

# THE ROBOT CRAWLER GRAPH PROCESS

ANTHONY BONATO, RITA M. DEL RÍO-CHANONA, CALUM MACRURY, JAKE NICOLAIDIS, XAVIER PÉREZ-GIMÉNEZ, PAWEŁ PRALAT, AND KIRILL TERNOVSKY

ABSTRACT. Information gathering by crawlers on the web is of practical interest. We consider a simplified model for crawling complex networks such as the web graph, which is a variation of the robot vacuum edge-cleaning process of Messinger and Nowakowski. In our model, a crawler visits nodes via a deterministic walk determined by their weightings which change during the process deterministically. The minimum, maximum, and average time for the robot crawler to visit all the nodes of a graph is considered on various graph classes such as trees, multi-partite graphs, binomial random graphs, and graphs generated by the preferential attachment model.

## 1. INTRODUCTION

A central paradigm in web search is the notion of a *crawler*, which is a software application designed to gather information from web pages. Crawlers perform a walk on the web graph, visiting web pages and then traversing links as they explore the network. Information gathered by crawlers is then stored and indexed, as part of the anatomy of a search engine such as Google or Bing. See [11, 18, 27] and the book [24] for a discussion of crawlers and search engines.

Walks in graph theory have been long-studied, stretching back to Euler’s study of the Königsberg bridges problem in 1736, and including the travelling salesperson problem [3] and the sizeable literature on Hamiltonicity problems (see, for example, [30]). An intriguing generalization of Eulerian walks was introduced by Messinger and Nowakowski in [25], as a variant of graph cleaning processes (see, for example, [2, 26]). The reader is directed to [9] for an overview of graph cleaning and searching.

In the model of [25], called the *robot vacuum*, it is envisioned that a building with dirty corridors (for example, pipes containing algae) is cleaned by an autonomous robot. The robot cleans these corridors in a greedy fashion, so that the next corridor cleaned is always the “dirtiest” to which it is adjacent. This is modelled as a walk in a graph. The robot’s initial position is any given node, with the initial weights for the edges of the graph  $G$  being  $-1, -2, \dots, -|E(G)|$  (each edge has a different value). At every step of the walk, the edges of the graph will be assigned different weights indicating the last time each one was cleaned (and thus, its level of dirtiness). It is assumed that each edge takes the same length of time to clean, and so weights are taken as integers. In such a model, it is an exercise to show that for a connected graph, one robot will eventually clean the graph (see [25]).

In the robot vacuum model, let  $s(G)$  and  $S(G)$  denote the minimum and maximum number of time-steps over all edge weightings, respectively, when every edge of a graph

---

Research supported by grants from NSERC, MITACS Inc. and Ryerson University.

$G$  has been cleaned. As observed in [25], if  $G$  is an Eulerian graph, then we have that  $s(G) = |E(G)|$ , and moreover the final location of the robot after the first time every edge has been cleaned is the same as the initial position. Li and Vetta [22] gave an interesting example where the robot vacuum takes exponential time to clean the graph. Let  $S_e$  be the maximum value of  $S(G)$  over all connected graphs  $G$  containing exactly  $e$  edges. It is proven in [22] that there exists an explicit constant  $d > 0$  such that, for all  $e$ ,  $S_e \geq d(3/2)^{e/5} - 1/2$ . Moreover,  $S_e \leq 3^{e/3+1} - 3$ . An analogous result was independently proven by Copper et al. [14] who analyzed a similar model to the robot vacuum. The “self-stabilization” found in robot vacuum is also a feature of so-called *ant algorithms* (such as the well-known *Langton’s ant* which is capable of simulating a *universal Turing machine*; see [17]). The robot vacuum model can be regarded as an undirected version of the *rotor-router* model; see [29, 31].

In the present work, we provide a simplified model of a robot crawler on the web, based on the robot vacuum paradigm of [22, 25] described above. We note that the paper is the full version (with full proofs and additional results) of the proceedings version of the paper [8]. In our model, the crawler cleans nodes rather than edges. Nodes are initially assigned unique non-positive integer weights from  $\{0, -1, -2, \dots, -|V(G)|+1\}$ . In the context of the web or other complex networks, weights may be correlated with some popularity measure such as in-degree or PageRank. The robot crawler starts at the dirtiest node (that is, the one with the smallest weight), which immediately gets its weight updated to 1. Then at each subsequent time-step it moves greedily to the dirtiest neighbour of the current node. On moving to such a node, we update the weight to the positive integer equalling the time-step of the process. The process stops when all weights are positive (that is, when all nodes have been cleaned). Note that while such a walk by the crawler may indeed be a Hamilton path, it usually is not, and some weightings of nodes will result in many re-visits to a given node. Similar models to the robot crawler have been studied in other contexts; see [20, 23, 29].

The paper is organized as follows. A rigorous definition of the robot crawler is given in Section 2. We consider there the minimum, maximum, and average number of time-steps required for the robot crawler model. The connections between the robot crawler and robot vacuum are discussed in Section 3. In Section 4, we give asymptotic (and in some cases exact) values for these parameters for paths, trees, and complete multipartite graphs. In Section 5, we consider the average number of time-steps required for the robot crawler to explore binomial random graphs. The robot crawler is studied on the preferential attachment model, one of the first stochastic models for complex networks, in Section 6.

Throughout, we consider only finite, simple, and undirected graphs. For a given graph  $G = (V, E)$  and  $v \in V$ ,  $N(v)$  denotes the neighbourhood of  $v$  and  $\deg(v) = |N(v)|$  its degree. For background on graph theory, the reader is directed to [30]. For a given  $n \in \mathbb{N}$ , we use the notation  $B_n = \{-n+1, -n+2, \dots, -1, 0\}$  and  $[n] = \{1, 2, \dots, n\}$ . All logarithms in this paper are with respect to base  $e$ . We say that an event  $A_n$  holds *asymptotically almost surely* (*a.a.s.*) if it holds with probability tending to 1 as  $n$  tends to infinity. All asymptotics throughout are as  $n \rightarrow \infty$  (we emphasize that the notations  $o(\cdot)$  and  $O(\cdot)$  refer to functions of  $n$ , not necessarily positive, whose growth is bounded).

For simplicity, we will write  $f(n) \sim g(n)$  if  $f(n)/g(n) \rightarrow 1$  as  $n \rightarrow \infty$  (that is, when  $f(n) = (1 + o(1))g(n)$ ).

## 2. DEFINITION AND PROPERTIES

We now formally define the robot crawler model and the various robot crawler numbers of a graph. The *robot crawler*  $\mathcal{RC}(G, \omega_0) = ((\omega_t, v_t))_{t=1}^L$  of a connected graph  $G = (V, E)$  on  $n$  nodes with an *initial weighting*  $\omega_0 : V \rightarrow B_n$ , that is a bijection from the node set to  $B_n$ , is defined as follows.

- (1) Initially, set  $v_1$  to be the node in  $V$  with weight  $\omega_0(v_1) = -n + 1$ .
- (2) Set  $\omega_1(v_1) = 1$ ; the other values of  $\omega_1$  remain the same as in  $\omega_0$ .
- (3) Set  $t = 1$ .
- (4) If all the weights are positive (that is,  $\min_{v \in V} \omega_t(v) > 0$ ), then set  $L = t$ , stop the process, and return  $L$  and  $\mathcal{RC}(G, \omega_0) = ((\omega_t, v_t))_{t=1}^L$ .
- (5) Let  $v_{t+1}$  be the dirtiest neighbour of  $v_t$ . More precisely, let  $v_{t+1}$  be such that
 
$$\omega_t(v_{t+1}) = \min\{\omega_t(v) : v \in N(v_t)\}.$$
- (6)  $\omega_{t+1}(v_{t+1}) = t + 1$ ; the other values of  $\omega_{t+1}$  remain the same as in  $\omega_t$ .
- (7) Increment to time  $t + 1$  (that is, increase  $t$  by 1) and return to 4.

If the process terminates, then define

$$\text{rc}(G, \omega_0) = L,$$

that is  $\text{rc}(G, \omega_0)$  is equal to the number of steps in the *crawling sequence*  $(v_1, v_2, \dots, v_L)$  (including the initial state) taken by the robot crawler until all nodes are clean; otherwise  $\text{rc}(G, \omega_0) = \infty$ . We emphasize that for a given  $\omega_0$ , all steps of the process are deterministic. Note that at each point of the process, the weighting  $\omega_t$  is an injective function. In particular, there is always a unique node  $v_{t+1}$ , neighbour of  $v_t$  of minimum weight (see step (4) of the process). Hence, in fact, once the initial configuration is fixed, the robot crawler behaves like a cellular automaton. It will be convenient to refer to a node as *dirty* if it has a non-positive weight (that is, it has not been yet visited by the robot crawler), and *clean*, otherwise.

The next observation that the process always terminates in a finite number of steps is less obvious.

**Theorem 1.** *For a connected graph  $G = (V, E)$  on  $n$  nodes and a bijection  $\omega_0 : V \rightarrow B_n$ ,  $\mathcal{RC}(G, \omega_0)$  terminates after a finite number of steps; that is,  $\text{rc}(G, \omega_0) < \infty$ .*

*Proof.* For a contradiction, suppose that there exists a connected graph  $G = (V, E)$  and initial weightings  $\omega_0$  such that  $\mathcal{RC}(G, \omega_0)$  runs forever. The node set  $V$  can be partitioned into two sets:  $V_{<\infty}$  consists of nodes that are visited a finite number of times (including those that are not visited at all);  $V_\infty = V \setminus V_{<\infty}$  consists of the nodes that are visited infinitely many times. Note that  $V_{<\infty} \neq \emptyset$ ; otherwise, the process would terminate. Moreover,  $V_\infty \neq \emptyset$  as the graph is finite but the process goes forever. Since  $G$  is connected, there exist  $w \in V_{<\infty}$  and  $u \in V_\infty$  such that  $uw \in E$ .

