Comparison of Random Walk Strategies for Ad Hoc Networks

Santpal S. Dhillon and Piet Van Mieghem

Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
P.O Box 5031, 2600 GA Delft, The Netherlands
Email: s.dhillon@ewi.tudelft.nl and P. VanMieghem@ewi.tudelft.nl

Abstract— **We study different variations of the random walk (RW) such as RW with memory, RW with look-ahead, RW using highest degree and RW proportional to the degree for random graphs. One of our insights is that comparison of different RW strategies based on the expected hopcount or weight is not sufficient. The expected hopcount for certain RW variations such as RW using highest degree is small. However, these strategies generally lead to infinite loops. Furthermore, the simulations show that RW using highest degree with look-ahead and memory is the most efficient algorithm for searching in random graphs.**

I. Introduction

Random walk (RW) is a routing or search algorithm that uses no topology information. Since RW does not use any topology information, the next hop is chosen uniformly among the neighbors of the node. Other variations of the RW use partial topology information such as the RW strategy where the next hop is chosen as the node with maximum degree, and the RW strategy where the probability of choosing the next node is proportional to the nodal degree. The term local search algorithm or path finding strategies is also used for different variations of RWs [1], [10].

Random walks (RWs) have numerous applications in ad hoc networks and overlay networks. Overlay networks such as Gia [3] and Gnutella[1] use RWs for resource discovery. RWs and its variations have also been used as a tool for searching on different graph topologies [1]. As a sampling technique, RWs have been used for providing membership services in ad hoc networks [5], [19] that provide the nodes in the network with a view of the other nodes in network and that are used by various applications such as location services, peer sampling services and random overlay constructions [19].

In this paper, we investigate the efficiency of searching with different RW strategies in ER random graphs $G_p(N)$ [12]. Since wireless ad hoc networks can be modeled by a geometric random graph [18], we believe our analysis is particularly important in the case of ad hoc networks. Our analysis shows that RW strategy using a combination of highest degree, look-ahead and complete memory is the most effective strategy for searching on random graph topologies. Also, for different RW strategies, if the probability of finding the destination is large, the expected hopcount for the particular strategy is also large and vice versa.

The rest of this paper is organized as follows. A literature overview of RWs and their applications is presented in Section 2. Section 3 explains our analysis of RWs and how our RW strategies differ from the previous search strategies. In section 4, we compute the hopcount distribution for a RW and a RW with no retracing of steps in a complete graph. Sections 5 and 6 present the analysis of RW strategies on random graphs and weighted random graphs respectively. Finally, we give the conclusions in section 7.

II. Literature Review

RWs have been studied as search algorithms on scale-free and power law graphs [1], [9], [10]. In [1] and [10], RW strategies where the next hop is chosen as the node with the highest degree and with no retracing of steps have been analyzed in terms of expected hopcount and weight. We show that strategies such as RW using highest degree and RW with no retracing of steps can lead to infinite loops. Therefore, the comparison of different RW strategies should also include an analysis

[1]The Gnutella 0.4 protocol specification, *http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf.*

of infinite loops, in addition, to the expected hopcount comparison.

Unstructured overlay networks such as Gia proposed by Chawathe *et al.* [3] and Gnutella build a random graph and use flooding or RWs to discover data stored at different nodes. RWs have been shown to induce lower overhead than constrained flooding used by the current versions of Gnutella [4], [14]. In the original Gia [3], the RWs were biased to prefer nodes with higher capacity but Castro *et al.* [2] have shown that preferring nodes with higher degree leads to a higher success rate and a lower delay. Thus, further improvements have been proposed to Gia in which RWs are biased towards the higher degree nodes [2]. Also, variations of RWs have been proposed in which there are no loops [2].

In mobile agent based routing, the mobile agents perform a RW or a variant of the RW. In Ant-Net, loop-erased RWs are used by the mobile agents [6]. Mobile agents using RW have been proposed for providing membership services for ad-hoc networks by Dolev *et al.* [8]. In [19], Bar-Yossef *et al.* develop a membership service for ad hoc networks based on RW using highest degree. They show that the performance of such membership service is superior to other existing membership services based on gossiping or flooding [19].

The analysis of RWs has been an active topic of research [7], [8], [11], [16]. Lovász [8] presents a detailed survey of the RWs. An exact analysis of RWs on graph topologies such as lattice, 1 dimension, torus has been studied in [7]. For a detailed mathematical analysis of RWs, we refer the reader to Chung [11].

III. Random Walks

The topology and link weight structure are essential for characterizing the network [15]. Therefore, we study the RW strategies under two distinct regimes. In the first case, we assume that all the link weights are 1. Thus, the RW strategies use only topological information such as degree for choosing one of the neighbors as the next hop. In the second case, we investigate different RW strategies on weighted random graphs. The weighted graphs offer a more realistic view of network. For example, the link capacities in ad hoc networks [2], [4] can be represented as weighted edges. In weighted graphs, the RW strategies can use link weights, in addition to any topological information, for choosing the next hop.

A. Random Walk with memory M

A first-in first-out (FIFO) list, called the memory list $\mathcal{M}$, is maintained. The memory list $\mathcal{M}$ contains the node identifiers n_j of the last M nodes visited during the RW, i.e. $\mathcal{M} = \{n_1, n_2, .., n_M\}$, where $M = \mid \mathcal{M} \mid$ represents the number of elements in the memory list $\mathcal{M}$. The next hop is chosen uniformly among the neighbors of the node that are not in the memory list $\mathcal{M}$. The RW$_\text{M}$ strategy is equivalent to the search queries proposed for overlay networks where the structure is used to ensure that nodes are visited only once during a query [2].

