, the optimized SKW algorithm with decoherence is depicted. Section 4 gives the numerical analysis and results of the algorithm with decoherence. Section 5 is the conclusion.

2. Optimized SKW algorithm

The data in the 2^n-sized database can be expressed as $\boldsymbol{x}=(x_1,x_2,\ldots x_n)$, $x_i=\{0,1\}$. $|\boldsymbol{x}|$ is its Hamming weight and $p(\boldsymbol{x})=|\boldsymbol{x}| \bmod 2$ is its parity bit. Then a map from 2^n to 2^{n+1} can be built, $\boldsymbol{x}\to\boldsymbol{x}'=\boldsymbol{x}||p(\boldsymbol{x})$, i.e., adding the parity bit at the end. Due to the oracle containing information of the marked state and $\boldsymbol{x}$ is one-to-one with $\boldsymbol{x}'$, we suppose $\boldsymbol{x}'$ can also be identified by the oracle.

Then $\boldsymbol{x}'$ can correspond to the vertex of an $n+1$ dimensional hypercube. They are connected only if their Hamming distance is 1. Assuming the marked vertex is $|0\rangle$, since the hypercube is symmetrical, its selection will not affect the final result. This database search problem is then equivalent to searching for a single marked vertex amongst the vertices on the hypercube.

In detail, the optimized SKW algorithm can be described by the repeated application of a unitary evolution operator UU' on the Hilbert space $H=H^{\mathrm{C}}\otimes H^{\mathrm{S}}$ of an $(n+1)$-dimensional hypercube. H^{C} represents the $(n+1)$-dimensional Hilbert space of the quantum coin, and H^{S} represents the 2^{n+1}-dimensional Hilbert space of the vertex. The state in H is $|d,\boldsymbol{x}\rangle$, where d specifies the state of the coin and $\boldsymbol{x}$ specifies the position on the hypercube. The unitary operators are

$$U=S\cdot C_0 \tag{1}$$

and

$$U'=S\cdot C', \tag{2}$$

where the shift operator is

$$S=\sum_{x=0}^{2^{n+1}-1}\sum_{d=1}^{n+1}|d,\boldsymbol{x}\oplus\boldsymbol{e}_d\rangle\langle d,\boldsymbol{x}|. \tag{3}$$

It will depend on the state of the coin d to perform a control transfer, mapping the state from $|d,\boldsymbol{x}\rangle$ to $|d,\boldsymbol{x}\oplus\boldsymbol{e}_d\rangle$, where $|\boldsymbol{e}_d\rangle=|0\cdots010\cdots0\rangle$ is a null vector except for a single 1 entry at the d-th component. The most remarkable fact about S is that it contains all information about the topology of the graph. The coin operator can be written as

$$C'=C_0\otimes I+(C_1-C_0)\otimes|0\rangle\langle 0|. \tag{4}$$

C_0 is generally chosen as the Grover operator, i.e.

$$C_0=G=2|S^C\rangle\langle S^C|-I. \tag{5}$$

It acts on the unmarked vertices, where

$$|S^C\rangle=\frac{1}{\sqrt{n+1}}\sum_{d=1}^{n+1}|d\rangle.$$

C_1 is generally chosen as $-I$, acting on the marked vertex.

The detail steps of the optimized SKW algorithm are as follows:

(i) Initialize the quantum computer to the equal superposition over all states, $|\psi_0\rangle = |S^C\rangle \otimes |S^S\rangle$. Using orthogonal projection P_e, $|\psi_0\rangle$ is mapped to the basis vectors that the Hamming weight of vertexes are even, and obtain the initial state $|\psi_0{}^{(e)}\rangle$.

(ii) Apply the evolution operator $UU' = S \cdot (C_0 \otimes I) \cdot S \cdot C'$ to the initial state $|\psi_0{}^{(e)}\rangle$ with $(\pi/4)\sqrt{2^n}$ times. C' is used to flip the phase of the marked state, $C'|S^C\rangle \otimes |x\rangle = (-1)^{f(x)}|S^C\rangle \otimes |x\rangle$, when $|x\rangle = |0\rangle$, $f(x) = 1$; when $|x\rangle \neq |0\rangle$, $f(x) = 0$.

(iii) Measure the quantum state. The probability to obtain the marked state $|S^C\rangle \otimes |0\rangle$ is $1 - O(1/(n+1))$.