Let  $U = N(u) \cap V_{<\infty}$  be the set of neighbours of  $u$  that are visited a finite number of times, and let  $T$  be the last time-step when some node from  $U$  was visited (that

is,  $\omega_t(v) = \omega_T(v)$  for any  $v \in U$  and  $t > T$ ). Note that  $\{w\} \subseteq U$ . Since  $u$  and all neighbours of  $u$  outside of  $U$  are visited infinitely many times, there exists time-step  $S > T$  such that the robot crawler arrives at node  $u$  and all nodes of  $N(u) \setminus U$  were visited at least once since time-step  $T$ . But then the process must move to some node of  $U$  as all the neighbours of  $u$  outside  $U$  are “cleaner” than those of  $U$ . This give us a desired contradiction as no node of  $U$  is visited after time-step  $T$ .  $\square$

The fact that every node in a graph will be eventually visited inspires the following definition. Let  $G = (V, E)$  be any connected graph on  $n$  nodes. Let  $\Omega_n$  be the family of all initial weightings  $\omega_0 : V \rightarrow B_n$ . Then

$$\text{rc}(G) = \min_{\omega_0 \in \Omega_n} \text{rc}(G, \omega_0) \quad \text{and} \quad \text{RC}(G) = \max_{\omega_0 \in \Omega_n} \text{rc}(G, \omega_0).$$

In other words,  $\text{rc}(G)$  and  $\text{RC}(G)$  the are minimum and maximum number of time-steps, respectively, needed to crawl  $G$ , over all choices of initial weightings. Now let  $\bar{\omega}_0$  be an element taken uniformly at random from  $\Omega_n$ . Then we have the average case evaluated as

$$\bar{\text{rc}}(G) = \mathbb{E}[\text{rc}(G, \bar{\omega}_0)] = \frac{1}{|\Omega_n|} \sum_{\omega_0 \in \Omega_n} \text{rc}(G, \omega_0).$$

Now, let us make some simple observations.

**Lemma 2.** *Let  $G$  be a connected graph of order  $n$ , maximum degree  $\Delta$ , and diameter  $d$ . Let  $C_n$  and  $K_n$  denote the cycle and the clique of order  $n$ , respectively.*

- (1)  $\text{rc}(G) \leq \bar{\text{rc}}(G) \leq \text{RC}(G)$ .
- (2)  $\text{rc}(K_n) = \bar{\text{rc}}(K_n) = \text{RC}(K_n) = n$
- (3)  $\text{rc}(C_n) = \bar{\text{rc}}(C_n) = \text{RC}(C_n) = n$ .
- (4)  $\text{rc}(G) = n$  if and only if  $G$  has a hamiltonian path.
- (5)  $\text{RC}(G) \leq n(\Delta + 1)^d$ .

*Proof.* We only prove item (5) since the proofs of the other items are straightforward. First, observe that if a node  $v$  is visited by the robot crawler  $\Delta + 1$  times within an interval of time-steps, then right after each of the first  $\Delta$  visits to  $v$  in that interval, the robot crawler must visit a new neighbour of  $v$  that was not yet visited in that interval (if any still exists). Therefore, since  $|N(v)| \leq \Delta$ , all neighbours of  $v$  must be visited at least once in that interval. In view of this, if a node  $v$  is cleaned  $(\Delta + 1)k$  times during the robot crawler process, then each of its neighbours must be cleaned at least  $k$  times. Suppose now that the robot crawler runs for  $n(\Delta + 1)^d$  or more steps. By the pigeonhole principle, at least one node  $v$  is cleaned at least  $(\Delta + 1)^d$  times. By inductively applying our previous conclusion, we deduce that every node at distance  $0 \leq t \leq d$  from  $v$  is visited at least  $(\Delta + 1)^{d-t} \geq 1$  times by the robot crawler. The proof follows as the diameter is  $d$ .  $\square$

### 3. CONNECTIONS BETWEEN THE ROBOT CRAWLER AND ROBOT VACUUM MODELS

In this section, we are going to show that our model is a generalization of the robot vacuum model studied by Messinger and Nowakowski in [25]. We will use the following notation for that model, analogous to the one defined in Section 2 for the robot crawler.

Let  $G = (V, E)$  be any connected graph. Given a bijection  $\omega_0 : E \rightarrow B_{|E|}$  and a node  $v_0 \in V$ , we denote by  $\mathcal{S}(G, \omega_0, v_0)$  the robot vacuum process that starts at node  $v_0$  and has initial weighting  $\omega_0$  of the edges;  $s(G, \omega_0, v_0)$  is the number of steps taken by the robot vacuum to visit all edges at least once;  $s(G)$  is the minimum of  $s(G, \omega_0, v_0)$  over all  $(\omega_0, v_0)$ ; and similarly  $S(G)$  is the maximum of  $s(G, \omega_0, v_0)$  over all  $(\omega_0, v_0)$ . Note that, in this model, steps are associated to edges instead of nodes and thus, the number of steps taken by the robot vacuum in  $\mathcal{S}(G, \omega_0, v_0)$  counts the total number of visits to the edges during the process. Moreover, the following generalization of the graph parameters  $\text{rc}(G)$  and  $\text{RC}(G)$  will turn out to be useful: for any  $W \subseteq V$ , let

$$\text{rc}(G, W) = \min\{\text{rc}(G, \omega_0) : \omega_0 \in \Omega_n \text{ such that } \mathcal{RC}(G, \omega_0) \text{ starts at some node of } W\},$$

$$\text{RC}(G, W) = \max\{\text{rc}(G, \omega_0) : \omega_0 \in \Omega_n \text{ such that } \mathcal{RC}(G, \omega_0) \text{ starts at some node of } W\}.$$

In particular,  $\text{rc}(G) = \text{rc}(G, V)$  and  $\text{RC}(G) = \text{RC}(G, V)$ . Moreover, for any sets of vertices  $\emptyset \neq W_1 \subseteq W_2 \subseteq V$ ,

$$\text{rc}(G) \leq \text{rc}(G, W_2) \leq \text{rc}(G, W_1) \leq \text{RC}(G, W_1) \leq \text{RC}(G, W_2) \leq \text{RC}(G).$$

Finally, for any  $k \in \mathbb{N}$ , a  $k$ -subdivision of  $G$ ,  $L_k(G)$ , is a graph that is obtained from  $G$  by replacing each edge of  $G$  by a path of length  $k$ .

The following theorem connects the robot crawler and robot vacuum models, showing how the parameters  $\text{rc}$  and  $s$  relate.

**Theorem 3.** *Let  $G = (V, E)$  be any connected graph and let  $k \in \mathbb{N} \setminus \{1\}$ . Then we have that*

$$\text{rc}(L_k(G), V) \in \{k \cdot s(G), k \cdot s(G) + 1\} \quad (1)$$

and

$$\text{rc}(L_k(G)) \in \{k \cdot s(G) - 1, k \cdot s(G), k \cdot s(G) + 1\}. \quad (2)$$

Note that in (1), we are restricting ourselves to crawling sequences that start from a node in  $V$ , which is the node set of  $G$  and thus, a strict subset of the node set of  $L_k(G)$ .

*Proof of Theorem 3.* Suppose  $G$  has  $n$  nodes and  $m$  edges. Let  $\omega_0 : E \rightarrow B_m$  be a bijection, and  $v_0 \in V$  be such that  $s(G, \omega_0, v_0) = s(G)$ . Recall that the node set of  $L_k(G)$  consists of  $V$ , *original* nodes of  $G$ , and  $(k-1)m$  *cloned* nodes ( $k-1$  nodes for each edge of  $G$ ) that are created during the subdivision process. We construct  $\hat{\omega}_0 : V(L_k(G)) \rightarrow B_{(k-1)m+n}$  as follows. Assign initial weight  $-(k-1)m - n + 1$  to  $v_0$  to insure it is the dirtiest node of  $L_k(G)$ . Next, we run  $\mathcal{S}(G, \omega_0, v_0)$ , the edge variant of the process on graph  $G$ . Each time an edge of  $G$  is visited for the first time, we assign initial weights to the corresponding cloned nodes of  $L_k(G)$ ; starting from the smallest available weight (that is,  $-(k-1)m - n + 2$ ) and then using consecutive integers. Initial weights for original nodes from  $V \setminus \{v_0\} \subseteq V(L_k(G))$  can be assigned arbitrarily from  $B_{n-1}$ .

In order to derive an upper bound for  $\text{rc}(L_k(G), V)$ , we will start the process on  $L_k(G)$  with initial weighting  $\hat{\omega}_0$  (and thus, from  $v_0$ ). An important property of  $L_k(G)$  is that all neighbours of any original node  $v \in V$  are cloned nodes associated with some edges of  $G$  (incident to  $v$  in  $G$ ). Hence, it is easy to see that the process on  $L_k(G)$  mimics the edge process on  $G$ : the robot crawler occupying some original node decides

where to go based on how dirty cloned nodes are (incident edges in the edge process); on the other hand, when the crawler enters some cloned node, it is immediately pushed through to the other original node. It follows that

$$\text{rc}(L_k(G), \hat{\omega}_0, v_0) \in \{k \cdot s(G), k \cdot s(G) + 1\}.$$

Note that each step of the edge process on  $G$  corresponds to  $k$  steps of the process on  $L_k(G)$ . Moreover, there are two possible endings of the process on  $L_k(G)$ . Suppose that the last step of the edge process on  $G$  is to move from  $v$  to  $u$ . The process on  $L_k(G)$  might finish at node  $u$  for the total number of steps  $k \cdot s(G) + 1$  or at the last cloned node corresponding to edge  $vu$  yielding  $k \cdot s(G)$ , depending whether  $u$  was visited earlier or not. In any case, we obtain that

$$\text{rc}(L_k(G), V) \leq k \cdot s(G) + 1.$$

Now, let us move to the lower bound for  $\text{rc}(L_k(G), V)$ . For a contradiction, suppose that there exists a bijection  $\hat{\omega}_0 : V(L_k(G)) \rightarrow B_{(k-1)m+n}$  that assigns the smallest weight to some  $v_0 \in V$  and such that  $\text{rc}(L_k(G), \hat{\omega}_0) < k \cdot s(G)$ . In order to construct the corresponding initial configuration  $\omega_0 : E \rightarrow B_m$  for the edge process, we use a similar reduction trick as before. We run  $\mathcal{RC}(L_k(G), \hat{\omega}_0)$ . Each time some cloned node of  $L_k(G)$  is visited for the first time, we assign an initial weight to the corresponding edge of  $G$ ; starting from the smallest weight (that is,  $-m + 1$ ) and then using consecutive integers. Arguing as before, we derive that  $\mathcal{S}(G, \omega_0, v_0)$  mimics  $\mathcal{RC}(L_k(G), \hat{\omega}_0)$  implying that  $s(G) \leq s(G, \omega_0, v_0) \leq \frac{1}{k} \text{rc}(L_k(G), \hat{\omega}_0) < s(G)$ . This contradiction implies that

$$\text{rc}(L_k(G), V) \geq k \cdot s(G),$$

and the proof of (1) follows.