Using memory, the loops are eliminated in RWs. The one hop loops can be prevented by using $M = 1$, both the two hop and one hop loops can be prevented by using $M = 2$ and so on. Thus, a complete memory $M = N - 1$ totally eliminates loops in the RWs. However, the introduction of memory $(M \geq 1)$ in RWs leads to a *deadlock*. Fig. 1 explains the concept of deadlock in a RW with memory. Let us assume that the RW has a memory $M = 3$ represented as $\mathcal{M} = \{n_1,\ n_2,\ n_3\}$ where n_3 represents the last node visited. Consider the situation where the RW enters the cluster **B** at node 1. Suppose that the RW leads to a path $1 \longrightarrow 2 \longrightarrow 3 \longrightarrow 4$ in the cluster **B**. At node 2, the memory list $\mathcal{M}$ contains $\{1\}$ and at node 3 the memory list $\mathcal{M}$ contains $\{1, 2\}$. At node 4, the memory list $\mathcal{M}$ contains $\{1, 2, 3\}$ and since all the neighbors of node 4 are in the memory list $\mathcal{M}$ and no self loops are allowed, the RW cannot move forward nor backward. Therefore, a deadlock prevents the RW from ever reaching the destination. The above analysis shows that there are two distinct regimes possible for RWs. Without memory, i.e. $M = 0$, the RWs can have loops but no deadlocks. For complete memory, the RWs can only have deadlocks and no loops. When the value of the memory M is such that $0 < M < N - 1$, the RWs can have loops and deadlocks. In the implementation of Gia, Castro *et al.* [2] have used RWs with complete memory as a query.

B. Random Walk with look-ahead

The RW with look-ahead (RW$_\text{LA}$) is a RW that uses information about the neighbors [17]. In RW$_{\text{LA}=1}$, the destination is chosen as the next node if it is among the neighbors of the node, else the next hop is chosen uniformly among all the neighbors of the node. The use of look-ahead 1 has been assumed in recent work [1], [9] in searching on graphs. In RW$_{\text{LA}=1}$, there are no deadlocks. The RW$_{\text{LA}=1}$ can also be used in combination with memory M (see Table I).

C. Random Walk using highest degree

In RW using highest degree (RW$_\text{HD}$), at any node u the next node v is chosen such that $d_v = \max\limits_{t \epsilon N_u} d_t$, where N_u

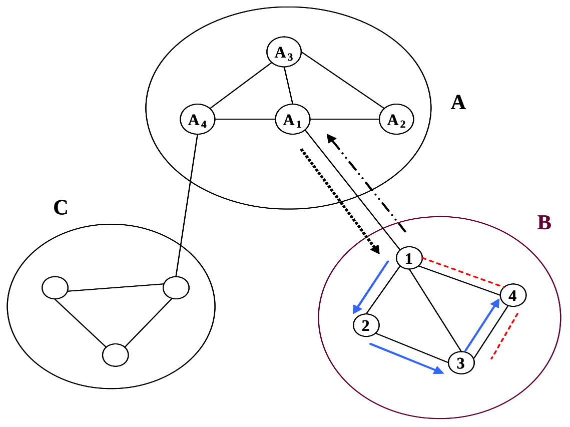

Fig. 1. Explanation of deadlock in (a) RW with memory: The path $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$ is shown by arrows. At node 4, the dashed lines indicate the links that cannot be used with memory $M \geq 3$. (b) RW using highest degree: Node A_1 chooses node 1 as next hop and node 1 chooses node A_1 as next hop leading to an infinite loop.

is the set of neighbors of u. If the degree of neighboring nodes is equal, then the next hop is chosen randomly among the equal degree neighbors. The RW_{HD} strategy can end in a deadlock. Fig. 1 illustrates the concept of deadlock in RW_{HD}. At node A_1, node 1 is chosen as the next node since it is the highest degree neighbor of node A_1. At node 1, node A_1 is chosen as the next hop since node A_1 is the highest degree neighbor of node 1. Thus, the RW_{HD} ends in an infinite loop or a deadlock. The RW_{HD} can be used in combination with different values of memory M and look-ahead (see Table I).

D. Random Walk proportional to the degree

The probability of choosing the neighboring node is proportional to the degree of the neighboring node (RW_{PD}). At node u, $q_{ut} \propto d_t$ where q_{ut} is the probability of choosing the neighbor t. By normalization, we obtain $q_{ut} = \frac{d_t}{\sum\limits_{t\epsilon N_u} d_t}$. There are no deadlocks in RW_{PD}. In a regular graph, the RW_{PD} is equivalent to RW.

E. Random Walk using minimum link weight

In RW using minimum link weight (RW_{W}), at any node u the next node v is chosen such that $w_{uv} = \min\limits_{t\epsilon N_u} w_{ut}$, where w_{ut} is the weight of link between node u and node t. Deadlocks in RW using minimum link weight are similar to the deadlocks occurring in RW_{HD}. Indeed, the degree can be considered as a special kind of link weight.

Table II gives the complexity of different RW strategies in random graphs. The convergence time is the worst case hopcount to reach a destination and is equal to the

TABLE I

EXPLANATION OF DIFFERENT RW STRATEGIES.

