SEARCHING WITH MULTIPLE RANDOM WALK QUERIES

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, P.VanMieghem@ewi.tudelft.nl

Abstract

We analyze the performance of searching with multiple random walk queries on Erdös-Rényi (ER) random graphs and power law graphs generated using preferential attachment. Our simulations show that searching with multiple random walk queries reduces message overhead as compared to flooding with sequence numbers. Moreover, the performance of searching by using multiple random walk queries is better in ER random graphs than in power law graphs grown by preferential attachment rule.

I Introduction

Efficient searching for resources and services is an important issue in various networks. There are two common methods employed for searching - flooding and random walks (RW). In wireless ad-hoc networks, reactive protocols such as AODV and DSR use flooding to locate the destination [16]. In web-graphs, search engines use breadth first search to perform a complete search of the web. However, to reduce the overhead of searching, agents or spiders based on RW, where the next hop is chosen uniformly among the neighbors of the node, or variations of the RW such as the RW strategy where the next hop is chosen as the node with maximum degree are widely used [14].

In peer-to-peer networks, both flooding and RW have been employed to locate services and resources [3, 4, 10]. RWs have been shown to induce lower overhead than constrained flooding for searching in peer-to-peer networks [4, 10]. Multiple RWs have been proposed for searching on unstructured peer-to-peer networks by Lv *et al.* [4]. However, the optimization of multiple RWs was not analyzed. Adaptive techniques based on RWs have been proposed for searching by Bisnik and Abhouzeid [15]. In the searching technique proposed in [15], the number of RW queries used for searching are varied depending on the previous performance of searching. Our work differs from previous approaches since we study the optimization between the number of queries and the time-to-live (TTL) of queries for different graph topologies.

We study the performance of different search strategies based on RWs on ER random graphs and power law graphs generated using preferential attachment. Both these graph topologies are important since ad-hoc wireless networks can be modeled as ER random graphs [11] while the web graphs and peer-to-peer networks can be modeled as power law random graphs [10]. We study both a single query and multiple queries to search for the destination. We do not consider dynamic topologies in this paper. In case of multiple RW queries, we study the optimal number of queries and the TTL of queries for ER random graphs and preferential attachment power law graphs. We also show an efficient way of searching graphs using RWs with no repetition of steps (i.e., memory). In addition, an optimal value of the memory M depends on the topology of the network.

Section 2 overviews the previous work. In Section 3, we explain different RW strategies and the parameters used for analysis of RWs. Section 5 presents the results for multiple queries. Finally, the conclusions and the summary of results is given in Section 6.

II Related Work

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, 10]. 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].

Different search algorithms for scale-free and power law graphs have been analyzed in [1, 8]. The term local search algorithm or path finding strategies is also used for different variations of RWs [1, 8]. In [1] and [8], RW strategies where the next hop is chosen as the node with the highest degree and without retracing of steps have been analyzed in terms of expected hopcount.

In mobile agent based routing, the mobile agents perform a RW or a variant of the RW while searching for the destination. In Ant-Net, loop-erased RWs are used by the mobile agents [19]. Mobile agents using RW have been proposed for providing membership services for ad-hoc networks by Dolev *et al.* [5]. As a sampling technique, RWs have been used for providing membership services in ad hoc networks [5, 12] that provide the nodes in the network with a view of the other nodes and that are used by various applications such as location services, peer sampling services and random overlay constructions [12]. Bar-Yossef *et al.* [12] 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 [12].

The analysis of RWs has also been an active topic of research

1-4244-1144-0/07/$25.00©2007 IEEE

[6, 7, 17]. For a detailed mathematical analysis of RWs, we refer the reader to Lovász [7].

III Search Strategies

A Random Walk and Random Walk with memory

In RW, the next hop is chosen uniformly among the neighbors of the node. A major shortcoming of RW is the existence of loops in the path while travelling from the source to the destination node. To prevent loops, the simplest method is to introduce memory in the RWs.

In RW with memory (RW_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 $\mathcal{M}$. In our implementation of the RW strategy with memory M (RW_M), the node identifier of the current node is not stored in the memory list $\mathcal{M}$ and no self-loops are allowed (The next hop cannot be chosen as the node itself.)

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. But the introduction of memory ($M \geq 1$) in RWs can lead to a *deadlock*. Therefore, introducing memory may remove the loops in the RW but can induce deadlocks. In the implementation of Gia, Castro *et al.* [2] have used a query in RWs which consists of all the previously visited hops. This is similar to using complete memory in our analysis. 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.

