Quantum Information and Computation, Vol. 21, No. 1&2 (2021) 0059-0079

ALGORITHMS FOR FINDING THE MAXIMUM CLIQUE BASED ON CONTINUOUS TIME QUANTUM WALKS

XI LI
School of Cyber Science and Engineering, Southeast University,Nanjing 211189, China.
230169107@seu.edu.cn

MINGYOU WU
School of Cyber Science and Engineering, Southeast University,Nanjing 211189, China.

HANWU CHEN
School of Computer Science and Engineering, Southeast University, Nanjing, 211189, China.
Key Laboratory of Computer Network and Information Integration (Southeast University)
Ministry of Education, Nanjing, 211189, China
hw_chen@seu.edu.cn

ZHIHAO LIU
School of Computer Science and Engineering, Southeast University, Nanjing, 211189, China.
Key Laboratory of Computer Network and Information Integration (Southeast University)
Ministry of Education, Nanjing, 211189, China
lzh@seu.edu.cn

Received July 26, 2020
Revised December 18, 2020

In this work, the application of continuous time quantum walks (CTQW) to the Maximum Clique (MC) problem was studied. Performing CTQW on graphs can generate distinct periodic probability amplitudes for different vertices. We found that the intensities of the probability amplitudes at some frequencies imply the clique structure of special kinds of graphs. Recursive algorithms with time complexity $O(N^6)$ in classical computers were proposed to determine the maximum clique. We have experimented on random graphs where each edge exists with different probabilities. Although counter examples were not found for random graphs, whether these algorithms are universal is beyond the scope of this work.

Keywords: The Maximum Clique Problem, Quantum Walks, Polynomial Algorithms
Communicated by: R Cleve & J Eisert

1 Introduction

The problem in finding the maximum clique (complete subgraph) is a Non-deterministic Polynomial Complete Problem (NP-complete problem)[1]. The optimal complexity of exact algorithms is $O(2^{0.249N})$[2]. Designing a polynomial algorithm for NP-complete problems on classical computers is normally difficult. Heuristic algorithms are significant considerations for graphs with large cardinality[3, 4]. In the literature, many approximation algorithms for the maximum clique problem are called sequential greedy heuristics. These heuristics generate a maximal clique through the repeated addition of a vertex into a partial clique, or the repeated deletion of a vertex from a set that is not a clique. As a sequential greedy

heuristic algorithm will terminate when a maximal clique is found, the probability of finding the maximum clique is relatively low. To improve the quality of the solution, the local search heuristics will continue to search the neighbor of the maximal clique. A well-known class of local search heuristics is k-interchange heuristics[5], which are based on the k-neighbor of a feasible solution. A common problem of local search heuristics is that only a locally optimal solution can be found. Some advanced search heuristics have been proposed to avoid that problem, such as genetic algorithms[6], tabu search[7, 8, 9], simulated annealing[10], and algorithms based on neural networks[11]. Advanced algorithms are usually applied in combination with other heuristic algorithms.

As the performance of quantum algorithms has been proven to be better than the classical algorithm in many situations[12, 13, 14], many scientists have turned to quantum algorithms for NP-complete problems[15, 16, 17, 18, 19]. With the advantage of quantum states, all possible solutions (combinations of vertices of a given graph) are encoded in an initial superposition state, and the optimal solution can be searched by a quantum evolution process in the previous quantum algorithms. The quantum algorithm asymptotically requires the square root of the number of operations that the classical algorithm requires[12, 15]. Apparently, graph structures are not adequately considered in these algorithms. The work of Noga Alon, Michael Kriv-elevich, and Benny Sudakov shows that the second eigenvector is related to the MC of random graphs[20]. Generally, connecting the structure of graphs with the NP-complete problem is unclear. In this work, the structure of the MC in the graph specifically refers to whether a vertex belongs to the MC. We mainly focus on the structure of center graphs because all kinds of graphs can be transformed to center graphs. A graph is called a center graph if there exists one so-called center vertex adjacent to all other vertices. We will describe how clique structure impacts the continuous time quantum walks (CTQW) of several special kinds of center graphs, and algorithms for the maximum clique problem will be proposed. In Farhi and Goldstones work, CTQW is defined as an evolution of a quantum system driven by the Laplacian matrix of a given graph[21]. With other physical models [22, 23], the Hamiltonian of the CTQW is defined as the adjacency matrix to the corresponding graph in this work. Then the state of CTQW is determined by $|\varphi(t)\rangle = e^{iAt}|\varphi(0)\rangle$, where A is the adjacency matrix of the given graph G and e^{iAt} is an evolution unitary operator. This operation exists in series form$e^{itA} = \sum_{s=0}^{\infty} \frac{(it)^s A^s}{s!}$. As A^s is characterized by the number of walks in the graphs, the CTQW does reflect the clique structures of several kinds of center graphs. The evolution can be estimated when eigenvectors and eigenvalues of a given adjacency matrix are obtained by numerical computation in $O(N^3)$ time on classical computers. The probability amplitude of CTQW is chosen as the critical feature to infer whether a vertex is a member of the maximum clique.

In this paper, the second section presents CTQW on center graphs. In the third section, an algorithm, named Algorithm A, with $O(N^6)$ time complexity based on CTQW for finding the maximum clique is introduced. In the fourth section, the probable error of Algorithm A is presented and a strategy for constructing a graph invalid for Algorithm A is described. In the fifth section, we give an algorithms, named Algorithm B, to fix problems with Algorithm A. The conclusions are the last section.

2 Clique structure and CTQW on the center graph

Generally, a graph is denoted as $G(V, E)$, consisting of a vertex set V and an edge set E. The set E is a subset of $V \times V$, which implies the connection relationship between any pair of vertices in V. Let the carnality of V equal to N, where the adjacency matrix of G is an $N \times N$ real symmetric matrix A, where $A_{jl} = 1$ if vertices v_j and v_l are connected, otherwise $A_{jl} = 0$.

Consider the CTQW on a given graph. One can associate every vertex v_j of the graph with a basis vector $|j\rangle$ in an N-dimensional vector space. The Hamiltonian of the system is

$$H = A, \tag{1}$$

If v_j is the initial state of the system, and the transition probability amplitude from v_j to v_l is $\alpha_{l,j}(t)$ or short format $\alpha_{l,j}$, then:

$$\alpha_{l,j}(t) = \langle l| e^{i\gamma At} |j\rangle, \tag{2}$$

The probability $\pi_{l,j}(t)$ or short format $\pi_{l,j}$ can be written as

$$\pi_{l,j}(t) = \left|\langle l|e^{i\gamma At}|j\rangle\right|^2. \tag{3}$$

The eigenvalues of A are denoted as $\lambda_n (n = 1, 2, \ldots, N)$, the eigenvalues are arranged in descending order, namely $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$. As A is a real symmetric matrix, the eigenvalues of A are all real. The unit-norm eigenvector corresponding to λ_n is denoted as $|\lambda_n\rangle$, producing:

$$\alpha_{l,j}(t) = \sum_n e^{i\lambda_n t} \langle l \mid \lambda_n\rangle \langle \lambda_n \mid j\rangle, \tag{4}$$

and

$$\pi_{l,j}(t) = \left|\sum_n e^{i\lambda_n t} \langle l \mid \lambda_n\rangle \langle \lambda_n \mid j\rangle\right|^2. \tag{5}$$

The real part of the amplitude $\alpha_{l,j}$ can be represented as:

$$R(\alpha_{l,j}) = \sum_n p_n \cos(\lambda_n t) \tag{6}$$

where $p_n = \langle l \mid \lambda_n\rangle \langle \lambda_n \mid j\rangle$. This implies that the real part of the amplitude of CTQW is a periodic function with N frequency components, and the frequency values are the eigenvalues of the adjacency matrix and the intensity of the frequency λ_n is p_n.