RW strategy	Next hop
RW	uniformly among the neighbors.
RW_{M}	uniformly among the neighbors that are not in the memory list M.
$\text{RW}_{\text{LA = 1}}$	destination, if it is a neighbor, else uniformly among the neighbors.
$\text{RW}_{\text{LA = 1; M}}$	destination, if it is a neighbor, else uniformly among the neighbors that are not in the memory list M.
RW_{HD}	neighbor with the highest degree.
$\text{RW}_{\text{HD; M}}$	highest degree neighbor that is not in the memory list M.
$\text{RW}_{\text{HD; LA = 1}}$	destination, if it is a neighbor, else highest degree neighbor.
$\text{RW}_{\text{HD; LA = 1; M}}$	destination, if it is a neighbor, else highest degree neighbor not in the memory list M.
RW_{PD}	probability of choosing the neighbor is proportional to the degree of neighbor. $\text{RW}_{\text{PD; M}}$ and $\text{RW}_{\text{PD; LA = 1}}$ use RW_{PD} with memory and look-ahead respectively.
$\text{RW}_{\text{PD; LA = 1; M}}$	destination, if it is a neighbor, else probability of choosing the node is proportional to the degree of the node but nodes in the memory list M are not considered.
RW_{W}	link with minimum weight.
$\text{RW}_{\text{W; LA = 1; M}}$	destination, if it is among the neighbors, else neighbor with minimum weight among the neighbors that are not in the memory list M.

cover time [8]. h represents the hopcount for the RW strategy and the maximum value of h is given by the convergence time for the particular RW strategy.

Some of the RW strategies listed in Table I have been compared in literature. Adamic *et al.* [9] have analyzed two RW strategies, i.e. RW with memory $M = 1$ and look-ahead 1, and RW using look-ahead 1, highest degree and complete memory. However, an analysis of deadlocks and how they affect the performance of different RW strategies is missing. Some of the other studies use alternative approaches to remove deadlocks. In [1], Kim *et al.* compare RW, RW with highest degree and RW proportional to the degree but remove loops in the paths while calculating the mean hopcount between the source and destination nodes. Thadakamalla *et al.*

TABLE II
COMPLEXITY OF DIFFERENT RW STRATEGIES.

RW strategy	convergence time	Memory overhead	Complexity
RW	$O(N\log N)$	$O(1)$	$O(h)$
RW_M	$O(N)$	$O(pN)$	$O(hMpN)$
RW_HD	$O(N)$	$O(pN)$	$O(hpN)$
RW_LA	$O(N\log N)$	$O(pN)$	$O(hpN)$

[10] compare RW strategies such as RW, RW using minimum link weight, RW using highest degree, look-ahead and complete memory on weighted networks. In their comparison, Thadakamalla *et al.* [10] have assumed that when a deadlock happens, the next hop is chosen as the previous hop.

IV. RANDOM WALK ON A COMPLETE GRAPH

The probability distribution of RW on a complete graph K_N follows a geometric distribution [8],

$$\Pr[H=m]=\frac{1}{N-1}\left(1-\frac{1}{N-1}\right)^{m-1} \tag{1}$$

We calculate the expression for RW_M on a complete graph K_N for $m\le M$. The probability that the destination is found in hop m equals the probability that destination has not been found in the last $m-1$ hops multiplied by the probability that the next chosen node is the destination. Let us denote the event that the RW_M process is not at the destination after $m-1$ hops by B_{m-1}. Then, $\Pr[B_{m-1}]=1-\sum_{j=1}^{m-1}\Pr\left[\hat{H}=j\right]$. Since $m-1$ nodes have been visited previously and no self loops are allowed, the next hop can be chosen as any of the remaining $N-((m-1)+1)$ nodes. Therefore, the probability that a chosen node is the destination is $\frac{1}{N-m}$. Hence,

$$\begin{aligned}\Pr\left[\hat{H}=m\right] &= \Pr[X_m=j|B_{m-1}]\cdot\Pr[B_{m-1}]\\ &= \frac{1}{N-m}\left(1-\left(\frac{m-1}{N-1}\right)\right)\\ &= \frac{1}{N-1}\end{aligned} \tag{2}$$

which means that the hopcount for RW_M is a uniform random variable on $[1,N-1]$ when $1\le m\le M$.

We now compute the probability distribution for RW_M for hopcount $>M$. Let us denote the event that the RW_M is not at the destination after M hops by B_M. Then, $\Pr[B_M]=1-\sum_{j=1}^{M}\Pr\left[\hat{H}=j\right]$. Since in K_N each node is connected to all other nodes and M nodes are in the memory list $\mathcal{M}$ and no self loops are allowed, the next hop can be chosen as any of the remaining $N-(M+1)$ nodes. Therefore, the probability that a chosen node is the destination is $\frac{1}{N-(M+1)}$. Hence,

$$\begin{aligned}\Pr\left[H_M=M+1\right] &= \Pr[X_{M+1}=j|B_M]\cdot\Pr[B_M]\\ &= \frac{1}{N-(M+1)}\left(1-\frac{M}{N-1}\right)\\ &= \frac{1}{N-1}\end{aligned}$$

The probability that the destination is found in hop $M+2$ can be computed as follows. The probability that the RW_M is in a non-destination after $M+1$ hops is $\Pr[B_{M+1}]=1-\sum_{j=1}^{M}\Pr\left[\hat{H}=j\right]-\Pr\left[H_M=M+1\right]$. The next hop can be chosen as any of the $N-(M+1)$ nodes that are not in the memory list $\mathcal{M}$. (The source node can now be chosen as the next node since it is no longer in the memory list $\mathcal{M}$.) The probability that a chosen node is the destination is $\frac{1}{N-(M+1)}$. Thus,

$$\begin{aligned}\Pr\left[H_M=M+2\right] &= \Pr[X_{M+2}=j|B_{M+1}]\cdot\Pr[B_{M+1}]\\ &= \frac{1}{N-1}\left(\frac{N-(M+2)}{N-(M+1)}\right)\end{aligned}$$

The probability distribution of RW with memory M for $m\ge M+1$ is,

$$\Pr\left[H_M=m\right]=\frac{1}{N-1}\left(\frac{N-(M+2)}{N-(M+1)}\right)^{m-M-1} \tag{3}$$

Thus, the probability distribution for a RW_M on complete graph K_N is a uniform random variable on $[1,M]$ and geometric variable on $[M+1,N-1]$.

