Exploring Low-degree Nodes First Accelerates Network Exploration

STEFANIA COSTANTINI, University of L'Aquila, Italy
PASQUALE DE MEO*, University of Messina, Italy
ANGELO GIORGIANNI and VALENTINA MIGLIORATO, University of Messina, Italy
ALESSANDRO PROVETTI, Birbeck, University of London, UK
FEDERICO SALVIA, University of Messina, Italy

We consider information diffusion on Web-like networks and how random walks can simulate it. A well-studied problem in this domain is Partial Cover Time, i.e., the calculation of the expected number of steps a random walker needs to visit a given fraction of the nodes of the network. We notice that some of the fastest solutions in fact require that nodes have perfect knowledge of the degree distribution of their neighbors, which in many practical cases is not obtainable, e.g., for privacy reasons. We thus introduce a version of the Cover problem that considers such limitations: Partial Cover Time with Budget. The budget is a limit on the number of neighbors that can be inspected for their degree; we have adapted optimal random walks strategies from the literature to operate under such budget. Our solution is called Min-degree (MD) and, essentially, it biases random walkers towards visiting peripheral areas of the network first. Extensive benchmarking on six real datasets proves that the—perhaps counter-intuitive strategy—MD strategy is in fact highly competitive wrt. state-of-the-art algorithms for cover.

1 INTRODUCTION

A number of data sources on the Web can be described as *network data,* i.e., collections of interrelated, often heterogeneous, objects (people, documents, multimedia objects and so on) tied by some kind of relationships. Important examples of network data are the friendship network in Facebook [Viswanath et al. 2009], mutual following relationships in a software development platform as GitHub [Rozemberczki et al. 2019] or co-purchase relationships between members of an e-commerce Web site like Amazon [Yang and Leskovec 2015]. In what follows we will speak interchangeably of networks and graphs [Newman 2018] as an ordered pair $G = \langle N, E \rangle$ consisting of a collection of nodes (associated to artificial or real entities) and edges (which capture relationships between nodes).

Random Walks [Lovász 1993] are an important class of algorithms to analyze the structure of large networks; in short, a random walk on a graph can be described as a random process which starts from one of the graph nodes and, in a sequential fashion, selects the next node to move according to some specified probability [Lovász 1993]. Random walks (RWs) have been applied in a broad range of graph analytic tasks such as the ranking of individuals in a social network [Newman 2005] and the segmentation of large virtual communities [De Meo et al. 2014; Pons and Latapy 2006]. One of the most important application of RWs is *network sampling* [Hu and Lau 2013]: a family of techniques that takes a graph $\mathcal{G}$ and seek to generate a representative subgraph $\mathcal{G}'$ which preserves some of the structural properties of G. Graph sampling has a wide spectrum of applications on the Web such as the identification of a sample of people to poll from an hidden population in sociological studies [Hu and Lau 2013], or the crawling of large Online Social Networks [Ahn et al. 2007; Catanese et al. 2011; Gjoka et al. 2011].

*Corresponding author

Authors' addresses: Stefania Costantini, stefania.costantini@univaq.it, University of L'Aquila, Department of Information Engineering, Computer Science and Mathematics, L'Aquila, 67100, Italy; Pasquale De Meo, pdemeo@unime.it, University of Messina, Department of Ancient and Modern Civilizations, Messina, 98166, Italy; Angelo Giorgianni; Valentina Migliorato, University of Messina, Department of Ancient and Modern Civilizations, Messina, 98166, Italy; Alessandro Provetti, ale@dcs.bbk.ac.uk, Birbeck, University of London, Department of Computer Science and Information Systems, London, WC1E 7HX, UK; Federico Salvia, University of Messina, Department of Ancient and Modern Civilizations, Messina, I-98166, Italy.

Many studies focused on estimating the *efficiency* of random walks and several parameters have been introduced so far [Aleliunas et al. 1979; Avin and Krishnamachari 2008; Ikeda et al. 2009; Kahn et al. 1989; Redner 2001]. A key parameter to assess the efficiency of a random walk is *partial cover time* [Avin and Brito 2004; Avin and Ercal 2005; Chupeau et al. 2015; Weng et al. 2017], which quantifies the time a RW takes to visit a given fraction of the nodes of G. Currently, the main focus in literature has been the "extremal" version of the problem, *cover time* [Aldous 1989] defined as the expected number of steps a RW needs to visit all nodes in G. Fewer studies have addressed cover time, mostly focusing on the boundary cover time for specific classes (e.g. regular graphs) [Kahn et al. 1989] or on heuristics [Abdullah et al. 2015; Ikeda et al. 2009].

We submit that in Web-based applications the (total) cover time may not be an interesting indicator vis-à-vis an optimized (or at least reduced) partial cover time. For instance, consider rumor spreading in Online Social Network: we are not worried if the rumor reached the entire population but we strive to spread the rumor to a sufficiently large sample of the whole population.

Of course, existing solutions for cover time might be extended to the partial cover time but they make assumptions which we believe are unrealistic in Web applications. For instance, the approach of [Ikeda et al. 2009] requires that a node knows the degrees of all of its neighbors to compute the probability that a random walk move from a node to one of its neighbours. In Web-based applications such as Online Social Networks, an individual may refuse to disclose the number of and identities of her/his friends for privacy reasons.

In this paper we introduce a new problem, called *Partial Cover Time with Budget* in which we wish to design a random walk whose partial cover time is as small as possible under the constraint that any node in the graph is allowed to query only a random sample of fixed size of its neighbors to retrieve their degrees.

We propose a new algorithm for reducing the partial cover time, called *Min-Degree* (in short, `MD`). The `MD` algorithm combines ideas from the literature in a novel way, which builds random walks displaying these two key properties: *(a)* RWs will preferentially visit unvisited nodes first and *(b)* among unvisited nodes, RWs will prefer transitioning to the lowest-degree node.

