

A Generic Search Strategy for Large-Scale Real-World Networks

Kurumida, Yuichi
Kyushu University

Ogata, Tsukasa
Fujitsu Network Technologies Limited | Kyushu University | Kyushu University

Ono, Hirotaka
Kyushu University

Sadakane, Kunihiro
Kyushu University

他

<https://hdl.handle.net/2324/14866>

出版情報 : Proceedings of the 1st international conference on Scalable information systems, 2006-05

バージョン :

権利関係 : © ACM, 2006. This is the author's version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version was published in ACM International Conference Proceeding Series; Vol. 152 Article No. 2, <http://doi.acm.org/10.1145/1146847.1146849>



A Generic Search Strategy for Large-Scale Real-World Networks

Yuichi Kurumida

Department of Electrical Engineering and
Computer Science
Kyushu University

6-10-1 Hakozaki, Fukuoka, Japan 812-8581
Email: kurumida@tcslab.csce.kyushu-u.ac.jp

Tsukasa Ogata¹

Hiroataka Ono

Kunihiko Sadakane and
Masafumi Yamashita

Department of Computer Science and
Communication Engineering
Kyushu University

6-10-1 Hakozaki, Fukuoka, Japan 812-8581

Email: tsukasa@tcslab.csce.kyushu-u.ac.jp

Email: {ono, sada, mak}@csce.kyushu-u.ac.jp

Abstract—We consider the following situation for a given large-scale network: Starting from an initial node we move to its neighbor node and repeat that until reaching a target node. How fast can we do this without any global topological information? This problem is considered “searching networks”, and several approaches have been proposed. In this paper, we present a general framework of search strategies, under which all of these existing approaches can be formalized. Our framework characterizes random search strategies by the following three parameters: memory for previously-visited nodes, look-ahead property and transition probability. Through computational simulations for large-scale networks with small-worldness and scale-freeness, we investigate the relationship between the effect of parameters of the strategies and the coefficients of networks such as the clustering coefficient. The comparison result provides a guideline to obtain good parameters of the strategies according to the diameter and the clustering coefficients of networks.

I. INTRODUCTION

Given a network/graph, we consider the following situation: Starting from an initial node, we move to its neighbor node and repeat that until reaching a target node as fast as possible. If we know the whole (global) topology of the network and can compute the shortest path from the global topological information, we can easily reach the target node via the shortest path. Actually, that assumption is natural for small-scale networks, such as real-world transportation networks, so many shortest path algorithms, e.g., Dijkstra’s algorithm, are well studied. On the other hand, for some types of networks, such as the WWW network and social networks, it is difficult to obtain the whole topological information of the networks. Usually, these networks have a large number of nodes and also easily change their topology; this implies that it is actually impossible to run a shortest path algorithm because such an algorithm requires to memorize all the topological information in general. How should we do for such networks only with limited memory and local topological information? In this paper, we present a generic search strategy, which is tunable by three parameters related to these factors, and the goal of the paper is to provide a guideline of designing a good search strategy.

To construct the generic search, we review some basic ideas about search with limited memory. One of the simplest approaches is to utilize the random walk. The standard random walk is the process which repeats the following procedure: A particle on a node randomly chooses one of its neighbor nodes with same probability, and moves to it (e.g., [15]). Obviously, the procedure requires only a small memory and local topological information. The standard random walk is intensively studied. For example, for any graph G , it is known that the expected number of steps to get from node u to another node v is bounded by $O(n^3)$, where n is the number of nodes of G , and the expected number of steps to visit all the nodes is also bounded by $O(n^3)$ [2]. This implies that if we use the standard random walk for search the expected number of running steps (i.e., the expected number of edges passed through from a start node to a target node) is also $O(n^3)$.

However, this time bound is not so good for the purpose of search, and actually bad for some case; it is known that there exists a graph whose expected running steps is $\Omega(n^3)$ [2], although such graphs are hardly seen in real-world. Anyway, the standard random walk is actually not suitable for network search because it sometimes takes stupid behaviors: a particle on v coming from u , a neighbor of v , may visit u again. That is, the standard random walk might be interesting from the theoretical viewpoint, but not practical as a search strategy. Actually, it is easy to avoid this behavior, and it is OK to add one-state memory in order to prohibit moves to the previous node. How is the effect of the prohibition? Furthermore, in the standard random walk, the probability of choosing a neighbor node for move is identical among the neighborhood nodes. Is this really best utilizing local topological information?

To answer these, many approaches have been proposed and their performance is investigated through computational simulations for networks with typical real-world network properties, say *small-worldness* and *scale-freeness*. S.-J. Yang proposes several types of search strategies based on random

¹He is currently working for Fujitsu Network Technologies Limited.

walk, but they are also equipped with small memories to avoid useless moves [19]. *No-back walk* is the simplest one and forbids to move to the previous node. *no-triangle-loop walk* and *no-quadrangle-loop walk* extend the no-back walk and prohibit the moves to not only the previous node but also the second previous node and the third previous node, respectively. Some random search strategies use not identical transition probabilities but biased ones instead. In such strategies, usually the transition probabilities are defined according to local topological information. For example, in *maximum connectivity first* strategy, the highest degree node among the current node's neighbors is searched first. Note that all of these approaches are equipped with the *look-ahead* property in which the particle on an adjacent node of the target can directly move to the target node in the next step instead of randomly choosing a next node.

As for the transition probability, recently, Ikeda et al. [12] has proposed β -random walk, in which transition probability is defined by the degree of the corresponding node and parameter β . They also give an optimal value of parameter β , and for this β , the hitting time is $O(n^2)$ and the cover time is $O(n^2 \log n)$ [12]. Recall that these of the standard random walk are both $O(n^3)$. Roughly speaking, the optimal parameter of β shows that choosing a lower degree node (with a higher probability) is good to achieve a fast random walk in general. On the other hand, the results of "maximum connectivity first policy" show that choosing a higher degree node is better under the graphs having scale-free property [14]. Is this due to the scale-freeness?