V. RANDOM WALKS ON RANDOM GRAPHS

In this section, we study the hopcount distribution of the RW on ER random graphs $G_p(N)$. In each simulation, 10^6 random graphs of the class $G_p(N)$ are constructed. The source and destination are chosen uniformly. Fig. 2 shows the pdf of the hopcount for RW between a source and a destination in a random graph without memory for $N=25$ and for different values of the link density p. Fig. 2 also shows the analytic result (5) for the complete graph. In general, the hopcount distribution for RW is close to a geometric random variable. Furthermore, the legend in Fig. 2 shows that the mean hopcount increases as the link density p is decreased and that for $p<1$, the pdf deviates from the geometric distribution (1).

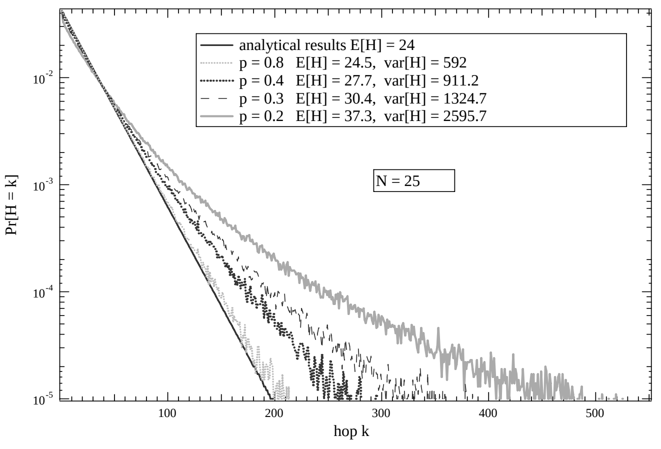

Fig. 2. The pdf of the hopcount for RW on a complete graph eq. (1) and a random graph for $N = 25$ and for different values of the link density p.

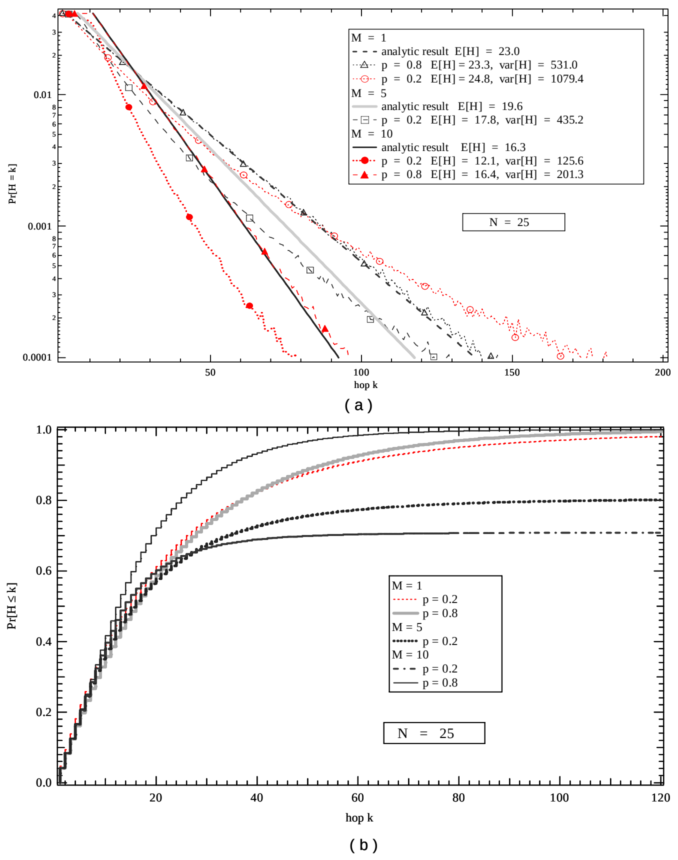

Fig. 3. (a) The pdf of the hopcount for $\text{RW}_{\text{M = 1}}$, $\text{RW}_{\text{M = 5}}$ and $\text{RW}_{\text{M = 10}}$ in a complete graph eq. (2 and 3) and a random graph for $N = 25$ and for different values of the link density p. (b) The distribution of the hopcount $\Pr[H \leq k]$.

Fig. 3 shows the pdf of the hopcount for $\text{RW}_{\text{M = 1}}$, $\text{RW}_{\text{M = 5}}$ and $\text{RW}_{\text{M = 10}}$ strategies on a random graph for $N = 25$ and for different values of the link density p. The introduction of memory in RWs leads to a reduced average hopcount at the expense of deadlocks. Fig. 3 (a) shows that the hopcount distribution for $\text{RW}_{\text{M = 1}}$, $\text{RW}_{\text{M = 5}}$ and $\text{RW}_{\text{M = 10}}$ is close to a geometric random variable. Moreover, for large values of the link density p, the hopcount distribution $(2, 3)$ for RW_{M} on a random graph is similar to hopcount distribution for RW_{M} on a complete graph. In Fig. 3, the average hopcount is calculated over the iterations in which there is no deadlock. However, the pdf is plotted over all the iterations. If the RW ends in deadlock, then the cumulative distribution function (cdf) of the hopcount distribution, $\Pr[H \leq k]$, does not tend to 1 as $k \to \infty$. Thus,

$$\Pr[\text{deadlock}] = 1 - \lim_{k\to\infty} \Pr[H \leq k] \tag{4}$$

shows the probability of ending in deadlocks. Fig. 3(b) shows that the probability of a deadlock increases with decreasing link density p.