We have conducted extensive validation tests on six real-life graph. On each we have compared the `MD` algorithm with four state-of-the-art algorithms and found it highly competitive.

This paper is organized as follows. In Section 2 we provide basic definitions and background results while in Section 3 we discuss the related literature. Section 4 describes the `MD` algorithm while we present the main findings of our experimental analysis in Section 5. Finally, we draw our conclusions in Section 6.

2 BACKGROUND

Let $\mathcal{G} = \langle N, E \rangle$ be an undirected and connected graph with $|N| = n$ nodes and $|m| = m$ edges. We say that G is of *order* n and *size* m.

For any node $i \in N$, let d_i be the degree of i, i.e., the number of edges incident onto i and let $N(i)$ be the set of neighbours of i, i.e, the set of nodes $j \in N$ for which the edge $\langle i, j \rangle$ belongs to E.

A Random Walk (in short, RW) on the graph G is the process of visiting the nodes of G in some sequential random order. The RW starts at some fixed node, and, at each step, it moves from the current node (say i) to the next one (say j) with probability (called *transition probability*) p_{ij}. We can collect p_{ij} transition probabilities into a matrix $\mathbf{P}$ called *transition matrix probability*.

A *Simple Random Walk - SRW* is a Random Walk such that the next node to visit is chosen uniformly at random from the set of neighbors of the current node. In other words, if the walk is at node i, then it will move to the node j in the next step with probability $p_{ij} = \frac{1}{d_i}$ if $j \in N(i)$ and $p_{ij} = 0$ otherwise.

Let us consider a RW starting from a node, say i: we say that the RW *covers* G if the RW visits at least once every node in G [Kahn et al. 1989]. For each node $i \in N$ we can define a random variable X_i which specifies the first time a RW starting from i covers G.

A long standing problem in random walk theory consists of estimating the expected value of X_i, for any node i from which the random walk starts visiting G.

More formally, we provide the following definition [Aldous 1989; Kahn et al. 1989]:

Definition 2.1 (Cover Time). Given a node i , the *cover time* $C_G(i)$ for the node i is defined as $C_G(i) = \mathbb{E}[X_i]$, i.e., it is the *expected number of steps* the random walk takes to visits all nodes in $\mathcal{G}$, provided that it starts from i.

The *maximum cover time* C_G is defined as:

$$C_G = \max_{i \in N} C_G(i) \tag{1}$$

The cover time of a graph represents thus a parameter to evaluate the efficiency of a random walk, i.e., to quantify how fast a random walk is in covering $\mathcal{G}$. The cover time of a graph (along with methods for bounding it) have been extensively investigated [Aleliunas et al. 1979; Chandra et al. 1996; Kahn et al. 1989; Matthews 1988], especially for Simple Random Walks.

One of the first result is due to Aleliunas *et al.* [Aleliunas et al. 1979] who showed that for any connected graph G the cover time $C(G)$ satisfies $C(G) < 2 \times m \times n$ which is bounded above by $O(n^3)$. Feige [Feige 1995a,b] improved the results of [Aleliunas et al. 1979] and, specifically, he showed that, for any connected graph G, the cover time satisfied the following condition:

$$(1 - o(1))\, n \log n < C_G < (1 + o(1))\, \frac{4}{27} n^3 \tag{2}$$

The lower bound occurs in case of a complete graph of order n (i.e. a graph in which any pair of nodes is connected by an edge) while the upper bound occurs for the so-called *lollipop graph*. In case of regular graphs (i.e., graphs in which nodes have the same degree), Kahn *et al.* [Kahn et al. 1989] proved that $C(G)$ is bounded above by $O(n^2)$.

In general, highly connected graphs display the lowest cover time; in contrast, if graph connectivity is poor or if bottlenecks exist in the graph, then we expect an increase in cover time.

In many Web-based applications, however, the cover time may not be a reliable indicator of the efficiency of a random walk. For instance, suppose we consider a virtual community and let us focus on the spreading of a rumor in that community; in general, it does not matter that low-degree nodes receive the rumor and it does not matter that the whole population receives the rumor. In many cases, it suffices to verify that a relatively large portion of the whole population has received that rumor and, thus, we are required to estimate the number of steps a walk takes before visiting a fraction τ (with $0 \leq \tau \leq 1$) of nodes in $\mathcal{G}$. Such an intuition is encoded in the notion of *partial cover time* [Avin and Brito 2004; Avin and Krishnamachari 2008]:

Definition 2.2 (Partial Cover Time). Let $\mathcal{G}$ be undirected and connected with order n and let i be a node in G and $\tau \in [0, 1]$. The *partial cover time* $PCT_G(\tau, i)$ for node $i \in N$ is the expected number of steps a random walk takes to visit

at least $\lfloor \tau \times |N| \rfloor$ nodes in $\mathcal{G}$, provided that the random walk starts from the node i. The *partial cover time* $PCT_G(\tau)$ is defined as follows.

$$PCT_G(\tau) = \max_{i \in N} PCT_G(\tau, i) \tag{3}$$

Some important bounds on $PCT_G(\tau)$ are possible, as in the the following.

Definition 2.3 (Hitting Time). Let $\mathcal{G} = \langle N, E \rangle$ be an undirected and connected graph. Given a pair of nodes $i \in N$ and $j \in N$, the *hitting time* $H_G(i, j)$ is defined as the *expected number of step* a random walk takes to get to j, provided that it starts from i. The *maximum hitting time* H_G is defined as:

$$H_G = \max_{i \in N, j \in N} H_G(i, j) \tag{4}$$

[Avin and Brito 2004] proved that for any graph $\mathcal{G}$ and $0 \leq \tau < 1$ we have that $PCT_G(\tau) \in \Theta(H_G)$. As a consequence, if $\mathcal{G}$ is such that $H_G \in O(n)$, then there exists a random walk which achieves a partial cover time which is also linear in the number n of graph nodes.