Based on these observations, in this paper, we present a generic search strategy: We characterize search strategies in terms of the following features: memory for previously-visited nodes, look-ahead property and transition probability. As for memory, *k-tabu* is a mechanism of prohibiting moves to k previously-visited nodes. *l-look-ahead* is the one of reaching the target node directly if it is within distance l from the current node. As for the transition probability, we adopt the one of the above β -simple random walk. By these capabilities, we define (k, l, β) -random walk as a generic search strategy, i.e., (k, l, β) -random walk is the search strategy that is based on the β -simple random walk but also has *k-tabu* and *l-look-ahead* capabilities. For example, under this framework, the search strategy based on the standard random walk is characterized as $(0, 0, 0)$ -random walk, and no back walk is characterized as $(1, 1, 0)$ -random walk, and so on. The aim of this paper is to provide a good perspective on performance analysis and comparison studies for the search strategies by using this characterization.

To this end, we perform computational experiments (simulations) in order to investigate the relationship between the performance and the parameters. In the computational simulations, we implement (k, l, β) -random walk and apply it with several parameters to large-scale networks with small-worldness and scale-freeness as real-world networks, and random graphs by Erdős and Rényi [9] as an artificial type of networks. The simulation results give a guideline for designing

good search strategies according to models of graphs.

The rest of the paper is organized as follows. Section II gives introductions about small-world and scale-free networks as real-world networks, their generating models, and also basic ideas of random walks. In section III, we give a brief survey about several existing search strategies, and then present our generic search strategy. Section IV gives the performance analysis/comparison of the strategies through computational simulations, and section V concludes this paper.

II. PRELIMINARIES

A. Graphs/Networks and their properties

Let $G = (V, E)$ be a finite undirected connected graph, where V is a set of n nodes (or vertices) and E is a set of n edges. In this paper, we also call G a *network*. For $u \in V$, $N(u)$ denotes the set of nodes adjacent to u (i.e., $N(u) = \{v \mid (u, v) \in E\}$), and $\deg(u) \stackrel{\text{def}}{=} |N(u)|$ is called u 's *degree*. A $x_0 - x_k$ *path* is a sequence of $(x_0, x_1, \dots, x_{k-1}, x_k)$ satisfying $(x_{i-1}, x_i) \in E$ for $i = 1, \dots, k$, and the *length* of the path is k . The *distance* $\text{dis}(u, v)$ between two nodes u, v is the length of a shortest $u - v$ path.

To characterize or quantify a given graph, various measures are proposed. Among them, we focus on the characteristic path length and the clustering coefficient, both of which are introduced by Watts et al. [18]. The *characteristic path length* of node u , denoted by $L_G(u)$, is the average distance from u to any other nodes $v \in V \setminus \{u\}$. The characteristic path length of graph G , denoted by L_G , is the average on $L_G(u)$. Formally,

$$L_G(u) \stackrel{\text{def}}{=} \frac{1}{n-1} \sum_{v \neq u} \text{dis}(u, v), \text{ and } L_G \stackrel{\text{def}}{=} \frac{1}{n} \sum_{u \in V} L_G(u).$$

The *clustering coefficient* of a node u , denoted by $C_G(u)$, is the proportion of the number of edges between the nodes within its neighborhood to the number of edges that could possibly exist between them. The clustering coefficient of graph G , denoted by C_G , is the average on $C_G(u)$. Formally,

$$C_G(u) \stackrel{\text{def}}{=} \frac{2E_u}{\deg(u)(\deg(u) - 1)}, \text{ and } C_G \stackrel{\text{def}}{=} \frac{1}{n} \sum_{u \in V} C_G(u).$$

where E_u is the number of edges between the neighbors of u , i.e., $|\{(v_1, v_2) \in E \mid v_1, v_2 \in N(u)\}|$.

In general, there is no strong relation between L_G and C_G . For example, C_G 's of both path graphs and complete binary trees are equal to 0, but L_G of the former is large ($O(n)$) while L_G of the latter is $O(\log n)$. Moreover, for complete graphs, both L_G and C_G are equal to 1. However, various kinds of real-world networks such as WWW [1], co-author relationship on science papers [16], airline routes [4] and biological metabolic networks [13], have a common feature about L_G and C_G , called *small-worldness*, whose definition is introduced by Watts et al. [18]. The small-world property is characterized as

$$L_G \approx L_{\text{rand}} \text{ and } C_G \gg C_{\text{rand}},$$

where $L_{rand} \sim \log(n)/\log(\bar{d})$ and $C_{rand} \sim \bar{d}/n$ (\bar{d} is the average degree of the graph) represent the average values of L_G and C_G of random graphs with almost the same size respectively. It is known that both of them are small if the number of edges is small.

On the other hand, these real-world networks are known to have another characteristics: It is reported that in many real-world networks [3], [4], [7], the degree distribution follows the power-law, i.e., the probability $P(d)$ that a node has degree d satisfies

$$P(d) \propto d^{-\gamma}.$$

The network whose degree distribution follows the power-law is called *scale-free network*. Barabási et al. [5] showed that the scale-free networks can be generated by two simple generating rules: *growth* and *preferential attachment*, as shown later.

In the next section, we review three major network generating models, ER, WS and BA, all of which are intensively studied.

B. Network models

In this subsection, we review three major network generating models, ER, WS and BA models. ER is a model of random graphs [8], WS is a model of generating small-world networks [18], and BA is a model of generating scale-free networks [5]. These are most typical generating models of the corresponding types of networks.

