THE ROBOT CRAWLER GRAPH PROCESS

ANTHONY BONATO, RITA M. DEL RÍO-CHANONA, CALUM MACRURY, JAKE NICOLAIDIS, XAVIER PÉREZ-GIMÉNEZ, PAWEŁ PRAŁAT, 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, 17, 26] and the book [23] 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, [29]). An intriguing generalization of Eulerian walks was introduced by Messinger and Nowakowski in [24], as a variant of graph cleaning processes (see, for example, [2, 25]). The reader is directed to [9] for an overview of graph cleaning and searching.

In the model of [24], 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 [24]).

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 [24], 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 [21] 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 [21] 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 [16]). The robot vacuum model can be regarded as an undirected version of the *rotor-router* model; see [28, 30].

In the present work, we provide a simplified model of a robot crawler on the web, based on the robot vacuum paradigm of [21, 24] 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 $\{0, -1, -2, \ldots, -|V(G)|+1\}$. 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 [19, 22, 28].

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 multi-partite 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 [29]. For a given $n \in \mathbb{N}$, we use the notation $B_n = \{-n+1, -n+2, \ldots, -1, 0\}$ and $[n] = \{1, 2, \ldots, n\}$. 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 $n \to \infty$ (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 $f(n)/g(n) \to 1$ as $n \to \infty$ (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* $\mathcal{RC}(G, \omega_0) = \big((\omega_t, v_t)\big)_{t=1}^{L}$ of a connected graph $G = (V, E)$ on n nodes with an *initial weighting* $\omega_0 : V \to B_n$, 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 ω_1 remain the same as in ω_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 $\mathcal{RC}(G, \omega_0) = \big((\omega_t, v_t)\big)_{t=1}^{L}$.
(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 ω_{t+1} remain the same as in ω_t.
(7) Increment to time $t + 1$ (that is, increase t by 1) and return to 4.

If the process terminates, then define
$$\mathrm{rc}(G, \omega_0) = L,$$
that is $\mathrm{rc}(G, \omega_0)$ is equal to the number of steps in the *crawling sequence* $(v_1, v_2, \ldots, v_L)$ (including the initial state) taken by the robot crawler until all nodes are clean; otherwise $\mathrm{rc}(G, \omega_0) = \infty$. We emphasize that for a given ω_0, all steps of the process are deterministic. Note that at each point of the process, the weighting ω_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 $\omega_0 : V \to B_n$, $\mathcal{RC}(G, \omega_0)$ terminates after a finite number of steps; that is,* $\mathrm{rc}(G, \omega_0) < \infty$.

Proof. For a contradiction, suppose that there exists a connected graph $G = (V, E)$ and initial weightings ω_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. □

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 Ω_n be the family of all initial weightings $\omega_0 : V \to B_n$. Then

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

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

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

Now, let us make some simple observations.

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

(1) $\mathrm{rc}(G) \le \overline{\mathrm{rc}}(G) \le \mathrm{RC}(G)$.
(2) $\mathrm{rc}(K_n) = \overline{\mathrm{rc}}(K_n) = \mathrm{RC}(K_n) = n$
(3) $\mathrm{rc}(C_n) = \overline{\mathrm{rc}}(C_n) = \mathrm{RC}(C_n) = n$.
(4) $\mathrm{rc}(G) = n$ *if and only if G has a hamiltonian path.*
(5) $\mathrm{RC}(G) \le n(\Delta + 1)^d$.

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 Δ 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 $|N(v)| \le \Delta$, 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 $0 \le t \le d$ from v is visited at least $(\Delta + 1)^{d-t} \ge 1$ times by the robot crawler. The proof follows as the diameter is d. □

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 [24]. 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 $\omega_0 : E \to B_{|E|}$ 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 ω_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 (ω_0, v_0); and similarly $S(G)$ is the maximum of $s(G, \omega_0, v_0)$ over all (ω_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 $\mathrm{rc}(G)$ and $\mathrm{RC}(G)$ will turn out to be useful: for any $W \subseteq V$, let

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

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

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

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 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*

