Random Walks: A Review of Algorithms and Applications

Feng Xia, *Senior Member, IEEE*, Jiaying Liu, Hansong Nie, Yonghao Fu, Liangtian Wan, *Member, IEEE*, and Xiangjie Kong, *Senior Member, IEEE*

***Abstract*—A random walk is known as a random process which describes a path including a succession of random steps in the mathematical space. It has increasingly been popular in various disciplines such as mathematics and computer science. Furthermore, in quantum mechanics, quantum walks can be regarded as quantum analogues of classical random walks. Classical random walks and quantum walks can be used to calculate the proximity between nodes and extract the topology in the network. Various random walk related models can be applied in different fields, which is of great significance to downstream tasks such as link prediction, recommendation, computer vision, semi-supervised learning, and network embedding. In this article, we aim to provide a comprehensive review of classical random walks and quantum walks. We first review the knowledge of classical random walks and quantum walks, including basic concepts and some typical algorithms. We also compare the algorithms based on quantum walks and classical random walks from the perspective of time complexity. Then we introduce their applications in the field of computer science. Finally we discuss the open issues from the perspectives of efficiency, main-memory volume, and computing time of existing algorithms. This study aims to contribute to this growing area of research by exploring random walks and quantum walks together.**

***Index Terms*—Random walks, quantum walks, algorithm, computational science.**

I. Introduction

A RANDOM walk is a random process in the mathematical space. It describes a path consisting of a succession of random steps in the mathematical space. It is firstly introduced by Pearson in 1905 [1]. Spitzer [2] gives a complete review of random walks for mathematical researchers and clearly presents the mathematical principles of random walks. Random walks can be used to analyze and simulate the randomness of objects and calculate the correlation among objects, which are useful in solving practical problems. It is fast becoming a key instrument in the fields of computer science, physics, chemistry, biology, economics, etc.

In the mathematical space, a simple random walk model is a random walk on a regular lattice, in which one point can jump to another position at each step according to a certain probability distribution. When it is applied on a specific network, the transition probability between nodes is positively relevant to their correlation strength. That is, the stronger their association is, the greater the transition probability is. After enough steps, we can obtain a random path that can describe the network structure.

The most typical random walk based algorithms in computer science area is PageRank [3]. It calculates the importance of web pages by walking randomly among them. Researchers have developed a series variants of PageRank, such as personalized PageRank [4], [5]. Researchers also improve the original random walk rules and propose some new algorithms, such as random walk with restart (RWR) [6], and lazy random walk (LRW) [7].

Quantum walks are first proposed by Aharonov *et al.* [8] in 1993. Quantum walks can be regarded as the counter part of classical random walks in quantum mechanics. The main difference between classical random walks and quantum walks is that quantum walks don't converge to some limiting distributions. They can spread significantly faster or slower than classical random walks because of quantum interference. Compared to classical random walk based algorithms, quantum walk based algorithms have lower time complexity [9]–[12]. They can provide an exponential speedup over any classical algorithm [9]. Quantum walk based algorithms can be roughly divided into two categories: discrete time based algorithms and continuous time based algorithms [13].

A random walk is implemented by utilizing the network topology, so it can also be used to calculate the proximity between nodes. For example, researchers have introduced algorithms based on random walks in the area of collaborative filtering [14]–[19]. Compared with other alternative approaches, random walk based algorithms can incorporate a great deal of contextual information. As same as collaborative filtering, link prediction and recommender system also aim to calculate the k-most-close nodes for the selected node. Hence, random walks are also effective in link prediction and recommendation system [20]–[27]. Random walks can also be applied in computer vision [7], [28]–[36], semi-supervised learning [37]–[41], network embedding [42], [43], and complex social network analysis [44]. Some researchers are also focusing on studying

Manuscript received June 3, 2019; revised September 16, 2019; accepted October 27, 2019. Date of publication November 25, 2019; date of current version March 25, 2020. This work was supported in part by the National Natural Science Foundation of China (61872054), and in part by the Fundamental Research Funds for the Central Universities under Grants (DUT19LAB23, DUT18JC09). *(Corresponding author: Xiangjie Kong.)*

F. Xia is with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116620, China, and also with the School of Science, Engineering and Information Technology, Federation University Australia, Ballarat, VIC 3353, Australia (e-mail: f.xia@ieee.org).

J. Liu, H. Nie, Y. Fu, L. Wan, and X. Kong are with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116620, China (e-mail: jiaying_liu@outlook.com; hansong.nie@outlook.com; 2871375215@qq.com; wan.liangtian.2015@ieee.org; xjkong@ieee.org).

Digital Object Identifier 10.1109/TETCI.2019.2952908

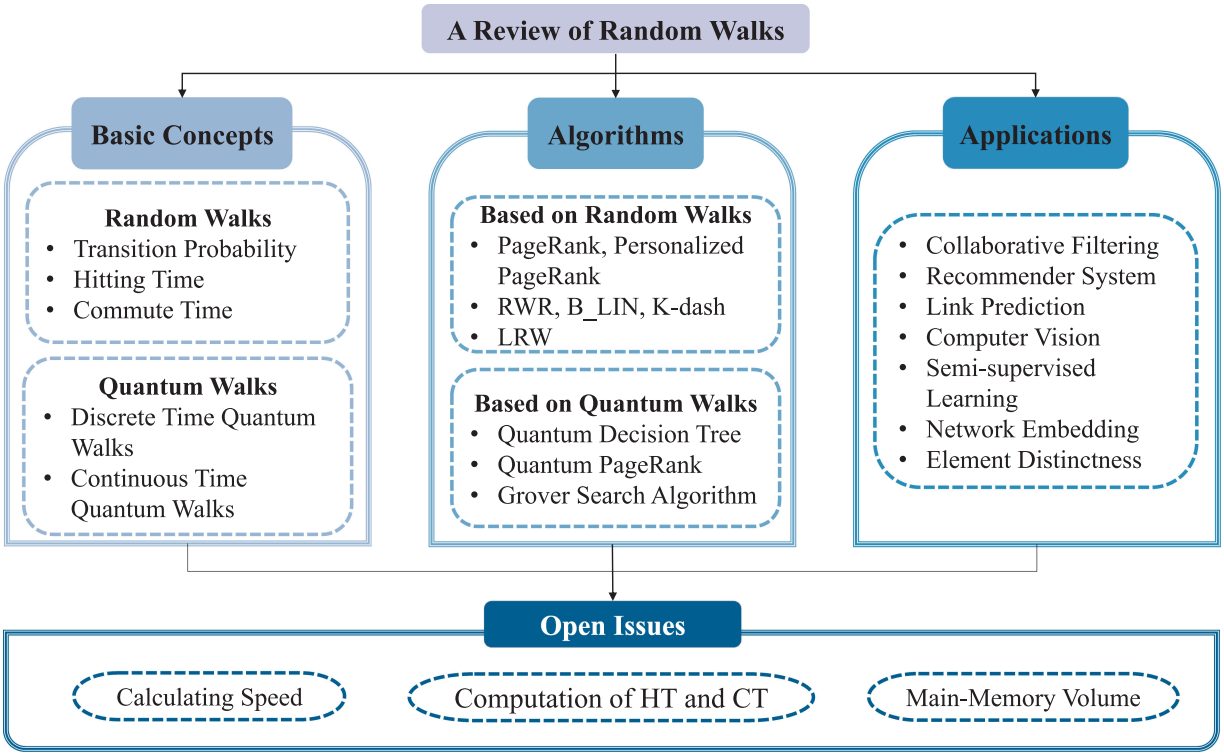


Fig. 1. The framework of random walks review.

applications of random walks on graphs [45], [46], text analysis [47], science of science [48], and knowledge discovery [49]. Quantum walks are often used to accelerate classical algorithms. It can be used to decision trees [10], search problems [11], [12], and element distinctness [50], [51].

In this paper, we provide a comprehensive review of random walks. To the best of our knowledge, it is the first time to review classical random walks and quantum walks together. We summarize random walks in the field of computer science from the perspectives of basic concept, algorithm and application. We also compare these algorithms systematically. In addition, some open issues of random walks and quantum walks are presented.

In the rest of the paper, we first introduce basic concepts and notations of classical random walks and quantum walks in Section II. In particular, we introduce quantum walks from two aspects: discrete time quantum walks and continuous time quantum walks. In Section III, we focus on illustrating some typical algorithms based on classical random walks and quantum walks. We also make an analytical comparison between these algorithms. In Section IV, we show the application scenarios of different algorithms and identify their advantages and disadvantages. Section V highlights the problems and future directions. Finally, the work is concluded in Section VI. The overall structure of this paper is summarized in Fig. 1.

