Comput Optim Appl (2015) 61:689–711
DOI 10.1007/s10589-015-9727-7

Potential induced random teleportation on finite graphs

Shui-Nee Chow · Xiaojing Ye · Haomin Zhou

Received: 22 April 2014 / Published online: 20 January 2015

Abstract We propose and analyze a potential induced random walk and its modification called random teleportation on finite graphs. The transition probability is determined by the gaps between potential values of adjacent and teleportation nodes. We show that the steady state of this process has a number of desirable properties. We present a continuous time analogue of the random walk and teleportation, and derive the lower bound on the order of its exponential convergence rate to stationary distribution. The efficiency of proposed random teleportation in search of global potential minimum on graphs and node ranking are demonstrated by numerical tests. Moreover, we discuss the condition of graphs and potential distributions for which the proposed approach may work inefficiently, and introduce the intermittent diffusion strategy to overcome the problem and improve the practical performance.

Keywords Random walk · Random teleportation · Potential · Intermittent diffusion · Gibbs distribution · Metropolis–Hastings algorithm · Finite graphs

1 Introduction

In this paper, we propose a generic optimization algorithm on finite graphs where each node is associated with a given potential value. The purpose is to find the node

S.-N. Chow · H. Zhou
School of Mathematics, Georgia Institute of Technology, Atlanta, GA 30332, USA
e-mail: chow@math.gatech.edu

H. Zhou
e-mail: hmzhou@math.gatech.edu

X. Ye (✉)
Department of Mathematics and Statistics, Georgia State University, Atlanta, GA 30303, USA
e-mail: xye@gsu.edu

which has the lowest potential value. The graphs of interests have extensively large size and hence exhaustive search of the optimal node is either impossible or extremely cost ineffective. Our method, which is an extension of random walk and teleportation, considers a randomized search on the graph with transition probability from node to node based on potential value changes. Recall that in classical settings, random walk on a graph defines a stochastic process in the discrete state space of the nodes. A random walk takes random successive movements on the graph as follows: starting from a node, select one of its neighbor nodes randomly and move there, and then choose a neighbor of this node at random and move again, etc. If the consecutive step is not restricted to adjacent nodes but can be some much "farther" (non-adjacent) nodes, random walk becomes random teleportation. Namely, random teleportation defines a stochastic process on the graph such that one can be transferred or teleported at random to a node which is not a neighbor of the current node in each step.

In the literature, random walk and teleportation on graphs or networks have received considerable attention in recent years due to their tremendous applications in a variety of scientific areas [3,4,13,22,28,31,33]. For instance, consider an electric network as a graph where each edge has unit or variable resistances, then characteristics of random walk, such as access time and commute time, reflect the properties of electric current flowing through the network [10,12,13,30]. In graph partition problem, the dynamics of random walk can reveal structures of complex networks and provide hints for optimal partitions [27,34]. In statistical simulations, random walk serves as a feasible algorithm for sampling from probability distributions [5,9,26]. More recently, random walk has become a ubiquitous tool to explore the properties of extremely large networks such as those in web search and social networks on the Internet [15,19,20,28]. For theory of classical random walk, such as hitting and commute time, covering time, and mixing rate, see, for example, [1,4,8,21,22]. For applications in contemporary sciences and technology including those mentioned above, see [3,11,18,28,33] and references therein.

In its classical settings, behaviors and properties of random walk are dependent only on graph structures such as node degrees (number of adjacent nodes). However, in many real-world applications, there are usually potential values associated to the nodes on graphs. For instance, the possible states of certain material can be modeled as nodes on a graph, and the potential of each node has physical meaning such as the energy configuration of the state represented by this node, [31]. In this case, it is of interests to find the optimal state (node) among extensively many possible ones such that the material is most stable or has best strength. In social networks, nodes and edges indicate people and their relationships, and the potential value of a node can be a virtual property that describes the importance or influence of an individual in the network [2,17]. Then we need to find the most influential individuals in a huge network quickly. For web search problem, a node is a webpage on the Internet and the potential measures how closely the webpage is related to a given search keyword or criterion, etc. In these cases, it is often of interests to locate the nodes with globally extremal potential values, or to provide a ranking of nodes according to their potential values and their connectivities to other nodes. Although in theory the graph structure and node potential distribution can be revealed by the eigensystem of the stochastic matrix of random walk, it is usually intractable to directly compute such systems numerically

due to the exponentially large sizes of graphs and networks in real-world applications. On the other hand, random walk and teleportation method, if properly designed, is feasible and usually efficient for such global optimization and ranking problems on large graphs, since it dose not require complete exploration of the entire graph but still can return optimal solution with high probability.

The new random walk and teleportation algorithm proposed in this paper is closely related to the Metropolis algorithm. The Metropolis or Metropolis–Hastings algorithm [16,24] is one of the earliest random walk methods and can be used to solve large scale optimization problem on graphs where node potential values are associated. In the most basic setting, suppose that one is currently at node x with potential $\phi(x) \in \mathbf{R}$, the Metropolis–Hastings algorithm first selects another node y randomly with uniform distribution. If $\phi(y) \leq \phi(x)$ then accept y as the next step automatically, otherwise accept y with probability $e^{-(\phi(y)-\phi(x))/\beta}$ for some prescribed parameter $\beta > 0$ which is usually referred to as temperature. Under this formulation, the stationary probability distribution of this walk is the Gibbs distribution $\pi(x) \propto e^{-\phi(x)/\beta}$. More generally, for any target stationary distribution π, the Metropolis–Hastings algorithm can convert the stochastic matrix $K(x, y)$, the probability of moving to y from x on the graph, of a base Markov chain to a stochastic matrix $M(x, y)$ of a π-reversible chain [6]. Hence, the Metropolis–Hastings algorithm can be used to construct a feasible chain such that, at the steady state, one has much larger chance to attain x of globally minimal potential since $\phi(x) < \phi(y)$ and the Gibbs distribution $\pi(x) \gg \pi(y)$ for any non-optimal node y if β is small enough. This property provides a theoretical foundation for finding optimal node on a large graph based on samplings.

By taking a closer look, we can see that the merit of Metropolis–Hastings algorithm is in the possibility of accepting a node with larger potential value and hence can guide the walker escape from local minimum. On the contrary, gradient or greedy method always selects the neighbor node of minimal potential and hence can get stuck at a local optimum easily. However, the Metropolis–Hastings algorithm does not distinguish between nodes as long as they have potential values no larger than that of the current node. This can be quite inefficient when the walker is already close to a global optimum, where it is desirable to go steepest descent direction. Therefore, the main purpose of this paper is to design a new random walk and teleportation algorithm that incorporates the merits of the Metropolis–Hastings algorithm and the gradient algorithm such that the walker can escape from local minima with moderate probability and reach the global optimum more easily.

In this paper, we also introduce an intermittent strategy to overcome the issue of the restrictive requirement on the noise or "temperature" parameter β. The choice of noise parameter β has been a major concern of sampling-based optimization algorithms on graphs. This parameter behaves as noise added to gradient descent (greedy search) method. Noise yields an analogue to diffusion as Brownian motion introduced to a deterministic system in continuous setting. To increase the ratio of probabilities of reaching global minimizer and other nodes, the noise β needs to be small in both of the Metropolis–Hastings algorithm and the proposed random walk and teleportation. However, small β yields slow convergence and hence is not efficient in practice. To overcome this difficulty, the simulated annealing adopts the Metropolis–Hastings algorithm with slowly decreasing $\beta \to 0$ [23,29,32]. The name annealing emulates

the physical process where a solid is slowly cooled down such that the structure is frozen at minimum energy configuration. It is obvious that dynamically decreasing β is critical to annealing efficiency. For a survey of existing cooling strategies, see [25]. The similar annealing approach can also be used for random walk and teleportation. In general the cooling rate is at logarithmic rate in time to guarantee convergence [14], and hence can be very inefficient in general. As an alternative, the intermittent diffusion strategy introduced in this paper switches β off periodically followed by greedy search. We show in theory the effectiveness of intermittent diffusion in this paper, and justify its practical performance in global optimization.