[Avin and Brito 2004] considered a partial cover time in the order of $O(n)$ as *optimal* and they provided some examples of graphs for which it is possible to design random walks achieving optimal partial cover time, namely i) the complete graph, ii) the star, iii) the hypercube, iv) the 3-dimensional mesh and v) random geometric graphs (i.e., undirected graphs where nodes belong to some metric space and the probability of an edge between two nodes decreases with their distance in that space).

3 RELATED WORKS

In this section we review some of the most popular techniques to reduce the cover time of a random walk.

3.1 Non-uniform transition probabilities

Some authors [Abdullah et al. 2015; Ikeda et al. 2009] suggested to use proper transition probabilities, which derive from the knowledge of the topology of $\mathcal{G}$, to reduce the cover time $C(\mathcal{G})$.

In detail, a very important result is due to Ikeda *et al.* [Ikeda et al. 2009], who considered a transition probability matrix $\mathbf{P}$ defined as follows:

$$p_{ij} = \begin{cases} \frac{d_j^{-1/2}}{\sum_{k \in N(i)} d_j^{-1/2}}, & \text{if } \langle i, j \rangle \in E \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

[Ikeda et al. 2009] proved that, *for any graph* G, a random walk in which transition probabilities follow Equation 5 has an hitting time in the order of $O(n^2)$ and a cover time in the order of $O(n^2 \log n)$. [Ikeda et al. 2009] proved also that a random walk whose transition probabilities obeyed Equation 5 were also *optimal* for graphs with an *arbitrary topology*, i.e., it is not possible to further reduce the cover time unless we restrict our attention on special classes of graphs. The results of Ikeda *et al.* [Ikeda et al. 2009] assumes that each node knows the degree of all its neighbors. In addition, observe that random walk in the framework of [Ikeda et al. 2009] are no longer simple because the walker may cross a node more than once; intuitively, such an approach works because a node tend to privilege low degree neighbours, thus favouring the exploration of regions of G which would be hard to reach.

Abdullah *et al.* [Abdullah et al. 2015] suggested as to use transition probabilities of the form $p_{ij} \propto 1/\min\left(d_i, d_j\right)$ and they called their choice *the minimum degree weighting scheme*. For this choice of transition probabilities, Abdullah *et al.* [Abdullah et al. 2015] proved that for every connected graph the hitting time is at most $6n^2$ that the cover time is at most $O(n^2 \log n)$. They further conjectured that if the minimum degree weighting scheme is applied, then every connected graph has cover time $O(n^2)$ but such a conjecture is still unverified to our knowledge.

3.2 Random Walks which prefer unvisited edges

An important research avenue to reduce cover time is to consider modified random walks which record the edges the random walk used to explore the graph G. More specifically, suppose that a particular step the random walk occupies a node i and let us consider the set of edges incident onto i. If there is at least an *unvisited edge* (i.e. an edge which has never been used by the random walk to explore $\mathcal{G}$), then the random walk picks one of the unvisited edges according to a prescribed rule $\mathcal{A}$; if there are no unvisited edges incident onto the node currently occupied by the random walk, then the random walk moves to a random neighbour.

The process above is called *E-Process* (or edge-process) [Berenbrink et al. 2015]. In the simplest case, the rule $\mathcal{A}$ is a uniform random choice over unvisited edges incident onto the node currently occupied by the walker but we do not exclude arbitrary choices of $\mathcal{A}$; as highlighted in [Berenbrink et al. 2015], the rule could be determined on-line by an adversary, or could vary from node to node.

An important approach to cite is due to Avin and Krishnamachari [Avin and Krishnamachari 2008], who explicitly focused on the reduction of the partial cover time. [Avin and Krishnamachari 2008] introduced the so-called *Random Walk with Choice*, or in short, $RWC(d)$ algorithm. The $RWC(d)$ algorithm is an extension of a standard random walk and, specifically, if we suppose that the random walk reaches a node i at the time step t, then the $RCW(d)$ algorithms performs the following steps:

(1) It selects, uniformly at random and with replacement, d of the neighbors of i, say $D(i)$ with $|D(i)| = d$.
(2) The random walk moves to the node j, selected according to the following rule:

$$j = \arg \min_{j \in D(i)} \frac{c_t(j) + 1}{d_j} \tag{6}$$

Here $c_t(j)$ counts the number of times the node j has been visited up to the time step t.

The parameter d is determined through experiments but in the special case $d = 1$ the $RWC(d)$ algorithm coincides with a Standard Random Walk.

4 APPROACH DESCRIPTION

We now present our approach, called *Min-Degree* (or, in short, `MD`) to reduce the partial cover time of an undirected and connected graph $G = \langle N, E \rangle$.

4.1 Main Features of the `MD` algoritmh

Previous research findings are relevant to design efficient strategies to navigate $\mathcal{G}$, i.e., strategies that use the lowest number of steps to visit a fraction τ of the nodes. For instance, the procedure proposed by [Ikeda et al. 2009] is *optimal* for the cover time, in the sense that if we would choose transition probabilities as in Equation 5 then we would obtain a random walk whose cover time is $O(n^2 \log n)$: the best lower bound for cover time we can hope for.

Unfortunately, the approach of [Ikeda et al. 2009] requires that a node knows the degrees of all of its neighbors. In the Social Web scenario (and, in general, in many Web related domains) such an assumption may be unrealistic: for instance, in real Online Social Networks, an individual may refuse to disclose the number and the identities of her/his friends for privacy reasons; in addition, for some applications, the time required for generating the full list of neighbors of a node could be unacceptably long.

We now introduce the new version of the problem, called *Partial Cover Time with Budget:* any node in the graph is allowed to query only a fixed number of neighbors to retrieve their degrees.

For the budget version of the problem we now define the MD algorithm. MD algorithm combines ideas from the literature, i.e., it builds random walks that have the following properties: *(a)* unvisited neighbors are preferred and *(b)* among unvisited neighbors lowest-degree nodes are preferred.

