Finding tree symmetries using continuous-time quantum walk*

Wu Jun-Jie(吴俊杰)[a)b)†], Zhang Bai-Da(张百达)[a)b)], Tang Yu-Hua(唐玉华)[a)b)],
Qiang Xiao-Gang(强晓刚)[c)], and Wang Hui-Quan(王会权)[a)b)]

[a)] *State Key Laboratory of High Performance Computing, National University of Defense Technology, Changsha* 410073, *China*
[b)] *School of Computer, National University of Defense Technology, Changsha* 410073, *China*
[c)] *Center for Quantum Photonics, H. H. Wills Physics Laboratory and Department of Electrical and Electronic Engineering, University of Bristol, Merchant Venturers Building, Woodland Road, Bristol BS8 1UB, UK*

(Received 8 September 2012; revised manuscript received 23 November 2012)

Quantum walk, the quantum counterpart of random walk, is an important model and widely studied to develop new quantum algorithms. This paper studies the relationship between the continuous-time quantum walk and the symmetry of a graph, especially that of a tree. Firstly, we prove in mathematics that the symmetry of a graph is highly related to quantum walk. Secondly, we propose an algorithm based on the continuous-time quantum walk to compute the symmetry of a tree. Our algorithm has better time complexity $O(N^3)$ than the current best algorithm. Finally, through testing three types of 10024 trees, we find that the symmetry of a tree can be found with an extremely high efficiency with the help of the continuous-time quantum walk.

Keywords: quantum walk, tree, symmetry, automorphism

PACS: 03.67.–a, 02.10.Ox, 89.20.Ff, 07.05.Tp **DOI:** 10.1088/1674-1056/22/5/050304

1. Introduction

Random walk is widely used in science to model many processes, such as molecular diffusion in physics, food selection of animals in biology, and fluctuation of stock price in economics. Its quantum counterpart, quantum walk, was firstly proposed by Aharonov, Davidovich, and Zagury. Compared with random walk, quantum walk has many advantages, including faster spreading.[1] Based on those advantages, a lot of new quantum algorithms have been proposed.[2–10] Therefore, many researches on quantum walk have been carried out.

In modern science, the graph is an important abstract model in many fields. Conversely, these fields also push forward the research of the graph theory. We investigate in this paper the symmetry of graph, which is very basic and important in the graph theory. It is related to the automorphism of the graph.[11–15]

This paper studies the relationship between quantum walk and the symmetry by performing quantum walk on a graph, especially a tree. The main contributions of our work are listed as follows. 1) We prove that the symmetry of a graph is highly related to quantum walk. More specifically, two symmetric vertices preserve equivalent probability amplitudes during a continuous-time quantum walk only if the quantum walk starts from a quantum state of equivalent probability amplitudes. 2) According to the above conclusion, we propose an algorithm based on the continuous-time quantum walk to compute the symmetry of a tree. Our algorithm firstly performs a continuous-time quantum walk on the tree, and then judges the common neighbor condition of the MAST algorithm,[16] and finally computes the symmetry according to the size of brethren sets. To our knowledge, it has better time complexity ($O(N^3)$) than the current best algorithm. 3) We find that the continuous-time quantum walk helps in finding the symmetry of a tree with an extremely high efficiency. We have performed continuous-time quantum walks on 24 Cayley trees, 5000 random trees, and 5000 scale-free trees.

The rest of this paper is organized as follows. Section 2 introduces quantum walk briefly. Section 3 proves the close relationship between the continuous-time quantum walk and the symmetry of a graph. Section 4 proposes a quantum walk algorithm for computing the symmetry. In Section 5, a lot of graphs are tested, and we demonstrate that the continuous-time quantum walk has a very high efficiency in finding the symmetry. We conclude in Section 6.

2. Preliminary and related work

Quantum walk is the quantum counterpart of random walk, which is an important model in many areas and has been widely studied.[17–20] There are two models of quantum walk, discrete-time quantum walk and continuous-time quantum walk. The discrete-time quantum walk introduces coin states just like the random walk, while the continuous-time quantum walk does not. They are both widely studied to construct quantum algorithms with better performance than that of the classical algorithms. A lot of algorithms have been developed in many applications, such as search, element

*Project supported by the National Natural Science Foundation of China (Grant No. 61003082).
†Corresponding author. E-mail: junjiewu@nudt.edu.cn
© 2013 Chinese Physical Society and IOP Publishing Ltd

distinctness, combinatorial optimization, and even universal computing.[2–9] A quantum computer with these quantum algorithms may have a better performance than the best classical computer, such as a high performance computer.[21,22] The quantum implementations of quantum walk are also widely studied.[23–29] Figure 1 compares the discrete-time quantum walk on a line with the random walk. Obviously, the quantum walk spreads faster than the classical one. After 100 steps, the random walker still has the highest probability to position at 0, while the quantum walker probably positions at 70 or −70. If we measure the position of the quantum walker at every step, the discrete-time quantum walk will degenerate into a random walk.