A. $RW_{LA = 1}$

In $\text{RW}_{\text{LA = 1}}$, each node has information about the neighbors and the destination is chosen as the next hop if it is among the neighbors (Table I). When the link density p is increased, the expected hopcount for $\text{RW}_{\text{LA = 1}}$ decreases (The mean node degree in a random graph [12] is $p(N-1)$). Additional simulations show that at large values of link density p, the hopcount distribution for $\text{RW}_{\text{LA=1}}$ is close to a geometric random variable with parameter p.

An estimate of the hopcount distribution for $\text{RW}_{\text{LA = 1; M = N-1}}$ on a random graph $G_p(N)$ is $\Pr[H = k] \cong p(1-p)^{k-1}$. In $\text{RW}_{\text{LA = 1; M = N-1}}$, the probability that the hopcount is one is equal to p i.e., the probability that the destination is directly connected to the source node. If the destination is not reached in hop 1, then the RW process is in one of the $N - 2$ nodes with a probability $1 - p$. With no retracing of steps allowed, the probability that the destination is connected directly to the current node is p. Hence, the probability that the hopcount is two equals $p(1 - p)$. This analysis can be extended to compute the general hopcount distribution for $\text{RW}_{\text{LA = 1; M = N-1}}$. However, the above analysis is an approximation since we neglect the probability of deadlocks and history of the links that disappear[2]. Intuitively, for large values of link density p, the $\Pr[H \geq 1]$ is small and the above analysis is

[2]Since no retracing of path is allowed, the probability of destination being connected to the current node is less than p.

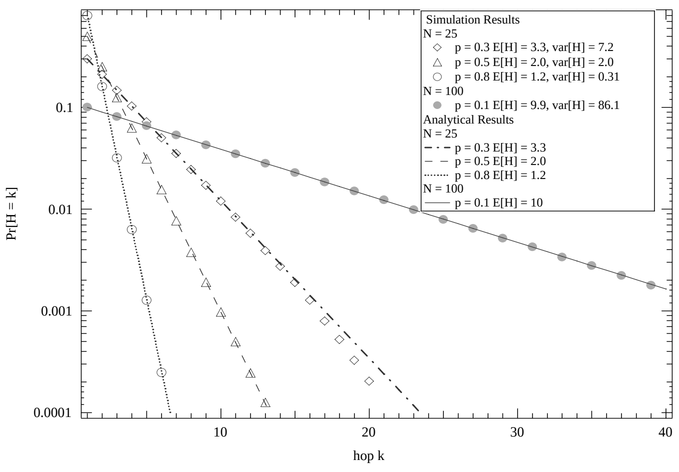

Fig. 4. Comparison of the expression $\Pr\left[H = k\right] \cong p\left(1-p\right)^{k-1}$ and simulation results for $\text{RW}_{\text{LA = 1; M = N-1}}$ on a random graph for $N = 25, 100$ and different values of the link density p.

a good approximation. Fig. 4 compares the simulation results and the analytical results for $\text{RW}_{\text{LA = 1; M = N-1}}$ for $N = 25, 100$ and different values of the link density p.

1) $RW_{LA = k}$: The $\text{RW}_{\text{LA = k}}$ is a RW strategy in which each node has topology information upto k hops, i.e. each node maintains a level set[3] L_N upto level k. If the destination is located within the level set L_N, a shortest path is chosen. However, if the destination is not located within the level set L_N, then the next node is chosen uniformly among the neighbors of the node. When each node maintains topology information of the whole network, the $\text{RW}_{\text{LA = k}}$ strategy leads to deterministic routing. Moreover, there is a phase transition in the performance of $\text{RW}_{\text{LA = k}}$ when k is equal to the average path length and for k greater than average path length, the performance of $\text{RW}_{\text{LA = k}}$ is similar to the shortest path routing [13].

When look-ahead $\geq$ 2, the probabilities that the hopcount is one and two can be calculated exactly for ER random graphs [15]: $\Pr\left[H_N = 1\right] = p$ and

$$\Pr\left[H_N = 2\right] = \left(1-p\right)\left(1-\left(1-p^2\right)^{N-2}\right) \qquad (5)$$

As the link density p and the network size N are increased, $\Pr[H_N > 2] = (1-p)\left[1-p^2\right]^{N-2}$ becomes negligible. Therefore, for large values of link density p, the performance of $\text{RW}_{\text{LA = 2}}$ is comparable to the Dijkstra's shortest path algorithm. Fig. 5 shows the comparison for $\text{RW}_{\text{LA = 2}}$ and the Dijkstra's shortest path

[3]Level set at level 1 is defined [15] as the set of nodes 1 hop away from the source node, level set at level 2 is defined as the set of nodes 2 hops away from the source node and so on.

