# Enhanced Random Walk with Choice: An Empirical Study

John Alexandris, Gregory Karagiorgos

***Abstract*—The Random Walk with $d$ Choice *(RWC($d$))* is a recently proposed variation of the simple Random Walk that first selects a subset of $d$ neighbor nodes and then decides to move to the node which minimizes the value of a certain parameter; this parameter captures the number of past visits of the walk to that node. In this paper, we propose the Enhanced Random Walk with $d$ Choice algorithm (ERWC($d, h$)) which first selects a subset of $d$ neighbor nodes and then decides to move to the node which minimizes a value $H$ defined at every node; this $H$ value is depending on parameter $h$ and captures information about past visits of the walk to that node and - with a certain weight - to its neighbors. Simulations of the Enhanced Random Walk with $d$ Choice algorithm on various types of graphs indicate beneficial results with respect to the Cover Time and Load Balancing. The graph types used are the Random Geometric Graph, Torus and Lollipop.**

***Index Terms*—Random Walk, Power of Choice, Cover Time, Load Balancing, Wireless networks.**

## I. Introduction

Recently there is a growing interest in random walk-based algorithms, especially for a variety of networking tasks (such as searching, routing, self stabilization and query processing in wireless networks, peer-to-peer networks and other distributed systems [1], [7]), due to its locality, simplicity, low overhead and inherit robustness to structural changes. Many wireless and mobile networks are subject to dramatic structural changes caused by sleep modes, channel fluctuations, mobility, device failures and other factors. Topology driven algorithms are inappropriate for such networks, as they incur high overhead to maintain up-to-date topology and routing information and also have to provide recovery mechanisms for critical points of failure. By contrast, algorithms that require no knowledge of network topology, such as random walks, are advantageous.

A random walk on a graph is the process of visiting the nodes of a graph in some sequential random order. The simple random walk starts at some fixed node (with uniform probability among all nodes in the network) and at each time instant, it moves to a randomly chosen neighbor of the currently visited node. The simple random walk is a totally uncontrolled process, which often shows unwanted behavior, like frequent revisits of already covered nodes and substantial delays in visiting the most isolated regions within a network.

Motivated by the need to *improve random walk-based performance over graphs*, researchers have proposed, to reduce the Cover Time of a random walk, various methodologies. A recently proposed methodology in [6] suggests that the use of multiple, parallel random walks starting from a fixed vertex of the graph will speed up the cover process. The authors prove that running many random walks in parallel yields a speed-up which is linear in the number of parallel walks. They also demonstrate that an exponential speed-up is sometimes possible, but that some natural graphs allow only a logarithmic speed-up.

A further methodology for reducing Cover Time on graphs is the use of Power of Choice. The essential idea behind the power of choice, is to make some decision process more efficient, by selecting the best among a small number of randomly generated alternatives. The most basic results [16] about the power of choice are as follows: Suppose that $n$ balls are placed into $n$ bins, with each ball being placed into a bin chosen independently and uniformly at random. Let the load of a bin be the number of balls in that bin after all balls have been thrown. What is the maximum load over all bins once the process is terminated? It is well known that, with high probability, the maximum load upon completion will be approximately $\frac{logn}{loglogn}$. We now state a surprising result proved in a seminal paper by Azar, Broder, Karlin, and Upfal [8]. Suppose that the balls are placed sequentially so that for each ball we choose 2 bins independently and uniformly at random and place the ball into the less full bin (breaking ties arbitrarily). In this case, the maximum load drops to $\frac{loglogn}{log2} + O(1)$ with high probability. If each ball has $d \geq 2$ choices instead, then the maximum load will be $\frac{loglogn}{logd} + O(1)$ with high probability. Having two choices hence yields a qualitatively different type of behavior from the single choice case, leading to an exponential improvement in the maximum load; having more than two choices further improves the maximum load by only a constant factor.

Chen Avin and Bhaskar Krishnamachari in their paper *The power of choice in Random Walks: An Empirical Study* [4] proposed to combine random walk on a graph with the Power of Choice. The Random Walk with $d$ Choice algorithm *(RWC($d$))* works in such a way that it selects $d$ neighbors uniformly at random and then chooses to step to the node $u$ of the $d$ neighbors than minimizes the fraction $\frac{c^t(u)+1}{\delta(u)}$, where $c^t(u)$ is the number of visits of the random walk at time $t$ at node $u$ and $\delta(u)$ is the degree of node $u$. If the graph is regular, the walk steps to the least-visited neighbor; if not, the walk steps to the node that is farthest away from its stationary distribution $\pi(u)$. For the complete graph the analytical results show that the cover time of RWC($d$) is reduced by a factor of $d$, compared with the cover time of the simple random walk. For general graphs the lack of Markov property suggests that the analytical results may be harder to obtain. The simulation-based study

Manuscript received March 13, 2012; revised April 11, 2012. The second author G. K. gratefully acknowledges support of the project Autonomic Network Architecture (ANA), under contract number IST-27489, which is funded by the IST FET Program of the European Commision.