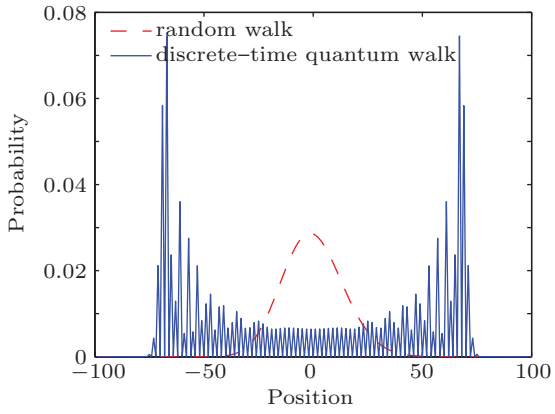


Fig. 1. (color online) Random walk and discrete-time quantum walk on a line after 100 steps.

The continuous-time quantum walk on a graph is defined as

$$|\psi_t\rangle = \mathrm{e}^{-\mathrm{i}\boldsymbol{A}t}|\psi_0\rangle, \tag{1}$$

where $|\psi_0\rangle$ is the initial quantum state, $|\psi_t\rangle$ is the quantum state at time t, and $\boldsymbol{A}$ is the adjacent matrix of the graph. Specially, many researchers use the Laplacian matrix instead of the adjacent one. Just like that in Ref. [30], we adopt the adjacent matrix in this paper. The $|\psi_0\rangle$ and $|\psi_t\rangle$ belong to a Hilbert space spanned by a group of basic states. Each basic state corresponds to a vertex of the graph. Thus, the state space is N-dimensional, where N is the number of the vertices.

Take the graph in Fig. 2 for example, its adjacent matrix is

$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Because the graph has eight vertices, the quantum state space is 8-dimensional. For example, the initial quantum state of a continuous-time quantum walk may be ($N = 8$ for Fig. 2)

$$|\psi_0\rangle = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}, \tag{2}$$

which has identical probability amplitudes, that means each vertex has an identical probability amplitude at the starting moment of the quantum walk. Thus, the quantum state at time t of the continuous-time quantum walk is

$$|\psi_t\rangle = \frac{1}{2\sqrt{2}} \mathrm{e}^{-\mathrm{i}\boldsymbol{A}t} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}.$$

More details about the continuous-time quantum walk can be found in Ref. [31].

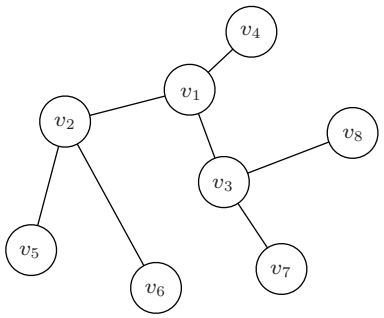


Fig. 2. A graph.

3. Relationship between quantum walk and graph symmetry

In this section, we study the relationship between quantum walk and graph symmetry. Firstly, we give the definitions of graph symmetry and automorphism.[16,32,33]

Definition 1 (automorphism) The $G(V,E)$ is a graph. A bijective mapping $f : V \to V$ is an automorphism of graph G if and only if $\forall v_1,\ v_2,\ (v_1,v_2) \in E \Leftrightarrow (f(v_1),f(v_2)) \in E$.

Definition 2 (symmetry) The symmetry of graph $G(V,E)$ is the number of all its automorphisms.

Definition 1 defines the automorphism of a graph. For example, $f_1, f_2, \ldots, f_8$ in Table 1 are automorphism mappings of the graph in Fig. 2. Specially, f_1 is a trivial automorphism mapping. Definition 2 defines the symmetry of a graph. The symmetry of the graph in Fig. 2 equals 8, all possible automorphism mappings are shown in Table 1.

Table 1. Automorphism mappings of the graph in Fig. 2.