The amplitude can also be represented as a form of sums, i.e.,

$$\alpha_{l,j}(t) = \sum_{\zeta=0}^{\infty} \frac{(it)^\zeta \left(A^\zeta\right)_{l,j}}{\zeta!}, \tag{7}$$

where $(A^\zeta)_{l,j}$ equals the number of walks of length ζ. Therefore, the CTQW can be determined by the number of walks in the graphs. As $(A^\zeta)_{l,j}$ can be denoted by the eigenvalues and eigenvectors, then Eq.7 can be converted to Eq.4 vice versa.

Consider a graph G and one of its vertices v_j and let $\mathbf{N}(j)$ denote the neighbors of v_j. The subgraph induced by vertex set $v_j \bigcup \mathbf{N}(j)$ is denoted as G_j. We call G_j the center subgraph of vertex v_j and the vertex v_j the center vertex of G_j. Note that the concept of the center graph is not completely the same as the concept of reference[25]. Two natural approaches can be used to transform a graph into a center graph or set of center graphs. The first way is to add a new vertex that connects to every vertex of the original graph. The second approach is to induce a set of center graphs $\{G_1, \ldots, G_N\}$ of the original graph G.

In this work, a center graph G_j is called the first kind of ideal center graph if there are two cliques in G_j and there is no edge connecting any pair of vertices $\{v_k, v_l\}$ when v_k and v_l are members of distinct cliques. An example of a center graph is shown in Fig.1.

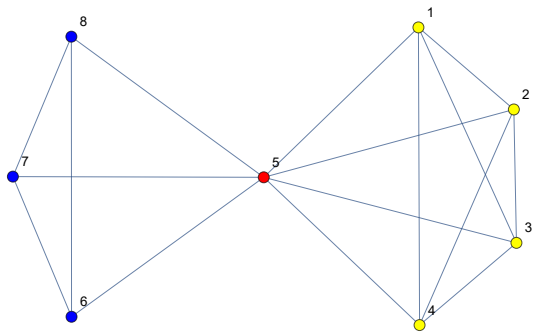

Fig. 1. The center graph G_5 of vertex 5. There exists two cliques, one is the subgraph induced by $\{1, 2, 3, 4, 5\}$, another is the subgraph induced by $\{5, 6, 7, 8\}$

It is difficult to obtain the analytical solutions of eigenvalues and eigenvectors even for the first kind of ideal center graph. A method of counting the number of walks is used to solve the CTQW. Let W_ζ denote the number of closed walks of length ζ starting from center vertex, denote the number of walks of length ζ starting from the center vertex j ending with one of the vertices in the MC, and H_ζ denote the number of walks of length ζ starting from the center vertex j ending with one of vertices not in the MC. It provides that:

$$\begin{cases} W_{\zeta+1} = (m_1 - 1) F_\zeta + (m_2 - 1) H_\zeta \\ F_{\zeta+1} = W_\zeta + (m_1 - 2) F_\zeta \\ H_{\zeta+1} = W_\zeta + (m_2 - 2) H_\zeta \end{cases} \tag{8}$$

where m_1 is the clique number and m_2 is the size of the remaining clique. Finding the solutions of Eq.(8) is equivalent to eigen decomposition of adjacency matrix A. However, the exact numerical solutions are complex and do not help to infer whether a vertex belongs to the maximum clique. We only need the relationships between the probability amplitudes of different vertices. In utilizing literature results[24, 26], we have

$$W_\zeta = \sum_{n=1}^{N} a_n \lambda_n^\zeta, \tag{9}$$

where $\sum_{n=1}^{N} a_n = 1$. Taking Eq.(9) into the second and third terms of Eq.(8), we have

$$F_{\zeta+1} = \sum_{n=1}^{N} a_n \lambda_n^{\zeta} + (m_1 - 2) F_{\zeta}, \tag{10}$$

and

$$H_{\zeta+1} = \sum_{n=1}^{N} a_n \lambda_n^{\zeta} + (m_1 - 2) H_{\zeta}. \tag{11}$$

Clearly, $m_1 - 2$ and $m_2 - 2$ are not eigenvalues of the first ideal center graph. Solving Eq.(10) and Eq.(11), we have

$$F_{\zeta} = \sum_{n=1}^{N} a_n \frac{(m_1 - 2)^{\zeta} - \lambda_n^{\zeta}}{m_1 - 2 - \lambda_n} \tag{12}$$

and

$$H_{\zeta} = \sum_{n=1}^{N} a_n \frac{(m_2 - 2)^{\zeta} - \lambda_n^{\zeta}}{m_2 - 2 - \lambda_n} \tag{13}$$

Hence the probability amplitude of v_l which is a vertex of MC is

$$\alpha_{l,j}(t) = \sum_{n=1}^{N} \frac{a_n \left(e^{i(m_1-2)t} - \mathrm{e}^{\mathrm{i}\lambda_n t}\right)}{m_1 - 2 - \lambda_n}, \tag{14}$$

and when v_k is not a member of the maximum clique, the probability amplitude is

$$\alpha_{l,j} = \sum_{n=1}^{N} \frac{a_n \left(e^{i(m_2-2)t} - \mathrm{e}^{\mathrm{i}\lambda_n t}\right)}{m_2 - 2 - \lambda_n}. \tag{15}$$

As $m_j - 2$ is not an eigenvalue for $j = 1, 2$, compare Eq.(4) with Eq.(15) to obtain

$$\sum_{n=1}^{N} \frac{a_n e^{i(m_j-2)t}}{m_j - 2 - \lambda_n} = 0. \tag{16}$$

Let $p_{l,n}$ denote the coefficient of $\alpha_{l,j}$ at eigenvalue λ_n. Then we have the follow theorem.

Theorem 1 *For the first kind of ideal center graph G_j, $v_j, v_l, v_k \in V(G_j)$, v_j is the center vertex, v_l is a member of the maximum clique of G_j, and v_k is not a member of the maximum clique. Then*

$$p_{l,1} > p_{k,1},$$

i.e.,

$$\left|\frac{1}{m_1 - 2 - \lambda_1}\right| > \left|\frac{1}{m_2 - 2 - \lambda_1}\right|$$

.

Theorem.1 is obvious since $\lambda_1 > m_1 - 2 > m_2 - 2$. Therefore, it is easy to determine the vertex that belongs to the maximum clique from the first kind of ideal center graph by using CTQW.

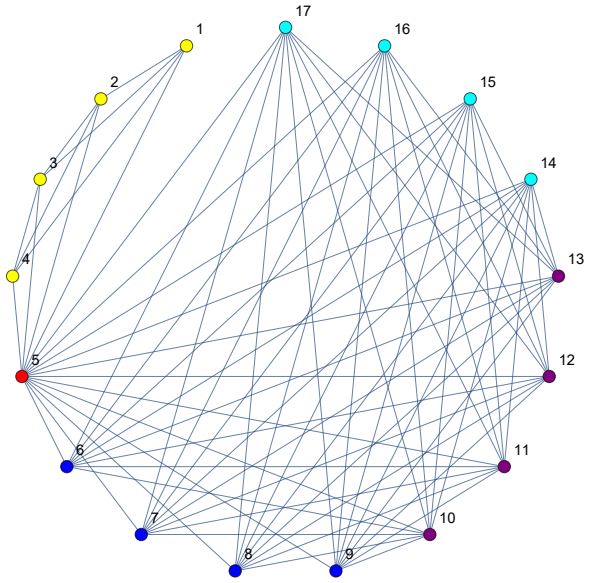

Fig. 2. The second kind of ideal center graph. There has no edge that connects the vertices in the MC and the vertices not in the MC. The subgraph induced by vertices {6,7,...,17} is a complete multi-partite graph and vertices in the same independent set share the same color.