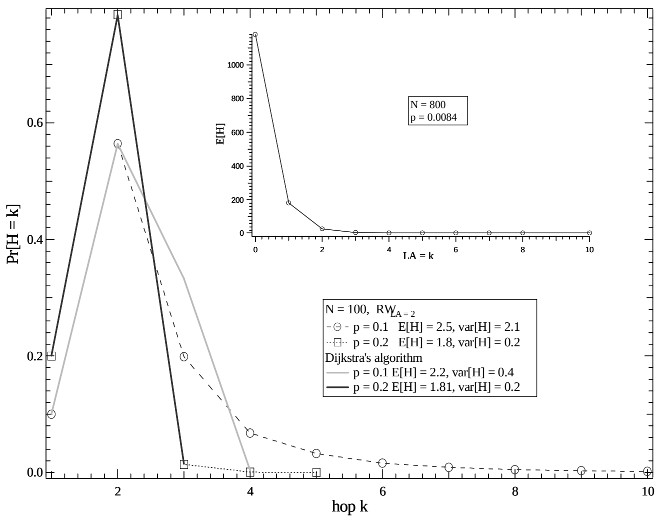

Fig. 5. Comparison of the $\text{RW}_{\text{LA = 2}}$ and the Dijkstra's shortest path algorithm for $N = 100$ and different values of the link density p. The insert shows expected hopcount for different values of look-ahead for $N = 800$ and $p = 0.0084$.

algorithm for $N = 100$ and different values of the link density p. The insert in Fig. 5 shows expected hopcount for different values of look-ahead for $N = 800$ and $p = 0.0084$. As shown in [15], the expected hopcount between any two nodes in ER random graphs can be approximated by $\frac{\log N}{\log p(N-1)}$. Therefore, for look-ahead greater than $\frac{\log N}{\log p(N-1)} = 3.5$, the expected hopcount for $\text{RW}_{\text{LA = k}}$ is constant and equal to the expected hopcount obtained by using Dijkstra's algorithm (see insert Fig. 5).

B. RW_{HD}

In RW_{HD}, the next hop is chosen as the neighboring node with the highest degree (Table I). Fig. 6 shows the simulation results for RW_{HD}, $\text{RW}_{\text{HD; M = N-1}}$, $\text{RW}_{\text{HD; LA = 1}}$, and $\text{RW}_{\text{HD; LA = 1; M = N-1}}$ for the random graph with $N = 25, 100$ and different values of the link density p. Fig. 6 shows that the RW_{HD} ends up in an infinite loop with a high probability. The expected hopcount $E[\text{hopcount}]$:

$$\text{RW}_{\text{HD}} < \text{RW}_{\text{HD; LA=1}} < \text{RW}_{\text{HD; LA=1; M=N-1}} < \text{RW}_{\text{HD; M=N-1}}$$

and the probability of deadlock for different strategies $\Pr[\text{deadlock}]$:

$$\text{RW}_{\text{HD}} > \text{RW}_{\text{HD; M=N-1}} > \text{RW}_{\text{HD; LA=1}} > \text{RW}_{\text{HD; LA=1; M=N-1}}$$

By comparing the expected hopcount for different RW strategies, it may be concluded that RW_{HD} performs best. However, the probability of deadlocks is maximum in

$\mathrm{RW}_{\mathrm{HD}}$. Therefore, considering both the mean hopcount and the probability of deadlocks, $\mathrm{RW}_{\mathrm{HD;\,LA\,=\,1;\,M\,=\,N\text{-}1}}$ performs best among the RW strategies compared. In conclusion, $\mathrm{RW}_{\mathrm{HD}}$ without look-ahead or memory is ineffective on random graphs.

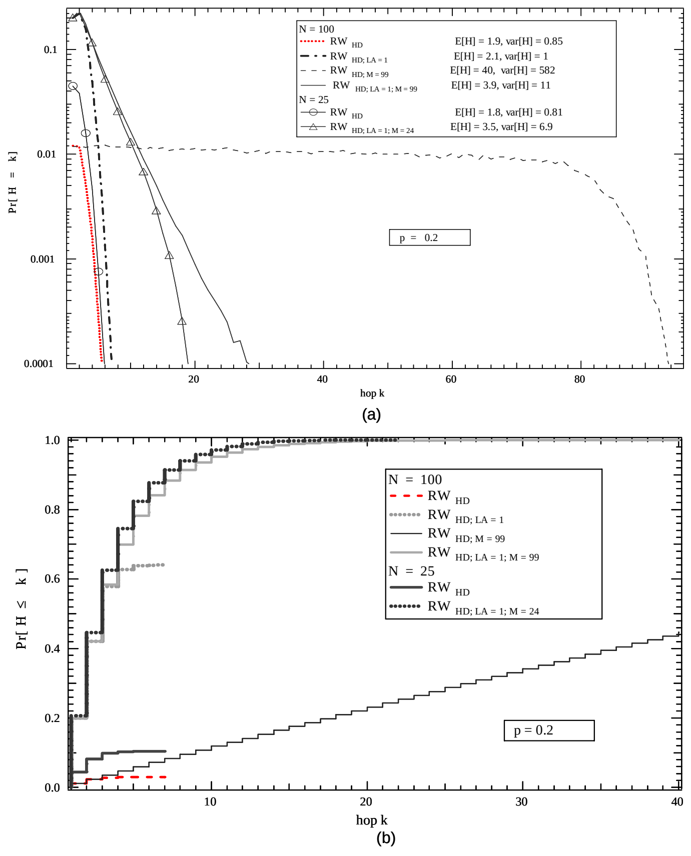

Fig. 6. (a) The pdf of the hopcount for $\mathrm{RW}_{\mathrm{HD}}$, $\mathrm{RW}_{\mathrm{HD;\,M\,=\,N\text{-}1}}$, $\mathrm{RW}_{\mathrm{HD;\,LA\,=\,1}}$ and $\mathrm{RW}_{\mathrm{HD;\,LA\,=\,1;\,M\,=\,N\text{-}1}}$ for $N = 25, 100$ and for different values of the link density p. (b) The distribution of the hopcount $\Pr[H \leq k]$

C. *Comparison of RW strategies for sparse graphs*

We compare the performance of RW strategies in sparse graphs i.e., random graphs with small link density p. When the link density p is large, the results for different RW strategies on a random graph $G_p(N)$ are similar to a complete graph K_N that can be solved analytically. Moreover, the average degree for most real networks like Gnutella [4] and Internet [9] is known to be small.