The remainder of this paper is organized as follows. In the next section we propose the potential induced random walk and random teleportation. We provide a comparison of the Metropolis–Hastings algorithm and the proposed random walk to reveal their major differences. In Sect. 3, we provide a continuous analogue of the random walk, and prove the convergence properties of the proposed random walk and teleportation. In Sect. 4, we introduce the intermittent diffusion strategy and justify its effectiveness in search of global minimum theoretically. Experimental results of global minimum search and node ranking on several graphs, effects of intermittent diffusion, and evaluations are presented in Sect. 5.

2 Proposed algorithm

2.1 Potential induced random walk

Let $G = (V, E)$ be a finite non-bipartite undirected graph with node set V and edge set E. The edge $e(x, y) = 1$ if edge $(x, y) \in E$ and 0 otherwise. Without likelihood of confusion, we also use notation $E = [e(x, y)]_{x,y\in V}$ as the adjacency matrix. Let $\phi(x) \in \mathbf{R}$ be the potential of node $x \in V$. In the remainder of this paper, $|V|$ is the total number of nodes in V, and $\mathcal{N}_x$ denotes the neighborhood of node x (including x, i.e. $e(x, x) = 1$) as follows,

$$\mathcal{N}_x := \{y \in V : e(x, y) = 1\}. \tag{2.1}$$

According to the potential function ϕ defined on V, we can further partition the neighborhood $\mathcal{N}_x$ to two disjoint subsets $\mathcal{N}_x^-$ and $\mathcal{N}_x^+$ defined by

$$\mathcal{N}_x^- = \{y \in \mathcal{N}_x : \phi(y) \le \phi(x)\} \text{ and } \mathcal{N}_x^+ = \{y \in \mathcal{N}_x : \phi(y) > \phi(x)\}. \tag{2.2}$$

Using the notation introduced above, we propose a random walk corresponding to Markov chain with transition probability

$$M(x, y) \propto e^{-(\phi(y)-\phi(x))/\beta}, \quad y \in \mathcal{N}_x. \tag{2.3}$$

More precisely, the algorithm proceeds as in Algorithm 1.

Algorithm 1 Potential induced random walk on graph

Given $G = (V, E)$ and potential $\phi : V \to \mathbf{R}$. Set $\beta > 0$ and start from initial node x.
repeat
Move to $y \in \mathcal{N}_x$ with probability

$$\frac{e^{-(\phi(y)-\phi(x))/\beta}}{\sum_{z\in\mathcal{N}_x} e^{-(\phi(z)-\phi(x))/\beta}}. \tag{2.4}$$

until Stopping criterion is satisfied.

Table 1 Comparison of unnormalized transition probability $M(x, y)$ of the Metropolis–Hastings algorithm and the Algorithm 1

	Metropolis–Hastings	Algorithm 1
$M(x, y) \propto$	$\min\left(1, e^{-(\phi(y)-\phi(x))/\beta}\right)$	$e^{-(\phi(y)-\phi(x))/\beta}$

Here $x \neq y \in \mathcal{N}_x$

Remark 2.1 As $\beta \to 0$, the proposed method is essentially the steepest descent gradient or greedy algorithm, i.e. in every step one moves to the node of lowest potential value in the neighborhood.

Remark 2.2 In the case of constant potential, i.e. $\phi \equiv$ constant, the proposed method reduces to classical random walk on graph. Note that there is slight difference to the definitions of classical random walk that staying at the current location is possible in Algorithm 1.

This proposed method is similar to the Metropolis–Hastings algorithm in the sense that it allows movement to nodes of larger potential values with positive probabilities. However, it also differs from the conventional Metropolis–Hastings algorithm as the proposed method distinguishes between the descent directions. Their difference in terms of transition probability is summarized in Table 1. From Table 1, one can readily see that the difference: in the proposed random walk Algorithm 1, X is more likely to take y if $\phi(y)$ is smaller; On the contrary, there is no preference among the nodes y as long as $\phi(y) \leq \phi(x)$ in Metropolis–Hastings algorithm. Therefore, the Algorithm 1 is expected to move to the global optimum faster than the Metropolis–Hastings if it is nearby. We remark here that the transition probability $M(x, y)$ of Metropolis–Hastings algorithm in Table 1 corresponds to a modified local version and its stationary distribution is not exactly Gibbs distribution $\pi(x) \propto e^{-\phi(x)/\beta}$ unless node degrees are taken into account.

2.2 Potential induced random teleportation

Classical random walk restricts the movement to neighbor nodes only, which can be quite inefficient in exploring large graphs. For instance, an arbitrary initial guess can be far from the global optimum, and hence taking consecutive steps to neighbor nodes only on the graph towards the global optimum requires long time. Recall that

the Metropolis–Hastings algorithm would have the capability of jumping to any node on the graph, we hence extend the proposed algorithm by adding n nodes randomly selected from the entire graph G to the original neighborhood $\mathcal{N}_x$ (but not eliminate repeated nodes) for consideration, and calculate the transition probability similar as (2.4). The modified algorithm essentially defines a random teleportation procedure and it is summarized in Algorithm 2 as follows.

Algorithm 2 Potential induced random walk and teleportation on graph

Given $G = (V, E)$ and potential $\phi : V \to \mathbf{R}$. Set $\beta > 0$ and start from initial node x.
repeat
Randomly select n nodes $\{y_1, \ldots, y_n\}$ from V (with equal probabilities), and move to $y \in \mathcal{N}_x$ or $y \in \{y_1, \ldots, y_n\}$ with probability

$$\frac{e^{-(\phi(y)-\phi(x))/\beta}}{\sum_{z\in\mathcal{N}_x} e^{-(\phi(z)-\phi(x))/\beta} + \sum_{i=1}^{n} e^{-(\phi(y_i)-\phi(x))/\beta}}. \tag{2.5}$$

until Stopping criterion is satisfied.

Remark 2.3 The teleportation (2.5) enables possibility to move to any nodes on the graph. This essentially converts any given graph to a complete graph where all nodes are connected to each other. Hence the graph structure becomes quite regular. Actually, the Google's PageRank web search algorithm also implements teleportation as the so-called damping factor, where a random walk between linked webpages can be directed to any webpage with a prescribed probability [19].

Based to the presentation of Algorithm 2, the transition probability $M(x, y)$ can be written as follows,

$$M(x, y) \propto \begin{cases} (1 + r)e^{-(\phi(y)-\phi(x))/\beta} & \text{if } y \in \mathcal{N}_x \\ re^{-(\phi(y)-\phi(x))/\beta} & \text{otherwise} \end{cases} \tag{2.6}$$

where $r = n/|V| \in [0, 1]$ is the portion of nodes that are considered as possible destinations in addition to nodes in $\mathcal{N}_x$ in each step.

2.3 Stationary distribution

To analyze the probabilistic properties of random walk and teleportation in Algorithms 1 and 2, we first define a set function K_β as follows,

$$K_\beta(W) := \sum_{x\in W} e^{-\phi(x)/\beta}, \tag{2.7}$$

where W is a subset of nodes in V. Then the transition probability M in (2.6) can be written as

$$M(x, y) = \frac{(1_{\mathcal{N}_x}(y) + r)e^{-\phi(y)/\beta}}{K_\beta(\mathcal{N}_x) + rK_\beta(V)} \tag{2.8}$$

where $1_{\mathcal{N}_x}$ is the indicator function of $\mathcal{N}_x$, i.e. $1_{\mathcal{N}_x}(y) = 1$ if $y \in \mathcal{N}_x$ and 0 otherwise. One can easily observe that the analysis includes the case of Algorithm 1 with $r = 0$. According to the detailed balance condition

$$p_x^* M(x, y) = p_y^* M(y, x), \tag{2.9}$$

we obtain the stationary distribution p^* as

$$p_x^* = C(K_\beta(\mathcal{N}_x) + rK_\beta(V))e^{-\phi(x)/\beta}, \tag{2.10}$$

where the normalizing constant C of p_x^* is defined by

$$\begin{aligned} C^{-1} &= \sum_{x\in V}(K_\beta(\mathcal{N}_x) + K_\beta(V))e^{-\phi(x)/\beta} \\ &= \sum_{x\in V} K_\beta(\mathcal{N}_x)e^{-\phi(x)/\beta} + K_\beta(V)\left(\sum_{x\in V} e^{-\phi(x)/\beta}\right) \\ &= \sum_{e(x,y)=1} e^{-(\phi(x)+\phi(y))/\beta} + r\left(\sum_{x\in V} e^{-\phi(x)/\beta}\right)^2 \\ &= q(E + rJ)q^T \end{aligned} \tag{2.11}$$

where $q = [e^{-\phi(x)/\beta}]_{x\in V} \in \mathbf{R}^{|V|}$ is a row vector, $E = [e(x, y)]_{x,y\in V}$ is the adjacency matrix, and J is the matrix of all ones of size $|V|$.