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

and

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

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 $\omega_0 : E \to B_m$ 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 $\hat{\omega}_0 : V(L_k(G)) \to B_{(k-1)m+n}$ 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 $\mathrm{rc}(L_k(G), V)$, 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

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

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

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

Now, let us move to the lower bound for $\mathrm{rc}(L_k(G), V)$. For a contradiction, suppose that there exists a bijection $\hat{\omega}_0 : V(L_k(G)) \to B_{(k-1)m+n}$ that assigns the smallest weight to some $v_0 \in V$ and such that $\mathrm{rc}(L_k(G), \hat{\omega}_0) < k \cdot s(G)$. In order to construct the corresponding initial configuration $\omega_0 : E \to B_m$ 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 $s(G) \leq s(G, \omega_0, v_0) \leq \frac{1}{k}\,\mathrm{rc}(L_k(G), \hat{\omega}_0) < s(G)$. This contradiction implies that

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

and the proof of (1) follows.

The upper bound for (2) is obvious as $\mathrm{rc}(L_k(G)) \leq \mathrm{rc}(L_k(G), V)$ 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 $\hat{\omega}_0 : V(L_k(G)) \to B_{(k-1)m+n}$ such that $\hat{\omega}_0$ assigns the smallest weight to e_0 and $\mathrm{rc}(L_k(G), \hat{\omega}_0) < k \cdot s(G) - 1$. 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 $0, 1, \ldots$, 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, $\mathrm{rc}(L_k(G)) < \mathrm{rc}(L_k(G), \hat{\omega}_0) < k \cdot s(G) - 1$; 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 $\mathrm{rc}(L_k(G), V) \geq k \cdot s(G)$.

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 $\mathrm{rc}(L_k(G), \hat{\omega}_0)$ steps, again contradicting $\mathrm{rc}(L_k(G), V) \geq k \cdot s(G)$. 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 $\mathrm{rc}(L_k(G), \hat{\omega}_0) + 1 < k \cdot s(G)$ steps, as usual contradicting the fact that $\mathrm{rc}(L_k(G), V) \geq k \cdot s(G)$. $\square$

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*

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

This shows that the model we consider in this paper is a generalization of the edge model introduced in [24]. Instead of analyzing $s(G)$ for some connected graph G, one can construct $L_3(G)$ and analyze $\mathrm{rc}(L_3(G))$ (or construct $L_2(G)$ and analyze $\mathrm{rc}(L_2(G), V)$).

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

Similarly, one can show the relationship between $S(G)$ and $\mathrm{RC}(L_k(G))$. 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*

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

and

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

Hence,

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

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

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

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 $\mathrm{rc}(P_n) = n$ and $\mathrm{RC}(P_n) = 2n-2$. In order to achieve the minimum, one has to start the process from a leaf of P_n. Regardless of ω_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 $n \in \mathbb{N}$,*

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

Proof. As it is evident that $\overline{\mathrm{rc}}(P_1) = 1$, we focus on $n \in \mathbb{N} \setminus \{1\}$. We label nodes as follows: $V(P_n) = \{v_1, v_2, \ldots, v_n\}$. 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, $i \in \{2, 3, \ldots, n-1\}$. 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

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

and the proof is finished. □

We next give the precise value of rc and 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*

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

where $\mathit{diam}(T)$ 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 $\mathrm{rc}(T)$ and $\mathrm{RC}(T)$ 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 $\omega : V \to B_n$, 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, $\omega_0 : V \to B_n$ 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*

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

Proof. As already mentioned, it is easy to see that the robot crawler process $\mathcal{RC}(T, \omega_0)$ behaves exactly as $DFS(T, \omega_0, v_0)$ 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,

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

and the proof follows. □

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 $d = \mathrm{diam}(T)$. 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

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

(Note that for this bound, we are not using the fact that path D has length $\mathrm{diam}(T)$, 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 $\mathrm{RC}(T)$.