f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
$v_1 \to v_1$	$v_1 \to v_1$	$v_1 \to v_1$	$v_1 \to v_1$	$v_1 \to v_1$	$v_1 \to v_1$	$v_1 \to v_1$	$v_1 \to v_1$
$v_2 \to v_2$	$v_2 \to v_2$	$v_2 \to v_2$	$v_2 \to v_2$	$v_2 \to v_3$	$v_2 \to v_3$	$v_2 \to v_3$	$v_2 \to v_3$
$v_3 \to v_3$	$v_3 \to v_3$	$v_3 \to v_3$	$v_3 \to v_3$	$v_3 \to v_2$	$v_3 \to v_2$	$v_3 \to v_2$	$v_3 \to v_2$
$v_4 \to v_4$	$v_4 \to v_4$	$v_4 \to v_4$	$v_4 \to v_4$	$v_4 \to v_4$	$v_4 \to v_4$	$v_4 \to v_4$	$v_4 \to v_4$
$v_5 \to v_5$	$v_5 \to v_6$	$v_5 \to v_5$	$v_5 \to v_6$	$v_5 \to v_7$	$v_5 \to v_7$	$v_5 \to v_8$	$v_5 \to v_8$
$v_6 \to v_6$	$v_6 \to v_5$	$v_6 \to v_6$	$v_6 \to v_5$	$v_6 \to v_8$	$v_6 \to v_8$	$v_6 \to v_7$	$v_6 \to v_7$
$v_7 \to v_7$	$v_7 \to v_7$	$v_7 \to v_8$	$v_7 \to v_8$	$v_7 \to v_5$	$v_7 \to v_6$	$v_7 \to v_5$	$v_7 \to v_6$
$v_8 \to v_8$	$v_8 \to v_8$	$v_8 \to v_7$	$v_8 \to v_7$	$v_8 \to v_6$	$v_8 \to v_5$	$v_8 \to v_6$	$v_8 \to v_5$

In graph theory, an automorphism mapping of a graph is a special kind of permutation of the vertices. It can be represented by a permutation matrix $\boldsymbol{P}$, which satisfies

$$\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{\mathrm{T}} = \boldsymbol{A}, \tag{3}$$

where $\boldsymbol{A}$ is the adjacent matrix of the graph.

A permutation matrix is a square binary matrix that has exactly one entry 1 in each row and each column and 0's elsewhere. A permutation matrix is an orthogonal matrix, i.e., $\boldsymbol{P}\boldsymbol{P}^{\mathrm{T}} = \boldsymbol{I}$, so the inverse matrix of a permutation matrix is its transpose, i.e., $\boldsymbol{P}^{-1} = \boldsymbol{P}^{\mathrm{T}}$. For example, the automorphism mapping f_2 corresponds to matrix permutation $\boldsymbol{P}_{f_2}$

$$\boldsymbol{P}_{f_2} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{4}$$

Theorem 1 Perform a continuous-time quantum walk on graph $G(V,E)$ from an initial quantum state of identical probability amplitudes $|\psi_0\rangle$ shown in Eq. (2). If f is an automorphism mapping of $G(V,E)$, the probability amplitudes of any vertex v ($v \in V$) and vertex $f(v)$ will always be kept equal during the walking.

Proof Let $\boldsymbol{A}$ be the adjacent matrix of $G(V,E)$, and $\boldsymbol{P}$ be the permutation matrix corresponding to automorphism mapping f. Thus, $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{\mathrm{T}} = \boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{-1}$.

Elements of $|\psi_0\rangle$ have identical probability amplitudes, any permutation of elements of $|\psi_0\rangle$ will not change it. Obviously, the inverse of a permutation matrix is still a permutation matrix. Thus, $\boldsymbol{P}^{-1}|\psi_0\rangle = |\psi_0\rangle$.

Let $|\psi_t\rangle$ be the quantum state at time t, so

$$\begin{aligned} |\psi_t\rangle &= \mathrm{e}^{-\mathrm{i}\boldsymbol{A}t}|\psi_0\rangle \\ &= \sum_{k=0}^{+\infty} \frac{(-\mathrm{i}t)^k \boldsymbol{A}^k}{k!}|\psi_0\rangle \\ &= \sum_{k=0}^{+\infty} \frac{(-\mathrm{i}t)^k (\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{-1})^k}{k!}|\psi_0\rangle \\ &= \sum_{k=0}^{+\infty} \frac{(-\mathrm{i}t)^k \boldsymbol{P}\boldsymbol{A}^k\boldsymbol{P}^{-1}}{k!}|\psi_0\rangle \\ &= \boldsymbol{P}\left[\sum_{k=0}^{+\infty} \frac{(-\mathrm{i}t)^k \boldsymbol{A}^k}{k!}\right]\boldsymbol{P}^{-1}|\psi_0\rangle \\ &= \boldsymbol{P}\left[\sum_{k=0}^{+\infty} \frac{(-\mathrm{i}t)^k \boldsymbol{A}^k}{k!}\right]|\psi_0\rangle \\ &= \boldsymbol{P}\mathrm{e}^{-\mathrm{i}\boldsymbol{A}t}|\psi_0\rangle \\ &= \boldsymbol{P}|\psi_t\rangle. \end{aligned} \tag{5}$$

Let p_i be the element of $|\psi_t\rangle$ corresponding to vertex v_i ($i = 1,\ldots,N$, N is the number of vertices), i.e.,

$$|\psi_t\rangle = \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_N \end{pmatrix}.$$

Obviously,

$$\boldsymbol{P}|\psi_t\rangle = \begin{pmatrix} f(p_1) \\ f(p_2) \\ \vdots \\ f(p_N) \end{pmatrix}.$$

Because $\boldsymbol{P}|\psi_t\rangle = |\psi_t\rangle$ proven in Eq. (5), $p_i = f(p_i)$ ($i = 1,\ldots,N$). Therefore, Theorem 1 is proved.