4.2 The MD algorithm

We now describe our MD algorithm. It takes as input an undirected and connected graph $\mathcal{G} = \langle N, E \rangle$ with $|N| = n$ nodes and $|E| = m$ edges, a threshold $\tau \in [0, 1]$, a starting node $i \in N$, and an integer budget B whose meaning will be clarified later. It returns the number of steps a random walk starting from i needs to visit, at least once; a subset of nodes of $\mathcal{G}$ consisting of $n_{\max} = \lfloor \tau \times |N| \rfloor$ nodes (see Algorithm 1 for a high level description).

Algorithm 1 The MD algorithm

$V \leftarrow \{i\}$
$x_i \leftarrow 1$
$n_{\max} \leftarrow \lfloor \tau \times |N| \rfloor$
$k \leftarrow i$
while $|V| < n_{\max}$ **do**
  $L(k) \leftarrow V - N(k)$
  if $|L_k| == 0$ **then**
    Draw a node j uniformly at random from $N(k)$
    $k \leftarrow j$
  else
    if $|L_k| \geq B$ **then**
      Draw a random sample $\hat{L}(k)$ of size B from L_k
      Let j be the smallest degree node in $\hat{L}(k)$
      $k \leftarrow j$
    else
      Let j be the smallest degree node in $L(k)$
      $k \leftarrow j$
    end if
  end if
  Add k to V
  $x_i \leftarrow x_i + 1$
end while
return x_i

MD uses an auxiliary variable x_i (which is set equal to 1 at the beginning) to record its progress. Let also k be an auxiliary variable storing the currently-visited nod (initially, of course, $k = i$). In addition, MD uses a set V to record the set of nodes already visited which, at the beginning, stores only the node i.

The MD algorithm is iterative and, at each iteration, it aims at adding a node to the set of visited nodes V; the algorithm stops as soon as the set V reaches cardinality $n_{\max} = \lfloor \tau \times |N| \rfloor$.

We thus describe the operations carried out within each iteration. Variable k contains the current node the walker is on and let $N(k)$ contain the set of neighbors of k.

MD will checks whether there are nodes in $N(k)$ which have not yet been visited; to do so it builds the set $L(k) = V - N(k)$.

If the cardinality of $L(k)$ is zero, then, there are no unvisited nodes in $N(k)$. Hence we select a random neighbour, say j, as in a Standard Random Walk.

In contrast, suppose that $|L(k)| > 0$, i.e., there is at least one of the neighbors of k which have not yet been visited. In this case, the MD algorithm has two options:

a) the set $L(k)$ contains at least B elements: the algorithm draws, uniformly at random, a subset $\hat{L}(k)$ of size B. The algorithm choose the lowest degree node from $\hat{L}(k)$ as the next node to move.
b) The set $L(k)$ contains no more than B elements: in this case, the algorithm chooses the lowest degree node in $L(k)$ as the next node to move to. In both the two cases, let j be the next node to visit. The algorithm MD renames the node j into k, which is the current on which the random walk is positioned.

The MD algorithm updates the set V by adding the node k and it increments by one the variable x_i. As previously noted, the process above stops if the cardinality of the set V reaches $n_{\max}$, x_i is returned as output.

As observed in Section 2, the number of steps a random walk starting from a node i takes to visit a fraction of nodes of G is a random variable X_i and, thus, the output of the MD algorithm is a realization, called x_i, of X_i. If we apply MD a large number of times, say T, we generate a sequence of observed values $x_i^1, \ldots, x_i^T$ and we take their average:

$$\rho(\tau) = \frac{1}{T} \sum_{\ell=1}^{T} x_i^{\ell} \tag{7}$$

By the Strong Law of Large Numbers [Ross 2006], we obtain that $\rho(\tau)$ converges to the actual partial cover time $PCT(\tau, i)$ (see Definition 2.2). In our experiments we found that $T = 10$ was sufficient to ensure convergence.

4.3 The role of the budget B

The budget B has a fundamental role in the MD algorithm that we wish to clarify in this section. When B is set to 1 the algorithm chooses, uniformly at random, one of the unvisited neighbors of the current node and, thus, it coincides with the Edge Process algorithm described in [Berenbrink et al. 2010].

It is instructive to consider the behaviour of the MD algorithm as B increases and, in detail, we wish to observe that if B is sufficiently large, then the MD algorithm would degenerate into a deterministic procedure. Specifically, let us suppose that the MD algorithm is currently visiting the node i; for a fixed value of B, say $B = B^{\star}$, let $L_i^{\star}$ be the set of nodes from which MD will choose the next node to move.

By construction, the MD algorithm selects the smallest degree node $n_{\min}^{\star} \in L_i^{\star}$. We ask for the probability p that $n_{\min}^{\star}$ coincide with the smallest degree node $n_{\min}$ in L_i.

The estimation of p depends on the node degree distribution and it will be experimentally discussed in Section 5.4; however, we expect that p will increase if the ratio $\frac{B^{\star}}{|L_i|}$ increases too. At the limit case $B^{\star} = |L_i|$ such a probability should be equal to one. Therefore, if $B^{\star}$ approaches to $|L_i|$, then MD would *always direct* the walk to a pre-specified node (namely the unvisited node of lowest degree) and, thus, it could be no longer considered a proper random process.

Table 1. Main features of the graphs used in our experimental evaluation

Dataset	Number of Nodes	Number of Edges	Clustering Coefficient	Diameter
Facebook-Pages	22 470	171 002	0.232	15
GitHub	37 700	289 003	0.013	7
BrightKite	58 228	214 078	0.172	16
Facebook Friendship	63 731	817 035	0.148	15
Flickr	105 938	2 316 948	0.089	9
Amazon	334 863	925 872	0.397	44

5 EXPERIMENTAL ANALYSIS

We have experimentally validated our MD algorithm by a comparative benchmark over a diversified set of six real datasets that are available in the public domain. We sought to address the following fundamental questions:

RQ$_1$ What is the optimal value for the budget B?