A center graph G_j is called the second kind of ideal center graph if it contains a maximum clique C and a complete multi-partite subgraph D with no edges connecting the subgraphs C and D. An example graph of this type is shown in Fig.2. For the example graph, the size of maximum clique of is five. Since the subgraph induced by vertices {6,7,...,17} is a complete multi-partite graph, adding edges will generate cliques with size not less than five.

For the second kind of ideal center graph, the number of walks can be determined by the following recursion equations:

$$\begin{cases} W_{\zeta+1} = (m_1 - 1) F_\zeta + z (m_2 - 1) H_\zeta \\ F_{\zeta+1} = W_\zeta + (m_1 - 2) F_\zeta \\ H_{\zeta+1} = W_\zeta + z (m_2 - 2) H_\zeta \end{cases} \tag{17}$$

where the variables W_ζ, F_ζ and H_ζ are defined analogously to the variables in equation (**??**). z is the number of vertices in each independent set. Comparing Eqs.(17) to Eqs.(8), one can find that the solution formats of Eqs.(17) are similar to the solutions of Eqs.(8), with just the replacement of $(m_2 - 2)$ by $z (m_2 - 2)$ in Eq.(15). Therefore, for a vertex v_l, which belongs to the maximum clique in the graph shown in Fig.2, it is provided that:

$$\alpha_{l,j}(t) = \sum_{n=1}^{N} \frac{a_n \left(e^{i(m_1-2)t} - \mathrm{e}^{i\lambda_n t} \right)}{m_1 - 2 - \lambda_n}, \tag{18}$$

and for a vertex which does not belong to the maximum clique, it provides:

$$\alpha_{l,j}(t) = \sum_{n=1}^{N} \frac{a_n \left(e^{i(z(m_2-2))t} - \mathrm{e}^{i\lambda_n t} \right)}{z (m_2 - 2) - \lambda_n}, \tag{19}$$

Similar to **Theorem.**1, the following theorem can be provided:

Theorem 2 *For the second kind of ideal center graph G_j, $v_j, v_l, v_k \in V(G_j)$, v_j is the center vertex, v_l is a member of the maximum clique of G_j, and v_k is not a member of the maximum clique. Then*

$$p_{l,1} < p_{k,1},$$

i.e.,

$$\left|\frac{1}{m_1 - 2 - \lambda_1}\right| < \left|\frac{1}{z\,(m_2 - 2) - \lambda_1}\right|,$$

if

$$m_1 - 2 < z\,(m_2 - 2)\,.$$

In the previous instances, there were no edges connecting vertices in different cliques. For general cases, multiple edges exist between vertices from distinct cliques. One of such instances is exhibited in Fig.3:

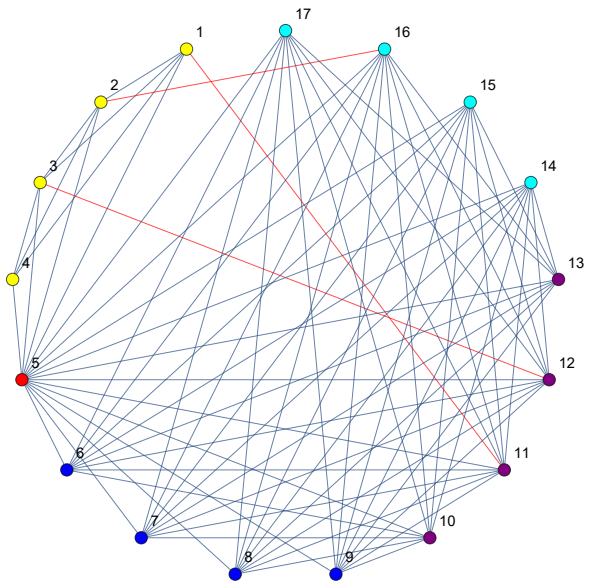

Fig. 3. An example graph. In this configuration, the maximum clique and the non-maximum clique connect by edges (1,11),(2,16),(3,12)

For general graphs, the method of counting the number of walks to determine the CTQW is as difficult as the eigen-decomposition of the adjacency matrix, and even when the eigen-decomposition is obtained, to directly deduce whether a vertex is a member of the maximum clique is still unknown. An intuitive idea is to generate a center subgraph of the original graph and find the maximum clique in that center subgraph. For instance, the maximum clique attached to vertex 3 of the graph shown in Fig.3 can be easily determined. The center graph of vertex 3 is exhibited in Fig.4. Note that the original center vertex 5 is not included in G_3.

The resultant graph in Fig.4 is the first kind of ideal center graph, and its maximum clique can be determined by **Theorem.**1. This shows that performing CTQW on center graphs in

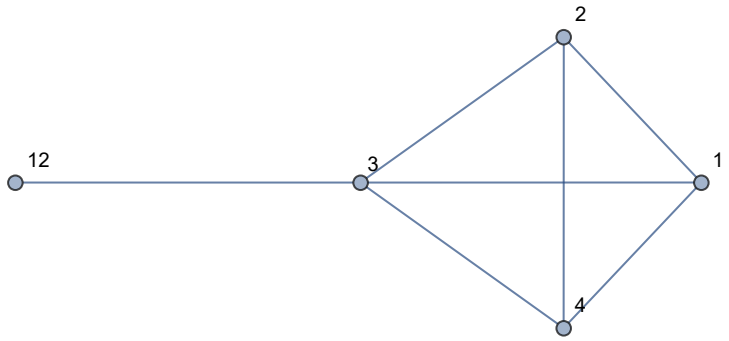

Fig. 4. The center subgraph of vertex 3. The graph is induced by vertex 3 and it′s neighbors except center vertex, namely {1,2,3,4,12}

an orderly way can help to understand the maximum clique. In this procedure, we repeatedly chose vertices and constructed associated center subgraphs. In the following section, $C(v_j, G)$ is used to denote the center subgraph of G whose center vertex is v_j or the procedure of constructing that center subgraph. The method regarding successive selection of a vertex to construct a center graph is presented in the next section.

3 A recursive algorithm for finding the maximum clique by CTQW

An algorithm, named Algorithm A, based on eigenvectors of the adjacency matrix of the graph for finding the maximum clique is proposed in this section. The intensities of the real part (or imaginary part) of the probability amplitude are used as the critical feature in selecting probable vertices. Algorithm A is described in the following pseudocode.

Algorithm A for finding the maximum clique

Require: Center graph G_s with its center vertex v_s
Ensure: The maximum clique $Clique(G_s)$

function ALGORITHM_A(G_s)
  Do eigen-decompositon on G_s, find v_{max} and v_{min}.
  $Clique(G_s) \leftarrow \{\}$
  $result1 \leftarrow Pick\_max(C(v_{max}, G_s))$
  $result2 \leftarrow \{\}$
  while $v_i \in V(C(v_{min}, G_s))$ **do**
    $Temp \leftarrow Pick\_max(v_i, C(v_{min}, G_s))$
    $result2 \leftarrow Max(result2, Temp)$
    $i \leftarrow i + 1$
  end while
  $result3 \leftarrow$ ALGORITHM_A($G_s - v_{min}$)
  $Clique(G_s) \leftarrow Max(result1, result2, result3)$
end function

In the pseudocode of Algorithm A, v_{min} denotes the vertex with the smallest intensity at the largest frequency λ_1, v_{max} denotes the vertex with the maximal intensity at the largest frequency, Max is a function which is used to find the biggest set. $Pick\_max$ is a subfunction

of Algorithm A and its step is shown in Table.3

The step of Pick_max

Require: Center graph G_s, center vertex v_s
Ensure: The maximum clique of G_s, denoted as $Clique(G)$;

1: Do eigen-decomposition on adjacent matrix of G_s, The largest eigenvalue is λ_1, corresponding eigenvector is $\mathbf{x_1}$ and x_1^j denotes the j-th component of $\mathbf{x_1}$;
2: Calculate the intensity of amplitude of every vertex v_k at the maximum frequency λ_1, denoted as p_k, $p_k = x_1^s x_1^k$;
3: Add vertex v_{max} to $Clique(G)$; Delete v_s and construct center graph $C(v_{max})$, if size of $C(v_{max})$ is not 1, then $v_s \longleftarrow v_{max}$ and turn to 1.