John Alexandris: is working as a software engineer and as a teacher in computer science at high school and university level, Mauromichaleon 32 Halandri, Athens, Greece, e-mail: johnalexgr@yahoo.gr

Gregory Karagiorgos: Department of Technology of Informatics and Telecommunications, A.T.E.I. of Kalamata / Branch of Sparta, Kladas, 23100 Sparta, Greece, e-mail: greg@teikal.gr

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

shows a consistent improvement in the cover time, cover time distribution and the load balancing at cover time for different graphs and different sizes. A surprising result is that for the 2-dimensional torus, choice seems to improve the cover time and the load on the most visited node by an unbounded factor. Specifically while the cover time of the n nodes torus is known to be $\Theta(n \log^2 n)$, the simulations showed that with $d = 2$, random walk with choice has lower cover time than the simple random walk on the hyper-cube, that is known to have optimal cover time $\Theta(n \log n)$.

In this paper we propose the *Enhanced Random Walk with $d$ Choice (ERWC($d, h$))* algorithm. We introduce, additionally to $d$, a new parameter $h > 1$. Let $H^t(v)$ be a value defined to node $v$ at time $t$ as follows:
$H^t(v) = (c^t(v) * h) + \sum_{u \in N(v)} (c^t(u) * 1)$ where $c^t(u)$ is the number of visits of the random walk at time $t$ at node $u$.

The main idea behind ERWC($d, h$) is to stir the course of the random walk based not only on the number of visits to the candidate node it is considering to move to (as it is the case with the RWC($d$)), but based on a more comprehensive metric that captures the level of past passage of the random walker through the broader region around the potential candidate node it is considering to move to. This way, it is expected that the random walk will not be stirred toward highly visited regions and, thus, it would be likely to enter less visited regions incurring less revisits to the nodes. In order to represent the intensity of visits to a region, we introduce the idea of recording the "trail" of a random walk through a region by increasing a counter of a visited node by $h > 1$ and that of its neighbors by 1. The more frequent and the closer the passage of a random walk from a certain node, the higher the accumulated value of its counter, $H$, would be expected to be. By selecting randomly $d$ nodes, the random walk selects d potential directions for moving toward in its next step. Among those $d$ possibilities, the random walk will consider first the subset of nodes never visited before and among them will select the one with the lowest value of $H$ divided by the degree of the node; if the latter subset is empty, it will select the neighbor (direction) with the lowest value of $H$ divided by the degree of the node. Through such moving rules, the random walk is stirred towards the unvisited nodes from the selected $d$, which is located in the least visited region (lowest value of $H$ divided by the degree of the node); or else, it is stirred towards the revisited node from the select $d$, which is located in the least visited region.

At any point in time $t$ the Enhanced Random Walk algorithm selects $d$ neighbors uniformly at random and then chooses to step to the unvisited node among them(if exists). If all the selected $d$ nodes have already been visited the random walk chooses to step to the node $u$ of the $d$ neighbors than minimizes the fraction $\frac{H^t(u)}{\delta(u)}$, where $\delta(u)$ is the degree of node $u$.

Our result, by using simulations, is that the ERWC($d, h$) algorithm outperforms the RWC($d$) algorithm in the following way:

1. For the Random Geometric and Torus graphs the performance improvement is in both Cover Time and Load Balancing at Cover Time.
2. For the Lollipop graphs the performance improvement is mainly at Cover Time. There is a small improvement in Load Balancing at Cover Time.

The rest of the paper is organized as follows: Section II gives background information and definitions on graphs, random walk algorithms and the metrics of interest, associated with random walks on graphs. Section III formally introduces the Enhanced Random Walk with $d$ Choice (ERWC($d, h$)) algorithm. In Section IV simulation results are presented for many types of graphs, including Random Geometric Graph, Torus, and Lollipop. Conclusions and Future Work are presented in Section V.