The upper bound for (2) is obvious as  $\text{rc}(L_k(G)) \leq \text{rc}(L_k(G), V)$  so we need to concentrate on the lower bound. For a contradiction, let us suppose that there exist some cloned node  $e_0 \in V(L_k(G)) \setminus V$  and bijection  $\hat{\omega}_0 : V(L_k(G)) \rightarrow B_{(k-1)m+n}$  such that  $\hat{\omega}_0$  assigns the smallest weight to  $e_0$  and  $\text{rc}(L_k(G), \hat{\omega}_0) < k \cdot s(G) - 1$ . Let  $v_0$  be the first original node cleaned during the process. Clearly, one can start the process at  $v_0$  and craft the initial weighting so that the robot crawler still mimics  $\mathcal{RC}(L_k(G), \hat{\omega}_0)$ ; the omitted cloned nodes would obtain weights  $0, 1, \dots$ , and the relative order of weights of the remaining nodes would be preserved. Let us call this process an *adjusted* one. There are some cases we need to consider.

Case 1: The original process re-cleans  $e_0$  before it terminates.

The adjusted process would terminate even earlier than  $\mathcal{RC}(L_k(G), \hat{\omega}_0)$ , which implies that it was not an optimal initial configuration and, in fact,  $\text{rc}(L_k(G)) < \text{rc}(L_k(G), \hat{\omega}_0) < k \cdot s(G) - 1$ ; we are not using this fact though. An important thing is that it starts from original node  $v_0$  as this contradicts the fact that  $\text{rc}(L_k(G), V) \geq k \cdot s(G)$ .

Case 2: The original process terminates at a neighbour of  $e_0$  but never re-cleans  $e_0$ .

Note that the process might terminate on an original node or a cloned one. Regardless of that, the observation is that the adjusted process would also terminate after  $\text{rc}(L_k(G), \hat{\omega}_0)$  steps, again contradicting  $\text{rc}(L_k(G), V) \geq k \cdot s(G)$ . Indeed, once the adjusted process reaches the previous terminating node, it cleans the omitted cloned nodes and then it stops.

Case 3: The original process terminates at a node that is not a neighbour of  $e_0$  and never re-cleans  $e_0$ .

Note that  $e_0$  is adjacent to some original node  $w_0$  different than  $v_0$ ; indeed, if this is not the case, then the (cloned) neighbour of  $e_0$  not visited at step 2 of the process is never cleaned. This time we can start the process at  $w_0$  and craft the initial weighting so that the robot crawler still mimics  $\mathcal{RC}(L_k(G), \hat{\omega}_0)$ . It terminates after at most  $\text{rc}(L_k(G), \hat{\omega}_0) + 1 < k \cdot s(G)$  steps, as usual contradicting the fact that  $\text{rc}(L_k(G), V) \geq k \cdot s(G)$ .  $\square$

From Theorem 3, we derive the following straightforward but important corollary.

**Corollary 4.** *Let  $G = (V, E)$  be any connected graph. Then we have that*

$$s(G) = \left\lfloor \frac{\text{rc}(L_3(G)) + 1}{3} \right\rfloor = \left\lfloor \frac{\text{rc}(L_2(G), V)}{2} \right\rfloor.$$

This shows that the model we consider in this paper is a generalization of the edge model introduced in [25]. Instead of analyzing  $s(G)$  for some connected graph  $G$ , one can construct  $L_3(G)$  and analyze  $\text{rc}(L_3(G))$  (or construct  $L_2(G)$  and analyze  $\text{rc}(L_2(G), V)$ ).

We mention that Theorem 3 is sharp; that is, all three values in (2) can be achieved. For example, for any  $k \in \mathbb{N} \setminus \{1\}$ ,

- (a)  $s(P_3) = 2$  and  $\text{rc}(L_k(P_3)) = k \cdot s(P_3) + 1$ ;
- (b)  $s(K_3) = 3$  and  $\text{rc}(L_k(K_3)) = k \cdot s(K_3)$ ;
- (c)  $s(K_4) = 7$  and  $\text{rc}(L_k(K_4)) = k \cdot s(K_4) - 1$ .

Similarly, one can show the relationship between  $S(G)$  and  $\text{RC}(L_k(G))$ . Since the proof is similar to the previous one, we leave it for the reader.

**Theorem 5.** *Let  $G = (V, E)$  be any connected graph and let  $k \in \mathbb{N} \setminus \{1\}$ . Then we have that*

$$\text{RC}(L_k(G), V) \in \{k \cdot S(G), k \cdot S(G) + 1\}$$

and

$$k \cdot S(G) \leq \text{RC}(L_k(G)) \leq k \cdot (S(G) + 1).$$

Hence,

$$S(G) = \left\lfloor \frac{\text{RC}(L_2(G), V)}{2} \right\rfloor.$$

As before, the result is sharp. For example, for any  $k \in \mathbb{N} \setminus \{1\}$ ,

- (a)  $S(C_3) = 3$  and  $\text{RC}(L_k(C_3)) = \text{RC}(C_{3k}) = 3k = k \cdot S(C_3)$ ;
- (b)  $S(P_3) = 3$  and  $\text{RC}(L_k(P_3)) = \text{RC}(P_{2k+1}) = 2 \cdot (2k+1) - 2 = 4k = k \cdot (S(P_3) + 1)$ .

#### 4. PATHS, TREES, AND COMPLETE MULTI-PARTITE GRAPHS

In this section, we analyze the robot crawler parameters for some deterministic classes of graphs. For the path  $P_n$  of length  $n - 1 \geq 2$ , we have that  $\text{rc}(P_n) = n$  and  $\text{RC}(P_n) = 2n - 2$ . In order to achieve the minimum, one has to start the process from a leaf of  $P_n$ . Regardless of  $\omega_0$  used, the process takes  $n$  steps to finish (see Lemma 2(4) and Theorem 7 for more general results). In order to achieve the maximum, the robot

crawler has to start from a neighbour of a leaf and a weighting that forces the process to move away from the leaf (again, see Theorem 7 for more general result).

**Theorem 6.** *For any  $n \in \mathbb{N}$ ,*

$$\overline{\text{rc}}(P_n) = \frac{3n}{2} - \frac{3}{2} + \frac{1}{n} \sim \frac{3n}{2}.$$

*Proof.* As it is evident that  $\overline{\text{rc}}(P_1) = 1$ , we focus on  $n \in \mathbb{N} \setminus \{1\}$ . We label nodes as follows:  $V(P_n) = \{v_1, v_2, \dots, v_n\}$ . With probability  $2/n$ , the process starts at one of the two endpoints of  $P_n$  (node  $v_1$  or  $v_n$ ). If this happens, the robot crawler moves along the path and the process terminates after  $n$  steps, regardless of a weighting used. With probability  $1/n$ , the process starts at some node  $v_i$ ,  $i \in \{2, 3, \dots, n-1\}$ . Then, by symmetry, with probability  $1/2$  it moves to  $v_{i-1}$ , continues all the way to  $v_1$ , and then walks again along the path reaching  $v_n$  after  $(i-1) + n$  steps. Otherwise, it moves first to  $v_{i+1}$  walking along the path as before, reaching the last node (that is, node  $v_1$ ) after  $(n-i) + n$  steps. It follows that

$$\begin{aligned} \overline{\text{rc}}(P_n) &= \frac{2}{n} \cdot n + \frac{1}{n} \sum_{i=2}^{n-1} \left( \frac{1}{2}(n+i-1) + \frac{1}{2}(2n-i) \right) \\ &= 2 + \left(1 - \frac{2}{n}\right) \left(\frac{3n}{2} - \frac{1}{2}\right) = \frac{3n}{2} - \frac{3}{2} + \frac{1}{n}, \end{aligned}$$

and the proof is finished.  $\square$

We next give the precise value of  $\text{rc}$  and  $\text{RC}$  for trees. The main idea behind the proof of this result is comparing the robot crawler to the Depth-First Search algorithm on a tree.

**Theorem 7.** *Let  $T = (V, E)$  be a tree on  $n \geq 2$  nodes. Then we have that*

$$\text{rc}(T) = 2n - 1 - \text{diam}(T) \quad \text{and} \quad \text{RC}(T) = 2n - 2,$$

where  $\text{diam}(T)$  is the diameter of  $T$ .

To prove Theorem 7, we will show a connection between the robot crawler and the Depth-First Search (DFS) algorithm for traversing a tree (or a graph in general). In this algorithm, one starts at the root node (selected arbitrarily) and explores as far as possible along each branch before backtracking. Using this observation, it will be straightforward to investigate  $\text{rc}(T)$  and  $\text{RC}(T)$  for a tree  $T$ .