II. Preliminaries and Notations

In this section, we will introduce the basic concepts and notations about random walks, including classical random walks and quantum walks. Table I lists the commonly used notations in this paper.

A. Classical Random Walks

A random walk is known as a random process. It describes a path consisting of a succession of random steps on some mathematical space, which can be denoted as $\{\xi_t, t = 0, 1, 2, \ldots\}$ where ξ_t is a random variable describing the position of a random walk after t steps. The sequence can also be regarded as a special category of Markov chain. In the initial state of a random walk, the position ξ_0 may be fixed or drawn from some initial distribution P_0 [45]. We can represent the distribution of position after t steps as follows:

$$P_t(i) = Pr(\xi_t = i) \tag{1}$$

where $P_t(i)$ is the probability that the random walk visits the position i after t steps. If the walk locates at the position i after t steps, the single step transition probability refers to the probability that the random walk can move to the position j at after the next step. It is represented as p_{ij} and can be calculated as:

$$p_{ij} = Pr(\xi_{t+1} = j | \xi_t = i). \tag{2}$$

Further, the t steps transition probability is defined as follows:

$$p_{ij}^{(t)} = Pr(\xi_t = j | \xi_0 = i). \tag{3}$$

From the perspective of graph representation, let $G = (V, E)$ be a connected graph, where V is the vertex set and E is the edge

TABLE I
DESCRIPTION OF NOTATIONS.

Notation	Description
$\{\xi_t, t = 0, 1, 2, ...\}$	the random variable in a sequence
P_0	the initial distribution of a random walk
t	the step number of a random walk
P_t	the distribution after t steps in a random walk
G	a connected graph
V	the set of nodes in G
E	the set of edges in G
n	the number of nodes in G
$A \in \mathbb{R}^{n\times n}$	the adjacency matrix of G
p_{ij}	the probability (single step) from node i to node j
$p_{ij}^{(t)}$	the probability (t steps) from node i to node j
$M \in \mathbb{R}^{n\times n}$	the matrix of transition probabilities of G
$D \in \mathbb{R}^{n\times n}$	a diagonal matrix
L	the Laplacian matrix of G
h_{ij}	the hitting time between node i and node j
C_{ij}	the commute time between node i and node j
$\mathcal{N}_i$	the neighbor set of node i
$\|\psi_{x_0}\rangle$	the wave-packet localizing around the position x_0
$\mathcal{P}$	the momentum operator
U_l	the unitary operator
S_z	the z component of the spin
$\|\uparrow\rangle, \|\downarrow\rangle$	two eigenstates of S_z
$\mathcal{H}_p$	the Hilberet space
$\mathcal{H}_c$	the coin-space
$\otimes$	tensor product
H	Hadamard coin
$\hat{H}$	an infinitesimal generator matrix and the Hamiltonian function

set. The adjacency matrix of G is denoted as $A \in \mathbb{R}^{n\times n}$, where n is the number of nodes in G. A_{ij} denotes the weight of edge from the node i to the node j. Then the transition probability (single step) from node i to node j on the graph can be defined as:

$$p_{ij} = \frac{A_{ij}}{\sum_{j\in V} A_{ij}}. \tag{4}$$

Further, let $M = (p_{ij})_{i,j\in V}$ be the matrix of transition probabilities on G. Then we can define D which denotes a diagonal matrix as:

$$D_{ii} = 1 \Big/ \sum_{j\in V} A_{ij}. \tag{5}$$

Thus we can redefine the transition probability matrix M of graph G as:

$$M = DA. \tag{6}$$

The rule of a random walk can be expressed as:

$$P_{t+1} = M^T P_t \tag{7}$$

where P_t can be viewed as a vector in $\mathbb{R}^{|V|}$. Its i-th element means the probability that the random walk from the initial node v_0 reaches the i-th node after t steps. We can calculate P_t as:

$$P_t = (M^T)^t P_0. \tag{8}$$

The Laplacian matrix of G can be defined as follows:

$$L = D - A. \tag{9}$$

Hitting time. Hitting time h_{ij} can be considered as the expected number of steps before node j visited in a random walk starting from node i [45]. The recursive definition of hitting time is as follows:

$$h_{ij} = \begin{cases} 1 + \sum_{k\in\mathcal{N}_i} p_{ik} h_{kj} & \text{if } i \neq j, \\ 0 & \text{if } i = j \end{cases} \tag{10}$$

where h_{kj} denotes the hitting time from node k to node j and node k is a direct neighbor of node i. p_{ik} is the transition probability from node i to node k. $\mathcal{N}_i$ is the neighbor set of node i [44].

The hitting time matrix is not symmetric even in a regular graph. Another important fact about hitting time is proved by Lovasz [45]: hitting time follows the triangle inequality.

Commute time. Commute time from node i to node j is defined as:

$$C_{ij} = h_{ij} + h_{ji} \tag{11}$$

which means the excepted number of steps in a random walk starting at i, before accessing the node j and then reaching the node i again [45]. In order to research the commute time on undirected graphs, Chandra *et al.* [52] give an electrical network view. They compare the commute time between two nodes on a graph to the resistance on an electrical network. They give some intuitions about commute time on undirected graphs:

- The smaller resistance can make the current go through more easily on electrical networks. The shorter commute time can make random walkers diffuse easier on undirected graphs.
- Commute time should be robust to small perturbation so that removing or adding a few resistances do not change much on an electrical network.

B. Quantum Views of Random Walks

The scalable quantum computer is a topical issue so that approaches of quantum computation are popular topics nowadays. Quantum walks are the corresponding part of classical random walks in quantum mechanics. The main difference between them is that quantum walks don't converge to some limiting distributions. Due to the quantum interference, quantum walks can spread significantly faster or slower than classical random walks. Existing literature gives us explicit introduction to quantum random walk in a comprehensive way [8], [13], [53], [54].

In quantum mechanics, let $|\psi_{x_0}\rangle$ denote a wave-packet which localizes around a position x_0. $\mathcal{P}$ is a momentum operator. The translation with length l of a particle can be expressed as the unitary operator U_l, which can be calculated as [13]:

$$U_l = \exp(-i\mathcal{P}\hat{h}) \tag{12}$$

where $\hat{h}$ is reduced Planck constant which is the smallest unit of to measure angular momentum. Meanwhile, it satisfies the following formula:

$$U_l \psi_{x_0} = \psi_{x_0 - l} \tag{13}$$

where we can set $\hat{h} = 1$ to simplify the notation.

We can assume that the particle has a spin-$1/2$ degree of freedom and represent the operator corresponding to the z component of the spin as S_z. The eigenstates of S_z are $|\uparrow\rangle$ and $|\downarrow\rangle$. A spin-$1/2$ particle can be described by a 2-vector:

$$|\Psi\rangle = (|\widetilde{\psi}^{\uparrow}\rangle, |\widetilde{\psi}^{\downarrow}\rangle)^T \tag{14}$$

where $|\widetilde{\psi}^{\uparrow}\rangle$ is the component of the wave-function of the particle in the spin-$|\uparrow\rangle$ space. $|\widetilde{\psi}^{\downarrow}\rangle$ is the component of the wave-function of the particle in the spin-$|\downarrow\rangle$ space.

The concept of quantum walks is firstly proposed by Aharonov *et al.* [8] in 1993. Kempe [13] presents two kinds of quantum walk including discrete time quantum walks and continuous time quantum walks. We will introduce an easy example in one-dimensional space to help readers quickly understand the basic ideas of discrete time quantum walks and continuous time quantum walks.

1) Discrete Time Quantum Walks: We can define a space $\mathcal{H} = \mathcal{H}_p \otimes \mathcal{H}_c$ for one dimensional quantum walks [13]. $\mathcal{H}_p$ denotes the Hilbert space which is spanned by the positions of the particle. For one dimensional Hilbert space, it can be represented as:

$$\mathcal{H}_p = \{|i\rangle : i \in Z\} \tag{15}$$

where $|i\rangle$ a particle localized at the position i. $\mathcal{H}_c$ denotes the coin-space which is spanned by two basic states $\{|\uparrow\rangle, |\downarrow\rangle\}$. The unitary operation S defines the conditional translation on space $\mathcal{H}$:

$$S = |\uparrow\rangle\langle\uparrow| \otimes \sum_i |i+1\rangle\langle i| + |\downarrow\rangle\langle\downarrow| \otimes \sum_i |i-1\rangle\langle i| \tag{16}$$

where $i \in Z$, $\otimes$ is the tensor product which separates two degrees of freedom, spin and space, and will allow us to view the resulting correlations between these two degrees of freedom more clearly [13]. S can realize the following equations:

$$S(|\uparrow\rangle \otimes |i\rangle) = |\uparrow\rangle \otimes |i+1\rangle, \tag{17}$$

$$S(|\downarrow\rangle \otimes |i\rangle) = |\downarrow\rangle \otimes |i-1\rangle. \tag{18}$$

It means that the particle jumps right if it has spin up and left if it has spin down.