## II. Background, Definitions and Metrics

Let $G = (V, E)$ be an undirected graph with $V$ the set of nodes and $E$ the set of edges. Let $|V| = n$ and $|E| = m$. For $v \in V$ let $N(v) = \{u \in V | (vu) \in E)\}$ be the set of neighbors of $v$ and $\delta(v) = |N(v)|$ the degree of $v$. A $\delta$-regular graph is a graph in which the degree of all nodes is $\delta$.

### A. Random Walk on Graphs

*1) Simple Random Walk:* The Simple Random Walk *(SRW)* is a walk where the next node to visit is chosen uniformly at random from the set of neighbors of a currently visited node. That is, when the walk is at node $v$ the probability to move in the next step to node $u$ is $P(v, u) = \frac{1}{\delta(v)}$ for $(v, u) \in E$ and 0 otherwise. If $\{v_t : t = 0, 1, 2, \ldots\}$ denotes the node visited by the SRW at step $t$ then the walk can be described with a Markov chain. The Simple Random Walk is attractive due to its simplicity and robustness, yet it lacks in performance inducing high Cover Time and bad Load Balancing.

*2) Random Walk with d Choice:* The Random Walk with $d$ Choice, RWC($d$) has been introduced in [4]. It is a walk whose next node to move is determined as follows: Let $v$ denote the node reached by the walk at time $t$; let $c^t(v)$ be the number of visits to node $v$ until time $t$; let $N(v)$ be the union of neighbor nodes connected to node $v$.
*RWC(d):* Upon visiting node $v$ at time $t$, the RWC($d$):

1. Selects $d$ nodes from $N(v)$ independently and uniformly at random.
2. Steps to node $u$ that minimizes $\frac{c^t(u)+1}{\delta(u)}$ (break ties in an arbitrary way)

The Random Walk with Choice has been shown to improve the Cover Time without losing the locality, simplicity and robustness of the random walk. This is an important goal, since it is directly related to the performance and energy usage of any random walk-based mechanism in a wireless network.

### B. Types of graphs used in simulation

We obtain simulations results for the types of graphs listed below:

1) *Random Geometric Graph, $G(n, r)$.* Random Geometric Graphs $G(n, r)$ result from placing $n$ points uniformly at random on the unit square and connecting two nodes if and only if their Euclidean distance is at most $r$. Recently, it has been proven that, when $r = \Theta(r_{con})$ then w.h.p. $G(n, r)$ has optimal Cover Time

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

$O(nlogn)$ and optimal Partial Cover Time $O(n)$ [5] for the simple random walk. $r_{con}$ grows as $O(\sqrt{\frac{logn}{\pi n}})$ and is the critical radius to guarantee connectivity w.h.p. [13]. The random geometric graphs have been widely used to model link connectivity and protocol behavior in randomly deployed wireless networks.

2) *Torus,* $T(n_1, n_2)$. A 2-dimensional torus $T(n_1, n_2)$ is a graph in which the vertices are arranged on a rectangular 2 dimensional array with dimensions $n_1$ and $n_2$ with the additional property that the vertices on opposite sides of the boundaries of the array are connected. The number of nodes $n$ is given by the equation $n = n_1 * n_2$. It is known that $T(n_1, n_2)$ has non-optimal Cover Time $\Theta(n \log^2 n)$ [10] for the simple random walk.
3) *Lollipop,* $L_{n_1,n_2}$. The Lollipop graph can be created by joining a complete graph $K_{n_1}$ to a boundary vertex of a path graph $P_{n_2}$. The number of nodes $n$ is given by the equation $n = n_1 + n_2$. It is known to have the worst case Cover Time of $O(n^3)$ [5] for the simple random walk.

### C. Metrics

In this subsection, we present a set of performance metrics associated with random walks on graphs, used later to validate the performance of the proposed algorithm. The *metrics of interest* are:

1) *Cover Time (CT).* The Cover Time $C_g$ of a graph $G$ is the maximum (over all starting nodes $v$) expected time taken by a random walk on $G$ to visit all nodes in $G$. Formally, for $v \in V$ let $C_v$ be the expected number of steps for the simple random walk starting at $v$ to visit all nodes in $G$, and the Cover Time of $G$ is $C_g = \max_v C_v$. The Cover Time of graphs have been widely investigated [15], [2], [10], [9], [3], [17], [5], [14]. It was shown by Feige in [11], [12] that for simple random walks $(1 + o(1))n \log n < C_g < (1 + o(1))\frac{4}{27}n^3$. Results for the Cover Time of specific graphs vary from optimal cover time $\Theta(nlogn)$ associated with the complete graph, to the worse case $\Theta(n^3)$ associated with the lollipop graph. The best known cases correspond to dense, highly connected graphs; on the other hand, when connectivity decreases and bottlenecks exist in the graph, the Cover Time increases.
2) *To measure Load Balancing at Cover Time we use Maximum Node Load (MNL).* At Cover Time every node $v \in V$ of the undirected graph $G = (V, E)$ is visited by the random walk $NL_v$ times. It is obvious that $NL_v \geq 1 \quad \forall v \in V$.
The Maximum Node Load is the infinite norm of the number of visits to every node at Cover Time. Formally: $MNL = ||NL||_\infty = \max_{v \in V} |NL_v|$
It is a metric representing how well the visits of the random walk are distributed over the nodes of the graph. The reason we use MNL value as a metric of Load Balancing is that the decrease of MNL value leads to better Load Balancing. This is an important metric primarily for nodes cooperating in wireless networks due to energy consumption limitations of these nodes. Each visit to a node is associated with a fixed energy amount required to handle transmission and reception of the random walk agent. At the same time, it is evident that the MNL metric is representative of the energy depletion in the network. It is expected that the more uniformly distributed the visits of the random walk over the nodes of the network, the more uniform is energy expenditure on behalf of each node.

## III. Enhanced Random Walk with Choice Algorithm

The Enhanced Random Walk with $d$ Choice, *ERWC($d, h$)*, is a walk whose next node to move to is determined as follows: Let $h > 1$ be an integer. Let $v$ be the node visited by the walk at time $t$; let $c^t(v)$ be the number of visits to node $v$ until time $t$; let $N(v)$ be the set of neighbor nodes to node $v$. Now we define $H^t(v)$ as follows:
$H^t(v) = (c^t(v) * h) + \sum_{u \in N(v)} (c^t(u) * 1)$

*ERWC($d, h$) Algorithm:* Upon visiting node $v$ at time $t$, the ERWC($d, h$):

1. Selects $d$ nodes from $N(v)$ independently and uniformly at random. Let $M(v, d)$ denote the selected set of neighbors of $v$, $M(v, d) \subseteq N(v)$.
2. Modify $M(v, d)$ as follows
$$M(v,d) = \begin{cases} u, & \forall u \in M(v,d) \text{ and } c^t(u) = 0 \\ M(v,d) & \text{remains the same otherwise.} \end{cases}$$
3. Steps to node $u \in M(v, d)$ that minimizes $\frac{H^t(u)}{\delta(u)}$ (break ties in an arbitrary way).
4. $c^t(u) = c^t(u) + 1$, $H^t(u) = H^t(u) + h$, and $H^t(k) = H^t(k) + 1$, $\forall k \in N(v) - \{u\}$.

Our main goal in this paper is to reduce the Cover Time, so it is an important discussion to find out a good $h$ value which minimize the ERWC($d, h$) Cover Time for all graph types instances with the same number of nodes.

*a) **Discussion to find out a good** $h$ **value**:* The ERWC($d, h$) algorithm uses the $H$ value at graph nodes to make decisions about subsequent random walk movements. In order to calculate $H^t(u)$ we have to calculate first a good value for $h$, which minimize the ERWC($d, h$) Cover Time for all graph types instances with the same number of nodes.

Solving such a problem requires us to find an analytical function for describing ERWC($d, h$) Cover Time with respect to parameter $h$, choice $d$ and graph type instance. However, such an analytical function is difficult to find. Instead we will propose an experimental way to find a good $h$ value for the ERWC($d, h$) which minimizes the Cover Time.

First we shall describe how we obtain Cover Time and MNL for the RWC($d$) and ERWC($d, h$) algorithms running on all graph types instances studied in this paper.

Given a graph instance $G = (V, E)$ and a random walk algorithm. Let $NS_v$ be the number steps needed by the random walk algorithm to cover $G$ starting at node $v$. Let $c(u)$ be the number of visits to node $u$, for each $u \in V$, when the random walk algorithm covers $G$. The Cover Time is $CT = \max_{v \in V}(NS_v)$ and the Maximum Node Load is $MNL = \max_{u \in V}(c(u))$ at Cover Time. We note that all graph instances with the same number of nodes are identical for all graphs types studied in this paper except of RGG.

