The robot crawler number of a graph*

Anthony Bonato[1], Rita M. del Río-Chanona[3], Calum MacRury[2], Jake Nicolaidis[1], Xavier Pérez-Giménez[1], Paweł Prałat[1], and Kirill Ternovsky[1]

[1] Ryerson University, Toronto, Canada,
[2] Dalhousie University, Halifax, Canada
[3] Universidad Nacional Autónoma de Mexico, Mexico City, Mexico

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 [10, 16, 25] and the book [22] 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, [28]). An intriguing generalization of Eulerian walks was introduced by Messinger and Nowakowski in [23], as a variant of graph cleaning processes (see, for example, [2, 24]). The reader is directed to [8] for an overview of graph cleaning and searching. In the model of [23], 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 $-1, -2, \ldots, -|E(G)|$ (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 [23]).

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 G has been cleaned. As observed in [23], 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 [20] gave an interesting example where the robot vacuum takes exponential time to clean the

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

graph. Let S_e be the maximum value of S over all connected graphs containing exactly e edges. It is proven in [20] 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. [13] who analyzed a similar model to the robot vacuum process. 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 [15]). The robot vacuum model can be regarded as an undirected version of the *rotor-router* model; see [27, 29].

In the present work, we provide a simplified model of a robot crawler on the web, based on the robot vacuum paradigm of [20, 23]. 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 [18, 21, 27].

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 process. We give asymptotic (and in some cases exact) values for these parameters for paths, trees, and complete multi-partite graphs. In Section 3, 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 4. We conclude with a summary and a list of open problems for further study. Due to lack of space, some of the proofs are omitted from this extended abstract and deferred to the extended version.

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 [28]. 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.

2 The robot crawler process: definition and properties

We now formally define the robot crawler process and the various robot crawler numbers of a graph. Some proofs are omitted owing to space constraints, and will appear in the full version of the paper. 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 $t + 1$ 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, but we omit the proof owing to space constraints.

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,* $\mathrm{rc}(G, \omega_0) < \infty$.

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

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

The following result is immediate. (Part 5. follows from the observation that, if a node v is cleaned by the robot crawler $\Delta + 1$ times within an interval of time-steps, then every neighbour of v must be cleaned at least once during that interval.)

Lemma 1. *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. $\mathrm{RC}(G) \le n(\Delta + 1)^d$.

The model introduced in [23] is analogous to the robot crawler process, in a way we make precise. For any connected graph $G = (V, E)$ and 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 shows the connection between the two models. Recall that $s(G)$ is the analogue of $\mathrm{rc}(G)$ in the robot vacuum model.

Theorem 2. *If $G = (V, E)$ is a connected graph, then*

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

Theorem 2 shows that, indeed, the model we consider in this paper is a generalization of the edge model introduced in [23]. Instead of analyzing $s(G)$ for some connected graph G, we may construct $L_3(G)$ and analyze $\mathrm{rc}(L_3(G))$.

Let us start with the following elementary example to illustrate the robot crawler parameters. 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 1(4) and Theorem 4 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 4 for more general result). By direct computation, we have the following result.

Theorem 3. *For any $n \in \mathbb{N}$,*

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

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 4. *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.

Now, let us move to more sophisticated example. 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 1(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 $\mathrm{RC}(K_n^k)$ the situation is slightly more complicated.

Theorem 5. *For any $k \in \mathbb{N} \setminus \{1, 2\}$ and $n \in \mathbb{N}$, we have 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 6. *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 6, 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

$$\Pr\left(|X - \mu| \geq \delta\mu\right) \leq 2\exp\left(-\frac{\delta^2 \mu}{3}\right). \tag{1}$$

(For example, see Corollary 2.3 in [17].) It is also true that (1) 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 (1) holds for the hypergeometric distribution with parameters N, n, and m, with expectation $\mu = nm/N$. (See, for example, Theorem 2.10 in [17].)

Now we are ready to state the important lemma which is used in the proof of Theorem 6. Its proof follows from the Chernoff's bound (1) for hypergeometric distributions, and is omitted.

Lemma 2. *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 of Theorem 6. 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 2, a random initial weighting is ε balanced with probability $1 - O(n^{-1})$, and for any initial weighting ω_0 we have $\mathrm{rc}(K_n^k, \omega_0) \leq \mathrm{RC}(K_n^k) = (k+1)n - 1 = O(n)$. Indeed,

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

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 $j \in [k], j \neq i$ ($u \in A_\ell$). Let us call V_i a *ℓ-crucial* partite set. Concentrating on non-crucial sets, we observe that for any $s \neq i$, 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 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 $f(\ell) \leq 0.6\ell/k$. 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 $\lfloor 2\ell/3 \rfloor \geq 6\varepsilon^{-2} k \log n$.