To explain Theorem 1, we show the continuous-time quantum walk on the graph in Fig. 2. In Table 2, the probability amplitudes at several moments are shown. At $t = 0$, the state is the initial quantum state that satisfies Eq. (2). Because there are automorphism mappings that connect v_2 and v_3, i.e., f_5, f_6, f_7, f_8 in Table 1, the probability amplitudes of v_2 and v_3 are kept equal as shown in Table 2.

Theorem 1 reveals the relationship between the continuous-time quantum walk and the symmetry of a graph. To understand the relationship better, we propose Hypothesis 1.

Table 2. Probability amplitudes of continuous-time quantum walk on Fig. 2.

	$t=0$	$t=0.1$	$t=0.2$	$t=0.3$	$t=0.4$
v_1	0.3536	$0.3412-0.1053$i	$0.3048-0.2061$i	$0.2461-0.2979$i	$0.1677-0.3770$i
v_2	0.3536	$0.3447-0.1053$i	$0.3187-0.2061$i	$0.2767-0.2979$i	$0.2206-0.3770$i
v_3	0.3536	$0.3447-0.1053$i	$0.3187-0.2061$i	$0.2767-0.2979$i	$0.2206-0.3770$i
v_4	0.3536	$0.3483-0.0349$i	$0.3326-0.0674$i	$0.3074-0.0952$i	$0.2735-0.1160$i
v_5	0.3536	$0.3483-0.0351$i	$0.3326-0.0684$i	$0.3074-0.0983$i	$0.2735-0.1232$i
v_6	0.3536	$0.3483-0.0351$i	$0.3326-0.0684$i	$0.3074-0.0983$i	$0.2735-0.1232$i
v_7	0.3536	$0.3483-0.0351$i	$0.3326-0.0684$i	$0.3074-0.0983$i	$0.2735-0.1232$i
v_8	0.3536	$0.3483-0.0351$i	$0.3326-0.0684$i	$0.3074-0.0983$i	$0.2735-0.1232$i

Hypothesis 1 Perform a continuous-time quantum walk on graph $G(V,E)$ from an initial quantum state of identical probability amplitudes $|\psi_0\rangle$ shown in Eq. (2). If the probability amplitudes of any two vertices v and v' ($v,v' \in V$) are always kept the same during the walking, there is an automorphism mapping f of $G(V,E)$ so that $v' = f(v)$ or $v = f(v')$.

For the graph in Fig. 2, Hypothesis 1 is right. For example, the probability amplitudes of v_5 and v_7 always are kept equal as shown in Table 2. Mappings f_5, f_6, and f_7 in Table 1 satisfy the requirement in Hypothesis 1, because $v_5 = f_5(v_7)$, $v_7 = f_5(v_5)$, $v_7 = f_6(v_5)$, and $v_5 = f_7(v_7)$. Notice that, a map g that only exchanges v_5 and v_7 ($g(v_i) = v_i$, $i = 1,2,3,4,6,8$, $g(v_5) = v_7$, and $g(v_7) = v_5$) is not an automorphism mapping. Thus, when computing the symmetry of a tree based on the quantum walk, the algorithm in the next section has to judge the common neighbor condition. We construct an algorithm for computing the symmetry of a tree based on Hypothesis 1. We find that all results of our algorithm for 10024 testing graphs show the rightness of Hypothesis 1, where the MAST algorithm[16] is used to check the results.

4. Quantum walk algorithm for finding the symmetry of a tree

In this section, we use the continuous-time quantum walk to study a special type of graph, the tree. The tree is an important kind of graph. The current best algorithm for computing the symmetry of a tree is proposed in Ref. [16]. In this section, we will show a new algorithm based on the continuous-time quantum walk. Our algorithm has better time complexity than the current best algorithm.

4.1. MAST algorithm

To explain our algorithm, we first briefly introduce the MAST algorithm in Ref. [16]. Zhang *et al.* have proven that the vertices of a tree can be divided into several sets of brethren, and two vertices are in an identical brethren set if and only if they satisfy the following two properties. 1) The two vertices has the same view of the graph, i.e., there is an automorphism mapping that maps one vertex to the other. 2) The two vertices has at least one common neighbor. Take the graph in Fig. 2 for example, there are five sets of brethren, i.e., $\{v_1\}$, $\{v_2,v_3\}$, $\{v_4\}$, $\{v_5,v_6\}$, and $\{v_7,v_8\}$.

Zhang *et al.* have proved that the symmetry of a tree can be computed based on the sizes of brethren sets. If a tree has n sets of brethren and the size of each brethren set is k_i ($i = 1,2,\ldots,n$), the symmetry of the tree is $\prod_{i=1}^{n} k_i!$. Thus, the symmetry of the graph in Fig. 2 is 8 ($1!\times 2!\times 1!\times 2!\times 2! = 8$).