3. The optimized SKW algorithm with decoherence

Quantum coherence will be disturbed by the influence of random events, which are usually modeled by some nonunitary disturbances, such as random measurements, or unitary disturbances, such as broken links.[25] These events are characterized by a rate defined in terms of a probability parameter δ. $t_0 = 1/\delta$ has been proved to be an important critical value.[25] When the time of quantum random walks is beyond it, the characteristics of probability distribution of quantum random walks will disappear and the classical behavior emerge. This paper considers that decoherence is caused by topological noise which breaks links randomly between connected sites of the hypercube. Currently, the effects of this decoherence model on quantum random walks have been investigated deeply, including on a line,[25] on a plane[26] and on a hypercube.[28]

In detail, each vertex in an n dimensional hypercube is connected with other n adjacent vertices. At each time step, the link between two vertices will be broken at probability δ. When it is broken, the graph is no longer a hypercube. So, the shift operator in the original algorithm needs to be redefined to include the possibility of broken links, as

$$\widetilde{S} = \sum_{x=0}^{2^n-1} \sum_{d=1}^{n} |d, \boldsymbol{x} \oplus \boldsymbol{e}_d(x)\rangle\langle d, \boldsymbol{x}|, \tag{6}$$

where

$$\boldsymbol{e}_d(x) = \begin{cases} \boldsymbol{e}_d & \text{if the link at site } x \text{ in direction } d \text{ is good,} \\ 0, & \text{if the link at site } x \text{ in direction } d \text{ is broken.} \end{cases}$$

The function $\boldsymbol{e}_d(x)$ only depends on x and d. If at site x the link in direction d is broken, there will be $|d, \boldsymbol{x} \oplus \boldsymbol{e}_d\rangle\langle d, \boldsymbol{x}| = |d, \boldsymbol{x}\rangle\langle d, \boldsymbol{x} \oplus \boldsymbol{e}_d| = |d, \boldsymbol{x}\rangle\langle d, \boldsymbol{x}|$. The probability flux which will originally cross the link can only be maintained *in situ*. The new shift operator $\widetilde{S}$ will preserve unitarity, no matter which links break.

The actual steps of the optimized SKW algorithm with decoherence are as follows:

i) Initialize the quantum computer to the equal superposition over all states, $|\psi_0\rangle = |S^C\rangle \otimes |S^S\rangle$. By orthogonal projection P_e, $|\psi_0\rangle$ is mapped to the basis vectors that the Hamming weight of vertexes are even, obtaining the initial state $|\psi_0{}^{(e)}\rangle$.

ii) Apply the evolution operator $\widetilde{U_i}\widetilde{U'_i}$ to the initial state $|\psi_0{}^{(e)}\rangle$ with t times, obtaining

$$|\psi_t\rangle = (\widetilde{U_1}\widetilde{U'_1})(\widetilde{U_2}\widetilde{U'_2}) \cdots (\widetilde{U_t}\widetilde{U'_t})|\psi_0{}^{(e)}\rangle.$$

The shift operator $\widetilde{S}$ varies randomly in every $\widetilde{U_i}$ and $\widetilde{U'_i}$ $(1 \leq i \leq t)$.

iii) Measure the quantum state. The probability to obtain the marked state $|S^C\rangle \otimes |0\rangle$ is $\widetilde{p}$.

Because the link breaks randomly, it is not practical to build an algorithm model by theoretical analysis to study effects on the algorithm. We will first investigate the characteristics of the algorithm through numerical simulations when it is in the presence of decoherence. There are $n \cdot 2^{n-1}$ links in the n-dimensional hypercube. Because the probability of each link being broken is δ, assuming per time step the number of broken links satisfies the Poisson distribution, i.e. $\delta \cdot n \cdot 2^{n-1}$ and they are selected randomly.

First, we study the relation between the number of iterations and the success rate of the algorithm when $n = 8$. We simulate situations for $\delta = 0$, $\delta = 5 \times 10^{-4}$, and $\delta = 0.01$ respectively. See Fig. 1.