The stationary distribution (2.10) explains how the random teleportation behaves under steady state. For comparison, we recall that the stationary distribution of Metropolis–Hastings algorithm, i.e. the Gibbs distribution, is

$$\pi(x) \propto e^{-\phi(x)/\beta}. \tag{2.12}$$

Compare the ratio of stationary probabilities at any two nodes x and y for the stationary distribution (2.10) and (2.12), we have

$$\frac{p_x^*/p_y^*}{\pi_x/\pi_y} = \frac{K_\beta(\mathcal{N}_x) + rK_\beta(V)}{K_\beta(N_y) + rK_\beta(V)}. \tag{2.13}$$

Let us consider Algorithm 1 with $r = 0$ for example, then the ratio (2.13) reduces to $K_\beta(\mathcal{N}_x)/K_\beta(N_y)$. Note that by definition there is

$$K_\beta(\mathcal{N}_x) = \sum_{z\in\mathcal{N}_x} e^{-\phi(z)/\beta}. \tag{2.14}$$

Therefore, when β is sufficiently small, there is a dominant term $e^{-\phi(x^*)}$ on the right side of (2.14), where x^* has the minimal potential value among nodes in N_x. Hence there is

$$K_\beta(\mathcal{N}_x) \approx e^{-\phi(x^*)}. \tag{2.15}$$

Thus, if $\mathcal{N}_x$ contains the global minimizer (for example if x itself is the global minimizer) but $\mathcal{N}_y$ does not, we have

$$K_\beta(\mathcal{N}_x) \approx e^{-\phi(x^*)} \gg e^{-\phi(y^*)} \approx K_\beta(\mathcal{N}_y), \tag{2.16}$$

and hence there is

$$\frac{p_x^*/p_y^*}{\pi_x/\pi_y} = \frac{K_\beta(\mathcal{N}_x)}{K_\beta(N_y)} \gg 1. \tag{2.17}$$

In other words, the stationary probability distribution p^* (2.10) by Algorithm 1 is more favorable to global minimizer x compared to Gibbs distribution π in (2.12) from the Metropolis–Hastings algorithm. This is a desirable property if the task is to find the global minimizer because the chance to reach this particular node is automatically increased.

The value of β determines how much the random walk relies on graph structure and potential distribution. If β is very large, then the variation of potential values has little impact and hence the proposed random walk reduces to classical random walk for which the stationary distribution is proportional to node degrees. In this case, the graph structure determines the behavior of random walk completely. When $\beta \to 0$, the walk goes steepest descent direction only and hence the potential distribution dominates the process. Therefore, a moderate value β yields a random walk that balances graph structure and potential distribution.

The degree of freedom of teleportation, which is described by $r = n/|V|$, also affects the stationary distribution. When random teleportation is enabled, there is $r > 0$ and both the numerator and denominator in (2.13) are bumped by $rK_\beta(V)$. Nevertheless, to remain feasibility of the algorithm, the value of r, or the size of additional teleport nodes n, cannot be large.

3 Convergence analysis

We have shown in the previous section that the random walk or teleportation has stationary distribution (2.10). Therefore, Algorithms 1 and 2 construct p^*-reversible Markov chain of nodes on graph G. As a consequence of Perron-Frobenius theorem, the chain converges to p^* exponentially fast depending on the second largest eigenvalue of transition matrix $M(x, y)$. Therefore the convergence rate, or the so-called mixing rate, depends on graph structure, particularly the spectral gap of the graph Laplacian matrix of G, the potential function ϕ, and temperature β. As the transition matrix $M(x, y)$ can have tremendously large size and values of entries vary a lot due to non-uniform distribution of node potentials for graphs in real-world applications, calculating its eigensystem to reveal these properties becomes difficult. In this section, we first provide a continuous time analogue of random walk and teleportation process, and derive the lower bound of convergence rate in an alternative way based on the continuous time dynamics of the probability evolution.

3.1 Continuous time analogue of discrete time random walks

We consider the random walk defined in Algorithm 1 first and the analysis can be easily carried out for random teleportation in Algorithm 2. Let $p(t) = [p_x(t)]_{x\in V}$ and $p^* = [p^*_x]_{x\in V}$ be row vectors in $\mathbf{R}^{|V|}$, where $p_x(t)$ is the probability that one is at node x at time step t, and p^*_x is the stationary distribution at node x. Then $p(t)$ satisfies

$$p(t+1) = p(t)M, \quad t = 0, 1, 2, \ldots, \tag{3.1}$$

and $p(t) \to p^*$ as $t \to \infty$. In this paper, we always choose starting node at random uniformly on the graph and hence the initial probability is $p(0) = (1, \cdots, 1)/|V|$. The random walk has a continuous time analogue as follows: once arrived at x, we draw time $t_{x,y} \sim \exp(M(x, y))$[1] for each $y \in \mathcal{N}_x$ independently, and move to y in time $t_{x,y}$ if $t_{x,y} = \min\{t_{x,z} : z \in \mathcal{N}_x\}$. Note that in general, suppose random variables $t_i \sim \exp(\alpha_i)$, $i = 1, \cdots, n$ are independent, where $\alpha_i > 0, \forall i$ and $\sum_{i=1}^n \alpha_i = 1$. Then $\mathbb{E}(\min\{t_i : i = 1, \cdots, n\}) = 1$ and $\text{Prob}(t_i \le t_j, \forall j \ne i) = \alpha_i$. Note that the coincidence $t_{x,y} = t_{x,y'}$ for $y \ne y'$ occurs with probability 0. This analogue is a continuous time Markov process. Moreover, $\mathbb{E}(\min\{t_{x,y} : y \in \mathcal{N}_x\}) = 1$ and $t_{x,y}$ has exactly chance $M(x, y)$ to be the minimal among $\{t_{x,z} : z \in \mathcal{N}_x\}$. Namely, one is expected to make a move in time 1 since arrived at x, and the probability of moving to y is $M(x, y)$. This is consistent with (3.1). For sufficiently long random walk and teleportations on large graphs, the discrete and continuous time versions are expected to behave quite similarly. Therefore, we use the continuous time evolution of probability $p(t)$ to estimate the convergence rate. In particular, $p(t)$ has derivatives with respect to time t and it follows conservation law etc.

3.2 Convergence results

For notation simplicity, we consider the convergence of random walk in Algorithm 1 only. The steps can be readily modified to show the convergence properties of random teleportation in Algorithm 2. We first derive the dependence of $p'(t)$ on $p(t)$, then use Gronwall's inequality to estimate convergence rate. In our derivation, we define

$$h_x(t) := \frac{p_x(t)}{p^*_x} - 1, \quad \forall x \in V. \tag{3.2}$$

Lemma 3.1 *The evolution of $p_x(t)$ satisfies*

$$p'_x(t) = \sum_{y\in\mathcal{N}_x} Ce^{-(\phi(y)+\phi(x))/\beta}\left(\frac{p_y(t)}{p^*_y} - \frac{p_x(t)}{p^*_x}\right), \tag{3.3}$$

where C is defined in (2.11) and depends only on G, ϕ, and β.

[1] An exponential random variable $T \sim \exp(\alpha)$ for $\alpha > 0$ has probability density function $p(t) = \alpha e^{-\alpha t}$ for $t > 0$.