4.2. Algorithm based on quantum walk

The key idea of the MAST algorithm is to divide vertices into some brethren sets. The dividing method of the MAST algorithm is recursively deleting vertices from the tree.[16] The key idea of our algorithm is using the quantum walk to divide the vertices. According to Theorem 1, the vertices in a brethren set must have identical probability amplitudes during the continuous-time quantum walk. So, we firstly perform a continuous-time quantum walk on the tree. Secondly, we place the vertices with identical probability amplitudes into one set, and all vertices are divided into several sets. Thirdly, the quantum walk continues and we repeat the partition until there is no new set partitioned. Finally, we partition these sets again according to the common neighbor condition.

Our algorithm is formalized in Algorithm 1. The ST in our algorithm is a set of simulation time. The CAT is a set of sets of vertices, and it is only a middle variable in our algorithm. We only choose N simulation time in ST (N is the number of vertices), because at most each vertex forms a single set in CAT. In our experiment, ST is set to $\{0.1,0.2,\ldots,0.N\}$. In Section 5, we will find that the symmetry of every tested graph is computed at only $t = 0.1$ and $t = 0.2$.

Let us take the graph in Fig. 2 for example to explain the process of Algorithm 1. In the first iteration of the first for loop, t is set to 0.1, $|\psi_t\rangle$ equals the vector corresponding to the $t = 0.1$ column in Table 2. Then, CAT is updated to $\{\{v_1\},\{v_2,v_3\},\{v_4\},\{v_5,v_6,v_7,v_8\}\}$ according to whether the vertices have identical probability amplitudes. In the second iteration, t is changed to 0.2, $|\psi_t\rangle$ is changed to the vector corresponding to the $t = 0.2$ column in Table 2. However, UpdateQW(CAT,$|\psi_t\rangle$) does not change CAT. Thus, the first for loop finishes. In the second for loop, the set $\{v_5,v_6,v_7,v_8\}$ will be partitioned into $\{v_5,v_6\}$ and $\{v_7,v_8\}$ because there is a com-

mon neighbor between v_5 and v_6, as well as between v_7 and v_8, but there is no common neighbor between v_5 and v_7, between v_5 and v_8, between v_6 and v_7, or between v_6 and v_8. Thus, CAT is changed to $\{\{v_1\},\{v_2,v_3\},\{v_4\},\{v_5,v_6\},\{v_7,v_8\}\}$ after finishing the second for loop. The last for loop will compute SYM, SYM $= 1!\times 2!\times 1!\times 2!\times 2! = 8$ according to CAT.

Algorithm 1: Continuous-time quantum walk algorithm for computing the symmetry of a tree.

Input: Tree $G(V,E)$
Output: Symmetry of the tree SYM

begin
  CAT $\longleftarrow \{V\}$; /* CAT is the set of sets of vertices */
  $N \longleftarrow$ size(V) ; /* N is the number of vertices */
  $|\psi_0\rangle \longleftarrow N^{-\frac{1}{2}}(1,1,\ldots,1)^{\mathrm{T}}$; /* $|\psi_0\rangle$ is the initial quantum state */
  /* ST is simulation time set, $\{0.1,0.2,\ldots,0.N\}$ in our experiment */
  for t **in** ST **do**
    /* Quantum walk on G at time t, and $\boldsymbol{A}$ is the adjacent matrix */
    $|\psi_t\rangle \longleftarrow \mathrm{e}^{-\mathrm{i}\boldsymbol{A}t}|\psi_0\rangle$;
    /* According to $|\psi_t\rangle$, partition CAT again, and isUpdated denotes whether CAT is partitioned again */
    isUpdated$\longleftarrow$UpdateQW(CAT, $|\psi_t\rangle$) ;
    if isUpdated == **false then**
      break ; /* Break the loop */
    end
  end
  for brethrenSet **in** CAT **do**
    /* According to whether there is at least a common neighbor between two vertices in brethrenSet, partition the set in CAT again */
    UpdateCN(CAT, brethrenSet) ;
  end
  SYM $\longleftarrow 1$;
  for brethrenSet **in** CAT **do**
    SYM $\longleftarrow$ SYM $\times$ factorial(size(brethrenSet)) ; /* factorial$(n) = n!$ */
  end
  return SYM ;
end

4.3. Complexity analysis

In this section, we analyze the complexity of our algorithm. We will find that our algorithm has better complexity than the current best algorithm in Ref. [16].

The main process of Algorithm 1 can be split into three parts. Each part contains a for loop. Although we set N elements in ST, only constant iterations are needed in the first for loop. Specially, we find that only one iteration is needed in all experiments in Section 5. The complexity of simulation of the continuous-time quantum walk on a graph in a classical computer is $\mathrm{O}(N^3)$. Thus, the complexity of the first for loop in Algorithm 1 is $\mathrm{O}(N^3)$. However if N iterations are needed instead of constant iterations, the worst complexity is $\mathrm{O}(N^4)$.