C is a unitary transformation which can rotate the spin in H_c. One of the most frequently used unitary transformation is Hadamard coin H [13]. Here is an example of H:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{19}$$

The Hadamard walk on Z is [13]:

$$\begin{aligned} |\uparrow\rangle \otimes |0\rangle &\xrightarrow{H} \frac{1}{\sqrt{2}}(|\uparrow\rangle + |\downarrow\rangle) \otimes |0\rangle \\ &\xrightarrow{S} \frac{1}{\sqrt{2}}(|\uparrow\rangle \otimes |1\rangle + |\downarrow\rangle \otimes |-1\rangle) \end{aligned}. \tag{20}$$

Then the single step quantum walk transformation can be defined as:

$$U = S \cdot (C \otimes I). \tag{21}$$

A quantum walk of t steps is defined as the transformation U^t.

2) Continuous Time Quantum Walks: The original purpose of continuous time quantum walks is to speed up algorithms using classical random walks. The concept of continuous time quantum walks is first presented by Farhi *et al.* [10] in 1998. The authors exploit quantum walks in the decision tree algorithm instead of classical random walks. Different from discrete time quantum walks, continuous time quantum walks don't need a coin-space $\mathcal{H}_c$ and take place entirely in the Hilbert space $\mathcal{H}_p$ [13]. The idea of continuous time quantum walks is from continuous time classical random walks. Kempe [13] gives another expression of the continuous time random walks as:

$$P(t) = exp(-\hat{H}t)P(0) \tag{22}$$

which is in analogy to Equation (8) and $\hat{H}$ is an infinitesimal generator matrix with similar structure to M.

The key idea proposed by Farhi *et al.* [10] is that the generator matrix $\hat{H}$ will become the Hamiltonian function of the process and generate an evolution $U(t)$ as follows:

$$U(t) = exp(-i\hat{H}t). \tag{23}$$

The connections between discrete quantum walks and continuous quantum walks are proposed by Strauch [55]. The author finds that discrete quantum walks can be transferred to the continuous quantum walks by the precise limiting procedure.

III. Algorithms Based on Random Walks

In this section, we will introduce some typical algorithms based on classical random walks and quantum walks.

A. Algorithms Based on Classical Random Walks

1) PageRank: PageRank is first proposed by Page *et al.* [3] in 1999. The purpose is to rank the web page in the *World Wide Web* (*WWW*). The network of web page is considered as a graph where web pages are considered as nodes. If there is a web page containing a hyperlink which points to another web page, then there should be a directed edge between these two nodes. The direction of the edge is as same as the web redirection. The most simple PageRank can be described by the following mathematical equation:

$$R(u) = c \sum_{v \in B_u} \frac{R(v)}{N_v} \tag{24}$$

where $R(u)$ is the rank of the web u. B_u is the set of pages pointing to page u, and c is a normalization parameter. Let $F(v)$

be the set of pages that v points to. N_v is the number of pages in $F(v)$.

The simple version corresponds to the standing probability distribution of a random walk on the network. When a random walk quickly converges to a limiting distribution on the set of nodes, it can be regarded as rapidly-mixing. It has been proved that a random walk can be rapidly-mixing on the graph of *WWW* [3]. The importance of a node can be regarded as the probability that the random walker reaches the node after long enough steps. The mathematical expression is:

$$R_{t+1} = M^T R_t \tag{25}$$

where R is the vector of PageRank, and M^T is the transition probability matrix.

To improve the convergence rate of PageRank, Kamvar *et al.* [56] present a novel algorithm called Quadratic Extrapolation for PageRank computation. It accelerates the convergence of the power method. The main strategy of the algorithm is to reduce the estimation of non-main eigenvectors periodically.

The result of PageRank is independent of the keywords searched by users. To solve this problem, Haveliwala *et al.* [5] present personalized PageRank:

$$R_{t+1} = (1-\alpha)M^T R_t + \alpha p \tag{26}$$

where α is a decay factor, p is the personalized PageRank vector which reflects the importance of each node in a graph for a specific user.

2) Random Walk With Restart: RWR is first proposed by Pan *et al.* [6] to calculate the affinity between node i and node j. Considering a random walk starting from node i, the walker can go back to node i with the probability c, which is the difference between RWR and classical random walks. Let $u_i(j)$ denote the steady-state probability that the random walker will visit node j. The formula is:

$$u_i = (1-c)M^T u_i + ce_i \tag{27}$$

where u_i is the probability distribution vector of RWR starting from node i. e_i is a vector whose entry that corresponds to node i equals 1, and the remaining elements being 0.

Let $G = \{V_1 \cup V_2, E\}$ denote a bipartite graph, where $V_1 = \{a_i | 1 \le i \le k\}$ and $V_2 = \{t_i | 1 \le i \le n\}$. k and n are the number of nodes in V_1 and V_2, respectively. The adjacency matrix A_B can be written as:

$$A_B = \begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix} \tag{28}$$

where A is k-by-n matrix. Using the bipartite structure, Sun *et al.* [57] propose that u_i should be calculated as:

$$u_i = (1-c)\begin{pmatrix} col\_norm(A)u_i(k+1:k+n) \\ col\_norm(A^T)u_i(1:k) \end{pmatrix} + ce_i \tag{29}$$

where $u_i(1:k)$ and $u_i(k+1:k+n)$ are the vectors of first k and last n elements of u_i, respectively. They only perform RWR on the partition that contains the query node. In other words, they present a local estimation of RWR by applying graph partition.

RWR is time-consuming when it is applied on large graphs. To fill this gap, Tong *et al.* [58] present a fast RWR by low-rank approximation. The authors first rewrite RWR as:

$$\begin{aligned} u_i &= c\hat{M}^T u_i + (1-c)e_i \\ &= (1-c)(I - c\hat{M}^T)^{-1} e_i \\ &= (1-c)Q^{-1}e_i \end{aligned} \tag{30}$$

where $Q = I - c\hat{M}^T$, $\hat{M}$ is the normalized weighted matrix associated with M. Let $\hat{M} = \hat{M}_1 + \hat{M}_2$, where $\hat{M}_1$ is the within-partition matrix and $\hat{M}_2$ is the cross-partition matrix. Then they propose B_LIN using SVD to calculate u_i as:

$$\begin{aligned} Q_1 &= (1 - c\hat{M}_1)^{-1} \\ \hat{M}_2 &= USV \\ \hat{\Lambda} &= (S^{-1} - cVQ_1^{-1}U)^{-1} \\ u_i &= (1-c)(Q_1^{-1}e_i + cQ_1^{-1}U\hat{\Lambda}VQ_1^{-1}e_i). \end{aligned} \tag{31}$$

Random walks are used to calculate the proximity between nodes and the specific node. If we want to find the top-k nodes, we can follow the method proposed by Fujiwara *et al.* [59] called K-dash to calculate the proximity of only selected nodes to find the top-k nodes. They first obtain the following equation:

$$\begin{aligned} u_i &= c\hat{M}^T u_i + (1-c)e_i \\ &= c(I - (1-c)\hat{M}^T)^{-1} e_i \\ &= cW^{-1}e_i \end{aligned} \tag{32}$$

where $W = I - (1-c)\hat{M}^T$, $\hat{M}$ is the column normalized weighted matrix associated with M. Since they don't need to calculate the proximity of all nodes, W is a sparse matrix, but W^{-1} may be dense. When the graph becomes large, it requires quadratic space to hold the inverse matrix which is unrealistic. Then they decompose W by LU decomposition to calculate u_i as follows:

$$\begin{aligned} W &= LU \\ u_i &= cU^{-1}L^{-1}e_i \end{aligned} \tag{33}$$

where the matrices L^{-1} and U^{-1} are lower and upper triangular, respectively.

3) Lazy Random Walk: LRW [7] is used to solve image segmentation problems. It first defines a graph on a given image, where every pixel is identified uniquely by a node. The similarity between node i and node j is defined as:

$$w_{ij} = \exp\left(-\frac{||g_i - g_j||^2}{2\sigma^2}\right) \tag{34}$$

where g_i is the image intensity value of node i. σ is the user defined parameter. The degree of each node is computed as:

$$d_i = \sum_{j \in V} w_{ij}. \tag{35}$$

TABLE II
COMPARISON ANALYTIC ON THE CLASSICAL ALGORITHMS.