Proof According to the conservation law, the change of probability $p_x(t)$ equals the difference of probabilities enter x and those flow out. Namely, we have

$$\begin{aligned} p_x'(t) &= \sum_{y\in\mathcal{N}_x\setminus\{x\}} p_y(t)M(y,x) - \sum_{y\in\mathcal{N}_x\setminus\{x\}} p_x(t)M(x,y) \\ &= \sum_{y\in\mathcal{N}_x} \big(p_y(t)M(y,x) - p_x(t)M(x,y)\big) \\ &= \sum_{y\in\mathcal{N}_x} \left(p_y^* M(y,x)\frac{p_y(t)}{p_y^*} - p_x^* M(x,y)\frac{p_x(t)}{p_x^*}\right) \\ &= \sum_{y\in\mathcal{N}_x} Ce^{-(\phi(y)+\phi(x))/\beta}\left(\frac{p_y(t)}{p_y^*} - \frac{p_x(t)}{p_x^*}\right), \end{aligned} \tag{3.4}$$

where in the last equality we used the fact that

$$p_x^* M(x,y) = p_y^* M(y,x) = Ce^{-(\phi(y)+\phi(x))/\beta} \tag{3.5}$$

by recalling the formulas of $M(x,y)$ and p_x^* in (2.8) and (2.10), respectively. □

Let $\lambda > 0$ be the spectral gap of graph G, i.e. λ is the second largest eigenvalue of $D - E$, the Laplace matrix of G, where $E = [e(x,y)]_{x,y\in V}$ is the adjacency matrix and D is the diagonal matrix so that $D_{xx} = \sum_y e(x,y)$. Then we have the following result.

Lemma 3.2 *Let* $p_{\max}^* = \max\{p_x^* : x \in V\}$, *then there is*

$$\frac{\lambda}{p_{\max}^*}\sum_{x\in V} h_x^2(t)p_x^* \le \frac{1}{p_{\max}^*}\sum_{e(x,y)=1}\big(h_x(t)-h_y(t)\big)^2 p_x^* \le \sum_{e(x,y)=1}\big(h_x(t)-h_y(t)\big)^2. \tag{3.6}$$

Proof Due to the Poincaré-type inequality (see, e.g., [7]), there is

$$\frac{\lambda}{p_{\max}^*}\mathrm{Var}_{p^*}(h(t)) \le \sum_{e(x,y)=1}\big(h_x(t)-h_y(t)\big)^2. \tag{3.7}$$

On the other hand, there is

$$\mathrm{Var}_{p^*}(h(t)) = \sum_{x\in V} p_x^*\left(h_x(t) - \frac{1}{|V|}\sum_{x\in V} h_x(t)\right)^2 = \sum_{x\in V} p_x^* h_x^2(t) - \left(\sum_{x\in V} p_x^* h_x(t)\right)^2, \tag{3.8}$$

where the second term on right vanishes:

$$\sum_{x\in V} p_x^* h_x(t) = \sum_{x\in V} p_x^*\left(\frac{p_x(t)}{p_x^*} - 1\right) = \sum_{x\in V}(p_x(t) - p_x^*) = 0, \tag{3.9}$$

since $\sum_x p_x(t) = \sum_x p_x^* = 1$. This completes the proof. □

Theorem 3.3 *Assume that $\phi^* := \max_{x\in V} \phi(x) < \infty$, then probability distribution $p_x(t)$ of the random walk defined in Algorithm 1 converges to the stationary distribution $p_x^* \propto K_\beta(N_x)e^{-\phi(x)/\beta}$ exponentially fast. Moreover, there is*

$$\left\| \frac{p(t)}{p^*} - 1 \right\|_{p^*} \le \left\| \frac{p(0)}{p^*} - 1 \right\|_{p^*} e^{-C\lambda t e^{-2\phi^*/\beta}/p^*_{\max}}. \tag{3.10}$$

where C is defined in (2.11) with $r = 0$ and depends only on G, ϕ, and β.

Proof The squared $(1/p^*)$-weighted distance L between $p(t)$ and p^* at $t > 0$ is defined as follows,

$$L(t) := \frac{1}{2}\left\| \frac{p(t)}{p^*} - 1 \right\|^2_{p^*} = \frac{1}{2}\sum_{x\in V}\left| \frac{p_x(t)}{p_x^*} - 1 \right|^2 \cdot p_x^* = \frac{1}{2}\sum_{x\in V} h_x^2(t)p_x^*. \tag{3.11}$$

Since $h_x'(t) = p_x'(t)/p_x^*$, by taking the derivative of L with respect to t, we have

$$L'(t) = \sum_{x\in V} p_x^* h_x(t)h_x'(t) = \sum_{x\in V} h_x(t)p_x'(t), \tag{3.12}$$

Due to the definition of h in (3.2) and Lemma 3.1, we can substitute $p_x'(t)$ in (3.12) by (3.3) and obtain

$$\begin{aligned} L'(t) &= \sum_{x\in V} h_x(t) \sum_{y\in\mathcal{N}_x} Ce^{-(\phi(y)+\phi(x))/\beta}(h_y(t) - h_x(t)) \\ &= C\sum_{e(x,y)=1} e^{-(\phi(y)+\phi(x))/\beta} h_x(t)(h_y(t) - h_x(t)) \\ &\quad + C\sum_{e(y,x)=1} e^{-(\phi(y)+\phi(x))/\beta} h_y(t)(h_x(t) - h_y(t)) \\ &= -C\sum_{e(x,y)=1} e^{-(\phi(y)+\phi(x))/\beta}(h_y(t) - h_x(t))^2 \\ &\le -Ce^{-2\phi^*/\beta}\sum_{e(x,y)=1}(h_y(t) - h_x(t))^2, \end{aligned} \tag{3.13}$$

since $\phi(y) + \phi(x) \le 2\phi^*$. Then from Lemma 3.2 we get

$$L'(t) \le -\frac{2C\lambda e^{-2\phi^*/\beta}}{p^*_{\max}} L(t). \tag{3.14}$$

Using Gronwall's inequality, we can readily deduce that $p(t)$ converges to p^* exponentially fast

$$\left\| \frac{p(t)}{p^*} - 1 \right\|_{p^*}^2 \leq \left\| \frac{p(0)}{p^*} - 1 \right\|_{p^*}^2 e^{-2C\lambda t e^{-2\phi^*/\beta}/p^*_{\max}}. \tag{3.15}$$

□

Corollary 3.4 *For constant potential, the probability $p(t)$ converges to stationary distribution $p^*_x \propto d(x)$, the degree of x, exponentially fast and*

$$\left\| \frac{p(t)}{p^*} - 1 \right\|_{p^*} \leq \left\| \frac{p(0)}{p^*} - 1 \right\|_{p^*} e^{-\lambda t/d^*}. \tag{3.16}$$

where d^* is the largest node degree in $G = (V, E)$.

Proof Without loss of generality we assume $\phi = 0$. Then $p^*_x = Cd(x)$. Applying Theorem 3.3 we can get the estimate. □

The results above can be easily extended to the random teleportation in Algorithm 2. We only present the results below since the proofs are very similar. Note that for the case of Algorithm 2, the graph G is completed due to the fact that random teleportation essentially allows moving to any node in G (although the probability of moving to a non-adjacent node could be very small). Therefore the edge set becomes $E = \{e(x, y) : x, y \in V\}$ and the spectral gap of G becomes $(N - 1)/N$.

Theorem 3.5 *Assume that $\phi^* := \max_{x\in V} \phi(x) < \infty$, then probability distribution $p_x(t)$ of the random walk defined in Algorithm 2 converges to the stationary distribution $p^*_x \propto (K_\beta(N_x) + rK_\beta(V))e^{-\phi(x)/\beta}$ exponentially fast. Moreover, there is*

$$\left\| \frac{p(t)}{p^*} - 1 \right\|_{p^*} \leq \left\| \frac{p(0)}{p^*} - 1 \right\|_{p^*} e^{-C(N-1)te^{-2\phi^*/\beta}/Np^*_{\max}}. \tag{3.17}$$

where C is defined in (2.11) and depends only on G, r, ϕ, and β.

For constant potentials, we can again assume $\phi(x) = 0, \forall x \in V$. Then $K_\beta(N_x) + rK_\beta(V) = d(x) + rN = d(x) + n$ and we have the following estimate.

Corollary 3.6 *For constant potential, the probability $p(t)$ converges to stationary distribution $p^*_x \propto (d(x) + n)$ exponentially fast and*

$$\left\| \frac{p(t)}{p^*} - 1 \right\|_{p^*} \leq \left\| \frac{p(0)}{p^*} - 1 \right\|_{p^*} e^{-(N-1)t/[N(d^*+n)]}. \tag{3.18}$$

where d^* is the largest node degree in $G = (V, E)$.