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Now we determine a simple method for finding a good $h$ value for the ERWC$(d,h)$ algorithm which minimize Cover Time. Let $d$ be the choice, $n = |V|$, and $N$ the number of graph instances. Given a graph type and fixed $d$, $n$, $N$. For each $h = 2,3,...,100$ using $N$ graph instances $G_i$ , $i = 1,2,...,N$ we calculate the mean Cover Time $mCT_h = \sum_{i=1}^{N}(CT_i)/N$. From these mean Cover Times we find a good $h$ value as the value resulting in lowest mean Cover Time i.e. $mCT_{h_{good}} = \min_{h=2,..,100}(mCT_h)$

Figure 1 presents an example of finding a good value for $h$ on $N = 200$ instances of RGG with $n = 900$ and $r = 2r_{con}$. The mean Cover Time of the ERWC$(d = 2,h)$ algorithm on the graph is shown for all values of $h \in [2,100]$. A good value for $h$ is 4 and is clearly in the region of low $h$ values.

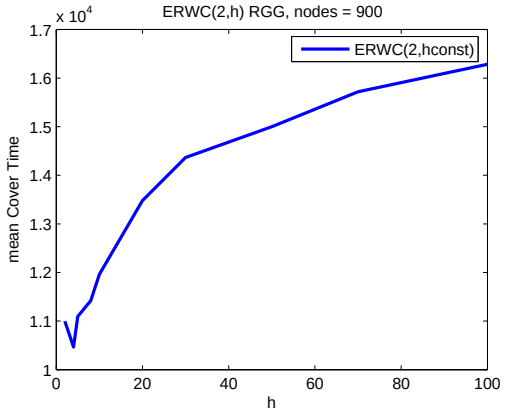


Fig. 1. Mean Cover Time for different $h$ on RGG $G(900, 2r_{con})$

| Graph type | Number of nodes $n$ | Good $h$ value |
|---|---|---|
| RGG,$2r_{con}$ | 900 | 4 |
| Torus | 900 | 3 |
| Lollipop | 100 | 2 |

TABLE I
GOOD $h$ FOR VARIOUS TYPES OF GRAPHS

In Table I can be seen the good $h$ values calculated for all graph types and sizes used in this paper.

## IV. EXPERIMENTAL RESULTS

In this section, we present comparative results for the performance of ERWC$(d,h)$ and RWC$(d)$ algorithms on RGG, Torus and Lollipop graphs with respect to Cover Time and Load Balancing at Cover Time. We obtain experimental results in the following way:

Let $d$ be the choice, $n = |V|$, $N$ the number of graph instances and $h$ be a previously calculated good value if the algorithm is the ERWC$(d,h)$. Given a graph type, an ERWC$(d,h)$ or RWC$(d)$ algorithm and fixed $d$, $n$, $N$, $h$ then for each graph instance $G_i$, $i = 1,...,N$ we obtain the $CT_i$, $MNL_i$. This way we get $CT_i$, $MNL_i$, where $i = 1,...,N$, for ERWC$(d,h)$ and RWC$(d)$ algorithms on the *same* $N$ graph instances of a graph type. The next step is to compare these results to show performance improvement of the ERWC$(d,h)$ algorithm over the RWC$(d)$ algorithm.

To compare the Cover Time results $CT_i$ we use the following statistics: $CT_{min} = \min_{i=1,..,N}(CT_i)$, $CT_{max} = \max_{i=1,..,N}(CT_i)$, $CT_{mean} = \sum_{i=1}^{N}(CT_i)/N$,

Furthermore, to compare the Maximum Node Load results we use the same statistics.

We present simulation results for ERWC$(d = 2,h)$ and RWC$(d = 2)$, using the appropriate good $h$ value for the enhanced random walk algorithm, on:

1) *Random geometric graphs*, $G(n,r)$ for $n = 900$ nodes and radius $r = 2r_{con}$.
2) *Torus graphs* $T(n_1,n_2)$ for $n = 900$ nodes, where $n_1 = n_2 = 30$.
3) *Lollipop graphs* $L_{n_1,n_2}$ for $n = 100$ nodes, where $n_1 = n_2 = 50$.

### A. Experimental results for Random Geometric Graphs (RGG)

This subsection presents simulation results for ERWC$(d = 2,h = 4)$ and RWC$(d = 2)$ algorithms on $G(n,2r_{con})$ with $n = 900$ and $N = 500$. We obtain our results $CT_i$, $MNL_i$, where $i = 1,2,...,N$, for both algorithms using the same $N$ instances of Random Geometric Graphs.