1) *ER model*: ER model is proposed by Erdős and Rényi and is the model that generates random graphs (networks) [8]. In this model, graphs are generated as follows: We first prepare n isolated nodes. Then, for each pair of nodes, we connect them with a given probability p_e . Thus, the expected number of the edges is $p_e n(n-1)/2$. The distribution of degree d is

$$P(d) = \binom{n-1}{d} p_e^d (1-p_e)^{n-1-d},$$

which is binomial with the average degree $\bar{d} = p_e(n-1)$. If $n \rightarrow +\infty$, this converges to

$$P(d) \rightarrow \frac{e^{-\bar{d}} \bar{d}^d}{d!},$$

which is the Poisson distribution.

2) *WS model*: WS model is proposed by Watts and Strogatz and is the model that generates small-world networks [18]. In this model, we prepare a regular lattice graph in which all the nodes are placed on a circle and then each node is connected to its k_w neighbors (See Figure 1 (left)).

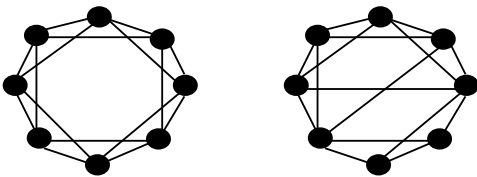


Fig. 1. A lattice graph (left) and a rewired one (right)

Then rewire each edge in E with a given probability p_w . “Rewiring the edge” means removing one end of the edge and connecting to another node uniformly chosen so as not to make any self-loops or multiple edges. (Figure 1 (right) is a graph after rewiring.). The number of the edges is $nk_w/2$.

The probability p_w decides C_G and L_G values, i.e., p_w determines if the resulting graph has the small-worldness. For example, if $p_w = 0$ the resulting graph is the lattice graph itself, which does not have the small-world property. If $p_w = 1$, the resulting graph is almost same as a random graph, which is not also small-world. It is known that for $0.01 < p_w < 1$ the resulting graphs have the small-worldness.

3) *BA model*: BA model proposed by Barabási and Albert is the model that generates scale-free networks [5]. This model is defined in two rules. (1) Growth: We first prepare the complete graph with m_0 nodes. At every time step, we add a new node and connect it to distinct $m' (\leq m_0)$ nodes randomly chosen from the existing nodes. (That is, m' edges are added.) (2) Preferential attachment: The probability that node u is chosen is $\deg(u) / \sum_{v \in V'} \deg(v)$, where V' is the set of nodes at the time. This process is repeated in t steps. Therefore, after t steps, the number of the nodes and the edges are respectively $m_0 + t$ and $m_0(m_0 - 1)/2 + m't$.

C. Random walk

In this subsection, we give basic ideas of random walks, which play an important role in this paper.

The random walk on finite graphs is the process that repeats moving a particle on a certain node to one of its neighbor nodes with some probability. In general, the “random walk” means the standard random walk in which the particle moves to one of the neighbor nodes with the same probability [6]. That is, the transition matrix $P = (p_{u,v})$ is given by

$$p_{u,v} = \begin{cases} 1/\deg(v) & (v \in N(u)) \\ 0 & (\text{otherwise}). \end{cases}$$

Note that the standard random walk uses only the degree information of the node on which the particle exists. The *hitting time* $H_G(u, v)$ is the expected number of steps in which the particle starting from u reaches v . The *hitting time* of graph G is defined by $H_G = \max_{u,v \in V} H_G(u, v)$. In the standard random walk, the hitting time of any graph G is bounded by $O(n^3)$ [10].

Recently, Ikeda et al. have generalized the standard random walk, and proposed β -simple random walk [12]. The β -simple random walk uses a generalized transition probability of the particle on u , which is based on not only $\deg(u)$ but also $\deg(v)$ of $v \in N(u)$. The transition matrix is given by

$$p_{u,v}^{(\beta)} = \frac{\deg^{-\beta}(u)}{\sum_{w \in N(u)} \deg^{-\beta}(w)},$$

where $\beta \in \mathbf{R}$. When $\beta = 0$, the β -simple random walk is identical to the standard random walk. In the β -simple random walk, it is proved that the hitting time of any graph G is bounded by $O(n^2)$ at $\beta = 0.5$ [12]. This β is actually optimal in the sense that for any transition matrix there exists a graph

whose hitting time is $\Omega(n^2)$. The $\beta = 0.5$ means that in the transition probability, nodes with lower degree are more preferable.

III. LOCAL SEARCH STRATEGIES

The goal of this paper is to provide a good search strategy on real-world networks, i.e., networks with small-worldness or scale-freeness. Now we formally define the search problem. We are given a network (graph), a start node $u \in V$ and a target node $v \in V$. The problem is to find v through a certain path $u \rightarrow v$ whose length is as short as possible.

As mentioned in section I, it is supposed that the size of the network is very large and sometimes the topology is also changeable. Therefore, the traditional memory-based searches such as the breadth-first search and shortest path algorithms are not applicable, and various search strategies based on “walk” are proposed. A general idea of walk-based searches is described as follows: We put a particle on the start node and repeatedly move it to one of the neighbor nodes until reaching the target node. In this setting, we are requested to lead the particle to the target node as fast as possible.

In this section, we briefly describe past walk-based strategies and then present our generic search strategy, (k, l, β) -random walk.