RQ$_2$ How efficient is the MD algorithm to find a partial cover of a graph G against other, state-of-the art, methods?

5.1 Dataset Description

We used six publicly-available benchmark graphs, whose main features are summarized in Table 1.

Facebook-Pages. This dataset was collected through the Facebook Graph API in November 2017 [Rozemberczki et al. 2019]. Nodes identify Facebook pages belonging to one of the following categories: politicians, governmental organizations, television shows and companies. Edges identify mutual "likes" between pages.

GitHub. This dataset was collected from the public GitHub API in June 2019 [Rozemberczki et al. 2019] and it describes a social network of GitHub developers. Nodes are developers who have starred at least 10 repositories and edges identify mutual follower relationships between them.

BrightKite. This dataset was obtained by collecting all the public check-in data between April 2008 to October 2010 for BrightKite, a location-based social networking Web site [Cho et al. 2011]. Nodes are associated with BrighKite members and edges specify friendship relationships.

Facebook Friendship. This dataset contains friendship data of Facebook users [Viswanath et al. 2009]. A node represents a user and an edge represents a friendship between two users.

Flickr. This dataset defines a graph in which nodes correspond to images from Flickr [McAuley and Leskovec 2012]. Edges are established between images which share some metadata, such as the same location or common tags to annotate an image.

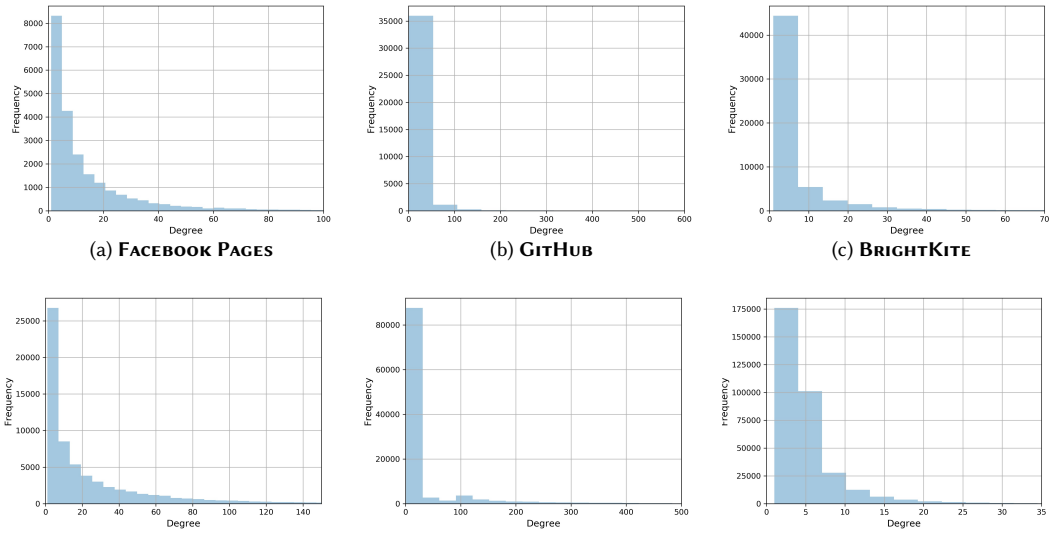
Amazon. This dataset defines the Amazon product co-purchasing network described in [Yang and Leskovec 2015]. Nodes represent products and edges connect commonly co-purchased products.

In Figures 1(a)- 1(f) we report node degree distribution for the datasets used in our tests.

We observe that node degree distribution is right-skewed for all datasets considered in our experimental trials.

Differences in observed distributions are likely to derive from the mechanisms regulating the formation and growth of each social network. For instance, the **GitHub** dataset collects mutual likes between GitHub members who are quite active as software contributors, and, thus, the average node degree is higher than in other social networks and approximately thousand nodes have a degree ranging from 50 to 110. Other datasets such as **Amazon** have edges

that represent the so-called *co-purchase* relationship. As expected, we observe that more than half of the nodes in the **Amazon** dataset display a degree less than five and the probability of observing a node with degree bigger than fifty is close to zero.



(a) **Facebook Pages** (b) **GitHub** (c) **BrightKite**

(d) Facebook (e) **Flickr** (f) **Amazon**

Fig. 1. Degree Distribution for the six datasets

5.2 Evaluation Metrics

We introduce the *normalized partial cover time* $C(\tau)$ *of an algorithm* as:

$$C(\tau) = \frac{\rho(\tau)}{n} \tag{8}$$

Here, the parameter $\rho(\tau)$ has been introduced in Equation 7 and it is normalized by the number n of graph nodes in order to make comparisons across graphs of different order possible.

Of course, the normalized partial cover time $C(\tau)$ increases (or, at least, it does not decrease) if τ increases. Given two methods M_1 and M_2 and a threshold $\overline{\tau} \in [0, 1]$, we say that the algorithm M_1 is *more efficient* than M_2 if the normalized partial cover time $C_1(\overline{\tau})$ associated with M_1 is less than the normalized partial cover time $C_2(\overline{\tau})$ associated with M_2.

5.3 Baseline Methods

We compared the `MD` algorithm with four baseline algorithms from the literature, namely:

- **Standard Random Walk, SRW**. This is the well-known random walk over an undirected and connected graph in which the walker selects the next node to move uniformly at random among its neighbors.
- **Edge-Process, EP**. This is the method described in [Berenbrink et al. 2015], and, unlike SRW, the random walk prefers unvisited edges to select the next node to reach.