To show the claim, 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 $\ell - \ell' \geq \ell/3$ 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

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

this phase lasts $2f(\ell)$ steps and all 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

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

It follows that 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, $\ell + f(\ell) \leq \ell + 0.6\ell/k = O(\log n)$ 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. □

3 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, [19]) 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 derive immediately the following result.

Corollary 1. *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 1(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 2. *Suppose* $pn \geq C\sqrt{n\log n}$, *for a sufficiently large constant* $C > 0$. *Then a.a.s. we have that*

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

Moreover, we give the following lower bound (whose proof is omitted here).

Theorem 7. *Suppose* $C\sqrt{n\log n} \leq pn \leq (1-\varepsilon)n$, *for constants* $C > 1$ *and* $\varepsilon > 0$. *Then a.a.s. we hae that*

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

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

Theorem 8. *Let* $p = p(n)$ *such that* $pn \gg \sqrt{n\log n}$. *Then a.a.s.*

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

The main ingredient to derive Theorem 8 is the following key lemma.

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

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

We are going to fix an initial weighting before exposing edges of the random graph. For a given initial weighting $\omega_0 : V \to B_n$, 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 3, we state the following useful claim that holds even for much sparser graphs (the proof is immediate by a standard Chernoff bound (1)).

Claim 1. Let $G = (V,E) \in \mathcal{G}(n,p)$ for some $p = p(n)$ such that $pn \gg \log n$. Let $\omega_0 : V \to B_n$ 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.

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 $\mathrm{rc}(\mathcal{G}(n,p), \omega_0)$ holds with the desired probability *unless* the claim is false. Now we can move to the proof of Lemma 3.

Proof of Lemma 3. 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 Claim 1 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 $n - 5 \log n/p \leq T_2 \leq n$. 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 Claim 1 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 Claim 1 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 $pn \ge 10\sqrt{n \log n}$, $25 \log n/p \le pn/4$ and so the property in Claim 1 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 (1), the probability that phase 4 ends because of property (b) and not (c) is at most

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

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

4 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 $(G_1^t)_{t \ge 0}$ 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:

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

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 $v_1', v_2', \ldots$ of nodes; the graph G_m^t if formed from G_1^{tm} by identifying nodes $v_1', v_2', \ldots, v_m'$ to form v_1, identifying nodes $v_{m+1}', v_{m+2}', \ldots, v_{2m}'$ 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 (1). Moreover, Pittel [26] 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 γ is the solution of $\gamma e^{1+\gamma} = 1$ (see Theorem 13 in [5]). Hence, the following result holds immediately from Theorem 4.

Theorem 9. *The following properties hold a.a.s. for any connected component G of G_1^n:*

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

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. $\mathrm{rc}(G_1^n) \sim \mathrm{RC}(G_1^n) \sim 2n$, 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.

Theorem 10. *A.a.s.* $\mathrm{rc}(G_2^n) \geq (1 + \xi + o(1))n$, *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 $\lfloor cn \rfloor \le t \le n$, let

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

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 $m\lfloor cn \rfloor \le t \le mn$, let

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

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

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

The conditional expectation is given by

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

Taking expectation again, we derive that

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

Hence, arguing as before, it follows that

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

Noting that $\mathbb{E}[Y_t] = \Theta(n)$ for any $\lfloor cn \rfloor \le t \le n$, 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; see Azuma-Hoeffding inequality (see, for example, [17]). Hence, we may assume that $Y_t \sim 2mn\sqrt{ct/n}$ for any $\lfloor cn \rfloor \le t \le n$.

The rest of the proof is straightforward. Note that, for a given $t = xn$ with $c \le x \le 1$, 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. $\square$

5 Conclusion and open problems

We introduced the robot crawler model, which is a simplified model of web crawling. We studied the minimum, maximum, and average time for the robot crawler process to terminate. We found exact values for these parameters in several graph classes such as trees and complete multi-partite graphs. We have successfully addressed the robot crawler model in binomial random graphs, and considered the rc parameter for preferential attachment graphs in the cases $m = 1, 2$.

Several problems concerning the robot crawler model remain open. We list some of these relevant to our investigation below.