Figure 2 is a diagram of sorted the $CT_i$ results for the Cover Time metric. It can be seen that ERWC$(2,4)$ algorithm outperforms RWC$(2)$ throughout the whole range of graph instances used in our simulations. One can thus verify that a random walk using the ERWC$(2,4)$ algorithm requires lower average number of steps to fully cover the RGG. This is an important result because it can be directly associated with reduced energy expenditures required by a wireless network to support the random walk.

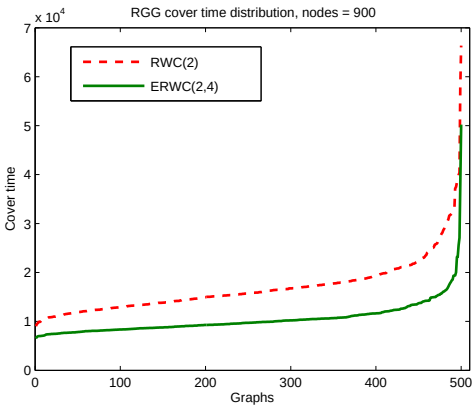


Fig. 2. Results for the Cover Time metric for ERWC$(2,4)$ and RWC$(2)$ on RGG

Figure 3 shows sorted the $MNL_i$ results for the Maximum Node Load at Cover Time for both algorithms. We can see that ERWC$(2,4)$ algorithm is significantly better than RWC$(2)$ algorithm with respect to Maximum Node Load. The numerical results, along with percentage reductions for both algorithms are presented in Table II. These results indicate a 38% reduction in required steps for Cover Time and 23.2% reduction of the $MNL_{mean}$ value in favor of ERWC(2,4) against RWC$(2)$ algorithm. Table II shows also percentage improvements associated with other measures,

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

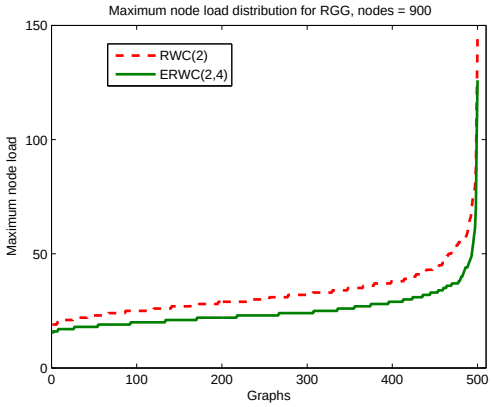


Fig. 3. Results for the Maximum Node Load metric for ERWC$(2,4)$ and RWC$(2)$ on RGG

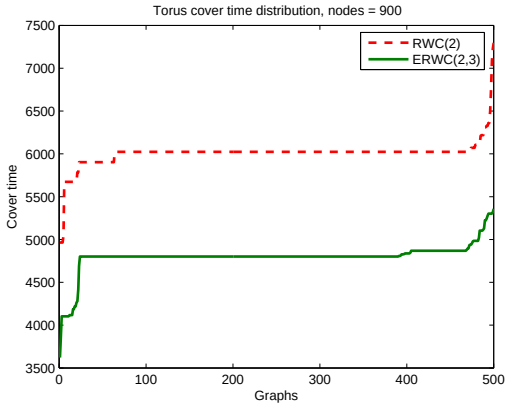


Fig. 4. Cover Time results for ERWC$(2,3)$ and RWC$(2)$ on Torus

| Statistics | $CT_{min}$ | $CT_{max}$ | $CT_{mean}$ |
|---|---|---|---|
| RWC(2) | 9036 | 66352 | 16896 |
| ERWC(2,4) | 6505 | 50195 | 10437 |
| Reduction(%) | 28% | 24% | 38% |
| Statistics | $MNL_{min}$ | $MNL_{max}$ | $MNL_{mean}$ |
| RWC(2) | 19 | 144 | 32.766 |
| ERWC(2,4) | 15 | 126 | 25.138 |
| Reduction(%) | 21% | 12.5% | 23.2% |

TABLE II
STATISTICS FOR CT AND MNL ON RGG GRAPHS

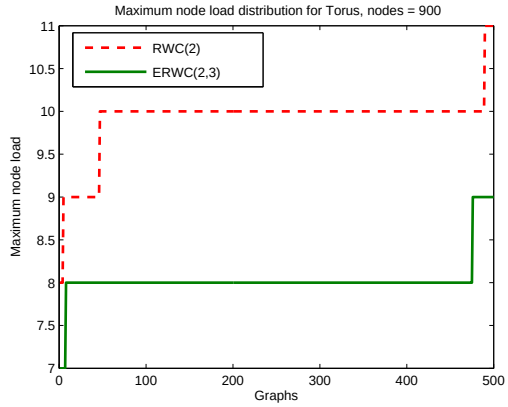


Fig. 5. Maximum Node Load results for ERWC$(2,3)$ and RWC$(2)$ on Torus

such as the 28% reduction for $CT_{min}$, 24% reduction for $CT_{max}$ and 21% for $MNL_{min}$, 12.5% for $MNL_{max}$. These results confirm the improved performance of random walk with ERWC$(d,h)$ algorithm over the random walk with RWC$(d)$ on RGGs.