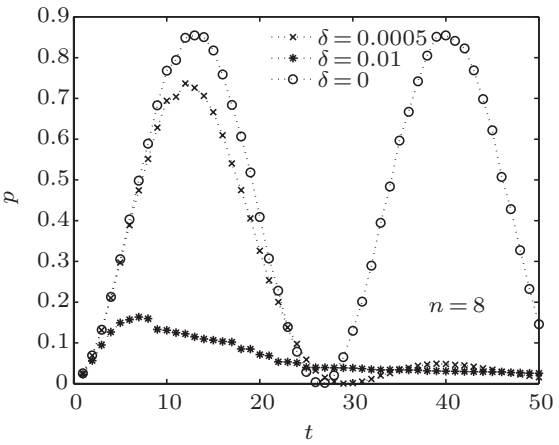


Fig. 1. Relation between the number of iterations t and the success rate of the algorithm p when $n = 8$. The asterisks line represents $\delta = 0.01$. The "x" line represents $\delta = 5 \times 10^{-4}$ and the circles line represents $\delta = 0$.

We can see the general relation between the number of iterations and the success rate of the algorithm from Fig. 1. The circles line represents that it needs 13 iteration times to reach the maximum probability when the decoherence rate is 0. Its success rate of the algorithm changes periodically. The x line represents that it needs 12 iteration times to reach the maximum probability when the decoherence rate is 5×10^{-4}, but it is smaller than the maximum probability obtained in the previous case, and the next peak of wave is even lower. So, the success rate of the algorithm no longer periodically changes. The asterisks line represents that the decoherence rate is 0.01.

Both the maximum probability and the required number of iterations are the smallest, and the success rate changes gently.

Then we study the effects of decoherence on the different databases. Figure 2 shows the relation between the number of iterations and the success rate of the algorithm when the decoherence rate is 5×10^{-4}. The database size is $N=2^n$ and $n=7,8,9$.

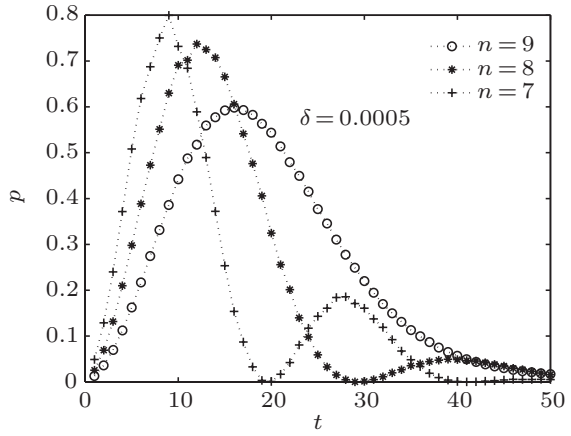


Fig. 2. Relation between the number of iterations t and the success rate of the algorithm p when $\delta=0.0005$. The circles line represents $n=9$. The asterisks line represents $n=8$ and the "+" line represents $n=7$.

It is shown that first the success rate increases with the increment of the number of iterations. The larger the database, the more the required number of iterations. However, the maximum probability that can be achieved is smaller.

Above we describe the optimized SKW algorithm with decoherence, making a preliminary study of the success rate of the algorithm and the number of iterations when it is in the presence of decoherence. The effects of the same decoherence rate on different databases have also been studied. Simulations show that the impact on the maximum probability is greater than that on the required number of iterations. For the same decoherence rate, the larger database will be subject to a greater impact. In the following we will further study the quantitative relationship between the success rate of the algorithm, the number of iterations, the decoherence rate, and the database size.

4. Numerical analyses and results

First we study the relationship between the decoherence rate and the maximum success rate of the algorithm. The size of databases are $2^7,2^8,2^9$ respectively. See Fig. 3.

It can be seen from Fig. 3 that the maximum success rate decreases exponentially with the increment of the decoherence rate. The larger the database, the faster the decrease. With curve fitting, we obtain the expression for the maximum success rate $p_{\max}$, the size of the database 2^n and the decoherence rate δ as

$$p_{\max}=p_0\cdot(\mathrm{e}^{-2^n\cdot\delta+b}+1-\mathrm{e}^{b}), \tag{7}$$

where p_0 is the success rate of the algorithm without decoherence. $-1<b<0$, and with the increment of n, b gradually tends to 0. So the expression is derived approximately as

$$p_{\max}\approx p_0\cdot\mathrm{e}^{-2^n\cdot\delta}. \tag{8}$$

We notice that the form of this expression is the same as the expression of hitting time probability to the opposite corner in Section 6 of Ref. [24], which is about the decoherence rate and the dimension of the hypercube. It means the impact of decoherence on them is similar.