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

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

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

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 $V_1, \ldots, V_k$ of size n by K_n^k. Note that for any $n \in \mathbb{N}$ and $k = 2$, we have that

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

Indeed, since K_n^2 has a hamiltonian path, $\mathrm{rc}(K_n^2) = 2n$ (see Lemma 2(4)). However, in fact, regardless of the ω_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, $\mathrm{rc}(K_n^k) = kn$. For any $k \in \mathbb{N} \setminus \{1, 2\}$ and $n \in \mathbb{N}$, it is straightforward to see that

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

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

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

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

Before we sketch the proof of Theorem 9, we need a definition. Suppose that we are given an initial weighting ω_0 of K_n^k. For any $\ell \in [kn]$, let A_ℓ be the set of ℓ cleanest nodes; that is,

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

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_ℓ^j is the number of nodes of V_j that are among ℓ the cleanest ones (in the whole graph K_n^k). Note that for a random initial weighing ω_0, the expected value of a_ℓ^j is ℓ/k. Let $\varepsilon > 0$. We say that ω_0 is *ε-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 ε-balanced, regardless of how small ε is. We will use the following version of *Chernoff's bound*. Suppose that $X \in \mathrm{Bin}(n, p)$ is a binomial random variable with expectation $\mu = np$. If $0 < \delta < 3/2$, then

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

(For example, see Corollary 2.3 in [18].) 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 Γ be a set of size n taken uniformly at random from set $[N]$. The random variable X counts the number of elements of Γ 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 [18].)

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 ω_0 be a random initial weighting of K_n^k. Then we have that ω_0 is ε-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 ε-balanced initial weighting ω_0, $\mathrm{rc}(K_n^k, \omega_0) = kn + O(\log n)$. This will finish the proof since, by Lemma 10, a random initial weighting is ε balanced with probability $1 - O(n^{-1})$, and for any initial weighting ω_0 we have $\mathrm{rc}(K_n^k, \omega_0) \le \mathrm{RC}(K_n^k) = (k+1)n - 1 = O(n)$. Indeed,

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

Let ω_0 be any ε-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_ℓ. 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 $j \in [k], j \ne i$ ($u \in A_\ell$). Let us call V_i a *ℓ-crucial* partite set. Concentrating on non-crucial sets, we observe that for any $s \ne i$, 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 ℓ-crucial set, are already visited; we will call such nodes *ℓ-dangerous*. Let $f(\ell)$ be the number of ℓ-dangerous nodes.

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

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

Proof. We run the process and stop at time T_ℓ when the robot crawler is about to move to the fist node of A_ℓ. We concentrate on the time interval from T_ℓ 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 ℓ-crucial set and ℓ-dangerous nodes. Clearly, there are $\ell - \ell' \ge \ell/3$ nodes in $A_\ell \setminus A_{\ell'}$. Since ω_0 is ε-balanced, the number of nodes of the ℓ-crucial set that belong to A_ℓ and $A_{\ell'}$ is at most $(1+\varepsilon)\ell/k$ and at least $(1-\varepsilon)\ell'/k$, respectively. Since

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

this phase lasts $2f(\ell)$ steps and all ℓ-dangerous nodes are cleaned. The claim now follows easily as one can use a trivial bound for the number of ℓ'-dangerous nodes. Regardless which partite set is ℓ'-crucial, since ω_0 is ε-balanced, we can estimate the number of nodes in ℓ'-crucial set that belong to $A_\ell \setminus A'_\ell$. Since ℓ'-dangerous nodes must be in $A_\ell \setminus A'_\ell$, we obtain that

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

It follows that ℓ' is good and the claim holds by induction. □

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_ℓ of the process, $\ell + f(\ell) \le$

$\ell + 0.6\ell/k = O(\log n)$ 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. □

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 $n \to \infty$, where $p = p(n)$ may and usually does depend on n.

It is known (see, for example, [20]) 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 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 $\mathrm{rc}(\mathcal{G}(n,p)) = n$.*
(2) *If $pn \leq \log n + \log\log n - \omega$, then $\mathrm{rc}(\mathcal{G}(n,p)) > n$.*

The next upper bound on $\mathrm{RC}(\mathcal{G}(n,p))$ 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*

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

Proof. Given two different nodes u, v, the number of common neighbours is distributed as $\mathrm{Bin}(n-2, p^2)$, 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. $\mathrm{RC}(\mathcal{G}(n,p)) \leq n^3$. □

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*

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

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 $m = \Theta(n)$.