B Graph Topologies

We use the ER random graph and the Barabási-Albert (BA) power law graphs for our analysis. Each node on a ER random graph is connected to any other node with probability p. The degree distribution for an ER random graph is given by [9]

$$\Pr[d = j] = \binom{N-1}{j} p^j \left(1 - p\right)^{N-1-j} \tag{1}$$

The BA-model for generating scale-free networks is defined in two steps [18]. Starting with small number (v_0) of disconnected nodes, at every timestep, a new node is added with l ($\leq v_0$) links. A new node connects to nodes already in the graph with probability $y = \frac{d_i}{\sum\limits_{v \in Z} d_v}$, where d_i is the degree of node i and Z is the number of nodes in the graph at a particular timestep.

After t timesteps the model leads to a random network with $N = t + v_0$ nodes and lt links. It has been shown in [18] that $\Pr[d \leq j] = 1 - \frac{l^2 t}{j^2 N}$. Thus, the probability that a node has degree j in this model follows a power law [18],

$$\Pr[d = j] = \frac{2l^2 t}{N} \frac{1}{j^3} = Aj^{-\gamma} \tag{2}$$

where the scaling exponent $\gamma = 3$ is independent of l.

The number of nodes with degree less than $\log N$ in BA model is $N \cdot \Pr[d \leq \log N] = N \left(1 - \frac{4}{(\log N)^2}\right)$ and the number of nodes with a large degree is small. On the other hand, in a.s. connected ER random graph where $p \geq \frac{\log N}{N}$, the average node degree is close to or greater than $\log N$.

C Definitions

A RW can be described as a finite Markov chain that is time-reversible [7]. The stochastic matrix $P = \Delta^{-1} A$, where $\Delta = diag\left(d_1, d_2, ..., d_N\right)$ is the degree matrix and A is the adjacency matrix, represents the transition matrix of the RW. It is known [7] that the RW has a unique stationary distribution π, such that $\pi P = \pi$, with $\pi_i = \frac{d_i}{2L}$. Let the RW start at node n_0. The node n_0 could be drawn from some initial distribution Y_0. Denote the sequence of random nodes by n_t $(t = 0, 1...)$. If we denote by Y_t the distribution of n_t i.e., $Y_t\left(i\right) = \Pr\left[n_t = i\right]$, the RW can be expressed as $Y_{t+1} = P^T Y_t$ and hence, $Y_t = \left(P^T\right)^t Y_0$. Thus, the probability that RW starting at i reaches node j in t steps is given by (i, j) entry of the matrix P^t [7].

It is known that a symmetric P matrix has n real eigenvectors with corresponding eigenvalues $1 = \lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n \geq -1$. Moreover, if we exclude bipartite graphs or reducible Markov chains, then $\mid \lambda_i \mid < 1$, for $i > 1$. The cover time C_N is the expected number of steps for RW to visit all nodes at least once [12],

$$C_N = O\left(\frac{\pi_{\min}^{-1} \log N}{1 - \lambda_2}\right) = O\left(\frac{N \log N}{1 - \lambda_2}\right) \tag{3}$$

where $\pi_{\min} = \frac{d_{\min}}{2L}$. Jonasson [6] showed that when $p \geq \frac{\log N}{N}$, w.h.p a random graph has the same cover time as the complete graph K_N i.e., the cover time is $N \log N$. Similarly, for BA power law graphs, Cooper and Frieze [17] proved that if $m \geq 2$, then w.h.p. the cover time $C_N \leq O\left\lceil \frac{2m}{m-1} N \log N \right\rceil + o\left(N \log N\right)$.

The number of hops required for uniform sampling by RW can be as low as number of samples in independent uniform sampling [10, 15]. Thus, if the RW starts at any node and makes k hops, and using each visited node as a sample point, approximately the same statistical properties can be achieved as k independent uniform samples. However, our simulations show that there are large differences in the behavior of RW as compared to independent uniform sampling, particularly, when the TTL is small.

IV Searching with multiple Random Walk queries

We show that using multiple queries reduces the overhead for searching as compared to flooding with sequence numbers.