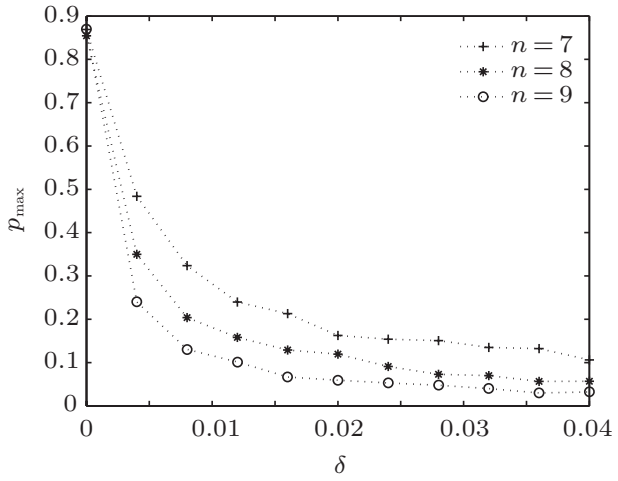


Fig. 3. Maximum success rate of the algorithm $p_{\max}$ versus the decoherence rate δ. The circles line represents $n=9$. The asterisks line represents $n=8$ and the "+" line represents $n=7$.

Correspondingly, the relationship between the decoherence rate and the required number of iterations is shown in Fig. 4. It can be seen from Fig. 4 that the required number of iterations decreases exponentially with the increment of the decoherence rate. The larger the database, the faster the decrease. With curve fitting, we obtain the expression for the required number of iterations t_0, the size of the database 2^n and the decoherence rate δ as

$$t_0=\frac{\pi}{4}\sqrt{2^n}\cdot(\mathrm{e}^{-2^{n-2}\cdot\delta+c}+1-\mathrm{e}^{c}), \tag{9}$$

where $-1<c<0$. With the increment of n, c gradually tends to 0. So the expression is derived approximately as

$$t_0\approx\frac{\pi}{4}\sqrt{2^n}\cdot\mathrm{e}^{-2^{n-2}\cdot\delta}. \tag{10}$$

Because $\delta\geq0$, $0<\mathrm{e}^{-2^{n-2}\cdot\delta}\leq1$ is established. Assuming d is a constant, when $\mathrm{e}^{-2^{n-2}\cdot\delta}=d$, we obtain $\delta=-\ln d/2^{n-2}$. Therefore, when $\delta=0$, the required number of iterations is $(\pi/4)\sqrt{2^n}$. When $0<\delta\leq-\ln d/2^{n-2}$, the required number of iterations is $(\pi/4\cdot\mathrm{e}^{2^{n-2}\cdot\delta})\sqrt{2^n}$. Its computing scale does not change. When $\delta>-\ln d/2^{n-2}$, the required number of iterations is $(\pi/4)\sqrt{2^n/\mathrm{e}^{2^{n-1}\cdot\delta}}$, which means the computing scale is dropped.

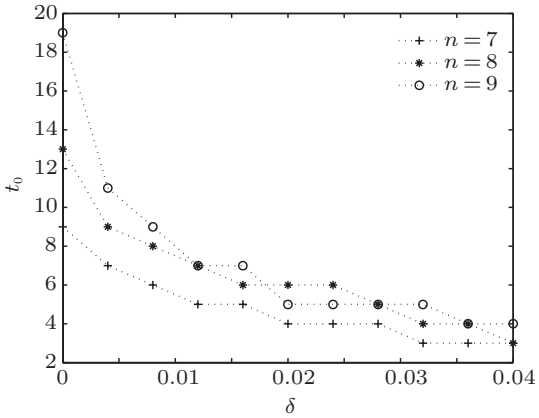


Fig. 4. Required number of iterations t_0 versus the decoherence rate δ. The circles line represents $n=9$. The asterisks line represents $n=8$ and the "+" line represents $n=7$.

From the above analyses, when the algorithm is in the presence of decoherence, both the maximum success rate and the required number of iterations will decrease. Therefore, the computational complexity of the algorithm may change. Here we define its computational complexity as the required number of iterations to obtain a success probability of p_0. If the probability of obtaining the marked state upon measurement after t iterations is p, then by repeating this procedure approximately p_0/p times, we can boost the overall success probability to p_0. Thus, the overall complexity of the algorithm is $O(t/p)$.[21] When it is in the presence of decoherence, let $t=t_0$ and $p=p_{\max}$. So the overall complexity of the algorithm is $O(t_0/p_{\max})$. Figure 5 shows the relationship between the decoherence rate δ and $t_0/p_{\max}$.