A. Past studies

One of the simplest search strategies is just to use the standard random walk as a search. However, since the standard random walk is memoryless, useless visits can be taken, i.e., the particle coming from u to w (not the target node) sometimes visits u again. One direction of the studies is to prohibit such useless visits by using small (short) memory. The *no-back walk* is the standard random walk with the capability of prohibiting to move to the previous node [19]. The *no-triangle-* and *no-quadrangle-* loop walk are the extensions of the no-back walk and they prohibit to move to the second and the third previous node as well, respectively [19]. The *self-avoiding walk* also has the identical transition probabilities and prohibits to move to all the nodes previously visited [11], [19].

Another research direction is to use a different transition probability. PRF chooses a node with the probability proportional to the degrees of neighbor nodes, and moves the particle deterministically to the target node if it is in the neighbor nodes [14]. The *maximum connectivity first* strategy moves the particle to one of the neighbor nodes whose degree is maximum [14]. It is reported that both methods, in which nodes with higher degree are preferable, have good performance for small-world networks. On the other hand, recall that in the β -simple random walk for a (general) graph has the best performance when $\beta = 0.5$, which means that nodes with lower degree are more preferable (see section II. C.). It is a little interesting, because these may imply that scale-free networks have special structures in terms of random walks.

Note that all these strategies has *look-ahead* property and move the particle directly to the target node if it is in the neighbor nodes. Among these, it was reported that the self-avoiding walk has good performance in the number of edges passed through [11], [19].

B. A generic search strategy: (k, l, β) -random walk

As shown above, many existing search strategies are based on random walks (which may use different transition probability) and also are equipped with memory for prohibiting moves and look-ahead property. Here, we present a general framework, which can formalize these strategies by three parameters: k -tabu, l -look-ahead and β -transition probability, where k, l are nonnegative integers and β is a real number.

The k -tabu is a mechanism that prohibits to move to the k nodes previously visited by memorizing (listing) them. (If all the neighbor node of the current position are listed, then move to one of the nodes according to the transition probability.) The list is easily implemented as a queue.

The l -look-ahead property is a mechanism that checks if the target node is within distance l from the current node. If yes, the particle can reach the target directly. This can be easily implemented by the simple bread-first search.

The k -tabu and l -look-ahead property can be combined with many search strategies. In this paper, we combine these properties with β -simple random walk, and call this (k, l, β) -random walk. The (k, l, β) -random walk is described as follows:

- Step 0 : Prepare an empty list with size k . (Called *tabu list*.)
- Step 1 : Use the l -look-ahead property. If it founds the target node, then output and halt.
- Step 2 : Prepare an empty *candidate list*. Check if each neighbor node is in the k -tabu list. If no, add it to the candidate list. After checking all the neighbor nodes, if the candidate list is still empty, then add all the neighbors to the candidate list.
- Step 3 : For all the nodes in the candidate list, compute the transition probability by

$$p_{u,v}^{(\beta)} = \begin{cases} \frac{\deg^{-\beta}(u)}{\sum_{w \in \text{candidate list}} \deg^{-\beta}(w)} & (v \in \text{candidate list}) \\ 0 & (\text{otherwise}). \end{cases}$$

Choose a node according to the probability. Add the chosen node to the tabu list and move to it. Goto Step 1.

By definition, the standard random walk and the β -simple random walk are characterized as $(0, 0, 0)$ -random walk and $(0, 0, \beta)$ -random walk, respectively. Also note that all the strategies explained in section III. A. can be characterized as (k, l, β) -random walk with certain parameters. Table I shows the correspondence of them with the values of three parameters.

TABLE I
EXISTING STRATEGIES AND (k, l, β) PARAMETERS

| past strategy | (k, l, β) -RW |
|--|------------------------|
| standard random walk | (0,0,0) |
| no-back walk [19] | (1,1,0) |
| no-triangle-loop walk [19] | (2,1,0) |
| no-quadrangle-loop walk [19] | (3,1,0) |
| self-avoiding walk [11], [19] | $(\infty, 1, 0)$ |
| preferentially self-avoiding walk [19] | $(\infty, 1, -1)$ |
| maximum connectivity first [14] | $(\infty, 1, -\infty)$ |

IV. COMPUTATIONAL SIMULATION

In this section, we investigate good search strategies of our (k, l, β) -random walk through computational simulations for networks with small-worldness or scale-freeness.

First, we explain how to prepare the networks for the simulations and how to set the (k, l, β) parameters.

A. Settings of the simulations

1) *Generated networks and their parameters:* We prepare three types of networks (graphs), ER, WS and BA. We use only the connected graphs out of the simple graphs generated by each model. For comparison, we construct the networks with the same numbers of nodes and edges. For this purpose, we set the parameters of each generating model so that a network with n nodes has about $2n$ edges and the average degree $\bar{d} = 4$. Therefore, the model-specific parameters are set as follows.

ER model: In a graph of ER model, the expected number of edges is equal to $p_e n(n-1)/2$, which should be $2n$. Hence we set the probability $p_e = 4/(n-1)$.

WS model: Since the average degree of the graph is 4, we set the degree of the initial regular lattice graph 4. As for the rewiring probability p_w , we need to specify some proper values to assure the small-worldness of the graph. To decide this, we use the metrics μ called *small-worldliness* which provides the quantitative measure of small-worldness, introduced by Walsh [17]:

$$\mu = (C_G/L_G)/(C_{rand}/L_{rand}).$$

As p_w grows from 0, rewired edges work as “shortcut”. Because of the shortcuts, L_G drops rapidly but C_G remains large, so μ grows fast. On the contrary, as p_w comes close to 1, C_G gets smaller while rewired edges affect L_G no longer, so μ also gets smaller; the network loses small-worldliness. Figure 2 shows μ plotted against p_w at $n = 1000$ and $k_w \in \{2, 4, 6, 10\}$, the degrees of lattice graphs. This figure shows μ is maximum when p_w is around 0.1 regardless of k_w . The similar tendencies are shown for other n 's and k_w 's, so we set $p_w = 0.1$.