4 Intermittent diffusion

The stationary distributions p^* in (2.10) is shown to be favorable to global minimum if β is small enough. However, the convergence to stationary distribution

can be slow if β is too small. In this case, the random walk can also get stuck near local minimum and requires extensively long time to escape. This yields unsatisfactory performance of the proposed random walk in practice. Alternatively, one can make β decrease gradually to 0 as in simulated annealing, however, the decay rate is in general too slow and hinders the practical efficiency of the algorithm.

To overcome this issue with β, we introduce the intermittent diffusion strategy that can significantly improve the efficiency of random walk for global optimum search. The intermittent diffusion implements multiple cycles of diffusion-greedy strategy to Algorithms 1 and 2. More specifically, in each cycle, Algorithms 1 and 2 first proceed with a constant β drawn from $[\beta_{\min}, \beta_{\max}] \subset \mathbf{R}^+$ until close to steady state, followed by greedy search to reach the nearest local minimum. The algorithm is summarized in Algorithm 3 below. The first part in each cycle i is an analogue to adding noise to the greedy approach so that one can move to nodes of larger potential values, similar to the diffusion as Brownian motion is added to a deterministic process. This step should not take long time due to the exponential convergence rate shown in Theorem 3.3. The second part, on the contrary, is completely greedy search since the noise is turned off, i.e. $\beta = 0$. This β switch-on-switch-off intermittent diffusion strategy is illustrated in Fig. 1.

Algorithm 3 Global minimizer search using Algorithm 2 implemented with intermittent diffusion

Given $G = (V, E)$ and potential $\phi : V \to \mathbf{R}$. Set $0 < \beta_{\min} < \beta_{\max}$, diffusion time $T > 0$, maximum number of diffusion cycles $k \in \mathbb{N}$, and initial node x.
for $i = 1, \cdots, k$ **do**
  Draw $\beta \sim \text{Uniform}(\beta_{\min}, \beta_{\max})$.
  Run Algorithm 2 with β for T steps and stop at $x \in V$.
  Run greedy search starting from x, and return a local minimum x_i.
end for
Set $x = \arg\min_{1 \le i \le k} \phi(x_i)$.

We now justify how such an intermittent diffusion strategy can possibly improve the practical performance of Algorithms 1 and 2. Recall that when β is switched on and the algorithm proceeded sufficiently long to be close to steady state p^*, the probability $c(\beta)$ that one is nearby the global minimum x^* and such that search can find x^* is at least $\sum_{z \in \mathcal{N}_{x^*}} p_z^*$. In other words, the probability that the global minimum is not located in one of such switch-on-switch-off cycle is $1 - c(\beta)$. Note that if $\beta_{\max}$ is small (not need to be that small as for $p^* \to \delta_{x^*}$), then the $c(\beta) \ge c^* > \sum_{z \in \mathcal{N}_y} p_z^* > 0$ for any non-optimum $y \in V$. Hence, if the cycle described above is repeated independently for k times with $\beta_i \in [\beta_{\min}, \beta_{\max}]$, $i = 1, \ldots, k$, then the probability that x^* is not returned by any of these k cycles is

$$\prod_{i=1}^{k} (1 - c(\beta_i)). \tag{4.1}$$

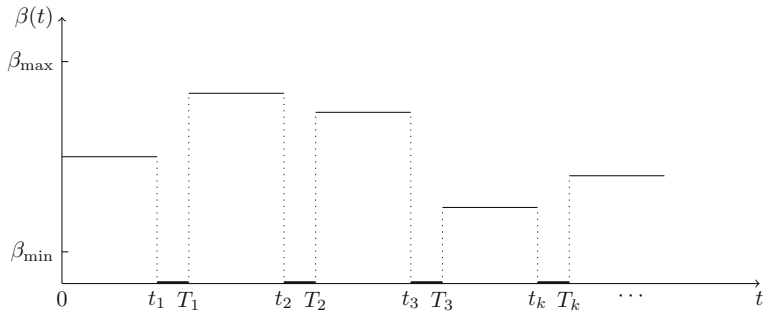

Fig. 1 Illustration of intermittent diffusion strategy. In each cycle of $(T_{k-1}, T_k]$, switch on $0 < \beta \in [\beta_{\min}, \beta_{\max}]$ during (T_{k-1}, t_k) to enable possibility of moving to nodes with higher potential, as an analogue of noise or diffusion process. Then switch to $\beta = 0$ and proceed gradient descent (greedy search) until a local minimum is returned at time T_k. Repeat such cycle for multiple times. The probability that the global minimum is returned by one of the cycles converges to 1 exponentially fast (at the order higher than any other local minima for $\beta_{\max}$ small enough) as $k \to \infty$

As a consequence, the probability that x^* is returned by at least one of such cycles is

$$1 - \prod_{i=1}^{k} (1 - c(\beta_i)) \geq 1 - (1 - c^*)^k \to 1, \tag{4.2}$$

as $k \to \infty$. Note that the convergence rate in (4.2) is much faster than the decay rate of β in simulated annealing. Therefore, by repeating the switch-on-switch-off strategy, we expect to reach the global minimum very quickly. In the next section, we show the significant improvements in efficiency gained by intermittent diffusion.

5 Experimental results

5.1 Experiment setting

We test the performance of the proposed method on a number of synthetic undirected graphs, and select three of the graphs with different structures and potential distributions to present the typical effects of the algorithm in this section. The first graph is a two-dimensional lattice of $10^3 \times 10^3$ mesh grid nodes, i.e. $|V| = 10^6$. The potential function has geographical mountain shape with 10 local minima and one unique global minimum. The second one is a 6-regular graph of $|V| = 10^6$: node x is connected to nodes $x - 3, x - 2, \ldots, x + 3$ for $x = 1, 2, \ldots, |V|$, and wrap to the other end if $x \geq |V| - 2$ and $x \leq 3$. The potential values of these nodes are shown in Fig. 2. The distribution of potential values has multiscale structure and exhibits many local minima. The search of global minimizer is considerably challenging in this case. The third graph is a 20-level complete binary tree with total of $|V| = 2^{20} - 1 \approx 1.05 \times 10^6$ nodes. The first four levels of $2^4 - 1 = 15$ nodes are plotted in Fig. 3. The potential value starts from 2 at node $x = 1$ and monotonically decays to -2 at $x = |V|$ in the order of node indices. Each of the 2^{19} leaves at the 20th level is a local minimum (has

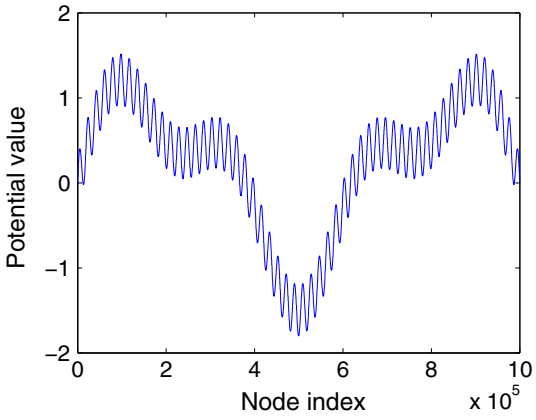

Fig. 2 Distribution of node potential value $\phi(x)$ of the 6-regular graph (Test 2 in Table 2, $|V| = 10^6$). Node x is connected to $x-3, \ldots, x+3$ (mod $|V|$), for $x = 1, \ldots, |V|$. There are a number of local minima and one unique global minimum

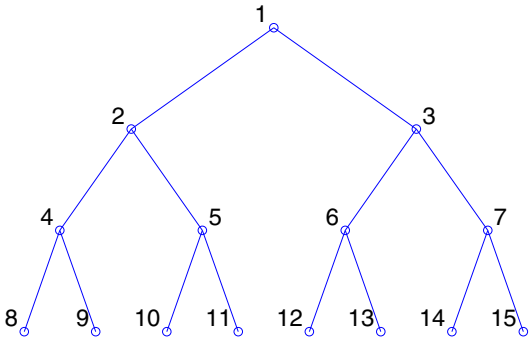

Fig. 3 The first four levels of the 20-level complete binary tree (Test 3 in Table 2, $|V| = 2^{20} - 1 \approx 1.05 \times 10^6$) with node indices. Potential value $\phi(x)$ decays linearly in order starting from node $x = 1$. There are 2^{19} local minima (the leaves), and one unique global minimum at the last node (the right most leaf)

Table 2 Tested graphs $G = (V, E)$ and their properties

Test	Graph name	$\lvert V\rvert$	Graph description
1	Lattice	10^6	$10^3 \times 10^3$ Mesh grid nodes
2	Regular	10^6	6-Regular graph
3	Tree	$2^{20} - 1$	20-Level complete binary tree

smaller potential value in its neighborhood), and the right most leaf, i.e. node $x = |V|$, is the unique global minimum. Such combination of tree structure and potential distribution is a hazard to any generic global optimization method on graphs without prior knowledge. These three graphs are summarized in Table 2.