Let us start with the following, recursive, definition of DFS. For a given connected graph  $G = (V, E)$  on  $n$  nodes, a weighting  $\omega : V \rightarrow B_n$ , and an initial node  $v_0 \in V$ , the *Depth-First Search* algorithm is defined as follows:

- (1) procedure  $DFS(G, \omega, v)$ ;
- (2) label  $v$  as *discovered*;
- (3) let  $\ell = |N(v)|$  be the number of neighbours of  $v$ ;
- (4) consider all neighbours of  $v$ ,  $(w_1, w_2, \dots, w_\ell)$ , in order yielded by  $\omega$ , starting from the dirtiest one; that is,  $\omega(w_1) < \omega(w_2) < \dots < \omega(w_\ell)$ ;
- (5) for each  $i = 1, 2, \dots, \ell$ ,  
if node  $w_i$  is not labelled as discovered, then recursively call  $DFS(G, \omega, w_i)$ .



As we run the DFS algorithm, we keep a global pointer to the node that is currently being processed on each recursive call. We say that the algorithm visits node  $v$ , every time that the pointer points to  $v$ . Note that a node  $v$  is first visited when it is labelled as discovered at step 2, and then again after returning from each recursive call at step 5. Therefore,  $v$  may be visited as many times as its degree (or its degree plus one, if  $v$  is the starting node).

We state a useful lemma.

**Lemma 8.** *Let  $T = (V, E)$  be any tree on  $n$  nodes,  $\omega_0 : V \rightarrow B_n$  be any initial weighting of the nodes. Consider tree  $T$  rooted at the node  $v_0$  with the smallest weight. Let  $P = P(\omega_0, v_0)$  be the (directed) path from the root  $v_0$  to some leaf of  $T$ , obtained by moving greedily at each step to the neighbour of largest weight until a leaf is reached. Then we have that*

$$\text{rc}(T, \omega_0) = 2n - 1 - |E(P)|.$$

*Proof.* As already mentioned, it is easy to see that the robot crawler process  $\mathcal{RC}(T, \omega_0)$  behaves exactly as  $\text{DFS}(T, \omega_0, v_0)$  with the only difference that the robot crawler stops at the last leaf without backtracking to the root  $v_0$  through path  $P$ . Since each edge of the path  $P$  is visited once and every other edge is visited twice,

$$\text{rc}(T, \omega_0) = 1 + 2(n - 1 - |E(P)|) + |E(P)| = 2n - 1 - |E(P)|,$$

and the proof follows.  $\square$

From Lemma 8 we obtain the desired result.

*Proof of Theorem 7.* Let  $D = (v_1, v_2, \dots, v_d)$  be a path of length  $d = \text{diam}(T)$ . By assigning  $\omega_{\max}(v_1) = 0$  and  $\omega_{\max}(v_2) = -n + 1$  (the other values can be assigned arbitrarily), we find that  $P = P(\omega_{\max}) = (v_2, v_1)$ , and so

$$\text{RC}(T) \geq \text{rc}(T, \omega_{\max}) = 2n - 1 - |E(P)| = 2n - 2.$$

(Note that for this bound, we are not using the fact that path  $D$  has length  $\text{diam}(T)$ , but only that the process starts at a node  $v_2$  that is adjacent to a leaf  $v_1$  and terminates at  $v_1$ .) Clearly, this is the largest value that can be achieved as  $|E(P)| \geq 1$ , and so the result holds for  $\text{RC}(T)$ .

Similarly, one can assign  $\omega_{\min}(v_1) = -n + 1$  and  $\omega_{\min}(v_i) = 2 - i$  for  $i = 2, 3, \dots, d$  (again, the other values can be assigned arbitrarily) to derive  $P = P(\omega_{\min}) = (v_1, v_2, \dots, v_d)$ . This time, we have

$$\text{rc}(T) \leq \text{rc}(T, \omega_{\min}) = 2n - 1 - |E(P)| = 2n - 1 - d.$$

As the bound is tight, the result holds for  $\text{rc}(T)$ .  $\square$

We finish this section by considering complete multi-partite graphs. For  $k \in \mathbb{N} \setminus \{1\}$  and  $n \in \mathbb{N}$ , denote the *complete  $k$ -partite graph* with partite sets  $V_1, \dots, V_k$  of size  $n$  by  $K_n^k$ . Note that for any  $n \in \mathbb{N}$  and  $k = 2$ , we have that

$$\text{rc}(K_n^2) = \overline{\text{rc}}(K_n^2) = \text{RC}(K_n^2) = |V(K_n^2)| = 2n.$$

Indeed, since  $K_n^2$  has a hamiltonian path,  $\text{rc}(K_n^2) = 2n$  (see Lemma 2(4)). However, in fact, regardless of the  $\omega_0$  used, the robot crawler starts at a node  $v_0$  and then oscillates

between the two partite sets visiting all nodes in increasing order of weights assigned initially to each partite set of  $K_n^2$ .

We next consider the case  $k \geq 3$ . Since  $K_n^k$  still has a hamiltonian path,  $\text{rc}(K_n^k) = kn$ . For any  $k \in \mathbb{N} \setminus \{1, 2\}$  and  $n \in \mathbb{N}$ , it is straightforward to see that

$$\text{rc}(K_n^k) = kn \text{ and } \text{RC}(K_n^k) = (k+1)n - 1.$$

Investigating  $\overline{\text{rc}}(K_n^k)$  appears more challenging. However, we derive the asymptotic behaviour.

**Theorem 9.** *For any  $k \in \mathbb{N} \setminus \{1, 2\}$ , we have that*

$$\overline{\text{rc}}(K_n^k) = kn + O(\log n) \sim kn.$$

Before we sketch the proof of Theorem 9, we need a definition. Suppose that we are given an initial weighting  $\omega_0$  of  $K_n^k$ . For any  $\ell \in [kn]$ , let  $A_\ell$  be the set of  $\ell$  cleanest nodes; that is,

$$A_\ell = \{v \in V_1 \cup V_2 \cup \dots \cup V_k : \omega_0(v) \geq -\ell + 1\}.$$

Finally, for any  $\ell \in [kn]$  and  $j \in [k]$ , let  $a_\ell^j = a_\ell^j(\omega_0) = |A_\ell \cap V_j|$ ; that is,  $a_\ell^j$  is the number of nodes of  $V_j$  that are among  $\ell$  the cleanest ones (in the whole graph  $K_n^k$ ). Note that for a random initial weighing  $\omega_0$ , the expected value of  $a_\ell^j$  is  $\ell/k$ . Let  $\varepsilon > 0$ . We say that  $\omega_0$  is  $\varepsilon$ -balanced if for each  $j \in [k]$  and  $6\varepsilon^{-2}k \log n \leq \ell \leq kn$ , we have that

$$\left| a_\ell^j - \frac{\ell}{k} \right| < \frac{\varepsilon \ell}{k}.$$

A crucial observation is that almost all initial weightings are  $\varepsilon$ -balanced, regardless of how small  $\varepsilon$  is. We will use the following version of *Chernoff's bound*. Suppose that  $X \in \text{Bin}(n, p)$  is a binomial random variable with expectation  $\mu = np$ . If  $0 < \delta < 3/2$ , then

$$\Pr(|X - \mu| \geq \delta\mu) \leq 2 \exp\left(-\frac{\delta^2\mu}{3}\right). \quad (3)$$

(For example, see Corollary 2.3 in [19].) It is also true that (3) holds for a random variable with the hypergeometric distribution. The *hypergeometric distribution* with parameters  $N$ ,  $n$ , and  $m$  (assuming  $\max\{n, m\} \leq N$ ) is defined as follows. Let  $\Gamma$  be a set of size  $n$  taken uniformly at random from set  $[N]$ . The random variable  $X$  counts the number of elements of  $\Gamma$  that belong to  $[m]$ ; that is,  $X = |\Gamma \cap [m]|$ . It follows that (3) holds for the hypergeometric distribution with parameters  $N$ ,  $n$ , and  $m$ , with expectation  $\mu = nm/N$ . (See, for example, Theorem 2.10 in [19].)

Now we are ready to state the important lemma which is used in the proof of Theorem 9.

**Lemma 10.** *Let  $\varepsilon > 0$  and  $k \in \mathbb{N} \setminus \{1, 2\}$ , and let  $\omega_0$  be a random initial weighting of  $K_n^k$ . Then we have that  $\omega_0$  is  $\varepsilon$ -balanced with probability  $1 - O(n^{-1})$ .*

*Proof.* Theorem 9 Let  $k \in \mathbb{N} \setminus \{1, 2\}$  and fix  $\varepsilon = 0.01$ . We will show that for any  $\varepsilon$ -balanced initial weighting  $\omega_0$ ,  $\text{rc}(K_n^k, \omega_0) = kn + O(\log n)$ . This will finish the proof since, by Lemma 10, a random initial weighting is  $\varepsilon$  balanced with probability  $1 -$

$O(n^{-1})$ , and for any initial weighting  $\omega_0$  we have  $\text{rc}(K_n^k, \omega_0) \leq \text{RC}(K_n^k) = (k+1)n - 1 = O(n)$ . Indeed,

$$\begin{aligned} \overline{\text{rc}}(K_n^k) &= \Pr(\omega_0 \text{ is } \varepsilon\text{-balanced}) (kn + O(\log n)) + \Pr(\omega_0 \text{ is not } \varepsilon\text{-balanced}) O(n) \\ &= (kn + O(\log n)) + O(1) = kn + O(\log n). \end{aligned}$$

Let  $\omega_0$  be any  $\varepsilon$ -balanced initial weighting. Fix  $\ell \in [kn]$  and let us run the process until the robot crawler is about to move for the first time to a node of  $A_\ell$ . Suppose that the robot crawler occupies node  $v \in V_i$  for some  $i \in [k]$  ( $v \notin A_\ell$ ) and is about to move to node  $u \in V_j$  for some  $j \in [k], j \neq i$  ( $u \in A_\ell$ ). Let us call  $V_i$  a  $\ell$ -crucial partite set. Concentrating on non-crucial sets, we observe that for any  $s \neq i$ , all the nodes of  $V_s \setminus A_\ell$  are already cleaned; otherwise, the robot crawler would go to such node, instead of going to  $u$ . On the other hand, it might be the case that not all nodes of  $V_i \setminus A_\ell$ , that belong to a  $\ell$ -crucial set, are already visited; we will call such nodes  $\ell$ -dangerous. Let  $f(\ell)$  be the number of  $\ell$ -dangerous nodes.