	Formula	Time Complexity	Time-Consuming
PageRank [3]	$R_{t+1} = M^T R_t$		Yes
personalized PageRank [5]	$R_{t+1} = (1-\alpha)M^T R_t + \alpha p$		Yes
RWR [6]	$u_i = (1-c)M^T u_i + ce_i$	$O(\|V\|^3)$	Yes
RWR on a bipartite graph [57]	$u_i = (1-c)\begin{pmatrix} col\_norm(A)u_i(k+1:k+n) \\ col\_norm(A^T)u_i(1:k) \end{pmatrix} + ce_i$	$O(\|V\|^2)$	No
B_LIN [58]	$u_i = (1-c)(Q_1^{-1}e_i + cQ_1^{-1}U\hat{\Lambda}VQ_1^{-1}e_i)$	$O(\|V\|^2)$	No
K-dash [59]	$u_i = cU^{-1}L^{-1}e_i$	$O(\|V\| + \|E\|)$	No
LRW [7]	$u_i = d_i \sum_{i=1}^{N=\|V\|} d_i$	$O(n\|V\|^2)$	Yes

The transition probability matrix is calculated as follows:

$$P_{ij} = \begin{cases} 1-\alpha & \text{if } i = j \\ \alpha \cdot w_{ij}/d_i & \text{if } i \sim j, \\ 0 & \text{if otherwise.} \end{cases} \tag{36}$$

where $i \sim j$ means the two nodes are adjacent nodes. α is a control parameter in the range $(0, 1)$. The equation means that the current node i in LRW will have the probability $(1-\alpha)$ to stay at node i and probability α to walk to the adjacent node. LRW will converges to a unique stationary distribution u as follows:

$$u_i = d_i \sum_{i=1}^{N=|V|} d_i. \tag{37}$$

Considering all of the above algorithms, PageRank, personalized PageRank, RWR and LRW are time-consuming on large graphs. Quadratic Extrapolation accelerates PageRank's convergence rate very well. Due to the personalized vector in personalized PageRank, it makes more sense to different users. Personalized PageRank and RWR have similar forms. RWR on a bipartite graph [57] converges faster, but has no generality. On the contrary, B_LIN and K-dash have fast convergence rate on any graph. K-dash calculates the proximity more exactly than B_LIN, because LU decomposition used in K-dash is not an approximation method like SVD used in B_LIN. Table II shows the differences among these algorithms.

B. Algorithms Based on Quantum Walks

In this section, we are going to introduce some algorithms based on two quantum walk models as mentioned above. We can find some different properties between quantum walks and classical random walks.

We will separate the algorithms into two categories depending on the model they use. The first category is based on the continuous time quantum walks, such as the quantum decision tree algorithm. The other is based on discrete time quantum walks, such as the quantum PageRank algorithm.

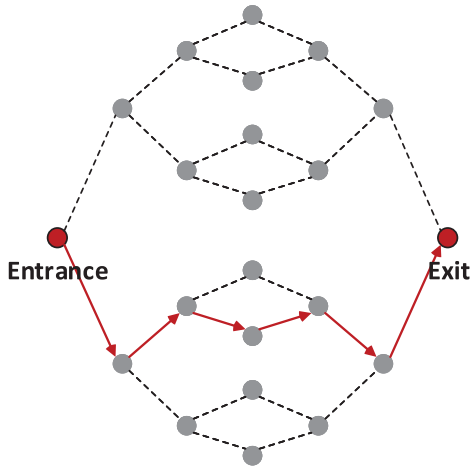


Fig. 2. Balance tree. We want to find the node named *Exit* by a classical random walk or quantum walk starting from *Entrance*.

1) Continuous Quantum Walk Based Algorithms: Fahri *et al.* [10] originally present the idea of continuous time quantum walks with the example of decision tree algorithm. They choose the approach that systematically explores the whole tree with a probabilistic rule.

The authors consider decision tree nodes as quantum states in Hilbert space. Then they constructs a Hamiltonian function $\hat{H}$ which determines the time evolution of the quantum system. With the basis of Hamiltonian function, the authors present the unitary time evolution operator shown in the Equation (23). They find that if classical random walk based algorithms require time polynomial in n to reach level n, quantum walks can also realize it. Moreover, if a tree is penetrable for a classical algorithm which requires time exponential in n, it is proved that the problem corresponding to the decision tree is solvable with this quantum algorithm in the polynomial time.

Childs *et al.* [9] construct an oracular problem which can be solved by a quantum walk in subexponential time. They first introduce a graph G_r consisting of two balanced binary trees of height n in Fig. 2. Then they modify the graph by randomly choosing a leaf on the left and connect it to a leaf on the right chosen at random in Fig. 3. Classical random walks or quantum

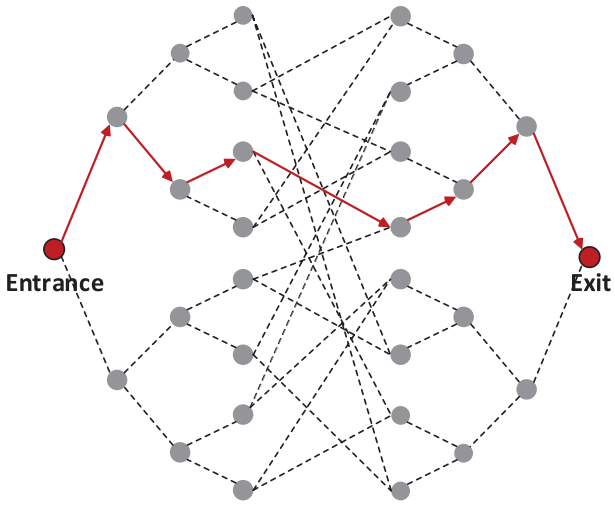


Fig. 3. Modified balance tree. We want to find the node named *Exit* by a classical random walk or quantum walk starting from *Entrance*.

walks go from *ENTRANCE* to *EXIT*. They define a Hamiltonian function $\hat{H}$ based on G's adjacency matrix to build a quantum walk on the graph. The equation of $\hat{H}$ is:

$$\langle n|\hat{H}|n'\rangle = \begin{cases} \gamma & n \neq n, nn' \in G \\ 0 & \text{otherwise} \end{cases} \tag{38}$$

where n and n' are nodes of G. nn' denotes the edge between node n and node n'. γ is the probability of moving to the next adjacent node. It is proved that the quantum walks are exponentially better than any classical random walks. However, this algorithm only finds the node named *EXIT* without finding a path from *ENTRANCE* to *EXIT*.

2) Discrete Quantum Walk Based Algorithms: In order to study the behavior of PageRank algorithm in the quantum network, Paparo *et al.* [60] present the quantum PageRank algorithm. They give an admissible class of quantum PageRank algorithms instead of a specific definition.

The authors exploit the idea of discrete time quantum walks. They define the coin space $\mathcal{H}_c$ and Hilbert space $\mathcal{H}_p$. The definition of coin space is similar to the one-dimension quantum walks.

$$\mathcal{H}_c = span\{|L\rangle, |R\rangle\}. \tag{39}$$

Since the PageRank algorithm is applied on the graph, the author defines the Hilbert space as the space of oriented edges:

$$H_p = span\{|i\rangle_1, |j\rangle_2 \mid i, j \in N\} \tag{40}$$

where N denotes all the nodes on the graph. The subscripts 1, 2 are used to show the direction [60].

The authors also reveal the properties of quantum PageRank algorithm in complex real world networks [61]. They find that the quantum PageRank algorithm can reveal the underlying topology of the network more univocally with respect to classical PageRank algorithms.

Considering the searching problem in the database, classical algorithms need $O(N)$ steps to find the target element, where N is the number of elements. Grover [11] proposes a novel quantum walk based algorithm to solve this problem. It is proved that the algorithm only takes $O(\sqrt{N})$ steps to find the same target. Affected by this, Shenvi *et al.* [12] propose a discrete quantum walk based algorithm. It can be regarded as a discrete walk on the hypercube and also achieve $O(\sqrt{N})$ searching time.

Since these algorithms are applied in different scenarios, it is hard to evaluate their performance. Compared quantum walk based algorithms with classical random walk based algorithms, computational complexity and convergence speed have been greatly improved in the quantum walks. The scope of the application is also more extensive for quantum walks. In addition, quantum walk based algorithms are better than classical random walk based algorithms in preserving the topology of network. Although the researches on quantum walks have increased in recent years [62]–[65], quantum walks from the perspectives of principle, mechanism and application are still worth exploring.

IV. Applications of Random Walks

Random walks have been successfully applied in various areas of computer science such as recommender system, computer vision, and network embedding. In this section, we select some major applications to illustrate the effectiveness and practicability of random walks in this section.

A. Collaborative Filtering

Collaborative filtering is a method of making automatic predictions about the interests of a user by collecting preferences from many users. It assumes that two people who have the same taste on one issue will have the same interest on the other issues.