For the second for loop, let there be k elements in CAT. Let these elements be $C_1, C_2, \ldots, C_k$ ($\sum_{i=1}^{k}$ size$(C_i) = N$). In an iteration for C_i ($i = 1,2,\ldots,k$), the complexity for finishing UpdateCN is $\mathrm{O}\left(\mathrm{size}(C_i)^2\right)$. Thus, the complexity of this loop is $\mathrm{O}\left(\sum_{i=1}^{k}\mathrm{size}(C_i)^2\right)$. Because

$$a_1^2 + a_2^2 + \ldots + a_k^2 \le (a_1 + a_2 + \cdots + a_k)^2$$

if $a_1, a_2, \ldots, a_k \ge 0$, $\sum_{i=1}^{k}\mathrm{size}(C_i)^2 \le \left(\sum_{i=1}^{k}\mathrm{size}(C_i)\right)^2 = N^2$. Thus, the worst complexity of this loop is $\mathrm{O}(N^2)$.

For the last for loop, in the worst case CAT has N elements. In this case, the complexity of this loop is $\mathrm{O}(N)$. Therefore, the complexity of Algorithm 1 is $\mathrm{O}(N^3)$. The current best algorithm for computing the symmetry of a tree is proposed in Ref. [16], and its complexity is $\mathrm{O}(N^4)$. In addition, the simulation of the continuous-time quantum walk can be achieved with complexity $\mathrm{O}(N^{3/2})$ using the Grover technology,[34] so the complexity of our algorithm can be reduced further.

5. Evaluation and analysis

In this section, we will use experiments to test the validity and efficiency of our algorithm proposed in the above section.

Firstly, we introduce our evaluation methodology. Then we report the experiment results.

5.1. Evaluation methodology

The validity denotes whether our algorithm can produce the right result, the symmetry of a tree. To validate this, we compare the results of our algorithm with that from the MAST algorithm.[16] The efficiency denotes how fast our algorithm can find the result, i.e., how many simulation steps of quantum walk are needed.

We test 10024 trees to validate and evaluate our algorithm. Three types of trees are chosen, just like those in Ref. [16]. One type is the Cayley tree. The Cayley tree, introduced by Bethe in 1935, is a kind of connected tree and widely studied in many fields. In a Cayley tree, each vertex has z neighbors, where z is called the coordination number. All vertices are arranged in shells around the root of the tree. A Cayley tree is denoted by two parameters, coordination number z and outermost shell number s. Figure 3 shows a Cayley tree with coordination number $z = 3$ and outermost shell number $s = 4$. Table 4 shows 24 Cayley trees tested in our experiment.

The other two types are random trees and scale-free trees. These two types of trees are generated by Algorithm 2, just like those in Ref. [16]. In line 8 of Algorithm 2, v_p is selected by two strategies according to the tree type. For the random tree, v_p is selected in the set of all existing vertices with identical probability. For the scale-free tree, it is selected with priority to existing vertices that have more neighbors. We test 5000 random trees and 5000 scale-free trees. Every type of trees contain 100 groups. Each group is composed of 50 trees with an identical number of vertices. The range of vertice number of groups is from 20 to 2000 with interval 20, i.e., 20, 40, 60, ..., 2000.

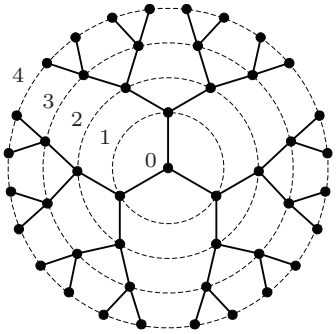


Fig. 3. A Cayley tree with $z = 3$ and $s = 4$.

Algorithm 2: Generating random or scale-free tree.

Input: Size of vertices of tree generated N
Output: Tree $G(V,E)$

1 **begin**
2 $V \longleftarrow v_1$; /* Only one single vertex in the tree */
3 $E \longleftarrow \emptyset$;
4 $n \longleftarrow 1$;
5 **while** $n < N$ **do**
6 $n \longleftarrow n+1$;
7 $V \longleftarrow V \cup \{v_n\}$; /* Add a new vertex v_n */
/* Add a edge linking v_n with an existing vertex v_p */
8 $E \overset{p}{\longleftarrow} E \cup \{v_p, v_n\}$;
9 **end**
10 **return** $G(V,E)$;
11 **end**

The configuration of the computer to implement our algorithm is shown in Table 3.

Table 3. Computer used.