Our goal is to control the function  $f(\ell)$ . We say that  $\ell$  is *good* if  $f(\ell) \leq 0.6\ell/k$ . Clearly,  $\ell = kn$  is good, as  $f(kn) = 0$ . We use the following claim.

*Claim.* If  $\ell$  is good, then  $\ell' = \lfloor 2\ell/3 \rfloor$  is good, provided that  $\lfloor 2\ell/3 \rfloor \geq 6\varepsilon^{-2}k \log n$ .

*Proof.* We run the process and stop at time  $T_\ell$  when the robot crawler is about to move to the first node of  $A_\ell$ . We concentrate on the time interval from  $T_\ell$  up to time-step  $T_{\ell'}$  when a node of  $A_{\ell'}$  is about to be cleaned. First, note that during the first phase of this time interval, the crawler oscillates between nodes of  $A_\ell \setminus A_{\ell'}$  that are not in the  $\ell$ -crucial set and  $\ell$ -dangerous nodes. Clearly, there are  $\ell - \ell' \geq \ell/3$  nodes in  $A_\ell \setminus A_{\ell'}$ . Since  $\omega_0$  is  $\varepsilon$ -balanced, the number of nodes of the  $\ell$ -crucial set that belong to  $A_\ell$  and  $A_{\ell'}$  is at most  $(1 + \varepsilon)\ell/k$  and at least  $(1 - \varepsilon)\ell'/k$ , respectively. Since

$$\frac{\ell}{3} - \left( \frac{(1 + \varepsilon)\ell}{k} - \frac{(1 - \varepsilon)\ell'}{k} \right) = \frac{\ell}{3} - \frac{(1 + 5\varepsilon)\ell}{3k} + O(1) \geq \left( \frac{k-1}{3} - 2\varepsilon \right) \frac{\ell}{k} > 0.64 \frac{\ell}{k} \geq f(\ell),$$

this phase lasts  $2f(\ell)$  steps and all  $\ell$ -dangerous nodes are cleaned. The claim now follows easily as one can use a trivial bound for the number of  $\ell'$ -dangerous nodes. Regardless which partite set is  $\ell'$ -crucial, since  $\omega_0$  is  $\varepsilon$ -balanced, we can estimate the number of nodes in  $\ell'$ -crucial set that belong to  $A_\ell \setminus A_{\ell'}$ . Since  $\ell'$ -dangerous nodes must be in  $A_\ell \setminus A_{\ell'}$ , we obtain that

$$f(\ell') \leq \frac{(1 + \varepsilon)\ell}{k} - \frac{(1 - \varepsilon)\ell'}{k} = \left( \frac{1}{2} + \frac{5}{2}\varepsilon \right) \frac{\ell'}{k} + O(1) < 0.53 \frac{\ell'}{k}.$$

It follows that  $\ell'$  is good and the claim holds by induction.  $\square$

To finish the proof, we keep applying the claim recursively concluding that there exists  $\ell < (3/2)6\varepsilon^{-2}k \log n = O(\log n)$  that is good. At time  $T_\ell$  of the process,  $\ell + f(\ell) \leq \ell + 0.6\ell/k = O(\log n)$  nodes are still dirty and every other node is visited exactly once. The process ends after at most  $2(\ell + f(\ell))$  another steps for the total of at most  $kn + (\ell + f(\ell)) = kn + O(\log n)$  steps.  $\square$

## 5. BINOMIAL RANDOM GRAPHS

The *binomial random graph*  $\mathcal{G}(n, p)$  is defined as a random graph with node set  $[n]$  in which a pair of nodes appears as an edge with probability  $p$ , independently for each pair of nodes. As typical in random graph theory, we consider only asymptotic properties of  $\mathcal{G}(n, p)$  as  $n \rightarrow \infty$ , where  $p = p(n)$  may and usually does depend on  $n$ .

It is known (see, for example, [21]) that a.a.s.  $\mathcal{G}(n, p)$  has a hamiltonian cycle (and so also a hamiltonian path) provided that  $pn \geq \log n + \log \log n + \omega$ , where  $\omega = \omega(n)$  is any function tending to infinity together with  $n$ . On the other hand, a.a.s.  $\mathcal{G}(n, p)$  has no hamiltonian cycle if  $pn \leq \log n + \log \log n - \omega$ . It is straightforward to show that in this case a.a.s. there are more than two nodes of degree at most 1 and so a.a.s. there is no hamiltonian path. Combining these observations, we immediately derive the following result.

**Corollary 11.** *If  $\omega = \omega(n)$  is any function tending to infinity together with  $n$ , then the following hold a.a.s.*

- (1) *If  $pn \geq \log n + \log \log n + \omega$ , then  $rc(\mathcal{G}(n, p)) = n$ .*
- (2) *If  $pn \leq \log n + \log \log n - \omega$ , then  $rc(\mathcal{G}(n, p)) > n$ .*

The next upper bound on  $RC(\mathcal{G}(n, p))$  follows from Lemma 2(5) and the fact that  $\mathcal{G}(n, p)$  has maximum degree at most  $n - 1$  and a.a.s. diameter 2 for  $p$  in the range of discussion.

**Corollary 12.** *Suppose  $pn \geq C\sqrt{n \log n}$ , for a sufficiently large constant  $C > 0$ . Then a.a.s. we have that*

$$RC(\mathcal{G}(n, p)) \leq n^3.$$

*Proof.* Given two different nodes  $u, v$ , the number of common neighbours is distributed as  $\text{Bin}(n - 2, p^2)$ , which by Chernoff bound (3) is at least 1 with probability  $1 - o(n^{-2})$  (by taking  $C$  large enough). Therefore, a.a.s. every pair of nodes of  $\mathcal{G}(n, p)$  has some common neighbour and hence, the diameter is at most 2. Moreover, trivially (and deterministically) the maximum degree is  $\Delta \leq n - 1$ . Therefore, by Lemma 2(5), a.a.s.  $RC(\mathcal{G}(n, p)) \leq n^3$ .  $\square$

Moreover, we give the following lower bound.

**Theorem 13.** *Suppose  $C\sqrt{n \log n} \leq pn \leq (1 - \varepsilon)n$ , for constants  $C > 1$  and  $\varepsilon > 0$ . Then a.a.s. we have that*

$$RC(\mathcal{G}(n, p)) \geq (2 - p + o(p))n.$$

*Proof.* Fix a node  $v$  of  $G \in \mathcal{G}(n, p)$ , and expose its neighbourhood  $K = N(v)$ . Let  $k = |K|$ ,  $M = V \setminus K$ , and  $m = |M|$ . We will show that the following properties hold a.a.s.

- (a)  $k \sim pn$  and  $m = \Theta(n)$ .
- (b) The subgraphs of  $G$  induced by  $K$  and by  $M$  (which we denote  $G[K]$  and  $G[M]$ , respectively) are both hamiltonian.
- (c) There is a pair  $C_K = (u_1, u_2, \dots, u_k)$  and  $C_M = (v_1, v_2, \dots, v_m)$  of Hamilton cycles of  $G[K]$  and  $G[M]$ , respectively, such that  $u_1 v_{m-1}$  and  $u_k v_m$  are edges in  $G$  (that is, edges  $v_{m-1}, v_m$  and  $u_k, u_1$  are contained in a 4-cycle).

If all of these properties hold, then we assign the initial weightings to nodes in the following order from dirtiest to cleanest:  $v_1, v_2, \dots, v_{m-1}, u_1, u_2, \dots, u_k, v_m, v$ . The robot crawler is then forced to first clean nodes  $v_1, v_2, \dots, v_{m-1}$ , next  $u_1, u_2, \dots, u_k$ , then  $v_m, v_1, v_2, \dots, v_{m-1}$  and finally  $u_1$  and  $v$ . Note that the only way of cleaning  $v$  is coming from a node in  $K$  at the previous time-step. This takes

$$(m-1) + k + m + 2 = 2n - k - 1 = (2 - p + o(p))n$$

steps, as required. (See Figure 1.)

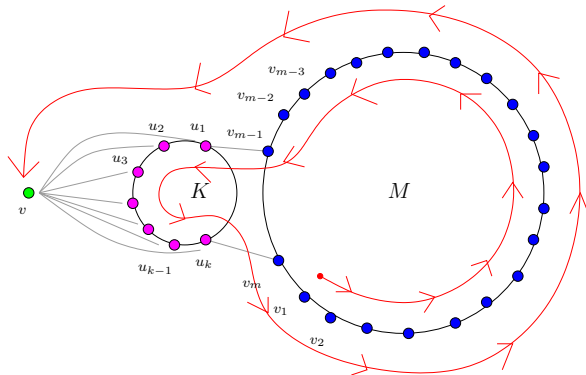


FIGURE 1. Example of a crawling sequence on a “typical” instance of  $\mathcal{G}(n, p)$  that takes  $2n - k - 1$  steps to clean all nodes.

Now it suffices to show that (a–c) hold a.a.s. The first statement in (a) follows trivially from Chernoff bound (3), without exposing any edges other than those incident to  $v$ . The second one uses the fact that  $p \leq 1 - \varepsilon$ . For (b), note that  $pk \sim p^2n \geq C \log n \geq C \log k$ , so the average degree in  $G[K]$  is at least  $C \log k$  and it is well known that  $G[K]$  is a.a.s. hamiltonian, provided that  $C > 1$  (recall the discussion before Corollary 11). For an analogous reason (with even more room to spare),  $G[H]$  is also a.a.s. hamiltonian. Finally, for (c), fix any Hamilton cycles  $C_K$  and  $C_M$  and any edge  $e = (x, y)$  in  $C_K$ . Let  $\{e_i = (v_{2i-1}, v_{2i}) : 1 \leq i \leq \lfloor m/2 \rfloor\}$  be a set of independent edges (matching) of  $C_M$ . An important observation is that the previous arguments did not expose edges between  $K$  and  $M$ . For each  $i$ , let  $A_i$  be the event that  $x$  is adjacent to  $x_i$  and  $y$  is adjacent to  $y_i$ . Events  $A_1, A_2, \dots, A_{\lfloor m/2 \rfloor}$  are independent and each one has probability of holding equal to  $p^2$ . The number of successful events is distributed as  $\text{Bin}(\lfloor m/2 \rfloor, p^2)$  and has expectation  $\Omega(np^2) = \Omega(\log n)$ . By Chernoff bound (3), we conclude a.a.s. that  $A_i$  holds for some  $i$ , and therefore property (c) holds after conveniently relabelling the nodes of  $C_K$  and  $C_M$ . (Note that there is room to spare as one needs the expected value to tend to infinity at any rate.)  $\square$

The rest of this section is devoted to the following result.