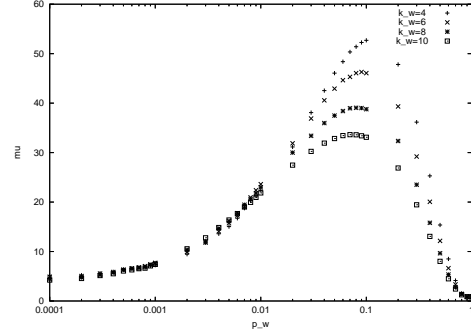


Fig. 2. small-worldliness μ

BA model: As shown in section II, the exact numbers of nodes and edges in BA model are $m_0 + t$ and $m_0(m_0 - 1)/2 + m't$, respectively. Given a number of nodes n , we set the time step $t = n$ and $m_0 = m' = 2$ so that the number of edges is about $2n$.

Table II shows examples of the average feature quantities of 20 networks of $n = 500$, which are used in the computational simulations. In the simulations, we prepare networks with n nodes for each generating model for $n \in \{50, 100, 200, 300, 500\}$.

TABLE II
FEATURE QUANTITIES

| | average deg | deg var | L_G | C_G | diameter |
|----|-------------|---------|-------|--------|----------|
| ER | 4.16 | 3.76 | 4.54 | 0.0063 | 9.40 |
| WS | 4.00 | 0.39 | 7.59 | 0.37 | 15.15 |
| BA | 3.99 | 24.0 | 3.76 | 0.043 | 7.00 |

2) *Parameter settings of (k, l, β) :* For the networks generated by those models with the parameters specified above, we run the (k, l, β) -random walks. The parameters used in the simulations are all combinations of $k \in \{0, 1, 2\}$, $l \in \{0, 1, 2\}$ and $\beta \in \{-1, -0.5, 0, 0.5, 1\}$. We denote this by $(k, l, \beta) \in (\{0, 1, 2\}, \{0, 1, 2\}, \{-1, -0.5, 0, 0.5, 1\})$. To compare their performance, we evaluate the number of edges passed through from a start node to a target node (i.e., the number of steps) for all pairs of start and target nodes. For each setting of parameters and start-target pair, we perform 50 trials of simulations and compute their average steps. The mean and max of the number of steps over all $\frac{1}{2}n(n-1)$ pairs for one network are called the *mean steps* and the *max steps*, respectively. The latter corresponds to the hitting time.

In summary, we implement (k, l, β) -random walks with $(k, l, \beta) \in (\{0, 1, 2\}, \{0, 1, 2\}, \{-1, -0.5, 0, 0.5, 1\})$, and apply them to 20 distinct networks with $n \in \{50, 100, 200, 300, 500\}$ nodes of ER, WS and BA models. For each combination of the parameters, we calculate the average of the results on them.

B. Results and discussions

We first see the overall tendencies of the simulation results.

Figures 3-8 show the results for $(k, l, \beta) \in (0, 1, \{-1, -0.5, 0, 0.5, 1\})$. In each figure, the mean or max steps are plotted against the number of nodes. From these results, we observe that the mean and max steps increase near-linearly with the number of nodes for every parameter. This tendency is wholly seen for other parameters (other figures are partially listed in appendix).

Tables III and IV show the mean and max steps for $n = 500$ and $\beta \in \{-1, -0.5, 0, 0.5, 1\}$. As for mean steps, if $l = 0$, namely in case without look-ahead property, the steps are smaller at $\beta > 0$ while at $l = 1$ they are smaller at $\beta < 0$. On the other hand, the max steps are smaller at $\beta > 0$ regardless of whether or not the look-ahead property is used.