We first compare the expected hopcount for different RW strategies in which there are no deadlocks. In these strategies, the final destination is always reached. The four strategies that are compared include RW, $\mathrm{RW}_{\mathrm{LA\,=\,1}}$, $\mathrm{RW}_{\mathrm{PD}}$ and $\mathrm{RW}_{\mathrm{PD;\,LA\,=\,1}}$. Fig. 7 shows the mean hopcount for these strategies for different values of N with constant number of neighbors, i.e. $p\,(N-1) \sim pN = 5$. The simulations indicate that the mean hopcount is linear in N for these strategies and the mean hopcount for the $\mathrm{RW}_{\mathrm{LA\,=\,1}}$ and the RW strategies are related by the expression $E[H_{\mathrm{RW}_{\mathrm{LA\,=\,1}}}] \sim \frac{E[H_{\mathrm{RW}}]}{pN}$. Moreover, for large values of the link density p, $E[H_{\mathrm{RW}}] \sim N$ and therefore $E[H_{\mathrm{RW}_{\mathrm{LA\,=\,1}}}] \sim \frac{1}{p}$ (see Section 5(A)).

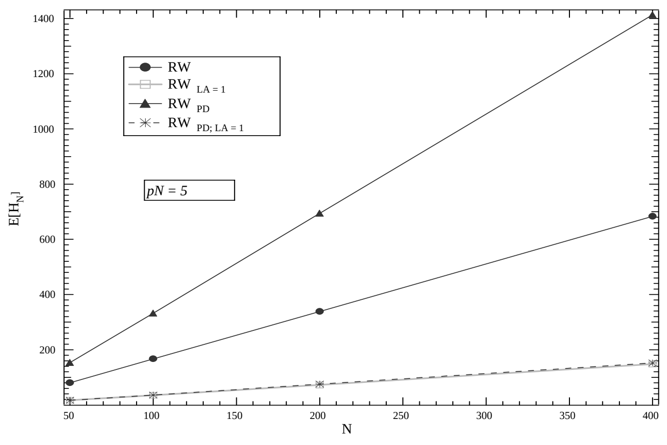

Fig. 7. The mean hopcount for different RW strategies without deadlocks as a function of N with constant number of neighbors ($pN = 5$).

We now compare different RW strategies with complete memory. The strategies that are compared include $\mathrm{RW}_{\mathrm{M\,=\,N\text{-}1}}$, $\mathrm{RW}_{\mathrm{LA\,=\,1;\,M\,=\,N\text{-}1}}$, $\mathrm{RW}_{\mathrm{PD;\,M\,=\,N\text{-}1}}$, $\mathrm{RW}_{\mathrm{HD;\,M\,=\,N\text{-}1}}$, $\mathrm{RW}_{\mathrm{PD;\,LA\,=\,1;M\,=\,N\text{-}1}}$ and $\mathrm{RW}_{\mathrm{HD;\,LA\,=\,1;\,M\,=\,N\text{-}1}}$. We also plot the *cdf* obtained with different RW strategies since with complete memory since the RW strategies may end in deadlocks. Fig. 8 shows the mean hopcount for these strategies for different values of N with constant number of neighbors $(p\,(N-1) \sim pN = 5)$, and the *cdf* of the hopcount distributions for $N = 200$ and $p = 0.025$. The results in Fig. 8 (a) are on log-log scale. $\mathrm{RW}_{\mathrm{HD;\,LA\,=\,1;\,M\,=\,N\text{-}1}}$ performs the best among the RW strategies compared. Thus, the $\mathrm{RW}_{\mathrm{HD}}$ performs well on random graphs provided that the search strategy uses look-ahead and memory. In terms of expected hopcount, the performance of $\mathrm{RW}_{\mathrm{LA\,=\,1;\,M\,=\,N\text{-}1}}$ is similar to $\mathrm{RW}_{\mathrm{HD;\,LA\,=\,1;\,M\,=\,N\text{-}1}}$. However, the probability of a deadlock in $\mathrm{RW}_{\mathrm{LA\,=\,1;\,M\,=\,N\text{-}1}}$ is much more than $\mathrm{RW}_{\mathrm{HD;\,LA\,=\,1;M\,=\,N\text{-}1}}$.

Additional simulation for ER random graphs with uniformly distributed link weights show that $\mathrm{RW}_{\mathrm{W;\,LA\,=\,1;\,M\,=\,N\text{-}1}}$ performs best in terms of expected weight while $\mathrm{RW}_{\mathrm{HD;\,LA\,=\,1;\,M\,=\,N\text{-}1}}$ performs best in terms