**Theorem 14.** *Let  $p = p(n)$  such that  $pn \gg \sqrt{n \log n}$ . Then a.a.s.*

$$\overline{\text{rc}}(\mathcal{G}(n, p)) = n + o(n).$$

The main ingredient to derive Theorem 14 is the following key lemma.

**Lemma 15.** *Let  $G = (V, E) \in \mathcal{G}(n, p)$  for some  $p = p(n)$  such that  $pn \gg \sqrt{n \log n}$ , and let  $\omega_0 : V \rightarrow B_n$  be any fixed initial weighting. Then with probability  $1 - o(n^{-3})$ , we have that*

$$\text{rc}(G, \omega_0) = n + o(n).$$

We are going to fix an initial weighting before exposing edges of the random graph. For a given initial weighting  $\omega_0 : V \rightarrow B_n$ , we partition the node set  $V$  into 3 types with respect to their initial level of dirtiness: *type 1* consists of nodes with initial weights from  $B_n \setminus B_{\lfloor 2n/3 \rfloor}$ , *type 2* with initial weights from  $B_{\lfloor 2n/3 \rfloor} \setminus B_{\lfloor n/3 \rfloor}$ ; the remaining nodes are of *type 3*. Before we move to the proof of Lemma 15, we state the following useful claim that holds even for much sparser graphs.

*Claim.* Let  $G = (V, E) \in \mathcal{G}(n, p)$  for some  $p = p(n)$  such that  $pn \gg \log n$ . Let  $\omega_0 : V \rightarrow B_n$  be any initial weighting. Then the following property holds with probability  $1 - o(n^{-3})$ . Each node  $v \in V$  has  $(1 + o(1))pn/3$  neighbours of each of the three types.

*Proof.* For the proof of the claim, let  $\omega = \omega(n) = pn/\log n$ ; note that  $\omega \rightarrow \infty$  as  $n \rightarrow \infty$ . Fix a node  $v \in V$  and one of the three types. The number of neighbours of  $v$  that are of the selected type is a random variable  $X$  with the binomial distribution  $\text{Bin}(n/3 + O(1), p)$  with  $\mathbb{E}[X] = pn/3 + O(1)$ . (The term  $O(1)$  comes from the fact that  $n$  does not need to be divisible by 3, and also the fact that  $v$  might also be of this type.) It follows from Chernoff bound (3) that

$$\begin{aligned} \Pr(|X - \mathbb{E}[X]| \geq (\omega^{-1/3})\mathbb{E}[X]) &\leq 2 \exp\left(-\frac{(\omega^{-1/3})^2 \mathbb{E}[X]}{3}\right) \\ &= 2 \exp\left(-\frac{(\omega^{1/3}) \log n}{9 + o(1)}\right) \\ &= o(n^{-4}). \end{aligned}$$

Hence, with probability  $1 - o(n^{-4})$ ,  $X \sim pn/3$  and the result holds by the union bound (as there are  $3n$  possibilities for selecting  $v$  and the type).  $\square$

We will use the claim in the proof of the main result but not explicitly; that is, we do not want to condition on the property stated in the claim. Instead, we uncover edges of the (unconditional) random graph (one by one, in some order) and show that the desired upper bound for  $\text{rc}(\mathcal{G}(n, p), \omega_0)$  holds with the desired probability *unless* the claim is false. Now we can move to the proof of Lemma 15.

*Proof.* Lemma 15 We consider four phases of the crawling process.

*Phase 1:* We start the process from the initial node (which is of type 1, since it has initial weight  $-n + 1$ ), and then we clean only nodes of type 1. The phase ends when the robot crawler is not adjacent to any dirty node of type 1; that is, when the crawler is about to move to a node of some other type than type 1 or to re-clean some node of type 1. An important property is that, at any point of the process, potential edges between the crawler and dirty nodes are not exposed yet. Hence, if  $x \geq 5 \log n/p$  nodes of type 1 are still dirty, the probability that this phase ends at this point is equal to

$$(1 - p)^x \leq \exp(-px) \leq n^{-5}.$$

Hence, it follows from the union bound that, with probability at least  $1 - n^{-4} = 1 - o(n^{-3})$ , this phase ends after  $T_1$  steps, where  $\lceil n/3 \rceil - 5 \log n/p \leq T_1 \leq \lceil n/3 \rceil$ , at most  $5 \log n/p$  nodes of type 1 are still dirty, and the other type 1 nodes are cleaned exactly once. Observe that during this phase we exposed only edges between type 1 nodes.

*Phase 2:* During this phase we are going to clean mostly nodes of type 2, with a few “detours” to type 1 nodes that are still dirty. Formally, the phase ends when the robot crawler is not adjacent to any dirty node of type 1 or 2; that is, when the crawler is about to move to a node of type 3 or to re-clean some node (of type 1 or 2). Arguing as before, we deduce that, with probability at least  $1 - o(n^{-3})$ , this phase ends after the total of  $T_2$  steps (counted from the beginning of the process), where  $\lceil 2n/3 \rceil - 5 \log n/p \leq T_2 \leq \lceil 2n/3 \rceil$ , at most  $5 \log n/p$  nodes of type 1 or 2 are still dirty, and the other type 1 or 2 nodes are cleaned exactly once.

Suppose that at the end of this phase some node  $v$  of type 1 is still dirty. This implies that  $v$  has at most  $10 \log n/p$  neighbours that are type 2. Indeed, at most  $5 \log n/p$  of them are perhaps not visited by the crawler yet; at most  $5 \log n/p$  of them were visited by the crawler but it did not move to  $v$  from them but went to some other of the at most  $5 \log n/p$  dirty nodes of type 1 instead. Since  $pn \geq 10\sqrt{n \log n}$ , we obtain that  $10 \log n/p \leq pn/10$  and so this implies that the property stated in the Claim is not satisfied. If this is the case, then we simply stop the argument. We may then assume that all nodes of type 1 are cleaned at this point of the process. Finally, let us mention that during this phase we exposed only edges between type 2 nodes, and between type 1 nodes that were dirty at the end of phase 1 and type 2 nodes.

*Phase 3:* This phase ends when the robot crawler is not adjacent to any dirty node; that is, when the crawler is about to re-clean some node. During this phase we are going to clean mostly nodes of type 3, with a few “detours” to type 2 nodes that are still dirty. Arguing as before, we deduce that, with probability at least  $1 - o(n^{-3})$ , this phase ends after the total of  $T_3$  steps, where  $n - 5 \log n/p \leq T_3 \leq n$ . Moreover, we may assume that at the end of this phase at most  $5 \log n/p$  nodes of type 3 are still dirty whereas all other nodes are cleaned exactly once; otherwise, the property stated in the Claim is not satisfied. As usual, the main observation is that during this phase we exposed only edges between type 3 nodes, and between type 2 nodes that were dirty at the end of phase 2 and type 3 nodes.

*Phase 4:* During this final phase we are going to re-clean (for the second time) some nodes of type 1, with a few “detours” to type 3 nodes that are still dirty. This phase ends when one of the following properties is satisfied:

- (a) all nodes are cleaned,
- (b) this phase takes more than  $20 \log n/p^2$  steps,
- (c) the robot crawler is not adjacent to any dirty node nor to any type 1 node that was cleaned only once, during phase 1 (note that these nodes have the smallest weights at this point of the process).

Recall that our goal is to show that either the property stated in the Claim is not satisfied or, with probability at least  $1 - o(n^{-3})$ , the phase ends when all nodes are

cleaned. From this it will follow that the process takes  $n + O(\log n/p^2) = n + o(n)$  steps with probability at least  $1 - o(n^{-3})$ , and the proof will be finished.

Suppose first that the phase ends because of property (c). It follows that the crawler occupies a node  $v$  that has at most  $25 \log n/p$  neighbours that are type 1: at most  $20 \log n/p$  of them were re-cleaned during this phase, and at most  $5 \log n/p$  of them were cleaned during phase 2. Since  $pn \geq 10\sqrt{n \log n}$ ,  $25 \log n/p \leq pn/4$  and so the property in the Claim is not satisfied. Hence, we may assume that the phase does not end because of (c).

Suppose now that the phase ends because of property (b) and that property (c) is never satisfied. This implies that all nodes visited during phase 4 must be different, since otherwise property (c) would hold. Moreover, the robot crawler can be adjacent to a dirty node at most  $5 \log n/p$  out of the first  $\lfloor 20 \log n/p^2 \rfloor$  steps in this phase, since each time this happens one dirty node will be cleaned in the next step, and there were at most  $5 \log n/p$  nodes of type 3 that were dirty at the end of phase 3. A crucial observation is that no edges between type 1 and type 3 nodes (and also no edges between dirty nodes of type 3) were exposed at the beginning of this phase. Using this we can estimate the probability that at the end of this phase some node is still dirty. Indeed, at each step, the probability that the robot crawler is adjacent to a dirty node (provided that some dirty node still exists) is at least  $p$ . Hence, using Chernoff bound (3), the probability that phase 4 ends because of property (b) and not (c) is at most

$$\Pr(\text{Bin}(\lfloor 20 \log n/p^2 \rfloor, p) \leq 5 \log n/p) \leq \exp\left(-\frac{(3/4)^2 20 \log n/p}{3 + o(1)}\right) = o(n^{-3}).$$

This shows that phase 4 does not stop because of property (b) with probability  $1 - o(n^{-3})$ , as required.  $\square$

We finish the section by proving Theorem 14.