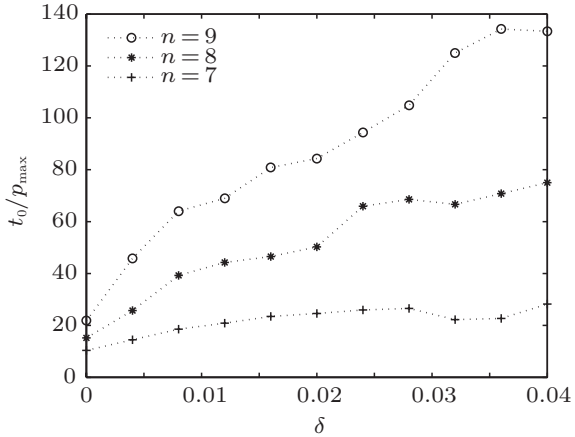


Fig. 5. $t_0/p_{\max}$ vs the decoherence rate δ. The circles line represents $n=9$. The asterisks line represents $n=8$ and the "+" line represents $n=7$.

Shown in Fig. 5, $t_0/p_{\max}$ increases with the increment of the decoherence rate. The larger the database, the faster the decrease. It suggests that decoherence will degrade the quantum effects of the algorithm and may lead to the algorithmic complexity increasing. Then, it will not speed up as expected. When the decoherence rate increases to a critical value, the quantum effects of the algorithm may disappear completely, which will not have any speed up compared with the classical search. We evaluate

$$O\left(\frac{t_0}{p_{\max}}\right) = O\left(\frac{(\pi/4)\sqrt{2^n}\cdot \mathrm{e}^{-2^{n-2}\cdot\delta}}{p_0\cdot \mathrm{e}^{-2^n\cdot\delta}}\right) = O(\sqrt{2^n}\cdot \mathrm{e}^{2^{n-2}\cdot 3\delta}). \tag{11}$$

When $\sqrt{2^n}\cdot \mathrm{e}^{2^{n-2}\cdot 3\delta}=2^n$, we obtain $\delta=\ln\sqrt{2^n}/3\cdot 2^{n-2}$. Because $\delta\geq 0$, $\mathrm{e}^{2^{n-2}\cdot 3\delta}\geq 1$ is found. Assuming a is a constant, when $\mathrm{e}^{2^{n-2}\cdot 3\delta}=a$, we obtain $\delta=\ln a/3\cdot 2^{n-2}$.

In summary, when $0<\delta\leq \ln a/3\cdot 2^{n-2}$, the complexity of the algorithm is $O(\sqrt{2^n})$. It still has a quadratic increase in speed. When $\ln a/3\cdot 2^{n-2}<\delta<\ln\sqrt{2^n}/3\cdot 2^{n-2}$, the algorithm can only produce a certain quantum increase in speed. The complexity of the algorithm is $O(\sqrt{2^n}\cdot \mathrm{e}^{2^{n-2}\cdot 3\delta})$, between $O(\sqrt{2^n})$ and $O(2^n)$. When $\delta=\ln\sqrt{2^n}/3\cdot 2^{n-2}$, the complexity of the algorithm is $O(2^n)$, which is the same with the classical search and has no quantum increase in speed. When $\delta>\ln\sqrt{2^n}/3\cdot 2^{n-2}$, the algorithmic complexity may be beyond the classical search.

Above investigations show that the broken-link-type decoherence will make the probability distribution on the hypercube tend to be uniform. This is in conflict with the purpose of marking the target by oracle in the search algorithm, which attempts to focus the probability distribution on the target. Although the decoherence will result in the decrease of the required number of iterations, its ultimate impact on the algorithm is negative from the computational complexity point of view.