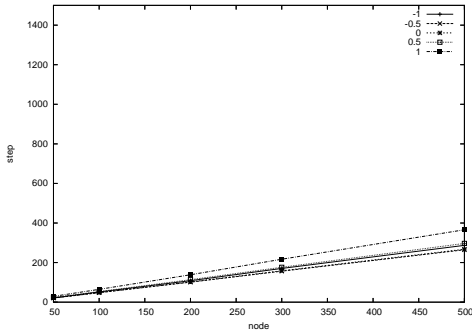


Fig. 3. mean steps : ER model, $(k, l)=(0,1)$

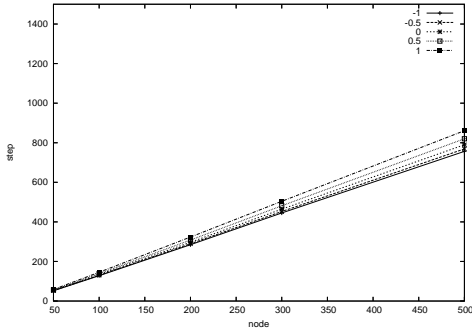


Fig. 4. mean steps : WS model, $(k, l)=(0,1)$

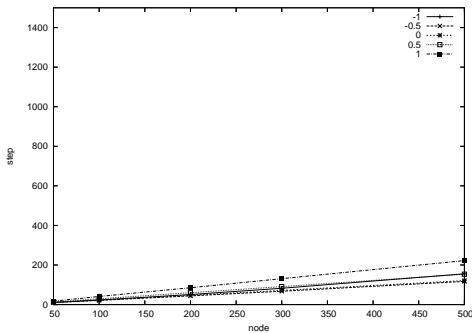


Fig. 5. mean steps : BA model, $(k, l)=(0,1)$

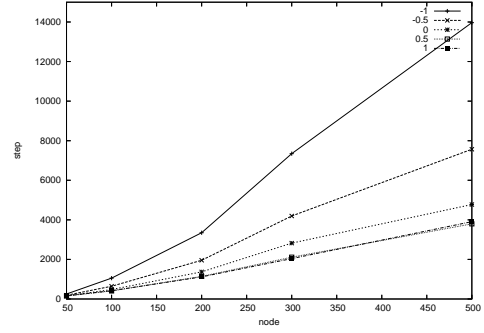


Fig. 6. max steps : ER model, $(k, l)=(0,1)$

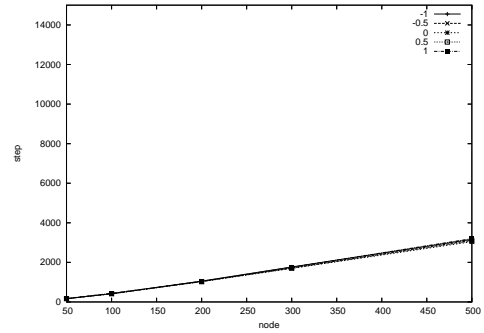


Fig. 7. max steps : WS model, $(k, l)=(0,1)$

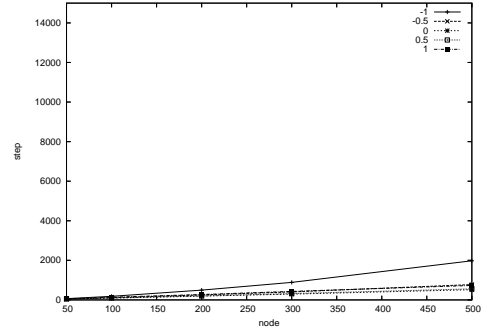


Fig. 8. max steps : BA model, $(k, l)=(0,1)$

Now, we review the relationship between steps and parameters k, l and β from Tables III and IV.

- $k = 0, l = 0$

In this case, the search strategy is equivalent to the β -simple random walk. Both the mean and max steps are minimum if $\beta \in \{0, 0.5, 1\}$ on each model. This might be derived from that in β -simple random walks on general finite graphs, the best β is 0.5 with respect to the hitting time, though $\beta = 0.5$ is not always the best in these cases.

Now we compare the results by the models. In WS model, the steps little depend on β , while in ER and BA models they greatly depend on β . The reason is that WS model networks have similarity to regular graphs

because they are rewired regular lattice graph whose rewiring probability p_w in our simulation is 0.1, very small. Namely, these networks seem to have almost same degrees, in which the transition probability of the β -simple random walk is identical regardless of β . The degree variance of WS networks in Table II supports this idea.

- $k \in \{1, 2\}, l = 0$

Both the mean and max steps at $\beta > 0$ are smaller than at $\beta < 0$ (also seen in Figures 9(a)-9(f) in appendix). The reason would be as follows: These cases $k > 0$ inherit the results for $k = 0$ and $l = 0$. Compared with the results at $k = 0$, the ones at $k = 1$ are greatly improved. On the contrary, at $k = 2$, the effect on ER and BA model is less, while on WS model the reduction effect is still observed. Networks of WS model, which have large C_G as shown in Table II, usually contain many triangles. Because of the triangles, the node second previously visited is likely to be adjacent to the current node and thus k -tabu may work effectively.

- $k \in \{0, 1, 2\}, l > 0$

Both the mean and max steps are reduced drastically. The effect of the l -look-ahead is prominent on BA model, followed by the ones on ER and WS models. This can be understood from the following reason: Networks of BA model have small L_G as shown in Table II, which implies that there are many nodes u that have many nodes within small distances. Since the look-ahead can check nodes with small distances without visiting, networks with small L_G can check more nodes; implicitly more nodes are visited.

As for the mean steps, those at $\beta < 0$ are rather smaller than at $\beta > 0$. This is also seen in Figures 3- 5. The search strategy with look-ahead property can check more nodes from the node with higher degree. Because of this, the search with small β (i.e., nodes with higher degrees have higher priority) can easily reach the target node. On the contrary, as for the max steps, those are at $\beta > 0$ smaller than at $\beta < 0$. Although we do not understand the exact reason, we consider it is because the performance of negative β strategies may greatly depend on locations of the start and the target nodes. For example, if the start node is located at an area of nodes with higher degrees and the target node is located at a sparse area, then the particle can hardly escape from the dense area. Thus, even if negative β strategies have good performance for the mean steps, in worst cases (max steps), they may require more steps to reach the target. The effect of k is similar to the one at $l = 0$.

It should be noted that negative β -random walks (higher degree nodes have higher priority) are not always better than positive β -random walks (lower degree nodes have higher priority), and only when $l \geq 1$ and for the mean steps, the former is better than the latter. These results imply that the goodness of other existing results (e.g., [14]) is not because

of the biased transition probability itself but because of the combination of the one and the look-ahead capability.