Searching (packet) overhead is defined as product of number of hops for each query (packet) and number of queries (packets) i.e., the total number of packets exchanged. A single RW query has an overhead and time to discover of $O(N \log N)$ for ER random graphs and BA power law graphs [6, 17]. Since the time to discover is large in RW, we split a single RW query into Q multiple queries each with a given TTL. Each RW query stops when the TTL is reached or the destination is located (In RW_M, the query also stops if there is a deadlock). Since each RW query makes at most TTL hops, the worst-case searching overhead is $Q \times TTL$. The expected number of hops are less since any query stops once the destination is found. However, there is a probability that a destination is never located by the multiple queries. Thus, with multiple queries we define the probability of success as the probability that the destination is located by at least one query out of the Q queries generated. In flooding with sequence numbers, the probability of success is defined as the probability that a destination is located with a given TTL. We want to minimize TTL and Q and maximize $\Pr[\text{success}]$.

To analyze the performance of multiple RW queries, we define the efficiency as the inverse of expected packet overhead needed to discover the destination node. The efficiency is normalized by multiplying by N. The gain of searching in one scheme over another is defined as the ratio of efficiency for the corresponding schemes.

$$\eta = \frac{\text{Number of iterations } (\Pr[\text{success}])}{\text{Total number of packets exchanged}} \times N \tag{4}$$

$$g\left(\frac{\text{RW}_\text{M}}{\text{RW}}\right) = \frac{\eta(\text{RW}_\text{M})}{\eta(\text{RW})} \tag{5}$$

Consider a single RW and RW_M on a complete graph K_N. The searching overhead can be approximated by the product of expected hopcount to discover destination and the number of iterations. Since in RW and RW_M, the expected hopcount is $N-1$ and $(N-1)/2$ respectively, the gain $g\left(\frac{\text{RW}_\text{M}}{\text{RW}}\right) = 2$ for a complete graph K_N. In flooding with $TTL = 1$, η is $N/(N-1)$ and with $TTL = 2$, η is $N/(N-1)^2$. Thus, the efficiency of flooding depends on the value of TTL and the efficiency decreases with TTL.

Table 1 and 2 show the efficiency for a single RW and RW_M query and flooding with sequence numbers for different values of N and p. The results for flooding in Tables 1 and 2 are for optimized values of TTL such that $\Pr[\text{success}]$ is close to 1. The gain obtained by using RW over flooding is significant, particularly, when the link density p is large. In addition, $\text{RW}_{\text{M=N-1}}$ is a more efficient way of searching than RW since the expected number of hops required to find destination or deadlock is less. However, the probability of deadlocks is high when the link density p is small. As the link density p is decreased, efficiency for both RW and RW_M decreases. This is in contrast to flooding where the efficiency increases as the link density is decreased.

A single RW leads to lower overhead to locate a destination than flooding with sequence numbers. However, the time to search for destination is much larger than in flooding. Since we want to maximize $\Pr[\text{success}]$ and minimize TTL, we split the single RW or RW_M into multiple queries with fixed TTL such that $Q \times TTL = N \log N$. Table 3 and 4 show the results for multiple RW and RW_M queries for ER random graph with $N = 400$ and link density $p = 0.015$ and 0.2 respectively. In Table 4, since the link density p is large, $\Pr[\text{success}] \simeq 1$ and is not shown.

Table 1: Efficiency of searching by using flooding, and a single RW or RW_M query for dense ER random graph ($pN = 80$) for different values of N

$pN = 80$					
N	η (RW)	η (RW_M)	η (flood)	$g(\frac{\text{RW}}{\text{flood}})$	$g(\frac{\text{RW}_\text{M}}{\text{RW}})$
100	1.17	1.82	0.015	78	1.55
200	1.08	2.16	0.03	36	2
400	1.05	1.997	0.06	17.5	1.90
800	1.017	2.0	0.12	8.5	1.98

Table 2: Efficiency of searching by using flooding, and a single RW and RW_M query for sparse ER random graph ($pN = 6$) for different values of N

$pN = 10$					
N	η (RW)	RW_M η(Pr[suc.])	η (flood)	$g\left(\frac{\text{RW}}{\text{flood}}\right)$	$g\left(\frac{\text{RW}_\text{M}}{\text{RW}}\right)$
100	0.84	1.6 (0.73)	0.14	6	1.89
200	0.84	1.54 (0.66)	0.21	4	1.83
400	0.82	1.47 (0.59)	0.11	7.5	1.79
800	0.8	1.4 (0.52)	0.14	5.7	1.76