CPU	Intel(R) core(TM) i5 CPU 760, 2.80GHz
Memory	DDR3, 4GB, 1333 MHz
Software	MATLAB v7.11.0.584 (R2010b)

5.2. Validity and efficiency

We find that all results of 10024 testing trees are correct through comparing SYM computed from our algorithm with that from the MAST algorithm. Table 4 shows the results of testing 24 Cayley trees. The smallest symmetry is 6 for the Cayley tree with $z = 3$ and $s = 1$, and the largest symmetry is 1.164×10^{231} for the Cayley tree with $z = 3$ and $s = 9$.

Specially, we find that every test for these 10024 trees needs only one step of the continuous-time quantum walk to divide the vertices into brethren sets, i.e., only two iterations run in the first for loop of Algorithm 1. In the first iteration with $t = 0.1$, all sets have been found. In the second iteration with $t = 0.2$, our algorithm only confirms that there is no new set partitioned indeed. The iter column in Table 4 shows the corresponding results from testing Cayley trees. That is to say, the continuous-time quantum walk finds the symmetry of a tree with an extremely high efficiency.

Table 4. Experiment results of Cayley trees, where z is the coordination number; s is the outermost shell number; N is the total number of vertices; SYM is the symmetry; iter is the number of iterations; and time is the clock time in units of s.

z	s	N	SYM	iter	time	z	s	N	SYM	iter	time
3	1	4	6.000×10^{0}	2	1.504×10^{-3}	4	4	161	6.984×10^{41}	2	3.329×10^{-2}
3	2	10	4.800×10^{1}	2	7.327×10^{-4}	4	5	485	7.663×10^{125}	2	5.611×10^{-1}
3	3	22	3.072×10^{3}	2	1.589×10^{-3}	5	1	6	1.200×10^{2}	2	7.581×10^{-4}
3	4	46	1.258×10^{7}	2	4.237×10^{-3}	5	2	26	9.555×10^{8}	2	1.918×10^{-3}
3	5	94	2.111×10^{14}	2	1.211×10^{-2}	5	3	106	3.841×10^{36}	2	1.499×10^{-2}
3	6	190	5.942×10^{28}	2	4.992×10^{-2}	5	4	426	1.003×10^{147}	2	3.960×10^{-1}
3	7	382	4.708×10^{57}	2	2.748×10^{-1}	6	1	7	7.200×10^{2}	2	7.791×10^{-4}
3	8	766	2.955×10^{115}	2	2.051×10^{0}	6	2	37	2.150×10^{15}	2	2.965×10^{-3}
3	9	1534	1.164×10^{231}	2	1.440×10^{1}	6	3	187	5.103×10^{77}	2	5.059×10^{-2}
4	1	5	2.400×10^{1}	2	7.333×10^{-4}	7	1	8	5.040×10^{3}	2	8.306×10^{-4}
4	2	17	3.110×10^{4}	2	1.237×10^{-3}	7	2	50	5.055×10^{23}	2	4.488×10^{-3}
4	3	53	6.771×10^{13}	2	5.031×10^{-3}	7	3	302	5.149×10^{143}	2	1.713×10^{-1}

Figure 4 shows the clock time used to run our algorithm when testing 5000 random trees and 5000 scale-free trees. A symbol in Fig. 4 denotes the average time of our algorithm to test trees with an identical number of vertices. Because the scale-free trees have larger symmetries than the random trees,[16] the size of brethren sets in a scale-free tree is larger than that in a random tree. So, the time used to compute the factorial at line 17 of Algorithm 1 for a scale-free tree is larger. Thus, the time used for the scale-free trees is larger than that for the random trees in Fig. 4.

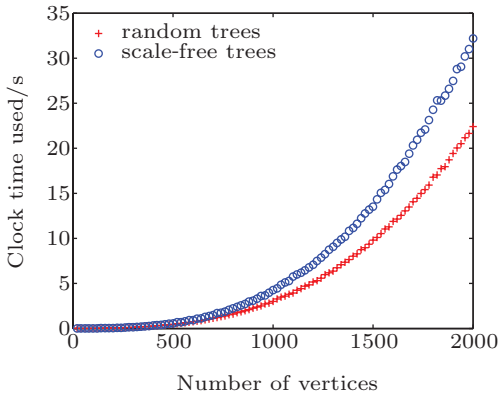


Fig. 4. (color online) Time used to run our algorithm for random trees and scale-free trees.

6. Conclusion

Quantum walk is widely studied to develop various quantum algorithms. This paper has studied the relationship between the continuous-time quantum walk and the symmetry of a graph. We have proved that they are closely related with each other. An algorithm based on the continuous-time quantum walk has been proposed for computing the symmetry of a tree. The complexity is reduced compared to the current best algorithm. Through testing 10024 trees, we have found that the continuous-time quantum walk helps in finding the symmetry of a tree with an extremely high efficiency.