5. Conclusion

This paper studies the effects of broken-link-type decoherence on the optimized SKW algorithm in detail. By defining the shift operator with the possibility of broken links, the optimized SKW algorithm with decoherence is depicted. Through simulations and analyses, we find this decoherence will decrease the maximum success rate and the required number of iterations. We obtain two expressions for the maximum success rate and the required number of iterations as a function of the decoherence rate and the database size. When $\delta=0$, the required number of iterations is $(\pi/4)\sqrt{2^n}$, which is the number of iterations of the ideal algorithm. When $0<\delta\leq -\ln d/2^{n-2}$, the required number of iterations is $(\pi/4\cdot \mathrm{e}^{2^{n-2}\cdot\delta})\sqrt{2^n}$. Its computing scale is still $\sqrt{2^n}$, but the coefficient decreases. When $\delta>-\ln d/2^{n-2}$, the required number of iterations is $(\pi/4)\sqrt{2^n/(\mathrm{e}^{2^{n-1}\cdot\delta})}$, which reduces the computing scale. Finally, we obtain the algorithmic complexity when it is in the presence of decoherence. The algorithmic complexity is $O(2^n)$, when the decoherence rate satisfies $0<\delta\leq \ln a/3\cdot 2^{n-2}$. The algorithmic complexity is $O(\sqrt{2^n}\cdot \mathrm{e}^{2^{n-2}\cdot 3\delta})$, when the decoherence rate satisfies $\ln a/3\cdot 2^{n-2}<$

$\delta < \ln\sqrt{2^n}/3\cdot 2^{n-2}$. When $\delta = \ln\sqrt{2^n}/3\cdot 2^{n-2}$, the algorithmic complexity is $O(2^n)$, which is the same with the classical search. When $\delta > \ln\sqrt{2^n}/3\cdot 2^{n-2}$, the algorithmic complexity may be beyond the classical search. The above conclusions show that the ultimate effects of broken-link-type decoherence on the optimized SKW algorithm is negative.

References

[1] Grover L 1996 *Proceedings of the 28th Annual ACM Symposium on Theory of Computing* (New York: ACM Press) p. 212
[2] Shor P W 1997 *SIAM J. Comput.* **26** 1484
[3] Schoning T 1999 *Foundations of Computer Science, the 40th Annual Symposium on IEEE*, pp. 410–414
[4] Motwani R and Raghavan P 1996 *Randomized Algorithms* (Cambridge: Cambridge University Press)
[5] Kempe J 2005 *Probability Theory and Related Fields* **133** 215
[6] Krovi H and Brun T A 2006 *Phys. Rev. A* **73** 032341
[7] Nayak A and Vishwanath A 2000 arXiv: 0010117v1 [quant-ph]
[8] Xue P and Zhang Y S 2013 *Chin. Phys. B* **22** 070302
[9] Qin H and Xue P 2014 *Chin. Phys. B* **23** 010301
[10] Zhang R, Xu Y Q and Xue P 2015 *Chin. Phys. B* **24** 010303
[11] Shenvi N, Kempe J and Whaley K B 2003 *Phys. Rev. A* **67** 052307
[12] Potocek V, Gábris A, Kiss T and Jex I 2009 *Phys. Rev. A* **79** 012325
[13] Li Y 2006 "Investigations on Quantum Random-Walk Search Algorithm", MS Thesis (Shanghai: East China Normal University) p. 34 (in Chinese)
[14] Zhang Y C, Bao W S, Wang X and Fu X Q 2015 *Chin. Phys. B* **24** 060304
[15] Childs A M, Cleve R, Deotto E, Farhi E, Gutmann S and Spielman D A 2003 *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, pp. 59–68
[16] Ambainis A 2007 *SIAM J. Comput.* **37** 210
[17] Krovi H, Magniez F, Ozols M and Roland J 2010 *Automata Languages and Programming* (Berlin/Heidelberg: Springer) pp. 540–551
[18] Childs A and Goldstone J 2004 *Phys. Rev. A* **70** 022314
[19] Tulsi A 2008 *Phys. Rev. A* **78** 012310
[20] Long G L, Li Y S, Zhang W L and Tu C C 2000 *Phys. Rev. A* **61** 042305
[21] Shenvi N, Brown K R and Whaley K B 2003 *Phys. Rev. A* **68** 052313
[22] Li Y, Ma L and Zhou J 2006 *J. Phys. A: Math. Gen.* **39** 9309
[23] Abal G, Donangelo R, Marquezino F L, Oliveira A C and Portugal R 2009 arXiv: 0912.1523v1 [quant-ph]
[24] Kendon V and Tregenna B 2003 *Phys. Rev. A* **67** 042315
[25] Romanelli A, Siri R, Abal G, Auyuanet A and Donangelo R 2005 *Physica A: Statistical Mechanics and Its Applications* **347** 137
[26] Oliveira A C, Portugal R and Donangelo R 2006 *Phys. Rev. A* **74** 012312
[27] Alagic G and Russell A 2005 *Phys. Rev. A* **72** 062304
[28] Marquezino F L, Portugal R, Abal G and Donangelo R 2008 *Phys. Rev. A* **77** 042312