For a given graph, we start from an initial node randomly drawn from the graph, then search for the global minimum using the following five methods: the Metropolis–Hastings (MH), global Metropolis–Hastings (GMH), random walk only (RW) as in Algorithm 1, random teleportation (RT) only, and the combination of random walk and teleportation (RWT) as in Algorithm 2. The MH method restricts movement to adjacent nodes with acceptance rate $\min(1, e^{-(\phi(y)-\phi(x))/\beta}|\mathcal{N}_x|/|\mathcal{N}_y|)$ which yields Gibbs distribution at steady state. The GMH method allows movements to any nodes on the graph with acceptance rate $\min(1, e^{-(\phi(y)-\phi(x))/\beta})$ and also yields Gibbs distri-

bution. The RT method implements teleportation in Algorithm 2 without using nodes in neighbor $\mathcal{N}_x$ in (2.5).

As the acceptance rate for nodes of higher potential depends on β, we use a large range of β values to see the performance of these five comparison methods. In particular, we use $\beta = 10^{-3}, 10^{-4}, \ldots, 10^{-7}$ for all methods in Test 1, and $\beta = 10^{-1}, 10^{-2}, \ldots, 10^{-5}$ for all methods in Tests 2 and 3. When β falls into these ranges, the performance of all methods appears to be near optimal.

For each graph and β value combination, we run each of the five methods 5,000 times, and count the success rate, i.e. the portion in 5,000 runs that the method found the global minimum. In each of the 5,000 runs, a method is restricted to take at most 10^4 node potential evaluations. For MH and GMH methods, this means that they can take 10^4 steps since they evaluate only one node potential in each step. For RW, RT, and RWT methods, they need to evaluate potentials of all neighbor nodes and/or random drawn farther nodes (teleport nodes). Therefore, RW is allowed to take 2,500 steps if there are four neighbors of each node on average, to count for a total of 10^4 node potential evaluations. For RT and RWT, we let them take 10 adjacent and/or teleport nodes in each step (for instance, take all four adjacent nodes and additional six teleport nodes for RWT), and hence restrict their maximal number of steps to 10^3, again a total of 10^4 node potential evaluations. Note that this counting is conservative for the proposed algorithms: when we move to an adjacent node, it is very likely that the potentials of some adjacent nodes of the new nodes have already been evaluated before, so one can save time evaluating them again and hence the efficiency of the proposed algorithms is further improved. We also point out that, at the same cost of 10^4 node potential evaluations, the brute-force or exhaustive search is expected to have only 1 % success rate.

5.2 Test results

The success rates of the five comparison algorithms in 5,000 runs on 2D lattice graph (Test 1 in Table 2) are shown in Fig. 4. Because of the regular structure of lattice and mild changes of potential distribution, random walk approach such as Metropolis–Hastings (MH), random walk only (RW) as in Algorithm 1 appears to work efficiently: they have about 30 % success rates to find the global minimum at the cost of 10^4 node potential evaluations for all β values. The reason is that the neighbor nodes provide useful potential descent information so that the walk follows the track to global minimum. Moreover, the combination of random walk and teleportation (RWT) is the most efficient algorithm among the five comparison methods: it is able to reach the global minimum in every run. This improvement is because of the teleportation step which helps the walk quickly jumps away from local minima to a much farther node with even lower potential value, where a global minimum is likely to be nearby. However, sole teleportation such as global Metropolis–Hastings (GMH) and random teleportation only (RT) completely ignores local graph structure and potential descent information and moves freely on the entire graph, and hence becomes very inefficient and attains low success rate in search of global minimum.

We now turn to the results on the 6-regular graph in Test 2 of Table 2. As described previously, the graph has relatively regular structure but the distribution of potential

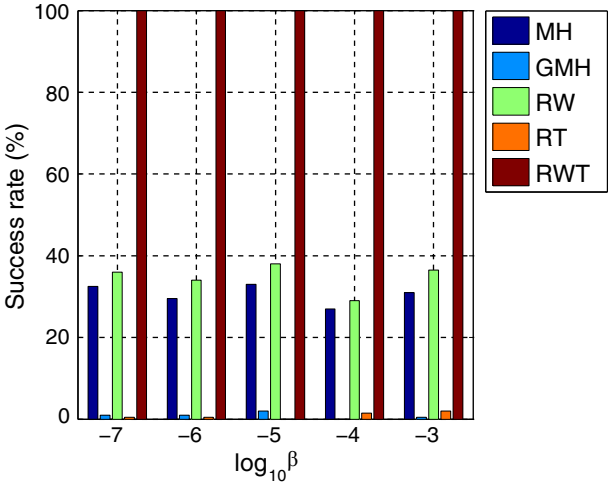

Fig. 4 Results of global minimum search on 2D lattice graph (Test 1 in Table 2, $|V| = 10^6$) using different β values. Success rates in 5,000 runs, each at the cost of 10^4 node potential evaluations, of the five methods: Metropolis–Hastings (MH), global Metropolis–Hastings (GMH), random walk only (RW), random teleportation (RT) only, combination of random walk and teleportation (RWT)

values causes challenges in global minimization due to the multiscale variations and many local minima as in Fig. 2. In this case, teleportation is necessary to escape from local minima. Meanwhile, random walk provides important potential descent information to attain local minimum in each potential valley. It is worth noting that, although random walk can also escape from local minima efficiently if β is sufficiently large, it can also pass the global minimum easily as it appears to be similar to local minima for such large β. Therefore, a combination of random walk with relatively small β and teleportation has benefits to both sides. This is justified by the outstanding performance of RWT in Fig. 5: RWT successfully locates the global minimum in up to 54 % of runs, whereas local search only methods (MH and RW) and sole teleportation methods (GMH and RT) have much lower success rates.

However, there exist certain types of graphs, for instance the tree graph in Test 3 of Table 2, that the proposed method would not work efficiently. The tree graph, as illustrated in Fig. 3, has extensively many local minima (the 2^{19} leaves in the 20th level), half of the nodes in V. Starting from any node in this tree, the right branch is a descent direction, e.g. $2 \to 5 \to 11$ in Fig. 3. Therefore, local search methods based on potential descent, such as MH and RW can be misled to local minimum easily. On the other hand, teleportation could not help much: even one teleports to a node on the right most branch in the mid-levels, it is very likely that a teleportation that occurs later will jump back to some node at lower left part of the tree since those nodes also have small potential values. For instance, even if teleportation first takes one to node 3, it may move to node 12 in the next step as shown in Fig. 3. Therefore, all the five methods perform very inefficiently and their success rates drop to only 1 % or lower as shown in Fig. 6. To sum up, if the local potential descent information lean too much to local minima, and the valley of global minimum does not exhibit enough attraction to

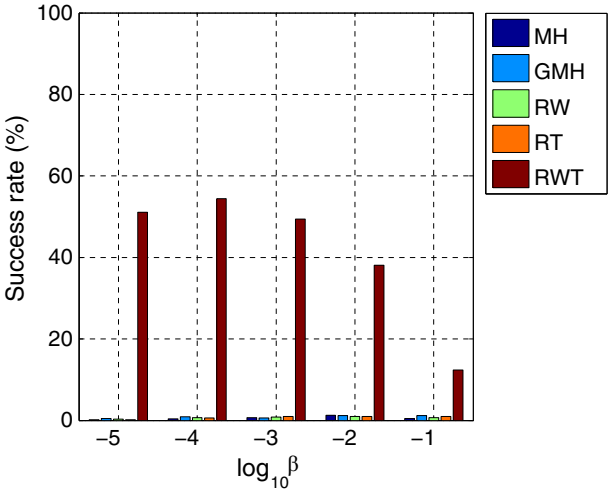

Fig. 5 Results of global minimum search on 6-regular graph (Test 2 in Table 2, $|V| = 10^6$) using different β values. Success rates in 5,000 runs, each at the cost of 10^4 node potential evaluations, of the five methods: Metropolis–Hastings (*MH*), global Metropolis–Hastings (*GMH*), random walk only (*RW*), random teleportation (*RT*) only, combination of random walk and teleportation (*RWT*)