1. Let G_n be the complete k-partite graph with partite sets of sizes $c_1 n, c_2 n, \ldots, c_k n$ for some constants $0 < c_1 \le c_2 \le \ldots \le c_k$. Derive the asymptotic behaviour of $\mathrm{rc}(G_n)$, $\overline{\mathrm{rc}}(G_n)$, and $\mathrm{RC}(G_n)$.
2. Theorem 8 holds for dense random graphs; that is, for $pn \gg \sqrt{n \log n}$. What about sparser random graphs?
3. Can the bound in Corollary 2 be improved? Is it true that $\mathrm{RC}(\mathcal{G}(n,p)) = O(n)$ for a wide range of p? Recall, in view of Theorem 7, that we cannot achieve $\mathrm{RC}(\mathcal{G}(n,p)) = (1 + o(1))n$, provided that $p < 1 - \varepsilon$ for some $\varepsilon > 0$.
4. Properties of the robot crawler remain open in the preferential attachment model when $m > 2$. Fix $m \ge 3$. Is it true that a.a.s. $\mathrm{rc}(G_m^n) \ge (1+\xi)n$ for some constant $\xi > 0$? Or maybe $\mathrm{rc}(G_m^n) \sim n$? It is possible that there is some threshold m_0 such that for $m \le m_0$, $\mathrm{rc}(G_m^n) \ge (1 + \xi)n$ for some constant $\xi > 0$ but $\mathrm{rc}(G_m^n) \sim n$ for $m > m_0$.

Our work with the robot crawler is a preliminary investigation. As such, 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 [11], preferential attachment graphs with increasing average degrees [14], or geometric models such as the spatially preferred attachment model [1, 12], geographical threshold graphs [9], 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(1)** (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.J. Nowakowski, *The Game of Cops and Robbers on Graphs*, American Mathematical Society, 2011.
9. M. Bradonjić, A. Hagberg, A. Percus, The structure of geographical threshold graphs, *Internet Mathematics*, **5** (2008) 113–140.
10. S. Brin, L. Page, Anatomy of a large-scale hypertextual web search engine, In: *Proceedings of the 7th International World Wide Web Conference*, 1998.
11. F. Chung, L. Lu. *Complex Graphs and Networks*, American Mathematical Society, August 2006.
12. C. Cooper, A. Frieze, P. Prałat, Some typical properties of the spatial preferred attachment model, *Internet Mathematics*, **10** (2014) 27–47.
13. 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.
14. C. Cooper, P. Prałat, Scale free graphs of increasing degree, *Random Structures and Algorithms* **38** (2011) 396–421.
15. A. Gajardo, A. Moreira, E. Goles, Complexity of Langton's ant, *Discrete Applied Mathematics* **117** (2002) 41–50.
16. M.R. Henzinger, Algorithmic challenges in web search engines, *Internet Mathematics* **1** (2004) 115–126.
17. S. Janson, T. Łuczak, A. Ruciński, *Random Graphs*, Wiley, New York, 2000.
18. S. Koenig, B. Szymanski, Y. Liu, Efficient and inefficient ant coverage methods, *Annals of Mathematics and Artificial Intelligence*, **31** (2001) 41–76.
19. J. Komlós, E. Szemerédi, Limit distribution for the existence of hamiltonian cycles in a random graph, *Discrete Mathematics*, **43(1)** (1983) 55–63.
20. Z. Li, A. Vetta, Bounds on the cleaning times of robot vacuums, *Operations Research Letters*, **38(1)** (2010) 69–71.
21. 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.
22. C.D. Manning, P. Raghavan, H. Schütze, *Introduction to Information Retrieval*, Cambridge University Press, 2008.
23. M.E. Messinger, R.J. Nowakowski, The Robot cleans up, *Journal of Combinatorial Optimization*, **18(4)** (2009) 350–361.
24. M.E. Messinger, R.J. Nowakowski, P. Prałat, Cleaning a network with brushes, *Theoretical Computer Science* **399** (2008) 191–205.
25. C. Olston, M. Najork, Web Crawling, *Foundations and Trends in Information Retrieval*, **4(3)** (2010) 175–246.
26. B. Pittel, Note on the heights of random recursive trees and random m-ary search trees, *Random Structures and Algorithms* **5** (1994) 337–347.
27. 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.
28. D.B. West, *Introduction to Graph Theory, 2nd edition*, Prentice Hall, 2001.
29. V. Yanovski, I. A. Wagner, A. M. Bruckstein, A distributed ant algorithm for efficiently patrolling a network, *Algorithmica*, **37** (2003) 165–186.