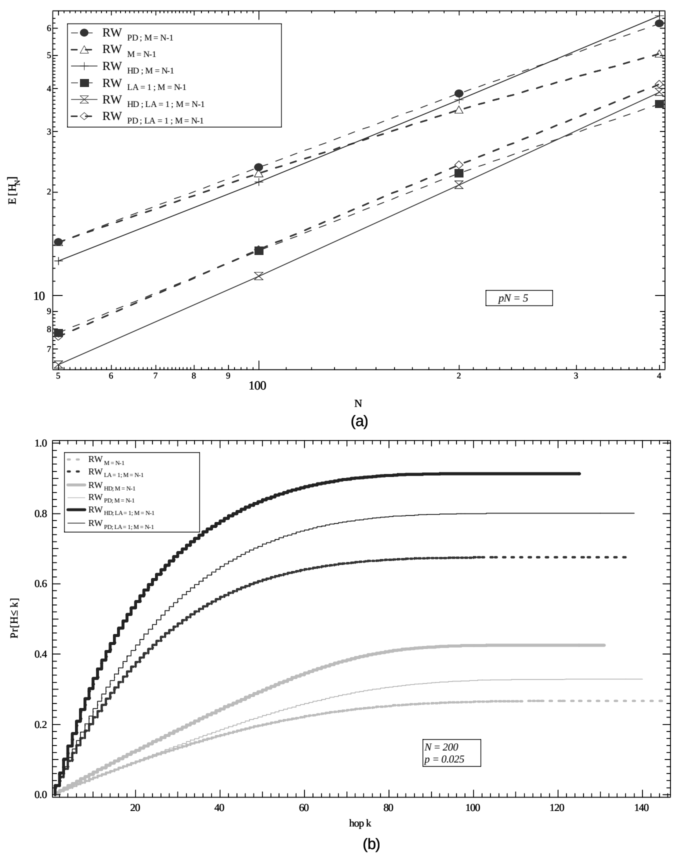

Fig. 8. (a) The mean hopcount for different RW strategies with complete memory as a function of N with constant number of neighbors $(pN = 5)$. (b) The distribution of the hopcount $\Pr[H \leq k]$ for different RW strategies for $N = 200$ and link density $p = 0.025$.

of expected hopcount. Moreover, the probability of deadlocks is least in $\text{RW}_{\text{HD; LA = 1; M = N-1}}$. Castro *et al.* [2] have shown that the $\text{RW}_{\text{HD; LA = 1; M = N-1}}$ strategy leads to a lower probability of deadlock as compared to the $\text{RW}_{\text{W; LA = 1; M = N-1}}$ strategy for a specific topology and link weight structure. Our simulations validate these observations for ER random graphs with uniformly distributed link weights.

VI. Conclusion

In this paper, we have analyzed RWs and different search strategies based on the RWs. The hopcount distribution for RW strategies based on memory and look-ahead is close to a geometric variable for random graphs. We also investigated the occurrence of deadlocks in various RW strategies. Deadlocks have been neglected in most of the previous search strategy comparisons. A mere comparison of the expected hopcount or weight of different RW strategies gives little insight into the performance of these strategies. The simulations showed that for RW strategies using highest degree or minimum link weight, the expected hopcount or weight is small but these strategies generally lead to deadlocks. In conclusion, RW strategies using highest degree or minimum link weight perform well when used in combination with memory or look-ahead.

References

[1] B. J. Kim, C. N. Yoon and S. K. Han And H. Jeong, *Path finding strategies in scale-free networks*, Physical Review E, vol. 65, 2002.

[2] M. Castro, M. Costa and A. Rowstron, *Peer-to-peer overlays: structured , unstructured, or both?*, Technical Report MSR-TR-2004-73, 2004.

[3] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham and S. Shenker, *Making Gnutella-like P2P Systems Scalable*, Proc. SIGCOMM Conference, 2003.

[4] Q. Lv, P. Cao, E. Cohen, K. Li and S. Shenker, *Search and Replication in Unstructured Peer-to-Peer Networks*, Proc. 16th ACM International Conference on Supercomputing, 2002.

[5] S. Dolev, E. Schiller and J. Welch, *Random Walk for Self-Stabilizing Group Communication in Ad-Hoc Networks*, Proc. 21st Symposium on Reliable Distributed Systems, 2002.

[6] G. Di Caro and M. Dorigo, *AntNet: Distributed Stigmetric Control for Communication Networks*, Journal of Artificial Intelligence Research 9, 1998.

[7] A. Borodin, Y. Rabani and B. Schieber, *Deterministic Many-to-Many Hot Potato Routing*, Proc. IEEE Transactions on Parallel and Distributed Systems, vol. 8, pp. 587-596, 1997.

[8] L. Lovász, *Random Walks on Graphs: A Survey*, Combinatorics, Paul Erdős is Eighty (Volume 2), János Bolyai Mathematical Society Budapest, pp. 353-398, 1996.

[9] L. A. Adamic, R. M. Lukose, A. R. Puniyani and B. A. Huberman, *Search in power-law networks*, Physical Review E, vol. 64, 2001.

[10] H. P. Thadakamalla, R. Albert and S. R. T. Kumara, *Search in weighted complex networks*, Physical Review E, vol. 72, 2005.

[11] F. R. K. Chung, *Spectral Graph Theory*, CBMS Conference on Recent Advances in Spectral Graph Theory, 1994 ISBN: 0-8218-0315-8.

[12] B. Bollobas, *Random Graphs*, Academic Press, 1985.

[13] S. Valverde and Ricard V. Solé, *Internet's Critical Path Horizon*, Eur. Phys. J. B. 38, pp. 242-252, 2004.

[14] C. Gkantsidis, M. Mihail and A. Saberi, *Random walks in peer-to-peer networks: Algorithms and Evaluation*, Performance Evaluation 63, pp. 241-263, 2006.

[15] P. Van Mieghem, *Performance Analysis of Communication Networks and Systems*, Cambridge University Press, 2006.

[16] D. J. Aldous and J. A. Fill, *Reversible Markov chains and Random walks on graphs*, (monograph in preparation).

[17] C. Gkantsidis, M. Mihail and A. Saberi, *Hybrid search schemes for unstructured peer-to-peer networks*, Proc. IEEE INFOCOM, 2005.

[18] R. Hekmat and P. Van Mieghem, *Degree Distribution and Hopcount in Wireless Ad-hoc Networks*, Proc. 11th IEEE International Conference on Networks, 2003.

[19] Z. Bar-Yossef, R. Friedman and G. Kliot, *RaWMS- Random Walk based Lightweight Membership Service for Wireless Ad Hoc Networks*, Proc. MobiHoc, pp. 238-249, 2006.