Much literature has recorded methods of collaborative filtering with successful demonstrations of Bayesian, nonparametric, linear methods, etc. All these methods are essentially the same. They all match the individual to others based on their choices and use combination of their experiences to predict future choices.

Brand *et al.* [15] introduce a random walk view to collaborative filtering. They want to study affinity relations on the association graph of a relational database to find out what products a customer wants to buy next.

Fig. 4 shows a fragment of the association graph. The authors study the expected behavior of random walks on the association graph and propose a novel measure of similarity based on the cosine correlation of two states in a random walk. One significant advantage of random walks view is that it can incorporate large amounts of contextual information. Compared with original measures by cross validation experiments, the authors prove that the new measure is more predictive and robust to perturbations.

Fouss *et al.* [16], [17] also use random walks in the movie collaborative recommendation. The authors exploit the graph structure of the relational database to calculate dissimilarity between elements in sets. They compare ten different scoring algorithms. Five of them are based on random walks: the average commute time (CT, normal and PCA-based), the average

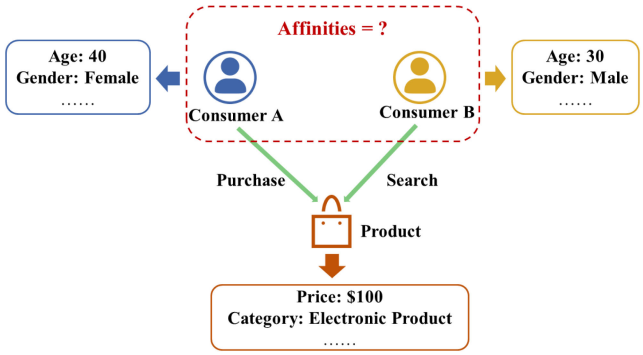


Fig. 4. An example of a customer-product association graph in a relational database. The affinities between pairs (customers) can be computed from statistics of a random walk based on the cosine correlation of two states on the entire graph.

first-passage time (one-way and return), and the pseudoinverse of the Laplacian matrix ($\mathbf{L}^+$).

They introduce a general procedure for computing similarity between elements of a relational database. The authors use movie recommendation as an example to show that ($\mathbf{L}^+$) almost always provides the best results in comparison with standard methods.

Yildirim *et al.* [18] propose a novel item-oriented algorithm called Random Walk Recommender. It is the first time to infer the transition probabilities between items based on their similarity. They first construct an item graph which captures the similarity of items between each other. Then they compute the rank values of items for each user by simulating a random walk on the graph. The rank values can be regarded as ratings between users and items. They prove their method performs significantly better than top-N algorithm [66] especially when the training data is sparse.

One of the biggest problems in collaborative filtering is the cold-start problem presented by Resnick [67]. It means that it is hard to do collaborative filtering for users who have rated only a very small number of items. Although there are some trust-based methods [68], [69] trying to solve the problem, the precision is not good enough.

Jamaliz *et al.* [19] propose a model called TrustWalker to solve this problem. They combine the trust-based and the item-based collaborative filtering approach to recommendation. It considers not only ratings of the target item, but also those of similar items.

Therefore, users in the trust network will keep a strong trust with the source user and we will get enough ratings at the same time, which will improve the precision of recommendations.

B. Recommender System

Recommender system is a subclass of information filtering system which attempts to predict users' ratings or preferences for items. It usually uses three ways to produce a list of recommendations: collaborative filtering, content-based filtering, and hybrid filtering.

Gori *et al.* [20] propose ItemRank, which is a scoring algorithm based on random walks. It can be used to rank products according to expected user preferences. They construct a correlation graph of movies. With the help of correlation graph, they can spread users' preferences. This procedure is similar to PageRank. Thus, it can be regarded as a biased version of PageRank designed to be applied to a recommender system.

Gori *et al.* [21] propose PaperRank algorithm based on random walks to solve the paper recommendation problem. Its structure is similar to ItemRank [20]. They utilize the model expressed by the citation graph and find out valuable papers related to research topics for researchers. Experiments on the ACM Portal Digital Library dataset demonstrate the outstanding performance of PaperRank.

Xia *et al.* [22] propose a method called CARE which incorporates author relations and historical preferences for scientific article recommendation. They assume that some researchers prefer to search articles published by the same authors to find articles they are interested in. The authors build a graph based on the information of co-authors' relationship. Then they employ the random walk with restart to generate a recommendation list. Compared with some baseline algorithms, the algorithm performs better in precision, F1-score, and recall.

Scholar collaboration is very important in academic research, but it is time-consuming to find a valuable collaborator. Xia *et al.* [23] propose the MVCWalker method based on random walks to find the most valuable collaborators. The authors use three academic factors to define link importance in academic social networks. Then they perform random walk with restart on the network to get the recommendation list of most valuable collaborators.

C. Link Prediction

Link prediction in a network refers to how to predict the possibility of links between two nodes in a network that have not yet been connected by network information. Lots of methods have been proposed to solve this problem [70], [71]. Liben-Nowell *et al.* [72] compare different methods in link prediction in detail, including hitting time, PageRank, and other variants of random walks.

The computation of hitting time and commute time is time consuming. To address this problem, Sarkar *et al.* [24] propose a truncated variant of commute time in the link prediction task. It utilizes the local structure of graphs. Then they propose an algorithm called GRANCH to find out which two nodes will have an edge in the near future. Experiments prove that GRANCH reduces the computation and storage while retaining the performance of methods.

Similarly, Liu *et al.* [25] propose two similarity indices for link prediction based on local random walk: the Local Random Walk index and the Superposed Random Walk index. While maintaining good prediction accuracy, they have lower time complexity.

Backstrom *et al.* [26] propose supervised random walks. It is a supervised learning task and ranks the nodes based on the network information including rich node and edge attributes. Its purpose is to learn the parameters of the function that assigns the strength of the edge such that a random walker is more likely to reach nodes to which new links will be created in future.

Link prediction also helps researchers find out the potential relation between miRNAs and diseases [27]. The authors consider the miRNA-Disease heterogeneous network as two overlapping sub-networks: miRNA similarity sub-network and diseases similarity sub-network. They employ random walk with restart to predict miRNA candidates that could potentially be associated with diseases. Cross validation and case analysis show that the method has good prediction performance.

D. Computer Vision

Computer vision is an interdisciplinary field that deals with how computers can be made to gain high-level understanding from digital images or videos. Its tasks include methods for acquiring, processing, analyzing, and understanding digital images, and extraction of high-dimensional data from the real world.

Meila *et al.* [28] present an approach of image clustering and segmentation based on random walks. The authors focus on pairwise (or similarity-based) clustering and image segmentation. They regard the pairwise similarities as edge flows in a Markov random walk and study the properties of the eigenvectors and values of the transition matrix.

Gorelick *et al.* [29] characterize the shape of a picture using random walks. For each internal pixel, they calculate the value reflecting the mean time required for a random walker beginning at the pixel to reach the boundary. With the help of these calculated values, they extract many properties of the silhouette, such as its part structure, rough skeleton, local orientation, convex part, and concave part.

Grady *et al.* [30], [31] propose a new algorithm for performing multi-label and interactive image segmentation. The interactive image segmentation means that the user has to label some pixels in the image manually. The algorithm can calculate the probability of the random walker which starts from an unlabeled pixel reaching the pre-labeled pixels. Therefore, a good image segmentation arises from the labels of all the pixels by assigning each pixel to a label with the maximum probability. But the algorithm has some problems, of which is that it requires user-specified seeds. To solve the above problem, Grady [32] proposes that combining a prior model into the energy minimization yields an extended random walkers algorithm. It can locate disconnected objects without user-specified labels.

Qiu *et al.* [33] exploit the properties of the commute time to develop a image clustering and segmentation method. By using the discrete Green's function of graphs, they analyze the cuts of the image from commute time. Qiu *et al.* [34] also use commute time to motion track. The main purpose of using commute time as proximity measure is to alleviate the effect of noise on the shape interaction matrix. Commute time is a more robust measure than raw proximity matrix when facing the noise on the shape interaction matrix. They calculate the commute time using the Laplacian eigensystem.

Shen *et al.* [7] propose a new image superpixel segmentation approach using LRW algorithm. The authors initialize the seed positions and run the LRW algorithm on the input image to obtain the probabilities of each pixel. Then the boundaries of initial superpixels are obtained with the help of probabilities and the commute time. The new algorithm can segment the weak boundaries and complicated texture regions very well.

Dong *et al.* [35] present a novel framework based on the sub-Markov random walk for interactive seeded image segmentation. It can be regarded as a traditional random walker with some new auxiliary nodes, that makes the framework more flexible. Under this framework, the authors design a new subRW algorithm with label prior to solve the segmentation problem of objects with thin and elongated parts.