Notice that the proposed algorithm in this paper is still a classical algorithm, which can be implemented on a classical computer, the term complexity occurred in this paper always represents the complexity of a classical algorithm on a classical computer, not on a quantum one. In Ref. [34], Douglas *et al*. disscussed how to implement this type of algorithms on a quantum computer, but this is not the focus of this paper. Thus, we did not discuss it. Our algorithm is only for computing the symmetry of a tree, but not for other types of graphs. That will be our future work. The relationship between the discrete-time quantum walk and the graph symmetry will also be studied.

References

[1] Kempe J 2003 *Contemporary Physics* **44** 307
[2] Shenvi N, Kempe J and Whaley K B 2003 *Phys. Rev. A* **67** 052307
[3] Childs A M and Goldstone J 2004 *Phys. Rev. A* **70** 022314
[4] Tulsi A 2008 *Phys. Rev. A* **78** 012310
[5] Somma R D, Boixo S, Barnum H and Knill E 2008 *Phys. Rev. Lett.* **101** 130504
[6] Childs A M 2009 *Phys. Rev. Lett.* **102** 180501
[7] Lovett N B, Cooper S, Everitt M, Trevers M and Kendon V 2010 *Phys. Rev. A* **81** 042330
[8] Berry S D and Wang J B 2011 *Phys. Rev. A* **83** 042317
[9] Gamble J K, Friesen M, Zhou D, Joynt R and Coppersmith S N 2010 *Phys. Rev. A* **81** 052313
[10] Qiang X, Yang X, Wu J and Zhu X 2012 *J. Phys. A: Math. Theor.* **45** 045305
[11] Forsyth D A and Ponce J 2002 *Computer Vision: A Modern Approach* (Upper Saddle River: Prentice Hall Professional Technical Reference)
[12] Kandel A, Bunke H and Last M 2010 *Applied Graph Theory in Computer Vision and Pattern Recognition* (1st edn.) (Berlin: Springer)
[13] Brügger A, Bunke H, Dickinson P and Riesen K 2008 (Perner P, ed.) *Lecture Notes in Computer Science* Vol. 5077 (Heidelberg: Springer) pp. 298–312
[14] Artymiuk P J, Spriggs R V and Willett P 2005 *Journal of the American Society for Information Science and Technology* **56** 518
[15] Fortin S 1996 *Technical Report 96-20, Department of Computing Science, University of Alberta*
[16] Zhang B D, Wu J J, Tang Y H and Zhou J 2012 *Physica Scripta* **86** 025006
[17] Zhang J Y, Sun W G and Chen G R 2012 *Chin. Phys. B* **21** 038901
[18] Lin F and Bao J D 2011 *Chin. Phys. B* **20** 040502
[19] Li K P 2010 *Chin. Phys. B* **19** 030519
[20] Hu K and Tang Y 2006 *Chin. Phys.* **15** 2782
[21] Yang X, Liao X, Xu W, Song J, Hu Q, Su J, Xiao L, Lu K, Dou Q, Jiang J and Yang C 2010 *Frontiers of Computer Science in China* **4** 445

[22] Yang X, Liao X, Lu K, Hu Q, Song J and Su J 2011 *Journal of Computer Science and Technology* **26** 344

[23] Broome M A, Fedrizzi A, Lanyon B P, Kassal I, Aspuru-Guzik A and White A G 2010 *Phys. Rev. Lett.* **104** 153602

[24] Zähringer F, Kirchmair G, Gerritsma R, Solano E, Blatt R and Roos C F 2010 *Phys. Rev. Lett.* **104** 100503

[25] Schreiber A, Cassemiro K N, Potoček V, Gábris A, Mosley P J, Andersson E, Jex I and Silberhorn C 2010 *Phys. Rev. Lett.* **104** 050502

[26] Douglas B L and Wang J B 2009 *Phys. Rev. A* **79** 052335

[27] Xu Y Y, Zhou F, Chen L, Xie Y, Xue P and Feng M 2012 *Chin. Phys. B* **21** 040304

[28] Peruzzo A, Lobino M, Matthews J C F, Matsuda N, Politi A, Poulios K, Zhou X, Lahini Y, Ismail N, Wörhoff K, Bromberg Y, Silberberg Y, Thompson M G and OBrien J L 2010 *Science* **329** 1500

[29] Karski M, Förster L, Choi J M, Steffen A, Alt W, Meschede D and Widera A 2009 *Science* **325** 174

[30] Tsomokos D I 2011 *Phys. Rev. A* **83** 052315

[31] Mülken O and Blumen A 2011 *Physics Reports* **502** 37

[32] Cameron P J 2004 *Automorphisms of Graphs, Topics in Algebraic Graph Theory* (Cambridge: Cambridge University Press)

[33] De Fraysseix H 1999 (J Kratochvíyl, ed.) *Lecture Notes in Computer Science* Vol. 1731 (Berlin: Springer) pp. 276–285

[34] Douglas B L and Wang J B 2008 *J. Phys. A: Math. Theor.* **41** 075303