TABLE III
MEAN STEPS

| $n = 500$ | $l = 0$ | $l = 1$ | $l = 2$ |
|--------------|----------------|-------------|------------|
| $\beta=-1$ | ER WS BA | ER WS BA | ER WS BA |
| $k = 0$ | 2040 1363 3505 | 287 757 156 | 62 428 15 |
| $k = 1$ | 1240 825 1693 | 199 458 99 | 46 263 12 |
| $k = 2$ | 1237 670 1602 | 199 373 96 | 46 215 11 |
| $\beta=-0.5$ | ER WS BA | ER WS BA | ER WS BA |
| $k = 0$ | 1267 1349 1381 | 265 769 117 | 64 437 17 |
| $k = 1$ | 863 819 993 | 190 467 86 | 47 269 13 |
| $k = 2$ | 861 663 980 | 190 378 85 | 47 219 13 |
| $\beta=0$ | ER WS BA | ER WS BA | ER WS BA |
| $k = 0$ | 973 1355 979 | 268 790 121 | 69 450 23 |
| $k = 1$ | 681 819 735 | 190 477 84 | 50 275 16 |
| $k = 2$ | 680 660 733 | 189 384 84 | 50 223 16 |
| $\beta=0.5$ | ER WS BA | ER WS BA | ER WS BA |
| $k = 0$ | 902 1380 1031 | 297 820 154 | 79 468 34 |
| $k = 1$ | 599 824 644 | 196 489 86 | 53 283 19 |
| $k = 2$ | 598 660 643 | 196 391 86 | 53 228 19 |
| $\beta=1$ | ER WS BA | ER WS BA | ER WS BA |
| $k = 0$ | 993 1426 1362 | 366 861 222 | 100 492 54 |
| $k = 1$ | 570 835 610 | 207 502 88 | 57 291 20 |
| $k = 2$ | 570 663 610 | 207 399 88 | 57 233 20 |

TABLE IV
MAX STEPS

| $n = 500$ | $l = 0$ | $l = 1$ | $l = 2$ |
|--------------|------------------|-----------------|---------------|
| $\beta=-1$ | ER WS BA | ER WS BA | ER WS BA |
| $k = 0$ | 53132 5378 24997 | 13967 3168 1979 | 2816 2506 399 |
| $k = 1$ | 14653 3238 8611 | 6667 1734 895 | 2084 1370 218 |
| $k = 2$ | 14543 2677 7970 | 6662 1343 855 | 2028 1099 214 |
| $\beta=-0.5$ | ER WS BA | ER WS BA | ER WS BA |
| $k = 0$ | 18048 4449 5921 | 7568 3119 776 | 2212 2416 242 |
| $k = 1$ | 6562 2638 2991 | 4228 1694 536 | 1643 1364 170 |
| $k = 2$ | 6475 2140 2939 | 4245 1287 522 | 1653 1088 170 |
| $\beta=0$ | ER WS BA | ER WS BA | ER WS BA |
| $k = 0$ | 7980 4016 2633 | 4775 3044 516 | 1927 2376 212 |
| $k = 1$ | 3509 2301 1740 | 3003 1701 396 | 1394 1363 151 |
| $k = 2$ | 3537 1834 1764 | 3048 1283 398 | 1414 1075 149 |
| $\beta=0.5$ | ER WS BA | ER WS BA | ER WS BA |
| $k = 0$ | 4886 3947 3174 | 3797 3078 561 | 1898 2412 248 |
| $k = 1$ | 2620 2153 1751 | 2518 1718 361 | 1274 1394 147 |
| $k = 2$ | 2660 1643 1756 | 2521 1277 358 | 1272 1065 143 |
| $\beta=1$ | ER WS BA | ER WS BA | ER WS BA |
| $k = 0$ | 4392 3976 7973 | 3908 3200 731 | 2124 2475 338 |
| $k = 1$ | 2376 2130 1708 | 2327 1711 343 | 1248 1372 149 |
| $k = 2$ | 2344 1605 1727 | 2266 1300 343 | 1218 1088 149 |

V. CONCLUSION

In this paper, we have presented a generic search strategy with three parameters k -tabu, l -look-ahead and β -transition probability, which formalizes existing random walk-based strategies on large-scale networks. One virtue of this framework is to provide a good perspective, which can make us understand relationship among the existing search strategies and also give some guidelines to design a good search strategy according to some graph quantities. Through the computational simulations for networks by standard graph generating models, ER, WS and BA, we discuss relationship between these three parameters in our framework and the properties of networks. The results of the simulations are summarized as follows: In terms of the number of steps, k -tabu and l -look-ahead property work more effectively on networks with large C_G and small L_G , respectively. As for the mean steps, the strategy without look-ahead would be better off moving to a node with a lower degree by priority, while the one with look-ahead would take the opposite way, which matches the results of the existing studies. On the contrary, concerning the max steps, moving to a node with a lower degree is better regardless of the usage of the look-ahead property.

There are several directions for further studies considered. One of the most important directions is to give theoretical analyses for random walks on small-world or scale-free networks. Another direction, quantifying the costs of tabu and look-ahead, also might be an interesting research topic.