*Proof.* Theorem 14 We say that a pair  $(G, \omega_0)$  is *good* if  $\text{rc}(G, \omega_0) \leq n + f(n)$ , where  $f(n) = o(n)$  sufficiently slowly. Otherwise, we call the pair *bad*. Lemma 15 implies that a random pair  $(G, \omega_0)$  is bad with probability at most  $qn^{-3}$  for some  $q = q(n) = o(1)$ . We say that a graph  $G$  is *dangerous*, if the fraction of initial weightings  $\omega_0$  such that  $(G, \omega_0)$  is bad over the total number  $n!$  of weightings is greater than  $\sqrt{q}n^{-3}$ . We conclude that

$$qn^{-3} \geq \Pr((G, \omega_0) \text{ is bad}) \geq \Pr(G \text{ is dangerous}) \sqrt{q}n^{-3},$$

so the probability that a graph in  $\mathcal{G}(n, p)$  is dangerous is at most  $\sqrt{q} = o(1)$ . Let us focus on graphs that are not dangerous. By Corollary 12 and the definition of dangerous, if  $G$  is not dangerous, then the contribution to  $\overline{\text{rc}}(G)$  from bad pairs  $(G, \omega_0)$  is at most  $\sqrt{q}n^{-3}n^3 = o(1)$ . The result follows since the contribution from good pairs is  $n + o(n)$ .  $\square$

Theorem 14 holds for dense random graphs; that is, for  $pn \gg \sqrt{n \log n}$ . The case of sparser random graph remains open. In addition, it is not clear whether the bound in Corollary 12 can be improved. For instance, is it true that  $\text{RC}(\mathcal{G}(n, p)) = O(n)$  for a wide range of  $p$ ? Recall, in view of Theorem 13, that we cannot achieve  $\text{RC}(\mathcal{G}(n, p)) = (1 + o(1))n$ , provided that  $p < 1 - \varepsilon$  for some  $\varepsilon > 0$ .



## 6. PREFERENTIAL ATTACHMENT MODEL

The results in Section 3 demonstrate that for the binomial random graph, for most initial weightings the robot crawler will finish in approximately  $n$  steps. We now consider the robot crawler on a stochastic model for complex networks. The *preferential attachment model*, introduced by Barabási and Albert [4], was an early stochastic model of complex networks. We will use the following precise definition of the model, as considered by Bollobás and Riordan in [5] as well as Bollobás, Riordan, Spencer, and Tusnády [6].

Let  $G_1^0$  be the null graph with no nodes (or let  $G_1^1$  be the graph with one node,  $v_1$ , and one loop). The random graph process  $(G_1^t)_{t \geq 0}$  is defined inductively as follows. Given  $G_1^{t-1}$ , we form  $G_1^t$  by adding node  $v_t$  together with a single edge between  $v_t$  and  $v_i$ , where  $i$  is selected randomly with the following probability distribution:

$$\Pr(i = s) = \begin{cases} \deg(v_s, t-1)/(2t-1) & 1 \leq s \leq t-1, \\ 1/(2t-1) & s = t, \end{cases}$$

where  $\deg(v_s, t-1)$  denotes the degree of  $v_s$  in  $G_1^{t-1}$ . (In other words, we send an edge  $e$  from  $v_t$  to a random node  $v_i$ , where the probability that a node is chosen as  $v_i$  is proportional to its degree at the time, counting  $e$  as already contributing one to the degree of  $v_t$ .)

For  $m \in \mathbb{N} \setminus \{1\}$ , the process  $(G_m^t)_{t \geq 0}$  is defined similarly with the only difference that  $m$  edges are added to  $G_m^{t-1}$  to form  $G_m^t$  (one at a time), counting previous edges as already contributing to the degree distribution. Equivalently, one can define the process  $(G_m^t)_{t \geq 0}$  by considering the process  $(G_1^t)_{t \geq 0}$  on a sequence  $v'_1, v'_2, \dots$  of nodes; the graph  $G_m^t$  if formed from  $G_1^{tm}$  by identifying nodes  $v'_1, v'_2, \dots, v'_m$  to form  $v_1$ , identifying nodes  $v'_{m+1}, v'_{m+2}, \dots, v'_{2m}$  to form  $v_2$ , and so on. Note that in this model  $G_m^t$  is in general a multigraph, possibly with multiple edges between two nodes (if  $m \geq 2$ ) and self-loops. For the purpose of the robot crawler, loops can be ignored and multiple edges between two nodes can be treated as a single edge.

It was shown in [6] that for any  $m \in \mathbb{N}$  a.a.s. the degree distribution of  $G_m^n$  follows a power law: the number of nodes with degree at least  $k$  falls off as  $(1 + o(1))ck^{-2}n$  for some explicit constant  $c = c(m)$  and large  $k \leq n^{1/15}$ . Let us start with the case  $m = 1$ , which is easy to deal with, since  $G_1^n$  is a forest. Each node sends an edge either to itself or to an earlier node, so the graph consists of components which are trees, each with a loop attached. The expected number of components is then  $\sum_{t=1}^n 1/(2t-1) \sim (1/2) \log n$  and, since events are independent, we derive that a.a.s. there are  $(1/2 + o(1)) \log n$  components in  $G_1^n$  by Chernoff's bound (3). Moreover, Pittel [28] essentially showed that a.a.s. the largest distance between two nodes in the same component of  $G_1^n$  is  $(\gamma^{-1} + o(1)) \log n$ , where  $\gamma$  is the solution of  $\gamma e^{1+\gamma} = 1$  (see Theorem 13 in [5]). Hence, the following result holds immediately from Theorem 7.

**Theorem 16.** *The following properties hold a.a.s. for any connected component  $G$  of  $G_1^m$ :*

$$\begin{aligned} \text{rc}(G) &= 2|V(G)| - 1 - \text{diam}(G) = 2|V(G)| - O(\log n), \\ \text{RC}(G) &= 2|V(G)| - 2. \end{aligned}$$

We may modify slightly the definition of the model to ensure  $G_1^n$  is a tree on  $n$  nodes, by starting from  $G_1^2$  being an isolated edge and not allowing loops to be created in the process (this is in fact the original model in [4]). For such variant, we would have that a.a.s.  $\text{rc}(G_1^n) \sim \text{RC}(G_1^n) \sim 2n$ , as the diameter would be negligible comparing to the order of the graph.

The case  $m \geq 2$  is more difficult to investigate. It is known that a.a.s.  $G_m^n$  is connected and its diameter is  $(1+o(1)) \log n / \log \log n$ , as shown in [5], and in contrast to the result for  $m = 1$  presented above. We managed to show that for the case  $m = 2$ , the robot crawler needs substantially more than  $n$  steps to clean the graph in this model. This immediately implies (in a strong sense) that  $G_2^n$  is not hamiltonian a.a.s. (See [16] for more results in this direction.)

**Theorem 17.** *A.a.s.  $\text{rc}(G_2^n) \geq (1 + \xi + o(1))n$ , where*

$$\xi = \max_{c \in (0, 1/2)} \left( \frac{2\sqrt{c}}{3} - c - \frac{c^2}{6} \right) \approx 0.10919.$$

*Proof.* Many observations in the argument will be valid for any  $m$  but, of course, we will eventually fix  $m = 2$ . Consider the process  $(G_m^t)_{t \geq 0}$  on the sequence of nodes  $(v_t)_{t \geq 0}$ . We will call node  $v_i$  *lonely* if  $\deg(v_i, n) = m$ ; that is, no loop is created at the time  $v_i$  is introduced and no other node is connected to  $v_i$  later in the process. Moreover,  $v_i$  is called *old* if  $i \leq cn$  for some constant  $c \in (0, 1)$  that will be optimized at the end of the argument; otherwise,  $v_i$  is called *young*. Finally,  $v_i$  is called  *$j$ -good* if  $v_i$  is lonely and exactly  $j$  of its neighbours are old.

Let us begin with the big picture for the case  $m = 2$ . Suppose that  $an$  nodes are young and 1-good,  $bn$  nodes are young and 2-good, and  $dn$  nodes are old and lonely (which implies that they are 2-good). Clearly, the robot crawler needs to visit all young nodes and all old and lonely ones, which takes at least  $(1 - c)n + dn$  steps. Observe that each time a young and 2-good node is visited, the crawler must come from an old but not-lonely node and move to another such one right after. Similarly, each time the crawler visits a young and 1-good node, it must come from or move to some node that is old but not lonely. It follows that nodes that are old but not lonely must be visited at least  $an/2 + bn + O(1)$  times. Hence, the process must take at least  $(1 - c + d + a/2 + b + o(1))n$  steps, and our hope is that it gives a non-trivial bound for some value of  $c \in (0, 1)$ .