The probability of success is very low in RW_M with only a single query when the link density p is small. However, when split into multiple queries, $\Pr[\text{success}]$ increases. As the TTL is decreased and Q is increased, the efficiency decreases for both RW and RW_M. The decrease in efficiency occurs with small TTL since most of the queries search only the neighboring nodes which have been visited already by other queries. There is also a decrease in efficiency because the number of queries and the TTL is fixed. Thus, multiple queries might locate the destination. The terminating conditions can be included which improve the efficiency but increases the complexity of searching algorithms. For example, a scheme is proposed in [4], where the query checks with the source node whether the destination is located. As shown by tables 3 and 4, the efficiency decreases by a factor of 3 as the TTL is decreased from 2400 to 120. Moreover, when the TTL is small, the gain obtained by using RW_M over RW is small. Therefore, only for large values of link density p and TTL, RW_M is a more efficient way of searching than RW. If we use $Q > 1$ with $TTL = 2400$, the efficiency decreases.

Figure 1 shows $\Pr[\text{success}]$ versus the searching overhead (number of packets exchanged) and the worst time to discover

Table 3: Efficiency of searching by multiple RW and RW_M queries for $p = 0.015$ and $N = 400$.

$pN = 6, N = 400$	RW		RW_M		$g(\frac{\text{RW}_\text{M}}{\text{RW}})$
$Q \times TTL = 2400$	η	Pr[suc.]	η	Pr[suc.]	
$Q = 1$	0.7	0.996	1.23	0.28	1.8
$Q = 20$	0.27	0.96	0.33	0.97	1.2
$Q = 120$	0.24	0.95	0.26	0.97	1.1
$Q = 400$	0.15	0.73	0.18	0.82	1.2

Table 4: Efficiency of searching using multiple RW and RW_M queries for $p = 0.2$ and $N = 400$.

	$pN = 80$		
$N = 400$	η (RW)	η (RW_M)	$g\left(\frac{\text{RW}_\text{M}}{\text{RW}}\right)$
$Q = 1, TTL = 2400$	1.05	1.997	2
$Q = 20, TTL = 120$	0.30	0.298	1
$Q = 120, TTL = 20$	0.285	0.285	1
$Q = 400, TTL = 6$	0.26	0.262	1

the destination. Since the size of network is not known *a priori*, we also show simulations for query split into multiple queries with a different TTL. We use a linearly increasing TTL and the maximum time to discover is given by maximum $TTL_{\max}$.

$$TTL(l) = TTL_{\min} + (l - 1) * add\_TTL \qquad (6)$$

Using (6), the add_TTL parameter can be expressed as $\frac{TTL_{\max} - TTL_{\min}}{Q-1}$. Figure 1 shows that efficiency of searching by RW and RW_M decreases with TTL. This is in contrast to the assumptions made in the analysis in [15], where searching with multiple queries and independent uniform sampling are assumed to be equivalent. Also, the linear query performs as good as sending multiple queries with a large $TTL(100)$. Thus, the simulations show that searching by using a single RW query with TTL is more efficient than sending Q RW queries with $TTL'(TTL = Q \times TTL')$ for ER random graphs.

Figure 2 shows the results for searching with multiple queries in BA power law graph. In these graphs, $\text{RW}_{\text{M=2}}$ gives the best performance in terms of reducing search overhead. $\text{RW}_{\text{M=N-1}}$ performs worse than $\text{RW}_{\text{M=2}}$ since many of the queries end in a deadlock. This reduces the efficiency of the $\text{RW}_{\text{M=N-1}}$ strategy and queries with larger TTL need to be sent to achieve the same probability of success as $\text{RW}_{\text{M=2}}$. Moreover, the improvement in performance of different RW strategies compared to flooding is limited. Even with a large $TTL = 1000$, the RW does not perform better than flooding.

Figure 3 compares the performance of searching with RW and RW_M in ER random graph and BA power law graph using the same $TTL = 400$. The results are for $N = 10000$ and for ER random graph the link density $p = 0.001$. Figure 3 also shows the results for searching for a high-degree destination node in BA power law graph (the average degree of the destination is 72). Searching in ER random graphs performs better than in BA power law graph. This can be attributed to the fact that most of the nodes in ER random graph have a larger degree than the degree of nodes in BA power law graph (section III.B). Thus, in BA power law graph, the RW makes large number of hops among the low degree nodes while searching for the destination node. Moreover, in BA power law graph, since the degree of uniformly chosen destination and source nodes is small, performance of searching for a uniformly chosen node is much worse than searching for a high degree node.