### B. Experimental results for Torus graphs

Figure 4 and Figure 5 present sorted the $CT_i$, $MNL_i$ results, where $N = 500$ and $i = 1, 2, ..., N$. These results were obtained for the algorithms ERWC$(d = 2, h = 3)$ and RWC$(d = 2)$, using the same $N$ instances of Torus graphs with $n = 900$. It can be seen in Figure 4 that the ERWC$(2,3)$ algorithm outperforms the RWC$(2)$ algorithm for all Torus graphs instances with respect to Cover Time. The flat line part of the results for Torus graphs are identical values for CT resulting due to the symmetry of the Torus graph. There are also reductions for the Maximum Node Load metric. Numerical values of the results and the corresponding percentage reductions are shown in Table III.

| Statistics | $CT_{min}$ | $CT_{max}$ | $CT_{mean}$ |
|---|---|---|---|
| RWC(2) | 4965 | 7297 | 6008.9 |
| ERWC(2,3) | 3621 | 5358 | 4799.4 |
| Reduction(%) | 27% | 26.5% | 20% |
| Statistics | $MNL_{min}$ | $MNL_{max}$ | $MNL_{mean}$ |
| RWC(2) | 8 | 11 | 9.92 |
| ERWC(2,3) | 7 | 9 | 8.03 |
| Reduction(%) | 12.5% | 18% | 19.6% |

TABLE III
STATISTICS FOR CT AND MNL IN TORUS GRAPHS

The ERWC$(2,3)$ algorithm achieves a significant 20% average reduction in required steps to Cover Time compared with RWC$(2)$ for the Torus graph. Furthermore, the average reduction with respect to MNL metric measures is 19.6% indicating improved Load Balancing characteristics. One can thus verify performance improvements for the ERWC$(d,h)$ based random walk proposed in this paper.

### C. Experimental results for Lollipop graphs

The last graph type used for comparing ERWC$(d = 2, h = 2)$ and RWC$(d = 2)$ algorithms is the Lollipop graph with $n = 100$. Figure 6 shows sorted the $CT_i$ results, where $N = 2000$ and $i = 1, 2, ..., N$. These results where obtained for both algorithms using the same $N$ graph instances of Lollipop graphs.

| Statistics | $CT_{min}$ | $CT_{max}$ | $CT_{mean}$ |
|---|---|---|---|
| RWC(2) | 3603 | 14605 | 6510.9 |
| ERWC(2,2) | 1813 | 7868 | 3544.7 |
| Reduction(%) | 49.6% | 46.1% | 45.5% |

TABLE IV
STATISTICS FOR CT IN LOLLIPOP GRAPHS

The reduction in $CT_{mean}$ in favor of ERWC$(2,2)$ measures 45.5%. Furthermore, Table IV shows reductions of 49.6% at $CT_{min}$ and 46.1% at $CT_{max}$. These results show a significant performance improvement in Cover Time of

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

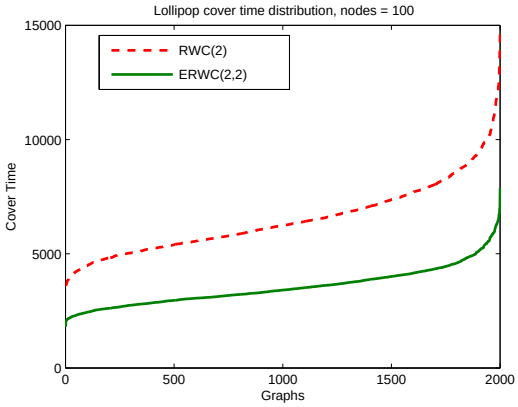


Fig. 6. Cover Time results for ERWC(2,2) and RWC(2) on Lollipop Graphs

ERWC$(2, 2)$ algorithm over the RWC$(2)$ algorithm on Lollipop graphs.

The performance improvement in Load Balancing on Lollipop graphs is small. For example there is a $2.99\%$ improvement with respect to $MNL_{mean}$, so we don't present any other results about the Maximum Node Load metric.