The probability that  $v_i$  is lonely is easy to estimate from the equivalent definition of  $G_m^n$  obtained in terms of  $G_1^{mn}$ . For  $i \gg 1$ , we derive that

$$\begin{aligned} \Pr(v_i \text{ is lonely}) &= \Pr(\deg(v_i, i) = m) \prod_{t=im+1}^{nm} \left( 1 - \frac{m}{2t-1} \right) \\ &\sim \exp \left( - \sum_{t=im+1}^{nm} \frac{m}{2t-1} + O \left( \sum_{t=im+1}^{nm} t^{-2} \right) \right) \\ &\sim \exp \left( - \frac{m}{2} \sum_{t=im+1}^{nm} t^{-1} \right) \sim \exp \left( - \frac{m}{2} \log \left( \frac{nm}{im} \right) \right) = \left( \frac{i}{n} \right)^{m/2}. \end{aligned}$$

We will also need to understand the behaviour of the following random variable: for  $\lfloor cn \rfloor \leq t \leq n$ , let

$$Y_t = \sum_{j \leq cn} \deg(v_j, t).$$

In view of the identification between the models  $G_m^n$  and  $G_1^{mn}$ , it will be useful to investigate the following random variable instead: for  $m \lfloor cn \rfloor \leq t \leq mn$ , let

$$X_t = \sum_{j \leq cmn} \deg_{G_1^t}(v'_j, t).$$

Clearly,  $Y_t = X_{tm}$ . It follows that  $X_{m \lfloor cn \rfloor} = Y_{\lfloor cn \rfloor} = 2m \lfloor cn \rfloor$ . Moreover, for  $m \lfloor cn \rfloor < t \leq mn$ ,

$$X_t = \begin{cases} X_{t-1} + 1 & \text{with probability } \frac{X_{t-1}}{2t-1}, \\ X_{t-1} & \text{otherwise.} \end{cases}$$

The conditional expectation is given by

$$\mathbb{E}[X_t | X_{t-1}] = (X_{t-1} + 1) \cdot \frac{X_{t-1}}{2t-1} + X_{t-1} \left(1 - \frac{X_{t-1}}{2t-1}\right) = X_{t-1} \left(1 + \frac{1}{2t-1}\right).$$

Taking expectation again, we derive that

$$\mathbb{E}[X_t] = \mathbb{E}[X_{t-1}] \left(1 + \frac{1}{2t-1}\right).$$

Hence, arguing as before, it follows that

$$\mathbb{E}[Y_t] = \mathbb{E}[X_{tm}] = 2m \lfloor cn \rfloor \prod_{s=m \lfloor cn \rfloor + 1}^{tm} \left(1 + \frac{1}{2s-1}\right) \sim 2cmn \left(\frac{tm}{cmn}\right)^{1/2} = 2mn \sqrt{ct/n}.$$

Noting that  $\mathbb{E}[Y_t] = \Theta(n)$  for any  $\lfloor cn \rfloor \leq t \leq n$ , and that  $Y_t$  increases by at most  $m$  each time ( $X_t$  increases by at most one), we obtain that with probability  $1 - o(n^{-1})$ ,  $Y_t = \mathbb{E}[Y_t] + O(\sqrt{n \log n}) \sim \mathbb{E}[Y_t]$  (using a standard martingale argument; for Azuma-Hoeffding inequality that is used here see, for example, [19]). Hence, we may assume that  $Y_t \sim 2mn \sqrt{ct/n}$  for any  $\lfloor cn \rfloor \leq t \leq n$ .

The rest of the proof is straightforward. Note that, for a given  $t = xn$  with  $c \leq x \leq 1$ , the probability that an edge generated at this point of the process goes to an old node is asymptotic to  $(2mn \sqrt{ct/n}) / (2mt) = \sqrt{cn}/t = \sqrt{c/x}$ . Moreover, recall that  $v_t$  is lonely with probability asymptotic to  $(t/n)^{m/2} = x$  for the case  $m = 2$ . It follows that

$$\begin{aligned} a &\sim \int_c^1 2\sqrt{c/x}(1 - \sqrt{c/x})x dx = \frac{4\sqrt{c}}{3} - 2c + \frac{2c^2}{3}, \\ b &\sim \int_c^1 (\sqrt{c/x})^2 x dx = c - c^2, \\ d &\sim \int_0^c x dx = \frac{c^2}{2}. \end{aligned}$$

Since

$$1 - c + d + a/2 + b \sim 1 + \frac{2\sqrt{c}}{3} - c - \frac{c^2}{6}$$

is maximized at  $c = \frac{\left((4+4\sqrt{5})^{2/3}-4\right)^2}{4(4+4\sqrt{5})^{2/3}} \approx 0.10380$ , the proof follows.  $\square$

Properties of the robot crawler remain open in the preferential attachment model when  $m > 2$ . Fix  $m \geq 3$ . Is it true that a.a.s.  $\text{rc}(G_m^n) \geq (1 + \xi)n$  for some constant  $\xi > 0$ ? Or maybe  $\text{rc}(G_m^n) \sim n$ ? It is possible that there is some threshold  $m_0$  such that for  $m \leq m_0$ ,  $\text{rc}(G_m^n) \geq (1 + \xi)n$  for some constant  $\xi > 0$  but  $\text{rc}(G_m^n) \sim n$  for  $m > m_0$ .

It would be interesting to study the robot crawler process on other models of complex networks, such as random graphs with given expected degree sequence [12], preferential attachment graphs with increasing average degrees [15], or geometric models such as the spatially preferred attachment model [1, 13], geographical threshold graphs [10], or GEO-P model [7].

## REFERENCES

- [1] W. Aiello, A. Bonato, C. Cooper, J. Janssen, P. Prałat, A spatial web graph model with local influence regions, *Internet Mathematics* **5** (2009) 175–196.
- [2] N. Alon, P. Prałat, N. Wormald, Cleaning regular graphs with brushes, *SIAM Journal on Discrete Mathematics* **23** (2008) 233–250.
- [3] D.L. Applegate, R.E. Bixby, V. Chvátal, W.J. Cook, *The Traveling Salesman Problem*, Princeton University Press, 2007.
- [4] A.L. Barabási, R. Albert, Emergence of scaling in random networks, *Science* **286** (1999) 509–512.
- [5] B. Bollobás, O. Riordan, The diameter of a scale-free random graph, *Combinatorica* **24** (2004) 5–34.
- [6] B. Bollobás, O. Riordan, J. Spencer, G. Tusnády, The degree sequence of a scale-free random graph process, *Random Structures and Algorithms* **18** (2001) 279–290.
- [7] A. Bonato, J. Janssen, P. Prałat, Geometric protean graphs, *Internet Mathematics* **8** (2012) 2–28.
- [8] A. Bonato, R.M. del Río-Chanona, C. MacRury, J. Nicolaidis, X. Pérez-Giménez, P. Prałat, K. Ternovsky, The robot crawler number of a graph, In: *Proceedings of WAW’15*, 2015.
- [9] A. Bonato, R.J. Nowakowski, *The Game of Cops and Robbers on Graphs*, American Mathematical Society, 2011.
- [10] M. Bradonjić, A. Hagberg, A. Percus, The structure of geographical threshold graphs, *Internet Mathematics*, **5** (2008) 113–140.
- [11] S. Brin, L. Page, Anatomy of a large-scale hypertextual web search engine, In: *Proceedings of the 7th International World Wide Web Conference*, 1998.
- [12] F. Chung, L. Lu. *Complex Graphs and Networks*, American Mathematical Society, August 2006.
- [13] C. Cooper, A. Frieze, P. Prałat, Some typical properties of the spatial preferred attachment model, *Internet Mathematics*, **10** (2014) 27–47.
- [14] C. Cooper, D. Ilcinkas, R. Klasing, A. Kosowski, Derandomizing random walks in undirected graphs using locally fair exploration strategies, *Distributed Computing*, **24** (2011) 91–99.
- [15] C. Cooper, P. Prałat, Scale free graphs of increasing degree, *Random Structures and Algorithms* **38** (2011) 396–421.
- [16] A. Frieze, X. Pérez-Giménez, P. Prałat, and B. Reiniger, Perfect matchings and Hamiltonian cycles in the preferential attachment model, preprint.
- [17] A. Gajardo, A. Moreira, E. Goles, Complexity of Langton’s ant, *Discrete Applied Mathematics* **117** (2002) 41–50.
- [18] M.R. Henzinger, Algorithmic challenges in web search engines, *Internet Mathematics* **1** (2004) 115–126.
- [19] S. Janson, T. Łuczak, A. Ruciński, *Random Graphs*, Wiley, New York, 2000.

- [20] S. Koenig, B. Szymanski, Y. Liu, Efficient and inefficient ant coverage methods, *Annals of Mathematics and Artificial Intelligence*, **31** (2001) 41–76.
- [21] J. Komlós, E. Szemerédi, Limit distribution for the existence of hamiltonian cycles in a random graph, *Discrete Mathematics*, **43** (1983) 55–63.
- [22] Z. Li, A. Vetta, Bounds on the cleaning times of robot vacuums, *Operations Research Letters*, **38** (2010) 69–71.
- [23] N. Malpani, Y. Chen, N. H. Vaidya, J. L. Welch, Distributed token circulation in mobile ad hoc networks, *IEEE Transactions on Mobile Computing*, **4** (2005) 154–165.
- [24] C.D. Manning, P. Raghavan, H. Schütze, *Introduction to Information Retrieval*, Cambridge University Press, 2008.
- [25] M.E. Messinger, R.J. Nowakowski, The Robot cleans up, *Journal of Combinatorial Optimization*, **18** (2009) 350–361.
- [26] M.E. Messinger, R.J. Nowakowski, P. Pralat, Cleaning a network with brushes, *Theoretical Computer Science* **399** (2008) 191–205.
- [27] C. Olston, M. Najork, Web Crawling, *Foundations and Trends in Information Retrieval*, **4** (2010) 175–246.
- [28] B. Pittel, Note on the heights of random recursive trees and random  $m$ -ary search trees, *Random Structures and Algorithms* **5** (1994) 337–347.
- [29] I. A. Wagner, M. Lindenbaum, A. M. Bruckstein, Efficiently searching a graph by a smell-oriented vertex process, *Annals of Mathematics and Artificial Intelligence*, **24** (1998) 211–223.
- [30] D.B. West, *Introduction to Graph Theory, 2nd edition*, Prentice Hall, 2001.
- [31] V. Yanovski, I. A. Wagner, A. M. Bruckstein, A distributed ant algorithm for efficiently patrolling a network, *Algorithmica*, **37** (2003) 165–186.

RYERSON UNIVERSITY

*E-mail address:* abonato@ryerson.ca

UNIVERSIDAD NACIONAL AUTÓNOMA DE MEXICO

*E-mail address:* ritamaria@ciencias.unam.mx

DALHOUSIE UNIVERSITY

*E-mail address:* hc509500@dal.ca

RYERSON UNIVERSITY

*E-mail address:* jnicolai@ryerson.ca

RYERSON UNIVERSITY

*E-mail address:* xperez@ryerson.ca

RYERSON UNIVERSITY

*E-mail address:* pralat@ryerson.ca

RYERSON UNIVERSITY

*E-mail address:* kirill.ternovsky@ryerson.ca