Li *et al.* [36] propose a visual tracking algorithm based on random walks on two graph models. Nodes and edges in the graph denote superpixels and the relationships between superpixels, respectively. They incorporate the structural information between target parts and similarity measurements into a structural model to improve the tracking accuracy. It is the first time that visual tracking is treated as Markov random walks [36].

E. Semi-Supervised Learning

Semi-supervised learning is a class of machine learning tasks and techniques. It uses a small amount of labeled data and a large amount of unlabeled data for training. Due to the less human effort and high precision, it is meaningful both in theory and in practice [37].

Zhu *et al.* [38] present a new approach of semi-supervised learning based on random walks. They do classification task in continuous state space rather than in the discrete label set. The intuition of the approach is that the data points should be labeled as same as their neighbors. The authors' strategy is to employ a real-valued function $f : V \to R$ on graph G and then to assign labels based on f. The function f provides a consistent probabilistic semantics. It is the basis of this semi-supervised classification method. The promising result has shown that the approach can improve the accuracy of classification by exploiting the structure of unlabeled data.

Szummer *et al.* [39] find that the partially labeled data may be in the sub-manifold space. The authors hope the measure can incorporate the structure of manifold and the density. Based on these considerations, they present a Markov random walk model to classify the data. The research [40] shows how to change the distance matrix into a Markov process and helps a lot with the construction of graph.

They classify node j with the label c when c maximizes the following formula:

$$c_k = \text{argmax}_c \sum_i P(c|i) P_{0|t}(i|j) \tag{41}$$

where $P_{0|t}(i|j)$ is the probability of a random walk from node i to node k, $P(c|i)$ can be estimated by two techniques: maximum likelihood with Expectation Maximization (EM) and maximum margin subject to constraints.

The parameter t in this approach is also important. It denotes the number of transitions which determines the smoothness of a random walk.

However, the choice of t can be tricky and subjective. To overcome this problem, Azran [41] presents the rendezvous algorithm.

The author also represents the data points as nodes of a graph and employ the random walk view to do classification.

Different from the work of Szummer *et al.* [39], the labeled points in rendezvous algorithm don't propagate, but absorb the states of the random walk. The probability that each unlabeled data is absorbed by different labeled points can be used to derive the distribution as the transition steps increase to infinity.

Hence, the rendezvous algorithm doesn't bother to choose a good value of the parameter t.

F. Network Embedding

Network embedding can encode nodes or edges to lower dimensional vector representations and keep network structure [73]. It is a promising direction for network representation and can be used to improve performance for downstream tasks.

Inspired by Word2Vec [74], Perozzi *et al.* [42] propose a new approach called DeepWalk for learning latent vector representations of nodes in a network. DeepWalk uses truncated random walks to extract local information of nodes. Analogy with language models, the sequence of nodes resulted from random walks can be regarded as sentences and the nodes in the network is equal to the words in vocabulary.

Perozzi also extend *SkipGram* and *Hierarchical Softmax* from Word2Vec to DeepWalk for reducing computation and speeding up convergence rate.

Grover *et al.* [43] find that current feature learning methods cannot adequately express the diversity of connection patterns in the network. Thus they propose Node2Vec which is a novel algorithmic framework for learning feature representations of nodes. It presents a flexible neighborhood sampling strategy based on random walks. In previous methods, considering a random walk that just walks from node v_i to node v_j, the single step transition probability of a random walk from node v_j to node v_k is based on the weight w_{jk} of edge (j,k). But node2Vec denotes the unnormalized transition probability p_{jk} as $p_{jk} = \alpha_{pq}(i,k)w_{jk}$, and

$$\alpha_{pq}(i,k) = \begin{cases} \frac{1}{p} & \text{if } d_{ik} = 0 \\ 1 & \text{if } d_{ik} = 1 \\ \frac{1}{q} & \text{if } d_{ik} = 2 \end{cases} \tag{42}$$

where d_{ik} is the length of shortest path between nodes v_i and v_k. p is the return parameter which controls the likelihood of revisiting a node in the walk. q is the in-out parameter [43].

Actually, the definition of α_{pq} can be regarded as a tradeoff between *breadth-first sampling* (BFS) and *depth-first sampling* (DFS) [43].

G. Element Distinctness

Element distinctness problem is to tell whether all the elements in a given sequence are distinct. More precisely, it can be described as "given a series of numbers $x_1, x_2, \ldots, x_N \in [M]$, are there $x_i, x_j \in M$ and $i \neq j$ such that $x_i = x_j$ [50]?" There is a simple classical algorithm to solve this problem with $Nlog(N) + O(N)$ comparisons. Buhrman *et al.* [50] present a quantum algorithm to speedup. Their algorithm gives an upper bound of computation cost $O(N^{3/4}log(N))$.

Ambainis [51] improves the quantum way to solve element distinctness with $O(N^{2/3})$ comparisons.The intuition of this optimal quantum algorithm is to construct a graph, and transform the element distinctness problem of finding a marked vertex in the graph. In order to search marked vertex efficiently, the author improves the Grover's quantum search algorithm [11], [75]. The author reuses the information that queries before, and search a marked vertex with $O(N^{2/3})$ comparisons instead of $O(N)$ comparisons in Grover's search algorithm. For the extensions of this algorithms, the authors propose that if we want to find k numbers which are equal in $x_1, x_2, \ldots, x_N$, we can get a quantum walk based algorithms with $O(N^{k/(k+1)})$ queries.

V. Open Issues

In this part, we will introduce some major problems of random walks. Most of them are caused by the growing real-world networks.

A. Speed of Random Walk Algorithms

The time complexity of random walk graph kernel is at least $O(n^3)$ or $O(m^2)$ for graph with n nodes and m edges [76]. In an artificially generated graph, this time complexity is acceptable. But it is a disaster on a real-world network since the number of nodes and edges is huge. It is also a challenge for random walk models of which the time complexity are at least $O(n^2)$. Researchers are already dealing with the problem. Kang *et al.* [76] propose ARK graph kernels with time complexity $O(n^2)$ or $O(m)$. There is a prerequisite for this graph kernel. The graph must have lower intrinsic ranks than the order of the graph.

Tong *et al.* [58] also realize the speed problem in random walk with restart. The random walk with restart algorithm is slow in query time or prohibitive on storage space.

The authors exploit the block-wise community-like structure and the linear correlations of the adjacency matrix of real-world networks. With these two properties, the authors devise B_LIN to make the random walk with restart faster. This approach not only saves a lot of storage space and computing time, but also preserves good performance.

As we can see, the main idea to cope up with speed of random walk algorithms is to obtain approximate computation instead of accurate computation. We still require more accurate approximate algorithms for random walks.

B. Problem of Main-Memory Volume

All the fast random walk graph kernels or algorithms are under the consumption that the whole graph can be fit in the main-memory. But with the rapid growth in the scale of the network, this condition can't be satisfied any more. One of the solutions is to divide the graph into several clusters.

There are studies providing some approaches for graph partition and clustering on giant network [77], [78]. One of the most popular method is METIS [78]. Since more and more researchers pay attention to the giant network problem, there is a more effective clustering algorithm for graph clustering and a better method to apply random walks on giant network with external memory [79]. The author calls the clustering method RWDISK. RWDISK has been proved to be a better way for graph partition on several famous datasets such as Digital Bibliography & Library Project (DBLP), Citeseer. But these methods still have an unacceptable time latency with respect to enormous graph. There are two ways to solve this issue, partition and using external memory.

C. Computation of Hitting and Commute Time

As we have mentioned, proximity measures play an important role in network analysis and beyond. The complexity of computing commute time is $O(n^3)$ which is prohibitive in large real-world graphs. There are some approximations of commute time to reduce the complexity [15], [24]. But we should be careful about these approximate approaches. They can't represent the structure of large real-world graphs or show the connectivity of nodes in large graphs.

Luxburg *et al.* [80] have shown that commute time can be approximated by simple formula with high accuracy when random geometric graphs (k-nearest neighbor graphs, ϵ-graphs, and Gaussian similarity graph) are large enough. More specifically, commute time H_{uv} can be represented by $1/d_u + 1/d_v$ in large graphs where d_u and d_v denote the degree of vertex u and vertex v, respectively. Thus, the approximations only consider the local density of two nodes rather than the structure information of the whole graph. The authors give two strategies to prove the result: one based on the flow argument of the electric network, and the other based on spectral argument. Both of them prove that approximations of commute time don't take into account any global properties of large graphs. In that case, the effectiveness of approximated commute time is doubtful. The computation of commute time in large graph is still a challenge.