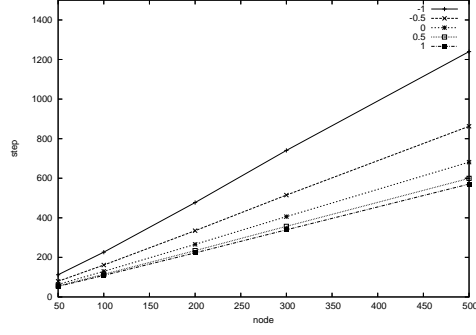
REFERENCES

- [1] R. Albert, H. Jeong and A.-L. Barabási, "The diameter of the world-wide web," *Nature*, Vol.401, 130–131, 1999.
- [2] D.J. Aldous, "On the time taken by random walks on finite groups to visit every state," *Z. Wahrsch. verw. Gebiete* 62, 361–393, 1983.
- [3] R. Albert and A.-L. Barabási, "Statistical Mechanics of Complex Networks," *Reviews Of Modern Physics*, Vol.74, 47–97, 2002.
- [4] L.A.N. Amaral, A. Scala, M. Barthélemy and H.E. Stanley, "Classes of small-world networks," *Proceeding of The National Academy of Sciences*, Vol.97, 11149–11152, 2000.
- [5] A.-L. Barabási, R. Albert and H. Jeong, "Mean-field theory for scale-free random networks," *Physica A*, Vol.272, 173–187, 1999.
- [6] G. Brightwell and P. Winkler, "Maximum Hitting Time for Random Walks on Graphs," *J.Random Structures and Algorithms*, Vol.3, 263–276, 1990.
- [7] H. Ebel, L.-I. Mielsch and S. Bornhold, "Scale-free topology of e-mail networks," *Physical Review E*, Vol.66, 1–4, 2002.
- [8] P. Erdős and A. Rényi, "On random graphs," *Publ. Math. Debrecen*, 6:290–297, 1959.
- [9] P. Erdős and A. Rényi, "On the Evolution of random graphs," *Publ. Math. Inst. Hung. Acad. Sci.*, Vol.5, 17–61, 1960.
- [10] U. Feige, "A tight upper bound on the cover time for random walks on graphs," *J.Random Structures and Algorithms*, Vol.6, 51–54, 1995.
- [11] C.P. Herrero, "Self-avoiding walks on scale-free networks," *Physical Review E*, Vol.71, 1–8, 2005.
- [12] S.Ikeda, I.Kubo and M.Yamashita, "Reducing the Hitting and the Cover Times of Random Walks on Finite Graphs by Local Topological Information," *Proceedings of The 2003 International Conference on VLSI*, 203–207, 2003.
- [13] H. Jeong, B. Tombor, R. Albert, Z.N. Oltvai and A.L. Barabási, "The large-scale organization of metabolic networks," *Nature*, Vol.407, 651–654, 2000.
- [14] B.I. Kim, C.N. Yoon, S.K. Han and H. Jeong, "Path finding strategies in scale-free networks," *Physical Review E*, Vol.65, 1–4, 2002.
- [15] R. Motowani and P. Raghavan, *Randomized Algorithms*, Cambridge, 1995.
- [16] S. Render, "How popular is your paper? An empirical study of the citation distribution," *Eur.Phys.J.B*, Vol.4, 131–134, 1998.
- [17] T. Walsh, "Search in a small world," *Proceeding of IJCAI*, 1172–1177, 1999.
- [18] D.J. Watts and S.H. Strogatz, "Collective dynamics of small-world networks," *Nature*, Vol.393, 440–442, 1998.
- [19] S. Yang, "Exploring complex networks by walking on them," *Physical Review E*, Vol.71, 1–5, 2005.

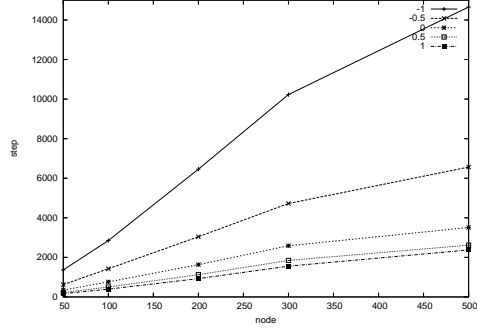
This appendix provides more results of the computational simulations that have been omitted due to the readability. It may be read to the discretion of the program committee.

APPENDIX

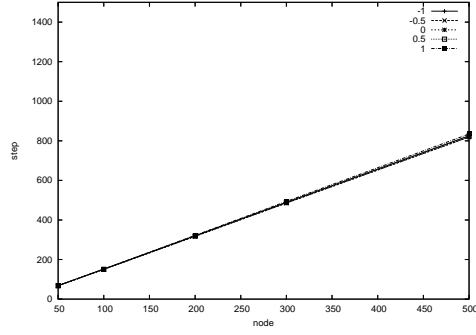
This appendix provides computational simulation results, which are not listed in the main body. The mean and max steps seem to increase in near-linear with the number of nodes. The right and left side figures show the mean and max steps respectively on each model for $(k, l, \beta) \in (\{1, 0, \{-1, -0.5, 0, 0.5, 1\}\})$. As we see in section IV, the mean and max steps at $\beta > 0$ are smaller than at $\beta < 0$ for $k = 1$.



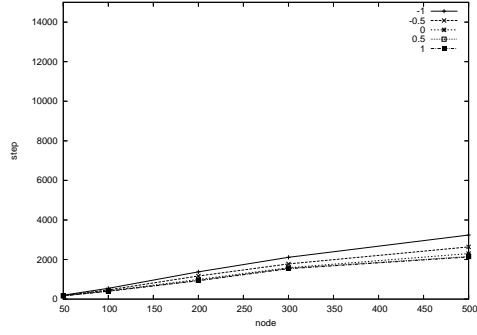
(a) mean steps : ER model



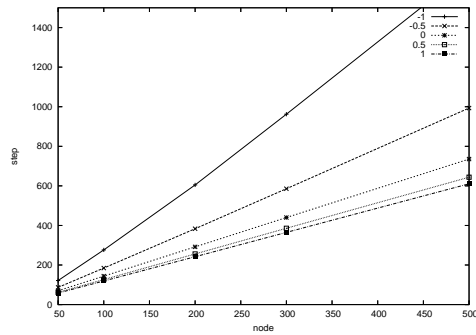
(b) max steps : ER model



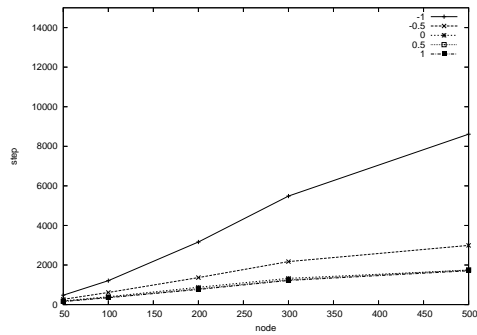
(c) mean steps : WS model



(d) max steps : WS model



(e) mean steps : BA model



(f) max steps : BA model

Fig. 9. Results for $(k, l, \beta) \in (1, 0, \{-1, -0.5, 0, 0.5, 1\})$