Algorithm A is inspired by **Theorem.**1, i.e., the frequency component of the largest eigenvalue λ_1 is used as the feature in choosing probable vertices belonging to the maximum clique. If the intensity p_k of vertex v_k is the largest among all other vertices, then it would be chosen as a probable member of the maximum clique. A tree-like diagram can be used to better understand the structure of the Algorithm A more clearly. In the tree-like diagram, the root denotes the algorithm, and the leaf node denotes the sub-modules of the algorithm.

As shown in Fig.5, there are three sub-modules in Algorithm A, and the last one is the recursive process. The first one corresponds to line 4 of the pseudocode in which the strategy of choosing the vertex with the largest intensity at frequency λ_1 in amplitude of CTQW is employed. Then, we wanted to delete the vertex with the weakest intensity at frequency λ_1, and this vertex is denoted as v_{min}. Since that vertex may be a member of the maximum clique, the second sub-module corresponding to line 6 through line 10 of the pseudocode of Algorithm A, is applied to find the maximum clique of v_{min}. The last sub-module is to recursively call the Algorithm A. Note that the size of the graph decreases by one since a vertex is deleted in the third module. The maximum clique of $Clique(G_s)$ is the largest clique found in these tree sub-modules.

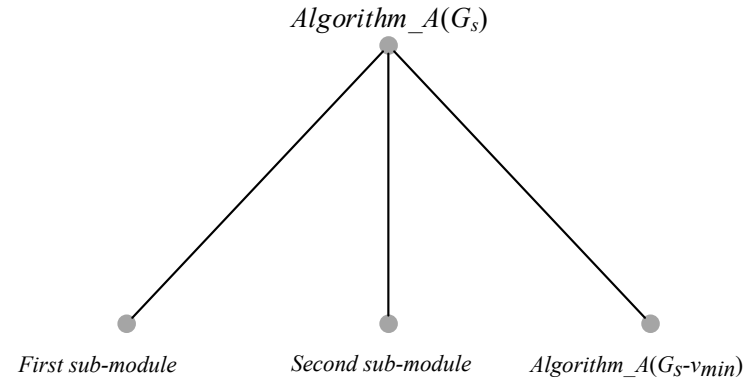

Fig. 5. The recursive algorithm for finding the maximum clique. There are three sub-modules in Algorithm A, and the third sub-module recursively calls the Algorithm A on a smaller subgraph.

Algorithm A has polynomial complexity. To observe this, we must solve the recursion of Algorithm A. Let $T(n)$ denote the complexity of Algorithm A, $Y_1(n)$ and $Y_2(n)$ denote

the complexity of the corresponding submodules in the tree-like diagram. Then the following recursion is satisfied.

$$T(n) = Y_1(n) + Y_2(n) + T(n-1) \tag{20}$$

As the complexity of the eigen-decomposition is $O(n^3)$, the $Pick\_max$ and $Y_1(n)$ are at most $O(n^4)$. Analogously, $Y_2(n)$ is at most $O(n^5)$ as the size of center subgraph is much less than n. Therefore, the worst complexity can be reduced to:

$$T(n) = O\left(n^5\right) + T(n-1) \tag{21}$$

Then from Eq.21, it provides that

$$T(n) = O\left(n^6\right) \tag{22}$$

Therefore, Algorithm A has polynomial complexity, and more importantly, it provides a new perspective to solve the maximum clique problem and contributes to the classification of graphs as well which will be discussed in the following section.

We have experimented on random graphs with different edge connecting probabilities on classical computers. In our experiments, counter examples, where the algorithm finds a sub-maximum clique, have not yet been determined. However, counter examples can be elaborately constructed. We will illustrate an approach for designing such a graph in the next section.

4 Probable counter examples of Algorithm A

It seems well established that phase transitions exist in many NP-complete problems, and yet the connection between NP-completeness and phase transitions is not a simple one[27]. In this section, we will see that a similar phenomenon appears when the graph has some special sub-structures which cause Algorithm A to return a non-maximum clique. To be more specific, when some sub-structures exist, Algorithm A will fail, and we will present an approach for constructing probable counter examples of Algorithm A in this section.

Let $W_\zeta^{v_j}$ denote the number of walks of length ζ from the center vertex v_s to the vertex v_j. From the previous section,

$$W_\zeta^{v_j} = \sum_{k=1}^{N} a_k^{v_j} \lambda_k^\zeta. \tag{23}$$

Where, $a_k^{v_j} = \langle v_s|\ \lambda_k\rangle \langle\lambda_k|\ v_j\rangle$. Therefore, the amplitude of CTQW is

$$p_{v_j} = \sum_{k=1}^{N} a_k^{v_j} e^{i\lambda_k t}.$$

In Algorithm A, $a_1^{v_j}$ of different vertices at the largest eigenvalue λ_1 are compared. For a large enough ζ, $a_1^{v_j}$ and $a_1^{v_h}$, $a_1^{v_j} > a_1^{v_h}$ if and only if $W_\zeta^{v_j} > W_\zeta^{v_h}$. This implies that if Algorithm A is invalid for a graph G, then every member of the maximum clique has a neighbor v_h that has the largest number of walks $W_\zeta^{v_h}$. In this case, no matter which vertex belonging to the maximum clique is chosen, v_h will be chosen in some layer of the recursion and Algorithm A spontaneously fails. For simplicity, assume all such subgraphs adjacent to every member of the maximum clique are the same, and it is a complete multi-partite graph with a degree far

larger than the size of the MC. For clarity, we propose a different kind of graph named as a base graph. A schematic diagram of a base graph is shown in Fig.6.

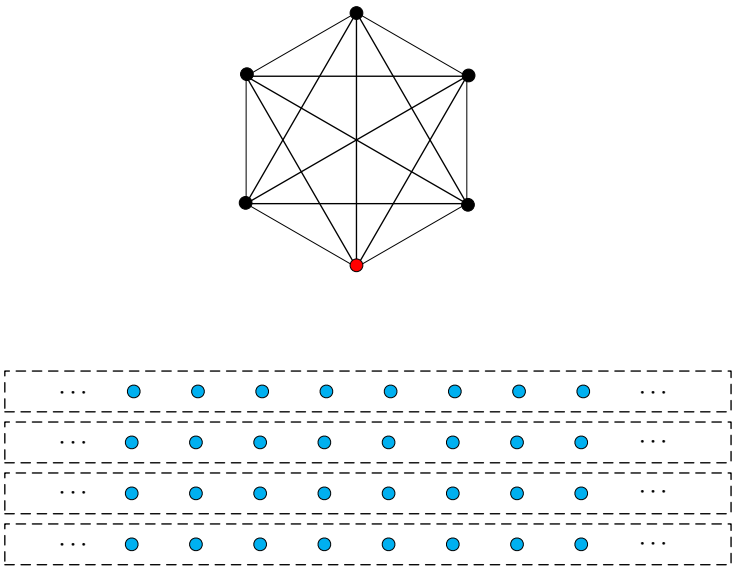

Fig. 6. Base graph. The graph contains a complete subgraph and a complete multi-partite subgraph. The number of independence set is not larger than the clique number, the number of vertices in the complete multi-partite subgraph is as great as possible.