VI. Conclusion

In this paper, we have presented an overview of random walks from the perspective of computer science, including classical random walks and quantum walks. We first introduce the basic knowledge and some algorithms of classical random walks and quantum walks in a comprehensible way. The typical random walk algorithms are PageRank and its variants. RWR and LRW are also reviewed. They are time-consuming when applied to large real-world graphs. Some methods are developed to accelerate convergence, such as Quadratic Extrapolation, B_LIN, K-dash. Then two types of algorithms based on quantum walks are discussed: continuous quantum walk based algorithms and discrete quantum walk based algorithms. We make comparisons between classical random walks and quantum walks and find that with the development of quantum computation, the quantum view of random walks accelerates the computation of random walk algorithms significantly.

Random walks can be used to calculate the proximity between two nodes and extract the network topology. It has been proved that random walks play an important role in many scenarios. We explore the applications of random walks in the field of computer science including collaborative filtering, computer vision, network embedding, and so on. Many problems with existing random walk based algorithms are caused by giant networks, such as slow convergence speed, insufficient storage capacity. Further research in this field would be of great help in theoretical and practical application of random walks.

References

[1] K. Pearson, "The problem of the random walk," *Nature*, vol. 72, no. 1867, 1905, Art. no. 342.

[2] F. Spitzer, *Principles of Random Walk*, vol. 34. Berlin, Germany: Springer, 2013.

[3] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," Stanford InfoLab, Tech. Rep. SIDL-WP-1999-0120, Nov. 1999. [Online]. Available: http://ilpubs.stanford.edu:8090/422/

[4] D. Fogaras, B. Rácz, K. Csalogány, and T. Sarlós, "Towards scaling fully personalized PageRank: Algorithms, lower bounds, and experiments," *Internet Math.*, vol. 2, no. 3, pp. 333–358, 2005.

[5] T. H. Haveliwala, "Topic-sensitive PageRank: A context-sensitive ranking algorithm for web search," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 4, pp. 784–796, Jul./Aug. 2003.

[6] J.-Y. Pan, H.-J. Yang, C. Faloutsos, and P. Duygulu, "Automatic multimedia cross-modal correlation discovery," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2004, pp. 653–658.

[7] J. Shen, Y. Du, W. Wang, and X. Li, "Lazy random walks for superpixel segmentation," *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1451–1462, Apr. 2014.

[8] Y. Aharonov, L. Davidovich, and N. Zagury, "Quantum random walks," *Phys. Rev. A*, vol. 48, no. 2, 1993, Art. no. 1687.

[9] A. M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann, and D. A. Spielman, "Exponential algorithmic speedup by a quantum walk," in *Proc. 35th Annu. ACM Symp. Theory Comput.*, 2003, pp. 59–68.

[10] E. Farhi and S. Gutmann, "Quantum computation and decision trees," *Phys. Rev. A*, vol. 58, no. 2, 1998, Art. no. 915.

[11] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. 28th Ann. ACM Symp. Theory Comput.*, 1996, pp. 212–219.

[12] N. Shenvi, J. Kempe, and K. B. Whaley, "Quantum random-walk search algorithm," *Phys. Rev. A*, vol. 67, no. 5, 2003, Art. no. 052307.

[13] J. Kempe, "Quantum random walks: An introductory overview," *Contemporary Phys.*, vol. 44, no. 4, pp. 307–327, 2003.

[14] G. Adomavicius and A. Tuzhilin, "Recommendation technologies: Survey of current methods and possible extensions," *Inf. Syst. Working Papers Ser.*, Jan. 2003.

[15] M. Brand, "A random walks perspective on maximizing satisfaction and profit," in *Proc. SIAM Int. Conf. Data Mining*, 2005, pp. 12–19.

[16] F. Fouss, A. Pirotte, and M. Saerens, "A novel way of computing similarities between nodes of a graph, with application to collaborative recommendation," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell.*, 2005, pp. 550–556.

[17] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, "Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 355–369, Mar. 2007.

[18] H. Yildirim and M. S. Krishnamoorthy, "A random walk method for alleviating the sparsity problem in collaborative filtering," in *Proc. ACM Conf. Recommender Syst.*, 2008, pp. 131–138.

[19] M. Jamali and M. Ester, "Trustwalker: A random walk model for combining trust-based and item-based recommendation," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 397–406.

[20] M. Gori, A. Pucci, V. Roma, and I. Siena, "Itemrank: A random-walk based scoring algorithm for recommender engines," in *Proc. Int. Joint Conf. Artif. Intell.*, 2007, vol. 7, pp. 2766–2771.

[21] M. Gori and A. Pucci, "Research paper recommender systems: A random-walk based approach," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell.*, 2006, pp. 778–781.

[22] F. Xia, H. Liu, I. Lee, and L. Cao, "Scientific article recommendation: Exploiting common author relations and historical preferences," *IEEE Trans. Big Data*, vol. 2, no. 2, pp. 101–112, Jun. 2016.

[23] F. Xia, Z. Chen, W. Wang, J. Li, and L. T. Yang, "MVCWalker: Random walk-based most valuable collaborators recommendation exploiting academic factors," *IEEE Trans. Emerg. Topics Comput.*, vol. 2, no. 3, pp. 364–375, Sep. 2014.

[24] P. Sarkar and A. Moore, "A tractable approach to finding closest truncated-commute-time neighbors in large graphs," 2012, *arXiv:1206.5259*.

[25] W. Liu and L. Lü, "Link prediction based on local random walk," *EPL (Europhysics Lett.)*, vol. 89, no. 5, 2010, Art. no. 58007.

[26] L. Backstrom and J. Leskovec, "Supervised random walks: Predicting and recommending links in social networks," in *Proc. 4th ACM Int. Conf. Web Search Data Mining*, 2011, pp. 635–644.

[27] Y. Liu, X. Zeng, Z. He, and Q. Zou, "Inferring microRNA-disease associations by random walk on a heterogeneous network with multiple data sources," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 14, no. 4, pp. 905–915, Jul./Aug. 2017.

[28] M. Meila and J. Shi, "A random walks view of spectral segmentation," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2001, pp. 177–182.

[29] L. Gorelick, M. Galun, E. Sharon, R. Basri, and A. Brandt, "Shape representation and classification using the Poisson equation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 1991–2005, Dec. 2006.

[30] L. Grady, "Random walks for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 11, pp. 1768–1783, Nov. 2006.

[31] L. Grady and G. Funka-Lea, "Multi-label image segmentation for medical applications based on graph-theoretic electrical potentials," in *Computer Vision and Mathematical Methods in Medical and Biomedical Image Analysis*. Berlin, Germany: Springer, 2004, pp. 230–245.

[32] L. Grady, "Multilabel random walker image segmentation using prior models," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2005, vol. 1, pp. 763–770.

[33] H. Qiu and E. R. Hancock, "Image segmentation using commute times," in *Proc. Brit. Mach. Vision Conf.*, 2005, pp. 929–938.

[34] H. Qiu and E. R. Hancock, "Robust multi-body motion tracking using commute time clustering," in *Proc. Eur. Conf. Comput. Vision*, 2006, pp. 160–173.

[35] X. Dong, J. Shen, L. Shao, and L. Van Gool, "Sub-Markov random walk for image segmentation," *IEEE Trans. Image Process.*, vol. 25, no. 2, pp. 516–527, Feb. 2016.

[36] X. Li, Z. Han, L. Wang, and H. Lu, "Visual tracking via random walks on graph model," *IEEE Trans. Cybern.*, vol. 46, no. 9, pp. 2144–2155, Sep. 2016.

[37] X. Zhu, "Semi-supervised learning literature survey," *Comput. Sci., Univ. Wisconsin–Madison*, Madison, WI, USA, vol. 2, Jul. 2008.

[38] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. 20th Int. Conf. Mach. Learn.*, 2003, pp. 912–919.

[39] M. Szummer and T. Jaakkola, "Partially labeled classification with Markov random walks," in *Proc. Advances Neural Inf. Process. Syst.*, 2002, pp. 945–952.

[40] N. Tishby and N. Slonim, "Data clustering by Markovian relaxation and the information bottleneck method," in *Proc. Advances Neural Inf. Process. Syst.*, 2001, pp. 640–646.

[41] A. Azran, "The rendezvous algorithm: Multiclass semi-supervised learning with Markov random walks," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 49–56.

[42] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 701–710.

[43] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 855–864.

[44] P. Sarkar and A. W. Moore, "Random walks in social networks and their applications: A survey," in *Social Network Data Analytics*. Berlin, Germany: Springer, 2011, pp. 43–77.

[45] L. Lovász *et al.*, "Random walks on graphs: A survey," *Combinatorics, Paul Erdos Eighty*, vol. 2, no. 1, pp. 1–46, 1993.