- **All Degrees, AD**. This is the method described in [Ikeda et al. 2009] and it assumes that a node knows the degree of all its neighbors. We recall that the AD method is *optimal for cover time*, i.e., it achieves a cover time of $O\left(n^2 \log n\right)$ independently of the topology of the graph.
- **Random Walks with Choice - RWC(d)**. This is the method described in [Avin and Krishnamachari 2008]; in compliance with recommendations provided in [Avin and Krishnamachari 2008] and after some experiments, we decided to set $d = 3$ because such a value of d offered the lowest $C(\tau)$.

All these methods have been described in Section 3. We also tried the method described in [Abdullah et al. 2015] but we found it had worse performance than other methods above and, thus, we do not report its results here.

5.4 Budget Tuning (RQ_1)

In this section we study the role of the budget B on our MD algorithm. Recall from Section 4.3 when B increases the probability p that will MD choose the smallest-degree node among the neighbors will increase accordingly; so for higher values of B MD could be no longer considered a random-search process.

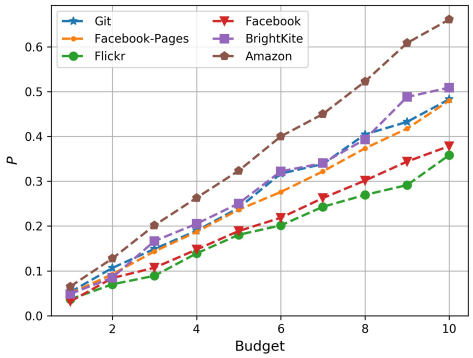


Fig. 2. Probability p of selecting the smallest degree node as function of the budget B.

Figure 2 reports the values of p as function of the budget B for all our datasets. Firstly, observe that, for all datasets under scrutiny, a value of $B = 10$ corresponds to a probability p ranging from 0.37 to 0.66: in other words, the MD algorithm has a high chance of discovering (and, thus, selecting) the smallest degree node even if it has at its disposal only ten nodes. Such a behavior depends on the degree distribution observable in many real-life graphs and, in particular, in the graphs considered in our study: node degree distribution is, in fact, right-skewed which implies that the vast majority of nodes displays a small degree (generally less than five). Therefore, a random sample of nodes in one of our graphs will, with high probability, contain one or more nodes of small degree; in many cases, the sample will also contain a node showcasing minimum degree.

A further observation is that p grows linearly with B in all datasets but its *rate of growth* differs across datasets: the steepest increase in p occurs for **Amazon**. Differences in slopes are attributable to the different node degree distribution we observe in each graph.

Finally, Figure 2 suggests that a value of $B = 5$ is generally reasonable because it implies a value of p always less than 0.32. Therefore, we set $B = 5$ for the experiments next.

5.5 Performance comparison (RQ_2)

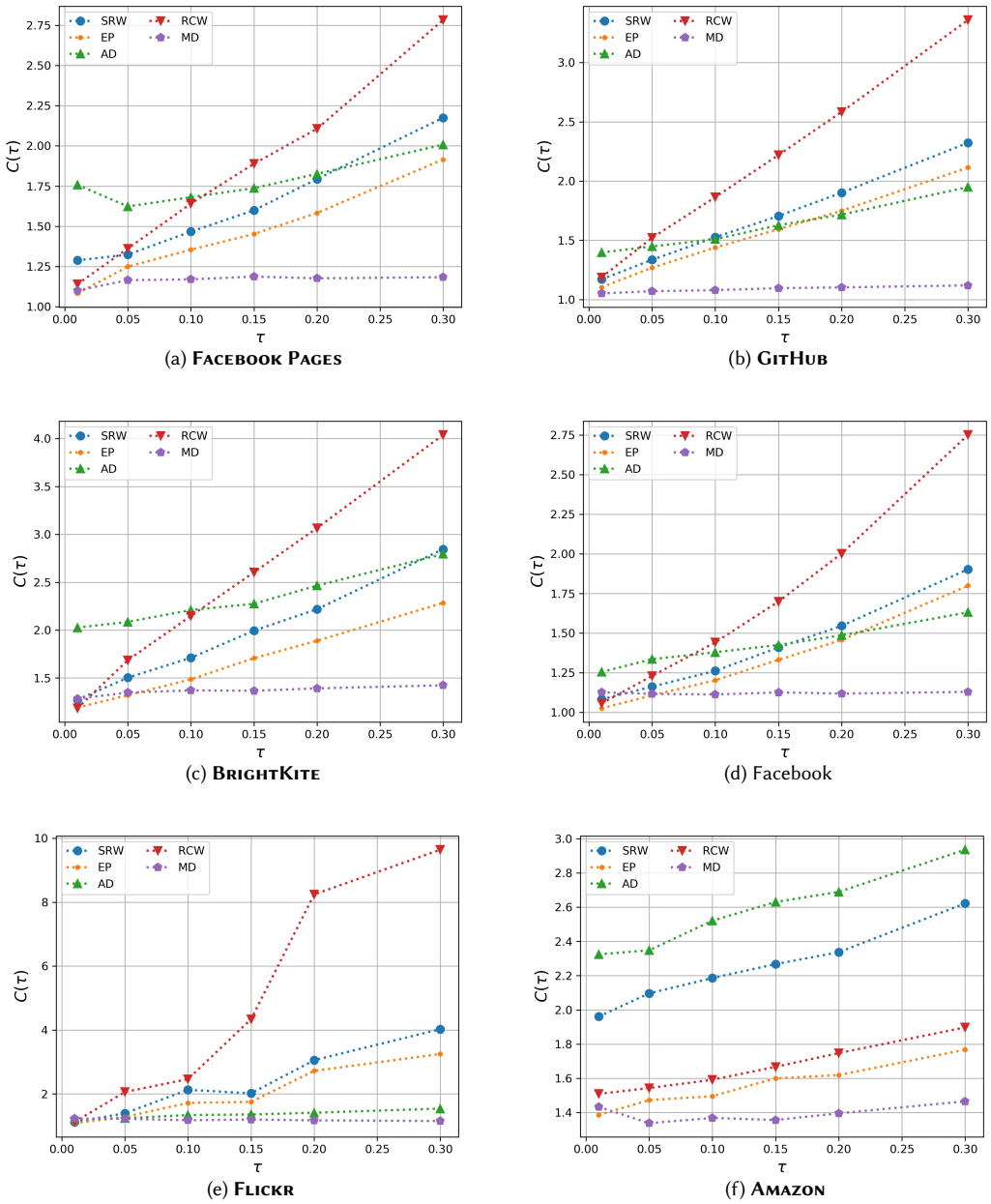
We used the normalized partial cover time $C(\tau)$ to compare the methods introduced in Section 5.3 and the MD algorithm.