The base graph consists of two elementary sub-graphs, one is the maximum clique at the upper of Fig.6, the other is a complete multi-partite graph containing all the light blue vertices in Fig.6. The vertex set in the same dashed box is not adjacent pairwise, i.e., it is an independent set. Also, vertices from different dashed boxes are fully adjacent. Edges can be added between the vertices of the maximum clique and the complete multi-partite graph. Firstly, a vertex, the red vertex in this instance, is chosen as the center vertex. Secondly, for an integer $P(P \geq 2)$, every combination with P vertices from the black vertices are connected to $q \cdot z \left(qz > \omega\left(G\right) - P, q + P + 1 < \omega\left(G\right)\right)$ common vertices from at least two partites of the complete multi-partite graph, where q is the number of partites, z is the number of vertices in each independent set, and $\omega\left(G\right)$ is the clique number. After the two procedures adding edges, every P vertices of the maximum clique have a common complete multipartite graph as their neighbor. For an simple instance $P = 2$, it means that every 2-vertex combination of the maximum clique has a common complete multi-partite graph as neighbor. The condition $qz > \omega\left(G\right) - P$ ensures that a non-maximum clique vertex will be chosen in the procedure of Algorithm A, therefore Algorithm A failed in this situation.

For illustrating the construction of a counterexample of Algorithm A, we will first give a feasible graph(Fig.7). As the amplitude of each gray vertex is bigger than the yellow vertex, the first sub-module of Algorithm A cannot find the exact maximum clique. And then, the second sub-module will check on a center subgraph with one yellow vertex as the center vertex. It is clear that the maximum clique can be accurately found by the second sub-module of Algorithm A. The center subgraph of one yellow vertex in the Fig.7 is shown in Fig.8.

In this instance, no pairs of members belonging to the maximum clique have common

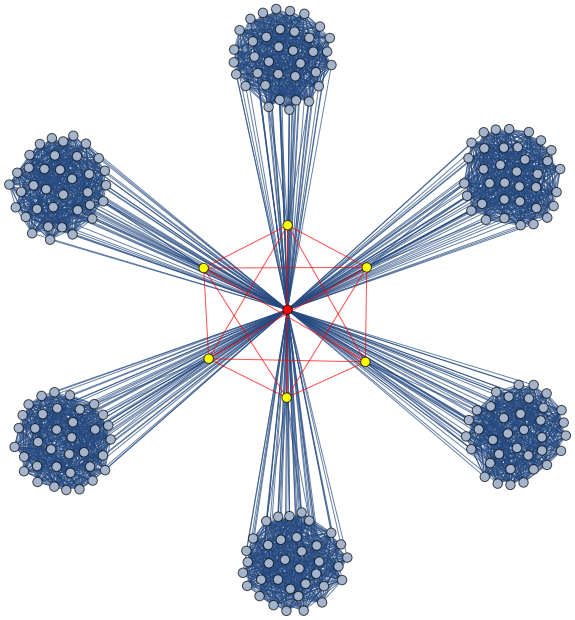

Fig. 7. A feasible graph. The red vertex is the center vertex, the yellow vertices are the members of the maximum clique, and each bulk of gray vertices is a complete multi-partite graph. Note that every yellow vertex is adjacent to a complete multi-partite subgraph but not all vertices of one bulk of the gray vertices

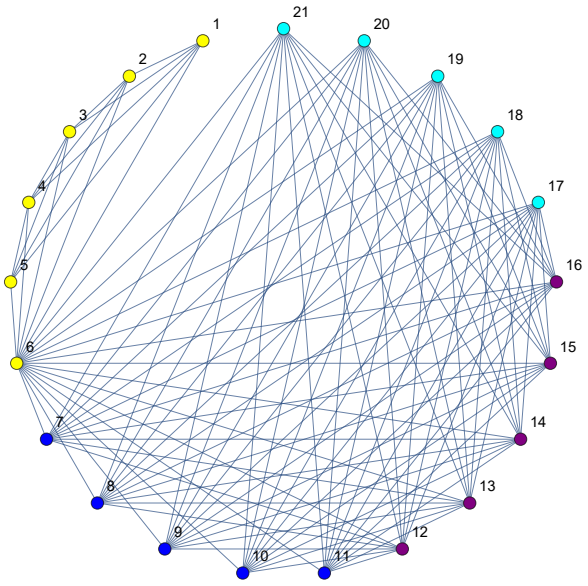

Fig. 8. One of the center subgraph of the feasible graph shown in 7. The yellow vertices are the members of the maximum clique, and one of them is the center vertex. The remaining vertices form a complete multi-partite graph where vertices in the same independet set are setted the same color.

neighbors. In the next instance shown in Fig.9, this kind of graph structure, namely common neighbors, will be added.

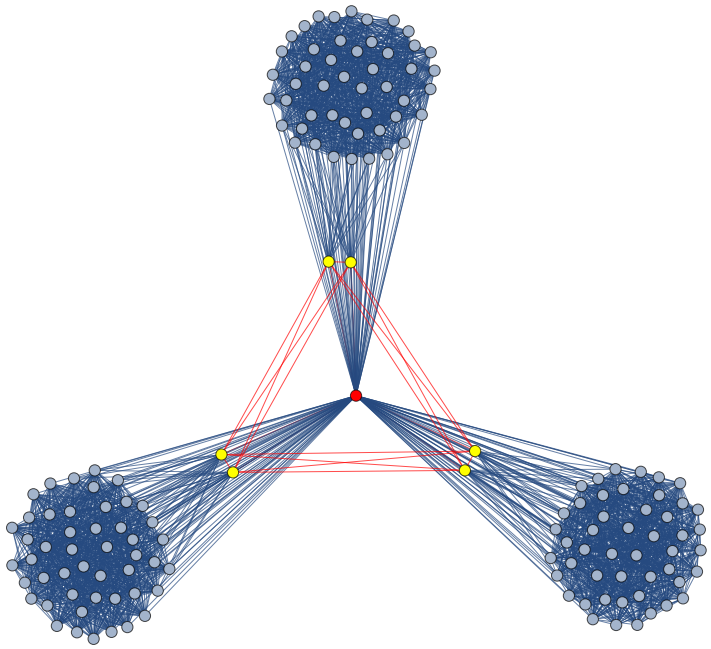

Fig. 9. Another feasible graph of Algorithm A. The red vertex is the center vertex, the six yellow vertices are the members of the maximum clique, the three groups of gray vertices form three complete multi-partite graphs. The yellow vertices are divided as three group, and every group of them are adjacent to a commom complete multi-partite subgraph.

As the number of edges attached to the yellow vertices is much less than the number attached to the gray vertices, then the first sub-module of the Algorithm A will find a clique in which all of its vertices are gray vertices in Fig.9. The second sub-module will check and find the exact maximum clique on the center subgraph with one yellow vertex as its center vertex, and this center subgraph is shown in Fig.10.

Two feasible graphs imply that every 2-vertex (or not less than 2) combination of the maximum clique vertices should have at least one common neighbor in counter examples. For convenience, we assume that all common neighbors are the same, it is a complete multi-partite graph, and denoted as G_c. As a lot of G_c must be drawn in figures, we use a square to denote it. By the method of adding edges in the base graph, a counter example is illustrated in Fig.11.

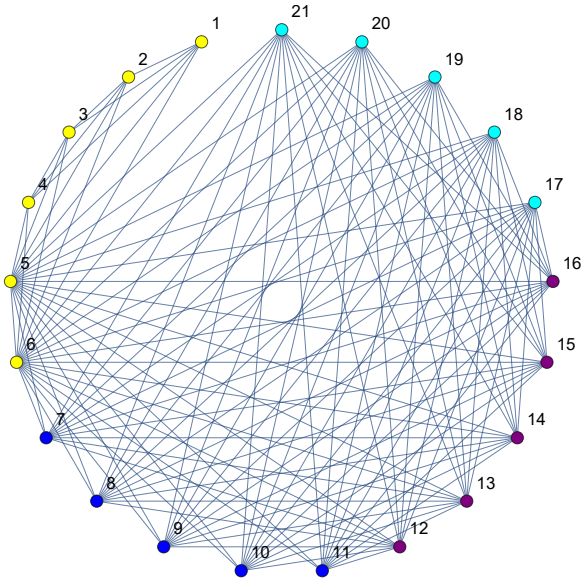

Fig. 10. The center subgraph of Fig.9. The yellow vertices are the members of the maximum clique, two of them are the both center vertices. The other vertices constitute a complete 3-partite subgraph, the vertices with the same color are in a common independent set.