[46] P. Sarkar and A. W. Moore, "Random walks in social networks and their applications: a survey," *Social Network Data Analytics*, Boston, MA, USA: Springer, 2011, pp. 43–77.

[47] D. R. Amancio, F. N. Silva, and L. D. F. Costa, "Concentric network symmetry grasps authors' styles in word adjacency networks," *EPL (Europhys. Lett.)*, vol. 110, no. 6, 2015, Art. no. 68001.

[48] D. R. Amancio, "Comparing the topological properties of real and artificially generated scientific manuscripts," *Scientometrics*, vol. 105, no. 3, pp. 1763–1779, 2015.

[49] H. F. de Arruda, F. N. Silva, L. D. F. Costa, and D. R. Amancio, "Knowledge acquisition: A complex networks approach," *Inf. Sci.*, vol. 421, pp. 154–166, 2017.

[50] H. Buhrman *et al.*, "Quantum algorithms for element distinctness," in *Proc. 16th Annu. IEEE Conf. Comput. Complexity*, 2001, pp. 131–137.

[51] A. Ambainis, "Quantum walk algorithm for element distinctness," *SIAM J. Comput.*, vol. 37, no. 1, pp. 210–239, 2007.

[52] P. G. Doyle and J. L. Snell, *Random Walks and Electric Networks*, vol. 22. Washington, DC, USA: Mathematical Association of America, 1984.

[53] S. E. Venegas-Andraca, "Quantum walks: A comprehensive review," *Quantum Inf. Process.*, vol. 11, no. 5, pp. 1015–1106, 2012.

[54] T. A. Brun, H. A. Carteret, and A. Ambainis, "Quantum to classical transition for random walks," *Phys. Rev. Lett.*, vol. 91, no. 13, 2003, Art. no. 130602.

[55] F. W. Strauch, "Connecting the discrete-and continuous-time quantum walks," *Phys. Rev. A*, vol. 74, no. 3, 2006, Art. no. 030301.

[56] S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub, "Extrapolation methods for accelerating pagerank computations," in *Proc. 12th Int. Conf. World Wide Web*, 2003, pp. 261–270.

[57] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos, "Neighborhood formation and anomaly detection in bipartite graphs," in *Proc. 5th IEEE Int. Conf. Data Mining*, 2005, pp. 418–425.

[58] H. Tong, C. Faloutsos, and J.-Y. Pan, "Fast random walk with restart and its applications," in *Proc. 6th Int. Conf. Data Mining*, 2006, pp. 613–622.

[59] Y. Fujiwara, M. Nakatsuji, M. Onizuka, and M. Kitsuregawa, "Fast and exact top-k search for random walk with restart," *Proc. VLDB Endowment*, vol. 5, no. 5, pp. 442–453, 2012.

[60] G. D. Paparo and M. Martin-Delgado, "Google in a quantum network," *Sci. Rep.*, vol. 2, 2012, Art. no. 444.

[61] G. D. Paparo, M. Müller, F. Comellas, and M. A. Martin-Delgado, "Quantum Google in a complex network," *Sci. Rep.*, vol. 3, 2013, Art. no. 2773.

[62] L. Xiao *et al.*, "Observation of topological edge states in parity–time-symmetric quantum walks," *Nature Phys.*, vol. 13, no. 11, 2017, Art. no. 1117.

[63] C. Cedzich, T. Geib, F. Grünbaum, L. Velázquez, A. Werner, and R. Werner, "Quantum walks: Schur functions meet symmetry protected topological phases," 2019, *arXiv:1903.07494*.

[64] K. Wang *et al.*, "Simulating dynamic quantum phase transitions in photonic quantum walks," *Phys. Rev. Lett.*, vol. 122, no. 2, 2019, Art. no. 020501.

[65] Z. Hou *et al.*, "Deterministic realization of collective measurements via photonic quantum walks," *Nature Commun.*, vol. 9, no. 1, 2018, Art. no. 1414.

[66] M. Deshpande and G. Karypis, "Item-based top-n recommendation algorithms," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 143–177, 2004.

[67] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An open architecture for collaborative filtering of netnews," in *Proc. ACM Conf. Comput. Supported Cooperat. Work*, 1994, pp. 175–186.

[68] J. A. Golbeck, "Computing and applying trust in web-based social networks," Ph.D. dissertation, 2005.

[69] R. Andersen *et al.*, "Trust-based recommendation systems: An axiomatic approach," in *Proc. 17th Int. Conf. World Wide Web*, 2008, pp. 199–208.

[70] L. Lü and T. Zhou, "Link prediction in complex networks: A survey," *Physica A, Statist. Mech. Appl.*, vol. 390, no. 6, pp. 1150–1170, 2011.

[71] M. Al Hasan and M. J. Zaki, "A survey of link prediction in social networks," in *Social Network Data Analytics*. Berlin, Germany: Springer, 2011, pp. 243–275.

[72] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *J. Amer. Soc. Inf. Sci. Technol.*, vol. 58, no. 7, pp. 1019–1031, 2007.

[73] P. Cui, X. Wang, J. Pei, and W. Zhu, "A survey on network embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 5, pp. 833–852, May 2019.

[74] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*.

[75] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp, "Quantum amplitude amplification and estimation," *Contemporary Math.*, vol. 305, pp. 53–74, 2002.

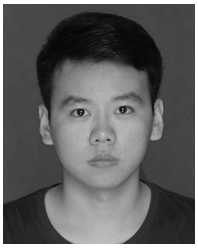
[76] U. Kang, H. Tong, and J. Sun, "Fast random walk graph kernel," in *Proc. SIAM Int. Conf. Data Mining*, 2012, pp. 828–838.

[77] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM J. Sci. Comput.*, vol. 20, no. 1, pp. 359–392, 1998.

[78] G. Karypis and V. Kumar, "A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices," Dept. Comput. Sci. Eng., Army HPC Res. Center, Univ. Minnesota, Minneapolis, MN, USA, 1998.
[79] P. Sarkar and A. W. Moore, "Fast nearest-neighbor search in disk-resident graphs," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2010, pp. 513–522.
[80] U. Von Luxburg, A. Radl, and M. Hein, "Hitting and commute times in large graphs are often misleading," 2010, *arXiv:1003.1266*.

Feng Xia (M'07–SM'12) received the B.Sc. and Ph.D. degrees from Zhejiang University, Hangzhou, China. He is currently an Associate Professor of Data Science, Federation University Australia, Ballarat, VIC, Australia, and on leave from Dalian University of Technology, Dalian, China, where he is a Full Professor. He has authored or coauthored two books and more than 200 scientific papers in international journals and conferences. His research interests include data science, big data, knowledge engineering, social computing, and systems engineering. He is a Senior Member of ACM.

Jiaying Liu received the B.Sc. degree in software engineering from the Dalian University of Technology, Dalian, China, in 2016. She is currently working toward the Ph.D. degree with the School of Software, Dalian University of Technology, Dalian. Her research interests include data science, big scholarly data, and social network analysis.



Hansong Nie received the B.Sc. degree in electronic information engineering from Dalian Maritime University, Dalian, China, in 2018. He is currently working toward the master's degree with the School of Software, Dalian University of Technology, Dalian. His research interests include big scholarly data, social network analysis, and science of success.

Yonghao Fu is currently studying as a Senior Student with Software School, Dalian University of Technology, Dalian, China. He will work toward the master's degree in his research interests, which include data mining and natural language processing.

Liangtian Wan (M'15) received the B.Sc. and Ph.D. degrees from the College of Information and Communication Engineering, Harbin Engineering University, Harbin, China, in 2011 and 2015, respectively. From October 2015 to April 2017, he was a Research Fellow of School of Electrical and Electrical Engineering, Nanyang Technological University, Singapore. He is currently an Associate Professor of School of Software, Dalian University of Technology, Dalian, China. He is the author of more than 40 papers published in related international conference proceedings and journals. His current research interests include social network analysis and mining, big data, array signal processing, wireless sensor networks, and compressive sensing. He has been serving as an Associate Editor for the IEEE ACCESS and *Journal of Information Processing Systems*.

Xiangjie Kong (M'13–SM'17) received the B.Sc. and Ph.D. degrees from Zhejiang University, Hangzhou, China. He is currently an Associate Professor with the School of Software, Dalian University of Technology, Dalian, China. He was a (Guest) Editor of several international journals, Workshop Chair or PC Member of a number of conferences. He has authored or coauthored more than 100 scientific papers in international journals and conferences (with more than 70 indexed by ISI SCIE). His research interests include computational social science, data science, and mobile computing. He is a Senior Member of CCF, and a Member of ACM.