The normalized partial cover time $C(\tau)$ obtained for values of τ ranging from 0.01 to 0.3 is reported in Figures 3(a)-3(f). The main findings of our experimental analysis can be summarized as follows:

- The MD algorithm significantly outperforms all other approaches if $\tau > 0.05$. In contrast, if $\tau \leq 0.05$ and we concentrate on **Brightkite**, **Facebook Friendship** and **Amazon** datasets, the MD algorithm is suboptimal, even if its normalized partial cover time $C(\tau)$ is very close to that of the best performing methods.
  The increase of $C(\tau)$ due to the increase τ in the MD algorithm is generally much slower than that experienced by other methods. We can therefore confirm the algorithmic idea underpinning MD, i.e., that biasing random walks toward low-degree and unvisited nodes actually accelerates the process of visiting a graph.
- Apart from our approach, the EP method performs very well if τ is small (i.e., it is smaller than 0.1); if τ is larger than 0.1 and we focus on the **Facebook** and **Flickr** datasets, the normalized partial cover time associated with the EP method deteriorates significantly but it is often significantly better than the normalized partial cover time observed for other methods. We can conclude, therefore, that the strategy of privileging unvisited edges yields a remarkable acceleration.
- In the SRW approach, we report an almost linear increase in $C(\tau)$ as τ increases too. If τ is smaller than 0.05, the SRW method is competitive with other methods, with the exception of the **Amazon** dataset. In general, poor performances of the SRW algorithm depends on the fact that the algorithm visits a node more than once and, thus, a larger number of steps are required before terminating.
- The AD method performs very well on the **Flickr** dataset: here, its normalized partial cover time is close to that of the MD algorithm and it is significantly smaller than the normalized partial cover time of all other methods. **Flickr** is also the most arduous dataset among those under scrutiny, i.e., the dataset on which all methods under investigation showcase the worst values of the normalized partial cover time. The AD method displays its worst performances on the **Amazon** dataset. That's not surprising: while the AD algorithm achieves, in the worst case, the optimal cover time for a graph of arbitrary topology, there are no guarantees that AD will is also be the most efficient choice for minimizing the partial cover time (and, thus, for normalized partial cover time) [Avin and Brito 2004]. Our experiments, therefore, prove that on real-life graphs the AD algorithm might not be competitive, if we goal is to minimize the (unbudgeted) normalized partial cover.
- With the exception of the **Amazon** dataset, the normalized partial cover time of the $RCW(d)$ algorithm is worse than that of all other methods. This result is somewhat surprising since the normalized partial cover time of the $RCW(d)$ algorithm is often worse than that of an SRW.
  It must be stressed, however, that the $RCW(d)$ approach has been designed to optimize the partial cover time for specific topologies such as regular graphs, grids, hypercubes or random geometric graphs (used to model wireless networks). Those topologies differ significantly from the topology of the graphs considered in our study (which display a high irregularity in the node degree distribution). Differences in graph topology have a big impact on the partial cover time and they explain the large values of $C(\tau)$ we observed for the $RCW(d)$ algorithm.

6 CONCLUSIONS

We have introduced a variation of the (Partial) Graph Cover Time problem that considers budgets, defined as a limit on the accessibility of neighbor nodes. We have designed an efficient random-walk solution which operates exactly



(a) **Facebook Pages** (b) **GitHub**

(c) **BrightKite** (d) Facebook

(e) **Flickr** (f) **Amazon**

Fig. 3. Values of the normalized partial cover time $C(\tau)$ as function of τ for the SRW, EP, AD, RCW and MD algorithms.

under the constraint that a node can access only a fraction of its neighbors. The MD algorithm introduce here favorably combines heuristic search ideas, namely the preference for unvisited nodes and, among those, for lowest-degree ones.

Experiments on six real-life graphs in the Social Web scenario have confirmed that MD is effective and can outperform state-of-the-art methods.

We plan to extend these results in several directions. For instance, we will assess how (and if) partial cover time $PCT_G(\tau, i)$ depends on the choice of the "start" node and, specifically, whether some node features associated with a start node v (such as the Betweenness Centrality or the Eigenvector Centrality of i) are in fact predictive of $PCT_G(\tau, v)$. Another direction of research is to parametrize the number of allowed visits to metadata on size and density of the input graph, e.g. with a log-log dependence between number of nodes and budget (which corresponds to considering MD optimal for $|\mathcal{G}| > 2^{16} \approx 64k$). Finally, we will propose an analogous of MD for the directed-graph case, which is the natural model for asymmetric relationships that we had previously explored such as trust networks [Agreste et al. 2015] and online negotiation [Costantini et al. 2013].

ACKNOWLEDGMENTS

This article is based upon work from COST Action DigForAsp CA17124, supported by COST (European Cooperation in Science and Technology. www.cost.eu

REFERENCES

M. Abdullah, C. Cooper, and M. Draief. 2015. Speeding up cover time of sparse graphs using local knowledge. In *Proc. of the International Workshop on Combinatorial Algorithms (IWOCA 2015)*. Springer, Verona, Italy, 1–12.

S. Agreste, P. De Meo, E. Ferrara, S. Piccolo, and A. Provetti. 2015. Trust Networks: Topology, Dynamics, and Measurements. *IEEE Internet Computing* 19, 6 (2015), 26–35.