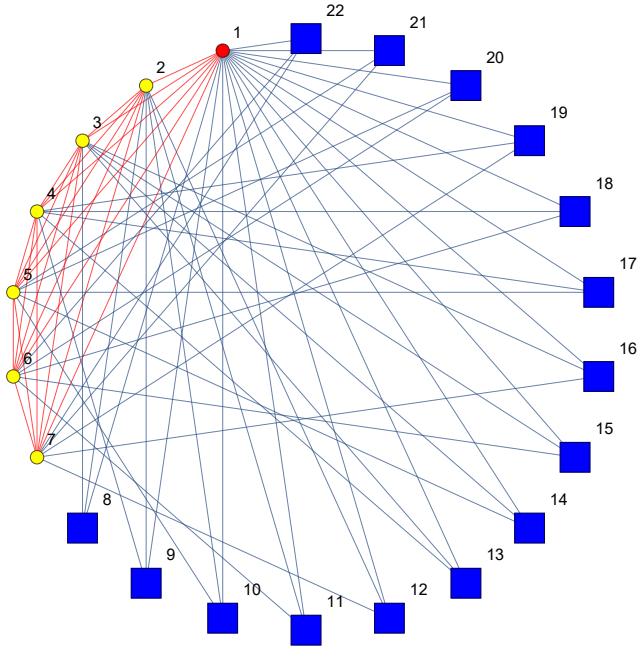

Fig. 11. A type of counter example of Algorithm A. The red vertex is the center vertex, the yellow vertices are the members of the maximum clique, the blue square is used to denote the G_c. The number of independent sets is confined by the size of the maximum clique. For instance, if the order of maximum clique is 6, then the number of independent sets in G_s is 2. If that number is not less than 3, then the cardinality of the maximum clique is not less than 6.

From the graph in Fig.11, one can find that no matter which vertex of the maximum clique is picked as the center vertex, the center subgraph has the format shown in Fig.12 and the maximum clique of this center subgraph cannot be found by the sub-modules of Algorithm A. Therefore, Fig.11 is a counter example of Algorithm A.

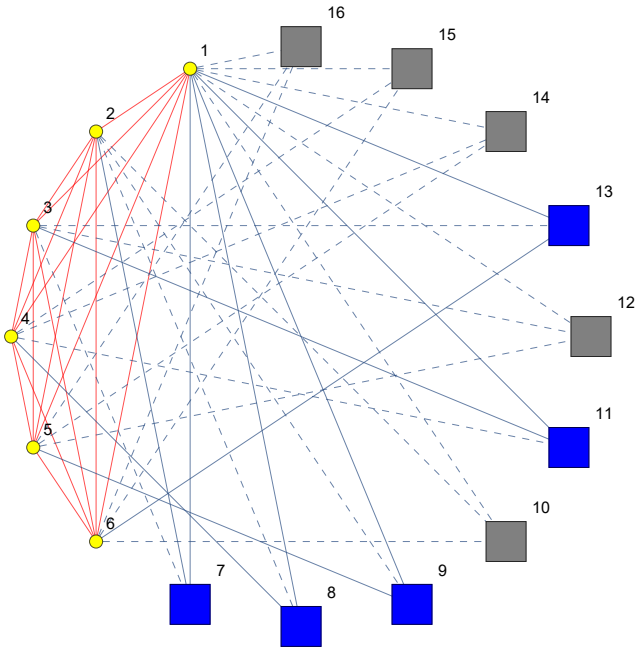

Fig. 12. A center subgraph of counter example shown in 11 of Algorithm A. The vertex 1 is the center vertex, the yellow vertices are the members of the maximum clique, the square is used to denote the G_c. The gray square means that G_c is optional, the blue means that square is necessary, and the dashed edge is optional

Although Algorithm A is not universal, it divides all graphs into two classes. The maximum clique of the first class of graphs can be accurately determined by Algorithm A but the second class of graphs cannot. Therefore, if an algorithm exists that can crack the second class of graphs, then the problem can be overcome. Algorithm B was designed to improve the performance of Algorithm A for the second class of graphs. We will further describe them in the next section.

5 Variational frequency selection algorithm for finding clique

In the Algorithm A, the frequency, namely the eigenvalue, for picking a probable vertex is settled as the largest eigenvalue of the center subgraph. In this section, we will show an approach where the frequency is not fixed.

Assume that v_s is a vertex with the smallest intensity at the largest frequency, and we want to find a maximum clique attached to v_s. In the center original graph, v_s has the smallest intensity at the largest frequency; however, v_s has the largest intensity at some frequency λ_j, and the other members of the maximum clique are more likely to occur at frequency λ_j. Then a vertex v_{ref} with the largest intensity at frequency λ_j except v_s is chosen, and v_{ref} is called the reference vertex. The CTQW on the center subgraph of v_s and the location of vertex

v_{ref} are used to determine the next reference vertex. The v_{ref} will become the new center vertex and a new reference vertex is used to find the next reference vertex. In this procedure, the frequency used to pick probable vertex is not fixed but dependent on the current center vertex, hence it is named as the variational frequency selection algorithm (VFSA). The VFSA is a sub-module of Algorithm B, which is used for a class of graphs not suitable for Algorithm A. The procedure for a VFSA is listed in the following table. To illustrate VFSA, we will

The steps of VFSA

Require: Center graph G_s, center vertex v_s, next center vertex v_{ref};
Ensure: The maximum clique of G_s, denoted as $Clique(G_s)$;

1: If G_s is a complete graph, then add the vertices of G_s to $Clique(G_s)$ and return $Clique(G_s)$, if not, add v_s to $Clique(G_s)$, turn to step.2
2: Do eigen-decomposition on adjacent matrix of G_s. Find a frequency, namely an eigenvalue f_{ref}, such that vertex v_{ref} can take the largest intensity among all frequencies.
3: Find vertex v'_{ref} which has the largest intensity at frequency f_{ref} in the neighbors of vertex v_{ref}
4: Delete v_s and let $v_s \longleftarrow v_{ref}$, $v_{ref} \longleftarrow v'_{ref}$, $G_s \longleftarrow C(v_{ref})$, turn to 1

present how VFSA works on a counter example of Algorithm A. The counter example is constructed by the method described in the previous section. For convenience, the counter graph is denoted as T. As the order of T is too large to exhibit here, we put it in Fig.A.1 of Appendix A. Via the eigen-decomposition on the T, the amplitude of each vertex is obtained. It is inconvenient to show all the amplitudes at every eigenvalue as the number of vertices in graph T is 206, so we will show amplitudes of some indispensable vertices and eigenvalues in the following table.