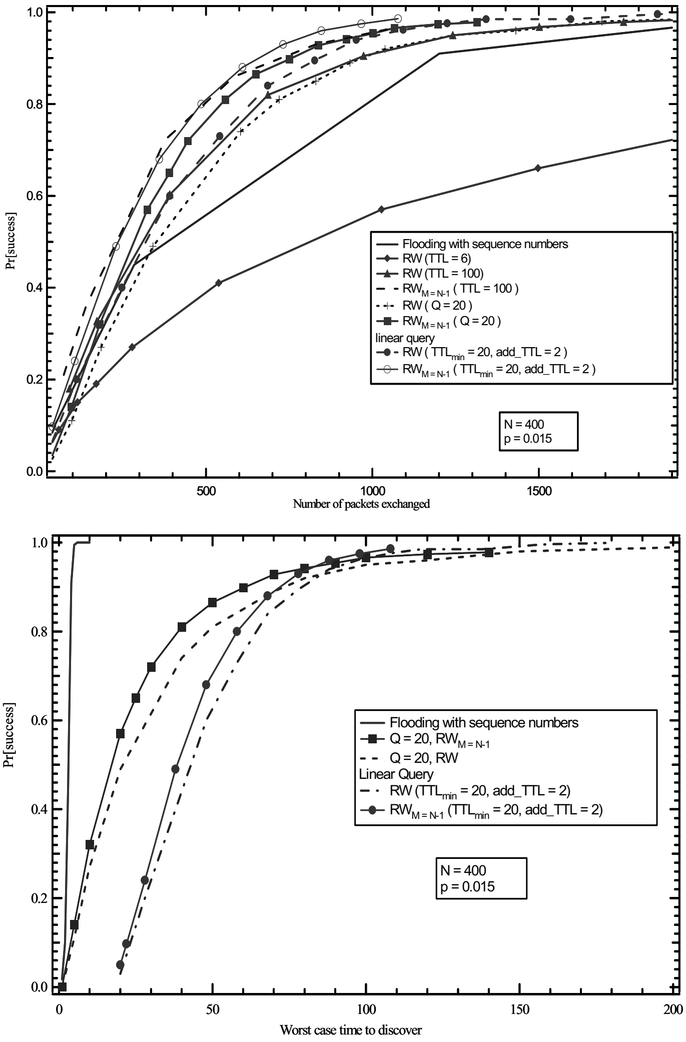

Figure 1: Performance of searching by using multiple queries in ER random graph for $N = 400$ and $p = 0.015$

V Conclusions

We have analyzed the performance of searching with multiple RW strategies in two types of graphs. The topology of graphs plays an important role in determining the performance of different search strategies. Searching with multiple RW queries performs better than flooding with sequence numbers in terms of overhead. The overhead can be reduced further by using