Y. Ahn, S. Han, H. Kwak, S. Moon, and H. Jeong. 2007. Analysis of topological characteristics of huge online social networking services. In *Proc. of the 16th International Conference on World Wide Web, WWW 2007*. ACM, Banff, Alberta, Canada, 835–844.

D. Aldous. 1989. An introduction to covering problems for random walks on graphs. *Journal of Theoretical Probability* 2, 1 (1989), 87–89.

R. Aleliunas, R. Karp, R. Lipton, L. Lovász, and C. Rackoff. 1979. Random Walks, Universal Traversal Sequences, and the Complexity of Maze Problems. In *Proc. of the Annual Symposium on Foundations of Computer Science (FOCS 1979)*. IEEE Computer Society, San Juan, Puerto Rico, 218–223.

C. Avin and C. Brito. 2004. Efficient and robust query processing in dynamic environments using random walk techniques. In *Proc. of the International Symposium on Information Processing in Sensor Networks*. Berkeley, California, 277–286.

C. Avin and G. Ercal. 2005. On the cover time of random geometric graphs. In *Proc. of the International Colloquium on Automata, Languages, and Programming (ICALP 2005)*. Springer, Lisbon, Portugal, 677–689.

C. Avin and B. Krishnamachari. 2008. The power of choice in random walks: an empirical study. *Computer Networks* 52, 1 (2008), 44–60.

P. Berenbrink, C. Cooper, R. Elsasser, T. Radzik, and T. Sauerwald. 2010. Speeding Up Random Walks with Neighborhood Exploration. In *Proc. of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010*. ACM Press, Austin, Texas, USA, 1422–1435.

P. Berenbrink, C. Cooper, and T. Friedetzky. 2015. Random Walks Which Prefer Unvisited Edges: Exploring High Girth Even Degree Expanders in Linear Time. *Random Structures And Algorithms* 46, 1 (2015), 36–54.

S. Catanese, P. De Meo, E. Ferrara, G. Fiumara, and A. Provetti. 2011. Crawling Facebook for social network analysis purposes. In *Proc. of the International Conference on Web Intelligence, Mining and Semantics, WIMS 2011*. ACM, Songdal, Norway, 52.

A. Chandra, P. Raghavan, W. Ruzzo, R. Smolensky, and P. Tiwari. 1996. The electrical resistance of a graph captures its commute and cover times. *Computational Complexity* 6, 4 (1996), 312–340.

E. Cho, S. Myers, and J. Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Diego, CA, USA, 1082–1090.

M. Chupeau, O. Bénichou, and R. Voituriez. 2015. Cover times of random searches. *Nature Physics* 11, 10 (2015), 844–847.

S. Costantini, G. De Gasperis, A. Provetti, and P. Tsintza. 2013. A heuristic approach to proposal-based negotiation: With applications in fashion supply chain management. *Mathematical Problems in Engineering* 2013, 896312 (2013).

P. De Meo, E. Ferrara, G. Fiumara, and A. Provetti. 2014. Mixing local and global information for community detection in large networks. *Journal of Computer and Systems Sciences* 80, 1 (2014), 72–87.

U. Feige. 1995a. A tight lower bound on the cover time for random walks on graphs. *Random Structures & Algorithms* 6, 4 (1995), 433–438.

U. Feige. 1995b. A tight upper bound on the cover time for random walks on graphs. *Random Structures & Algorithms* 6, 1 (1995), 51–54.

M. Gjoka, M. Kurant, C. Butts, and A. Markopoulou. 2011. Practical recommendations on crawling online social networks. *IEEE Journal on Selected Areas in Communications* 29, 9 (2011), 1872–1892.

P. Hu and W. Lau. 2013. A survey and taxonomy of graph sampling. *arXiv preprint arXiv:1308.5865* (2013).

S. Ikeda, I. Kubo, and M. Yamashita. 2009. The hitting and cover times of random walks on finite graphs using local degree information. *Theoretical Computer Science* 410, 1 (2009), 94–100.

J. Kahn, N. Linial, N. Nisan, and M. Saks. 1989. On the cover time of random walks on graphs. *Journal of Theoretical Probability* 2, 1 (1989), 121–128.

L. Lovász. 1993. Random walks on graphs. *Combinatorics* 2, 1 (1993), 1–46.

P. Matthews. 1988. Covering problems for Brownian motion on spheres. *The Annals of Probability* 16, 1 (1988), 189–199.

J. McAuley and J. Leskovec. 2012. Image Labeling on a Network: Using Social-Network Metadata for Image Classification. In *Proc. of the European Computer Vision Conference, ECCV 2012*. Florence, Italy, 828–841.

M. Newman. 2005. A measure of betweenness centrality based on random walks. *Social networks* 27, 1 (2005), 39–54.

M. Newman. 2018. *Networks*. Oxford university press.

P. Pons and M. Latapy. 2006. Computing Communities in Large Networks Using Random Walks. *J. Graph Algorithms Appl.* 10, 2 (2006), 191–218.

S. Redner. 2001. *A guide to first-passage processes*. Cambridge University Press.

S. Ross. 2006. *A first course in probability*. Pearson Prentice Hall Upper Saddle River, NJ.

B. Rozemberczki, C. Allen, and R. Sarkar. 2019. Multi-scale Attributed Node Embedding. *arXiv preprint arXiv:1909.13021* (2019).

B. Viswanath, A. Mislove, M. Cha, and K. Gummadi. 2009. On the evolution of user interaction in Facebook. In *Proc. of the 2nd ACM workshop on Online social networks*. Barcelona, Spain, 37–42.

T. Weng, J. Zhang, M. Small, F. Bijarbooneh, and P. Hui. 2017. Partial cover time that is sublinear in the number of targets on complex networks: a universal law. *arXiv preprint arXiv:1701.03259* (2017).

J. Yang and J. Leskovec. 2015. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems* 42, 1 (2015), 181–213.