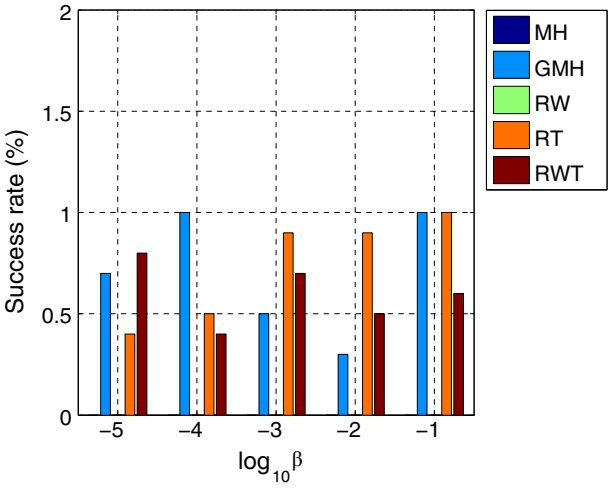

Fig. 6 Results of global minimum search on complete binary tree graph (Test 3 in Table 2, $|V| = 2^{20} - 1 \approx 1.05 \times 10^6$) using different β values. Success rates in 5,000 runs, each at the cost of 10^4 node potential evaluations, of the five methods: Metropolis–Hastings (*MH*), global Metropolis–Hastings (*GMH*), random walk only (*RW*), random teleportation (*RT*) only, combination of random walk and teleportation (*RWT*). Note that all methods succeeded ≤ 1 %

maintain descent search, then the proposed method may perform badly. Nevertheless, we show later in this section that the intermittent diffusion strategy can significantly improve the performance of the proposed RWT algorithm to overcome this problem.

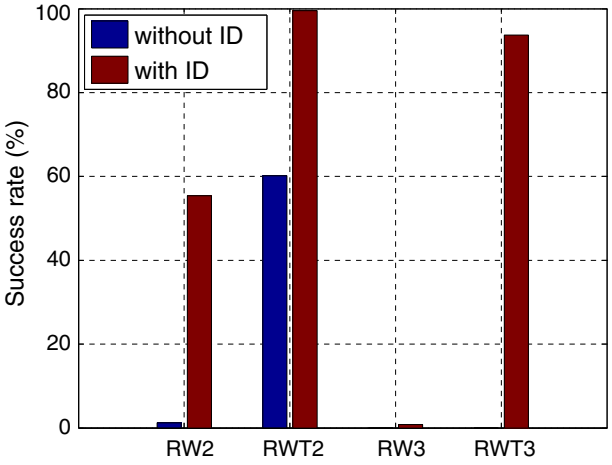

Fig. 7 Effects of intermittent diffusion (ID) to random walk (RW) and combination of random walk and teleportation (RWT) on the 6-regular graph and binary tree graph (Tests 2 and 3 in Table 2 respectively, $|V| \approx 10^6$). The results are named RW2, RWT2, and RW3, RWT3, respectively. *Bar heights* are the success rates in percentage (%) in 5,000 runs of global minimum search by RW and RWT without and with ID at the same cost of 10^4 node potential evaluations

5.3 Improvement using intermittent diffusion

We have introduced the concept of intermittent diffusion in Sect. 4 and illustrated its strategy in Fig. 1. To demonstrate its effect, we select the random walk only (RW) and the combination of random walk and teleportation (RWT), implement them using intermittent diffusion and test on the same 6-regular graph and binary tree graph as in Tests 2 and 3 in Table 2. More specifically, we let RW and RWT proceed 10 cycles of search, each using 10^3 node potential evaluations with β values randomly drawn from $[10^{-4}, 10^{-2}]$ followed by a greedy search as illustrated in Fig. 1, instead of a search that uses up 10^4 evaluations straightly. Therefore, the cost of using intermittent diffusion remains almost the same (the difference is a marginal increase of cost in greedy search step which in total is only few percents of the overall cost). We compare the best success rates obtained by RW and RWT in Tests 2 and 3 above with their new success rates using intermittent diffusion strategy (marked "+ID"), and present the results in Fig. 7.

For the 6-regular graph, both the RW and RWT methods obtain significant improvements on search efficiency by using intermittent diffusion as shown in Fig. 7. In particular, RW used to have only 1.20 % success rate but now it is bumped to 60.18 % at the same cost. RWT can also locate the global minimum in almost every run after intermittent diffusion is implemented. The reason is that, although the potential values exhibit multiscale structure and multiple local minima, the valley of global optimum is still detectable by both methods and more significant than valleys of other local minima. Hence, both RW and RWT have large chance to visit the valley when β is switched on, and the follow up greedy search in each cycle can locate the global minimum effectively.

Fig. 8 Ranking of top five nodes in 2D lattice graph (Test 1 in Table 2, $|V| = 10^6$) by the comparison algorithms. *Horizontal axes* show the node indices from 1 to $|V|$. Marker heights correspond to the rankings of the nodes (higher marker indicates higher ranking). From *top to bottom*, the *six plots* show the rankings given by potential values, and the five comparison algorithms: Metropolis–Hastings (MH), global Metropolis–Hastings (GMH), random walk only (RW), random teleportation (RT) only, combination of random walk and teleportation (RWT)

The effect of intermittent diffusion becomes subtle in the tree graph (Test 3 in Table 2). According to the graph structure shown in Fig. 3 and the potential distribution, the greedy search works only at the nodes on the right most branch. For the random walk only (RW) approach, once it started from some node at the lower left part of the tree, it is very difficult to climb back step by step (every step is a potential ascend direction and hence consecutive successes have very low probability) to upper levels and reach a node on the right most branch. Therefore, the performance of RW does not improve even with intermittent diffusion. On the other hand, for the combination of random walk and teleportation (RWT) algorithm, there is good chance to reach some node on the right most branch due to the teleportation procedure, but difficult to remain on that branch for long time. Therefore, the periodic greedy search prevents jumping away from that branch and hence RWT avoids consistently missing the global minimum as before. This is justified by the significant improvement on success rate using intermittent diffusion as shown in Fig. 7.

5.4 Node ranking

An important problem in exploring large graphs is to provide rankings of nodes base on limited samplings. In this test, we show how the comparison algorithms rank the nodes in the graphs with given potential values. For each of the 5,000 runs, we list the ranking of nodes based on the times that they are visited during each run. Let $n_x^{(1)} \in \{0, 1, \cdots, 5000\}$ be the number of runs where node x is ranked number 1, then we define $x^* = \arg\max_{x \in V} n_x^{(1)}$ as the node of overall rank 1. Similarly, we can define $n_x^{(2)}$ and determine the node of overall rank number 2, etc. We show the overall ranking results by the five comparison algorithms for the 2D lattice graph and 6-regular graph in Figs. 8 and 9, respectively. For reference, we also provide the node rankings according to potential values on the top of each figure. Note that the gaps

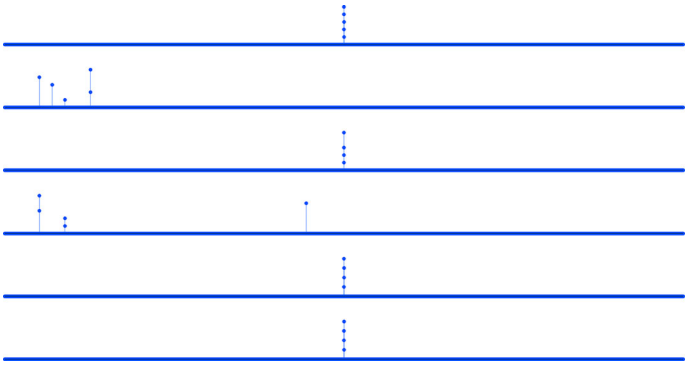

Fig. 9 Ranking of top five nodes in 6-regular (Test 2 in Table 2, $|V| = 10^6$) by the comparison algorithms. *Horizontal axes* show the node indices from 1 to $|V|$. Marker heights correspond to the rankings of the nodes (higher marker indicates higher ranking). From *top to bottom*, the *six plots* show the rankings given by potential values, and the five comparison algorithms: Metropolis–Hastings (MH), global Metropolis–Hastings (GMH), random walk only (RW), random teleportation (RT) only, combination of random walk and teleportation (RWT)

in some ranking results are due to inappropriate display of plot. For instance, the gap between the first and third nodes in the last ranking (bottom one) in Fig. 8 implies that the node ranked number 3 is also ranked as the second, i.e. it is also the node that are most frequently ranked as number 2. Similarly, in the last two rankings in Fig. 9, the missing top 1 nodes imply that the nodes ranked the second are also ranked the first, i.e. they are also the nodes ranked as top 1 most frequently by the two algorithms.