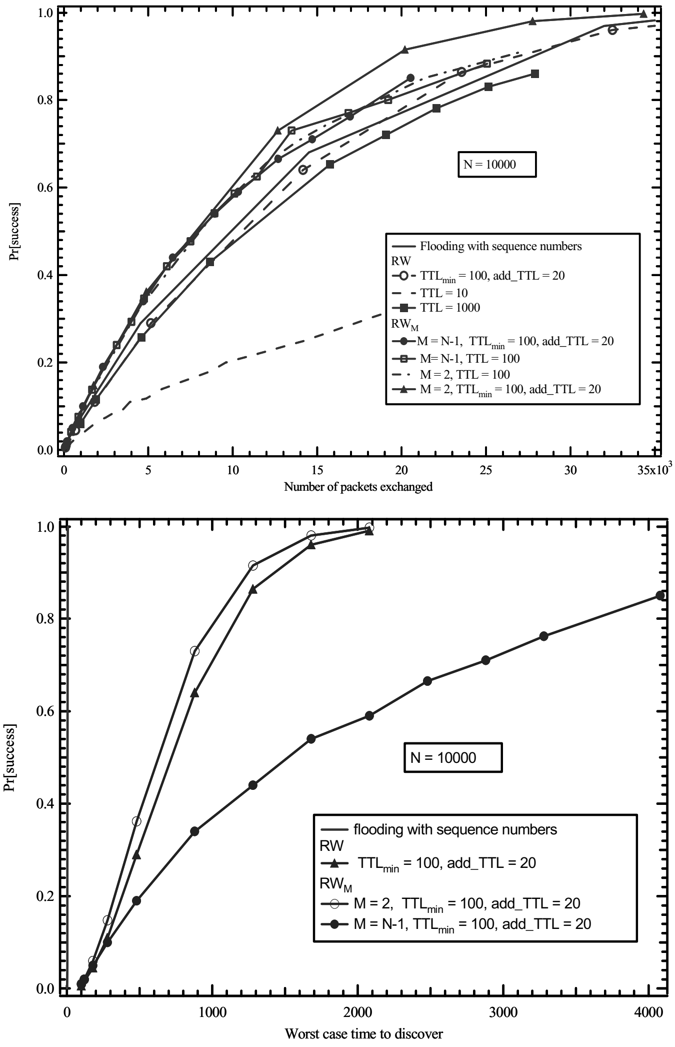

Figure 2: Performance of searching using RW and $\mathrm{RW_M}$ with multiple queries for BA power law graph $(N = 10000)$.

multiple $\mathrm{RW_M}$ queries. However, the searching efficiency decreases with TTL for both RW and $\mathrm{RW_M}$. The performance of searching with multiple RW and $\mathrm{RW_M}$ queries is better in ER random graphs than in BA power law graphs. Moreover, in BA power law graph, only small values of memory M improve performance of searching using multiple RW queries.

Acknowledgement

This research was supported by the Dutch Ministry of Economic Affairs under the Innovation Oriented Research Program (IOP GenCom, QoS for Personal networks @ Home) and partially supported by European Union CONTENT NoE (FP6-IST-038423).

References

[1] B.J. Kim, C.N. Yoon, 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.

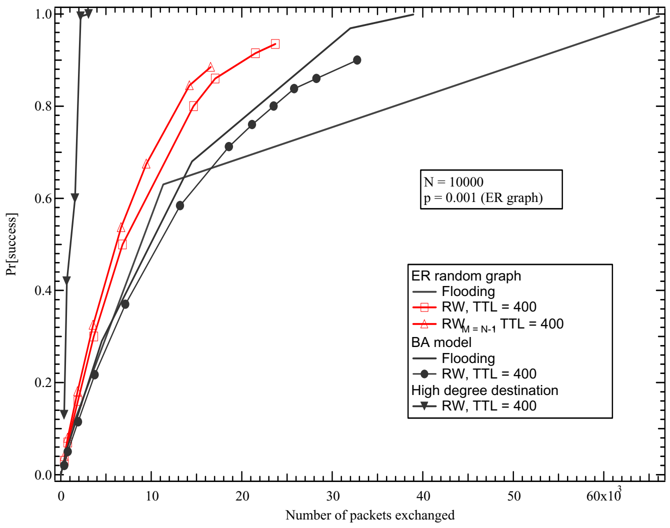

Figure 3: Comparison of searching by multiple RW and $\mathrm{RW_M}$ queries on ER random graph and BA power law graph. $(N = 10000$ and for ER random graph $p = 0.001)$

[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] J. Jonasson, "On the cover time of random walks on random graphs", *Combinatorics, Probability and Computing*, 7, pp. 265-279, 1998.

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

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

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

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

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

[12] 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.

[13] S.S. Dhillon and P. Van Mieghem, "Comparison of Random Walk strategies for Ad-hoc Networks", *Proc. 6th Annual Mediterranean Ad Hoc Networking Workshop*, pp. 204-211, 2007.

[14] C. Cooper and A. Frieze, "Crawling on web graphs", *Proc. 34th annual ACM symposium on Theory of computing*, pp. 419 - 427, 2002.

[15] N. Bisnik and A. A. Abouzeid, "Optimizing Random Walk Search Algorithms in P2P Networks", *Computer Networks*, 2006.

[16] S.R. Das, C.E. Perkins and E. E. Royer, "Performance Comparison of Two On-Demand Routing Protocols for Ad Hoc Networks", *Proc. INFOCOM*, pp. 3-12, 2000.

[17] C. Cooper and A. Frieze, "The cover time of the preferential attachment graph", *Journal of Combinatorial Theory*, Series B 97, 269-290.

[18] A. Barabási, R. Albert and H. Jeong, "Mean-field theory for scale-free random networks", *PhysicaA* 272, pp. 173-187, 1999.

[19] S.S. Dhillon and P. Van Mieghem, "Performance analysis of the AntNet algorithm", *Computer Networks*, vol 51, Issue 8, pp. 2104-2125, 2007.