Table 1. The amplitude of CTQW of ten vertices. The vertex 201 is the center vertex, vertices $202, 203, 204, 205, 206$ are members of the maximum clique, and vertices $198, 199, 200$ are not.

v_i	198	199	200	201	202	203	204	205	206
$\lambda \approx -1.88$	0.024	0.024	0.024	0.912	0.039	0.039	0.039	0.039	0.039
$\lambda \approx 6.44$	0	0	0	0	0	0	0	0	0
$\lambda \approx 6.75$	-0.06	-0.06	-0.06	0.004	0.022	0.022	0.022	0.022	0.022
$\lambda \approx 103.04$	0.010	0.010	0.010	0.019	0.006	0.006	0.006	0.006	0.006

From the table, one can see that the amplitude of each vertex in the maximum clique at the principle eigenvalue is 0.006 which is the least among all vertices. Hence, a center subgraph with one of 202 to 206 as its center vertex will be checked. Without loss of generality, assuming the picked center vertex is vertex 202, then the program will find a reference frequency that vertex 202 has the largest amplitude, which is $\lambda \approx -1.88$ in the Table.1. There are 4 vertices, namely vertices $203, 204, 205, 206$, which have the largest amplitude at $\lambda \approx -1.88$ and the corresponding amplitude is 0.039. One of these vertices, say 203, will be picked as the next center vertex v_{ref} and the other vertices not belonging to the maximum clique are excluded. Then in the next step, the center subgraph of vertex 202 will be checked, and the next center vertex is 203. Note that the order of the new center subgraph changes to 69, vertex 202 to 206 have new labels 65 to 69, respectively. The center subgraph is shown in Fig.A.2 of Appendix A, and some amplitudes of the center subgraph are exhibited in table.2. In this turn of VFSA,

we will not check the largest eigenvalue, but the reference frequency $f_{ref} = 2.63$ where vertex 66(corresponding the original vertex 203) has the largest amplitude will be checked. At $f_{ref} = 2.63$, any member of the maximum clique takes the largest amplitude, therefore one of them will be picked. This procedure is repeated until there are no vertices.

Table 2. The amplitude of CTQW of partial vertices. The vertex 65 is the center vertex, vertices $65, 66, 67, 68, 69$ are members of the maximum clique, and vertices $61, 62, 63, 64$ are not.

v_i	61	62	63	64	65	66	67	68	69
$\lambda \approx -2.12$	-0.024	-0.024	-0.024	-0.024	0.913	-0.104	-0.104	-0.104	-0.104
$\lambda \approx 2.63$	-0.004	-0.004	-0.004	-0.004	0.034	0.088	0.088	0.088	0.088
$\lambda \approx 3.53$	0	0	0	0	0	0	0	0	0
$\lambda \approx 34.49$	0.028	0.028	0.028	0.028	0.054	0.016	0.016	0.016	0.016

By adding sub-module VFSA to Algorithm A, algorithm B is obtained. The procedure of algorithm B is illustrated in the following pseudocode.

Algorithm B for finding the maximum clique

Require: Center graph G_s with its center vertex v_s
Ensure: The maximum clique $Clique(G_s)$

function ALGORITHM_B(G_s)
  Do eigen-decompositon on G_s, find v_{max}, v_{min} and v_{ref}
  $Clique(G_s) \leftarrow \{\}$
  $result1 \leftarrow Pick\_max(C(v_{max}, G_s))$
  while $v_i \in V(C(v_{min}, G_s))$ **do**
    $Temp \leftarrow Pick\_max(C(v_i, C(v_{min}, G_s)))$
    $result2 \leftarrow Max(result2, Temp)$
    $i \leftarrow i + 1$
  end while
  $result3 \leftarrow VFSA(C(v_{min}, G_s), v_{min}, v_{ref})$
  $result4 \leftarrow$ ALGORITHM_B($G_s - v_{min}$)
  $Clique(G_s) \leftarrow Max(result1, result2, result3, result4)$
end function

In the pseudocode $C(v_{min}, G_s)$ means the subcenter graph of v_{min} which is a subgraph of G_s, and analogously $C(v_i, C(v_{min}, G_s))$ denotes the subcenter graph of v_i which is a subgraph of $C(v_{min}, G_s)$. The tree-like diagram of algorithm B is showed in the Fig13. The first and second sub-modules in Fig.5 are the same with Algorithm A, and the third module is the VFSA (variational frequency selection algorithm). The inputs of VFSA are $C(v_{min}, G_s)$, v_{min} and v_{ref}, where v_{min} signifies the vertex with the weakest intensity at the frequency λ_1. Let f_{ref} denote the frequency where v_{min} takes the largest intensity in graph G_s. The vertex v_{ref} is the auxiliary adjacent vertex of v_{min} and has the largest intensity except v_{min} at frequency f_{ref}. Vertex v_{ref} acts as the central vertex in the subsequent procedure. Since f_{ref} and v_{ref} can be determined when v_{min} is given, we regard them as two implicit parameters and do not show them in the Fig.13. The tree-like diagram of algorithm B is shown in Fig.13.

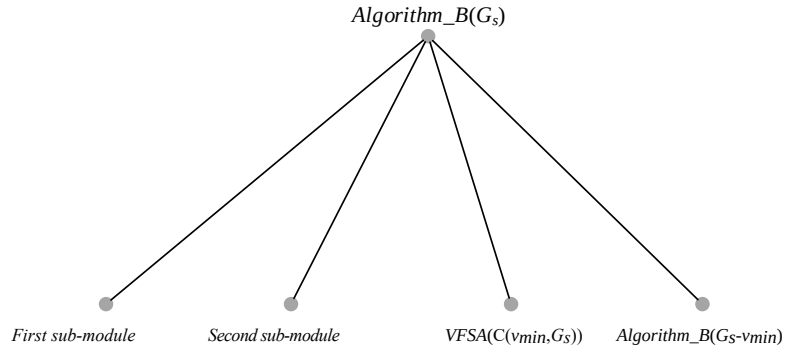

Fig. 13. The tree-like diagram of algorithm B. There are four sub-modules in algorithm B, and the fourth sub-module recursively calls the algorithm B on a smaller subgraph.

The results of algorithm B on random graphs are shown in Fig.14. The expected value of clique number $\omega(G)$ of graphs with order 20 to 100 are given in Fig.14, where for each order there are $100,000$ random graphs are sampled and the $\omega(G)$ is is the average of that $100,000$ graphs. In the numerical expirements, the clique number of every graph is obtained algorithm B and then checked by classical exact algorithm branch-bound method for insuring the result of algorithm B is accurate. The curves in Fig.14 are consistent with theorem in work of Grimmett and McDiarmid[28] and work of Bollobas and Erdos[29].

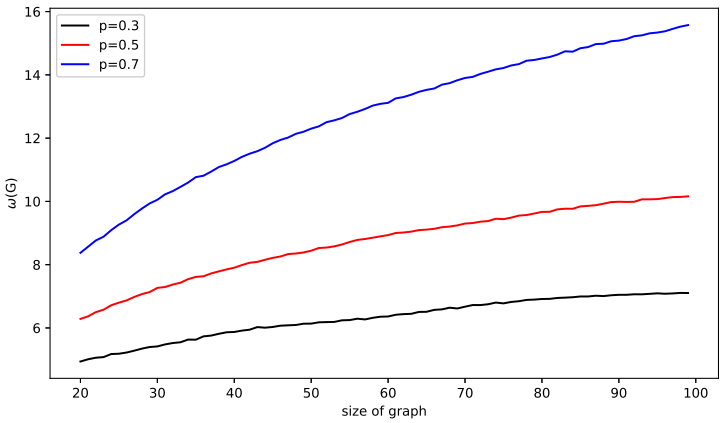

Fig. 14. The expected value of clique number. Each color of curve represent a specifical p.