## V. Conclusions

In this work, the Enhanced Random Walk with Choice algorithm is introduced as an improved version of the Random Walk with Choice algorithm. ERWC$(d, h)$ algorithm is formally defined and compared against RWC$(d)$ via simulations over three different graph types, including Random Geometric, Torus, and Lollipop graphs. Comparative results are presented for both ERWC$(d, h)$ and RWC$(d)$ algorithms, which are based on the Cover Time metric and Maximum Node Load metric.

Our simulation results indicate significant savings in Cover Time up to $38\%$ for Random Geometric Graphs, $20\%$ for Torus graphs and $45.5\%$ for Lollipop graphs. Furthermore, the Load Balancing properties of the ERWC$(d, h)$ algorithm are better than those of RWC$(d)$ with reductions for Maximum Node Load in the order of $23.2\%$ for Random Geometric Graphs and $19.6\%$ for Torus graphs.

Our plans for future work include running simulations on other types of graphs to verify performance improvement of the ERWC$(d, h)$ algorithm over RWC$(d)$. Furthermore, the analytical description of Cover Time for both algorithms is an open issue for future research.

## References

[1] M. Alanyali, V. Saligrama, and O. Sava, *A random-walk model for distributed computation in energy-limited network* In Proc. of 1st Workshop on Information Theory and its Application, San Diego, 2006.

[2] D.J. Aldous, *Lower bounds for covering times for reversible Markov chains and random walks on graphs* J.Theoret. Probab., 2(1): 91−100, 1989

[3] R. Aleliunas, R. M. Karp, R. J. Lipton, L. Lovász, and C. Rackoff, *Random walks, universal traversal sequences, and the complexity of maze problems* In 20th Annual Symposium on Foundations of Computer Science (San Juan, Puerto Rico, 1979), pages 218 − 223. IEEE, New York, 1979.

[4] Chen Avin and Bhaskar Krishnamachari, *The Power of Choice in Random Walks: An Empirical Study*, The 9th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, (MSWiM), Malaga, Spain, October 2006.

[5] Chen Avin and Gunes Ercal, *On The Cover Time of Random Geometric Graphs*. In Proceedings. Automata, Languages and Programming, 32nd International Colloquium, ICALP-05, pages 677 − 689, 2005.

[6] Noga Alon, Chen Avin, Mchail Koucky, Gady Kozma, Zvi Lotker and Mark R. Tuttle (2011). *Many Random Walks Are Faster Than One*. Combinatorics, Probability and Computing, 20, pp 481-502 doi:10.1017/S0963548311000125

[7] Chen Avin and C. Brito, *Efficient and robust queryprocessing in dynamic environments using random walk techniques*. In Proc. of the third international symposium on Information processing in sensor networks, pages 277 − 286, 2004.

[8] Y. Azar, A. Broder, A. Karlin, and E. Upfal, *Balanced allocations*, In Proceedings of the 26th ACM Symposium on the Theory of Computing, pages 593 − 602, 1994.

[9] A. Broder and A. Karlin, *Bounds on the cover time* J. Theoret. Probab., 2: 101 − 120, 1989.

[10] A. K. Chandra, P. Raghavan, W. L. Ruzzo, and R. Smolensky, *The electrical resistance of a graph captures its commute and cover times* In Proc. of the twenty-first annual ACM symposium on Theory of computing, pages 574 − 586. ACM Press, 1989.

[11] U. Feige, *A Tight Upper Bound on the Cover Time for Random Walks on Graphs*, Random Struct. Alg. 6 (1995), 51 − 54.

[12] U. Feige, *A Tight Lower Bound on the Cover Time for Random Walks on Graphs*, Random Struct. Alg. 6 (1995), 433 − 438.

[13] P. Gupta and P.R. Kumar, *Critical power for asymptotic connectivity in wireless networks*. In Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H.Fleming, 1998, 547 − 566.

[14] Johan Jonasson and Oded Schramm, *On the Cover Time of Planar Graphs in Electronic Communications* in Probability 5 (2000) 85−90.

[15] P. Matthews, *Covering problems for Brownian motion on spheres*, Ann. Probab., 16(1): 189 − 199, 1988

[16] Michael Mitzenmacher, *The Power of Two Choices in Randomized Load Balancing*, PhD Thesis, 1996.

[17] D. Zuckerman, *A technique for lower bounding the cover time*, In Proc. of the twenty-second annual ACM symposium on Theory of computing, pages 254 − 259. ACM Press, 1990.

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)