As we can see, based on very limited samplings on the graph, the proposed algorithms can generate moderately accurate rankings of the nodes related to their potential values (as compared to the references in the first row of Figs. 8, 9). In particular, the results by global Metropolis–Hastings algorithm (GMH) and the random teleportation only algorithm (RT) are very similar to that of reference ranking. This is because that these two methods only take the potential value into account, but not graph structures. In contrast, the other methods also consider the neighbor information (local graph structure and potentials of neighbor nodes) and hence their ranking results do not follow ranking of potentials exactly.

6 Concluding remarks

We designed a new potential induced random walk and teleportation algorithm on finite graphs. The algorithm can be used to search for nodes with extreme potentials, and provide ranking of nodes according to their potentials efficiently. In each step, the algorithm chooses a subset of nodes, including the neighbors of the current node and some non-neighbor teleportation nodes on the graph, and moves to one of them according to transition probability determined by the gaps between potential values of neighbor/teleportation nodes and the current node. The algorithmintegrates the structure of the graph with the merit of gradient descent methods so that nodes with lower potential have higher probability to be reached, and hence to obtain the nearby

optimum more efficiently. Meanwhile, the algorithm also allows teleportation to nodes far away to avoid getting stuck at local minima. Convergence rate to the steady state of the proposed random teleportation is derived. For very difficult searching problems, we introduce an intermittent diffusion strategy which can significant improve the random teleportation algorithm. Theoretical and numerical studies are carried out to justify the performance of the proposed algorithm.

The proposed algorithm is aimed to solve generic optimization problem defined on graph. Therefore, it has the potential to solve a large number of applications, such as problems of in RNA folding or information propagation on large scale social media networks, where the possible states or variables are discrete and can be described as nodes on a graph. Although the graphs formed in these applications often have extensively large sizes, the proposed algorithm provides an effective way to reach the optimum node with high probability.

Acknowledgments This work was supported in part by NSF Grants Faculty Early Career Development (CAREER) Award DMS-0645266 and DMS-1042998, and ONR Award N000141310408.

References

1. Abdullah, M., Cooper, C., Frieze, A.: Cover time of a random graph with given degree sequence. Discrete Math. **312**(21), 3146–3163 (2012)
2. Aggarwal, C.C.: Social Network Data Analytics. Springer, New York (2011)
3. Aldous, D.: Applications of random walks on finite graphs. In: Selected Proceedings of the Sheffield Symposium on Applied Probability (Sheffield, 1989), volume 18 of IMS Lecture Notes Monograph Series, pp. 12–26. Institute of Mathematics and Statistics, Hayward, CA, (1991)
4. Aldous, D., Fill, J.: Reversible Markov Chains and Random Walks on Graphs. Monograph in Preparation (1999). Book in preparation
5. Berg, B.: Introduction to Markov Chain Monte Carlo Simulations and Their Statistical Analysis (2004). arXiv:cond-mat/0410490
6. Billera, L., Diaconis, P.: A geometric interpretation of the Metropolis–Hastings algorithm. Stat. Sci. **16**(4), 335–339 (2001)
7. Bobkov, S., Tetali, P.: Modified logarithmic sobolev inequalities in discrete settings. J. Theor. Probab. **19**(2), 289–336 (2006)
8. Burioni, R., Cassi, D.: Random walks on graphs: ideas, techniques and results. J. Phys. A Math. Gen. **38**(8), R45 (2005)
9. Casella, G., George, E.: Explaining the Gibbs sampler. Am. Stat. **46**(3), 167–174 (1992)
10. Chandra, A., Raghavan, P., Ruzzo, W., Smolensky, R., Tiwari, P.: The electrical resistance of a graph captures its commute and cover times. Comput. Complex. **6**(4), 312–340 (1996)
11. Coppersmith, D., Doyle, P., Raghavan, P., Snir, M.: Random walks on weighted graphs and applications to on-line algorithms. J. ACM **40**(3), 421–453 (1993)
12. Doyle, P., Snell, L.: Random Walks and Electric Networks (Carus Mathematical Monographs). Mathematical Assn of America, first printing edition (1984)
13. Doyle, P.G., Snell, J. L.: Random Walks and Electric Networks, 3 (2000). arXiv:math/0001057
14. Geman, S., Geman, D.: Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Trans. Pattern. Anal. Mach. Intell. **6**(6), 721–741 (1984)
15. Gjoka, M., Kurant, M., Butts, C., Markopoulou, A.: A Walk in Facebook: Uniform Sampling of Users in Online Social Networks (2011). arXiv:0906.0060
16. Hastings, W.: Monte carlo sampling methods using Markov chains and their applications. Biometrika **57**(1), 97–109 (1970)
17. Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the spread of influence through a social network. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 137–146. ACM, New York (2003)

18. Klein, D., Palacios, J., Randic, M., Trinajstic, N.: Random walks and chemical graph theory. J. Chem. Inf. Comput. Sci. **44**(5), 1521–1525 (2004)
19. Langville, A., Meyer, C.: Deeper inside pagerank. Internet Math. **1**(3), 335–380 (2004)
20. Langville, A.N., Meyer, C.D.: Google's PageRank and Beyond: The Science of Search Engine Rankings. Princeton University Press, Princeton, NJ (2009)
21. Lawler, G.F., Limic, V.: Random Walk: A Modern Introduction, vol. 123. Cambridge University Press, Cambridge (2010)
22. Lovász, L.: Random walks on graphs: a survey. Comb. Paul Erdős Eighty **2**(1), 1–46 (1993)
23. Meer, K.: Simulated annealing versus metropolis for a TSP instance. Inf. Process. Lett. **104**(6), 216–219 (2007)
24. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculations by fast computing machines. J. Chem. Phys. **21**, 1087 (1953)
25. Nourani, Y., Andresen, B.: A comparison of simulated annealing cooling strategies. J. Phys. A Math. Gen. **31**, 8373–8385 (1998)
26. Richey, M.: The evolution of Markov chain Monte Carlo methods. Am. Math. Mon. **117**(5), 383–413 (2010)
27. Rosvall, M., Bergstrom, C.: Maps of random walks on complex networks reveal community structure. Proc. Natl. Acad. Sci. **105**(4), 1118–1123 (2008)
28. Sarkar, P., Moore, A.W.: Random walks in social networks and their applications: a survey. In: Aggarwal, C.C. (ed.) Social Network Data Analytics, pp. 43–77 (2011). Springer, US
29. Suman, B., Kumar, P.: A survey of simulated annealing as a tool for single and multiobjective optimization. J. Oper. Res. Soc. **57**(10), 1143–1160 (2006)
30. Tetali, P.: Random walks and the effective resistance of networks. J. Theor. Probab. **4**(1), 101–109 (1991)
31. van Kampen, N.: Stochastic Processes in Physics and Chemistry, 3rd edn. North Holland, Amsterdam (2007)
32. Wegener, I.: Simulated annealing beats metropolis in combinatorial optimization. In: Automata, Languages and Programming, p. 61 (2005)
33. Weiss, G.: Aspects and Applications of the Random Walk (Random Materials and Processes S.). North-Holland, Amsterdam (1994)
34. Weinan, E., Li, T., Vanden-Eijnden, E.: Optimal partition and effective dynamics of complex networks. Proc. Natl. Acad. Sci. USA **105**(23), 7907–7912 (2008)