6 Conclusion

In this paper, we show that the clique structure of a graph is related to the CTQW. For some ideal graphs, the frequency intensity of the probability amplitude of CTQW is a good feature to directly speculate whether a vertex is a member of the maximum clique or not. As the frequencies of CTQW are the eigenvalues of the adjacency matrix, the clique structure is related to the eigenvalues and the corresponding eigenvectors. For general graphs, this feature is not obvious, and one cannot directly find the maximum clique. To reveal the

hidden maximum clique, we propose two recursive algorithms, Algorithm A and algorithm B, using CTQW with $O(N^6)$ time complexity to find the maximum clique on graphs. It seems that Algorithm A is valid for random graphs via numerical experiments, but counter examples can be elaborately constructed. For such graphs whose maximum clique cannot be found by Algorithm A, we propose algorithm B to be fixed. The counter examples of algorithm B were not found or elaborately constructed. Although the worst time cost is $O(N^6)$ which is very large, this can be improved by releasing some rules of vertex selection. In addition, if the amplitude of one of the vertices of the maximum clique is always bigger than the other vertices for the second class of graphs, then algorithm B is universal, but this is beyond the scope of this article and will be undertaken in future work.

Acknowledgments

This work is supported by the National Natural Science Foundation of China(Grant Nos. 61871120 and 61502101), Natural Science Foundation of Jiangsu Province, China (Grant No.BK20191259), the Six Talent Peaks Project of Jiangsu Province (Grant No. XYDXX-003), and the Fundamental Research Funds for the Central Universities.

References

1. Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.
2. John M Robson. Finding a maximum independent set in time O(2n/4). Technical report, Technical Report 1251-01, LaBRI, Université Bordeaux I, 2001.
3. Marcello Pelillo. *Heuristics for maximum clique and independent setGraph coloring; Greedy randomized adaptive search procedures; Heuristics for maximum clique and independent set; Replicator dynamics in combinatorial optimization HEURISTICS FOR MAXIMUM CLIQUE AND INDEPENDENT SET*, pages 951–963. Springer US, Boston, MA, 2001. ISBN 978-0-306-48332-5.
4. Geng, Xiutang and Xu, Jin and Xiao, Jianhua and Pan, Linqiang. A simple simulated annealing algorithm for the maximum clique problem. *Information ences* , 177(22):5064–5071, 2007.
5. Aarts E, Lenstra JK (eds) Local search in combinatorial optimization. *Journal of the Operational Research Society*, 50(2):273-284, 1997.
6. Carter B, Park K. How good are genetic algorithms at finding large cliques: An experimental study. Techn Report Comput Sci Dept Boston Univ no. BU-CS-93-015 1993.
7. Friden C, Hertz A, de Werra M. STABULUS: A technique for finding stable sets in large graphs with tabu search. Computing 42:3544 1989.
8. Gendreau A, Salvail L, Soriano P. Solving the maximum clique problem using a tabu search approach. Ann Oper Res 41:385403 1993.
9. Soriano P, Gendreau M. Diversification strategies in tabu search algorithms for the maximum clique problem. Ann Oper Res 63:189207 1996.
10. Kirkpatrick S, Gelatt CD, Vecchi MP. Optimization by simulated annealing. *Science*, 220671680. 1983.
11. Lin F, Lee K. A parallel computation network for the maximum clique problem. Proc. 1st Internat. Conf. Fuzzy Theory Tech. 1992.
12. Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.
13. Peter W Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. Ieee, 1994.
14. Lov K Grover. Quantum mechanics helps in searching for a needle in a haystack. *Physical review letters*, 79(2):325, 1997.

15. Jérémie Roland and Nicolas J Cerf. Adiabatic quantum search algorithm for structured problems. *Physical Review A*, 68(6):062312, 2003.
16. Nicolas J Cerf, Lov K Grover, and Colin P Williams. Nested quantum search and structured problems. *Physical Review A*, 61(3):032303, 2000.
17. Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.
18. William M Kaminsky and Seth Lloyd. Scalable architecture for adiabatic quantum computing of np-hard problems. In *Quantum computing and quantum bits in mesoscopic systems*, pages 229–236. Springer, 2004.
19. Andrew M Childs, Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. Finding cliques by quantum adiabatic evolution. *arXiv preprint quant-ph/0012104*, 2000.
20. Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. *Random Structures & Algorithms*, 13(3-4):457–466, 1998.
21. Edward Farhi and Sam Gutmann. Quantum computation and decision trees. *Physical Review A*, 58(2):915, 1998.
22. Matthias Christandl, Nilanjana Datta, Artur Ekert, and Andrew J Landahl. Perfect state transfer in quantum spin networks. *Physical review letters*, 92(18):187902, 2004.
23. John King Gamble, Mark Friesen, Dong Zhou, Robert Joynt, and SN Coppersmith. Two-particle quantum walks applied to the graph isomorphism problem. *Physical Review A*, 81(5):052313, 2010.
24. Dragoš M Cvetković, Michael Doob, and Horst Sachs. *Spectra of graphs: theory and application*, volume 87. Academic Pr, 1980.
25. Danail Bonchev, Alexandru T Balaban, and Ov Mekenyan. Generalization of the graph center concept, and derived topological centric indexes. *Journal of Chemical Information and Computer Sciences*, 20(2):106–113, 1980.
26. Piet Van Mieghem. *Graph spectra for complex networks*. Cambridge University Press, 2010.
27. Brian Hayes. Computing science: Can't get no satisfaction. *American Scientist*, 85(2):108–112, 1997. ISSN 00030996. URL `http://www.jstor.org/stable/27856726`.
28. Grimmett, G. R. and Mcdiarmid, C. J. H. On colouring random graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, 77(2):313–324, 1975.
29. Bollobas B , Erds P . Cliques in random graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, 80(3):419–427, 1997.

Appendix A

A counter example graph T of Algorithm A.

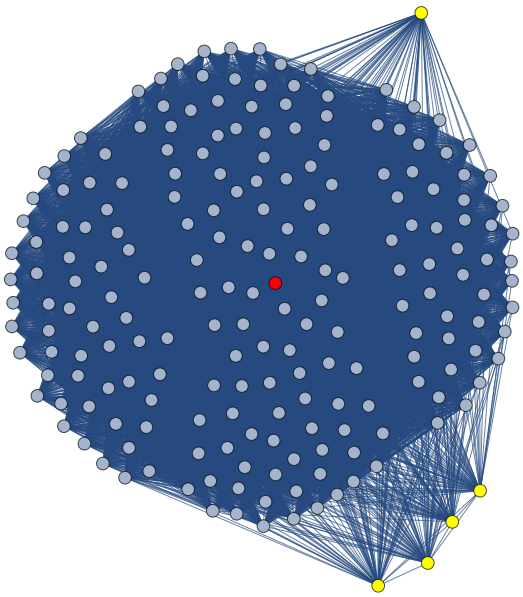

Fig. A.1. A counter example graph T with 206 vertices. The red vertex is the center vertex, the yellow vertices are the members of the maximum clique, the induced subgraph of other gray vertices is a complete multi-partite graph.

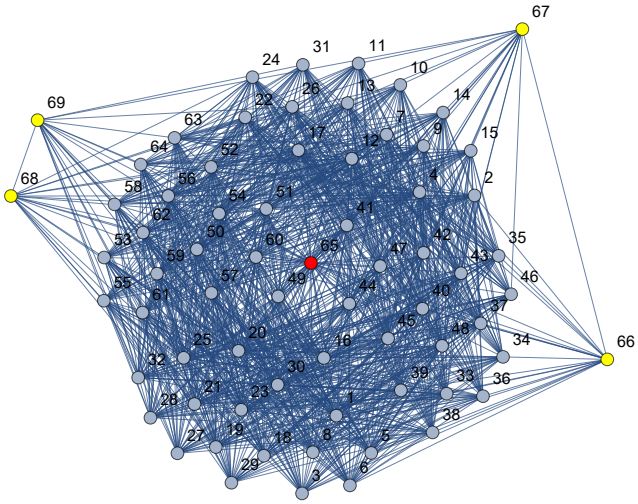

Fig. A.2. Center subgraph of graph T in Fig.A.1. The vertex 65 is the center vertex, the yellow vertices belong to the maximum clique, the other gray vertices together to be a a complete multi-partite graph.