ARTICLE IN PRESS

Discrete Applied Mathematics ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

The robot crawler graph process

Anthony Bonato [a], Rita M. del Río-Chanona [b], Calum MacRury [c], Jake Nicolaidis [a], Xavier Pérez-Giménez [d], Paweł Prałat [a,*], Kirill Ternovsky [a]

[a] *Ryerson University, Canada*
[b] *Universidad Nacional Autónoma de Mexico, Mexico*
[c] *McGill University, Canada*
[d] *University of Nebraska-Lincoln, USA*

ARTICLE INFO

Article history:
Received 27 October 2015
Received in revised form 11 November 2017
Accepted 24 January 2018
Available online xxxx

Keywords:
Deterministic walk
Graph searching
Random graph
Preferential attachment model

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 $-1, -2, \ldots, -|E(G)|$ (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 [25]).

* Corresponding author.
E-mail addresses: abonato@ryerson.ca (A. Bonato), ritamaria@ciencias.unam.mx (R.M. del Río-Chanona), hc509500@dal.ca (C. MacRury), jnicolai@ryerson.ca (J. Nicolaidis), xperez@ryerson.ca (X. Pérez-Giménez), pralat@ryerson.ca (P. Prałat), kirill.ternovsky@ryerson.ca (K. Ternovsky).

https://doi.org/10.1016/j.dam.2018.01.018
0166-218X/

Please cite this article in press as: A. Bonato, et al., The robot crawler graph process, Discrete Applied Mathematics (2018), https://doi.org/10.1016/j.dam.2018.01.018.

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 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 analysed 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 $\{0, -1, -2, \ldots, -|V(G)| + 1\}$. 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 [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 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 [30]. 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* $\mathcal{RC}(G, \omega_0) = \left((\omega_t, v_t)\right)_{t=1}^{L}$ 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 $\mathcal{RC}(G, \omega_0) = \left((\omega_t, v_t)\right)_{t=1}^{L}$.
(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.

Please cite this article in press as: A. Bonato, et al., The robot crawler graph process, Discrete Applied Mathematics (2018), https://doi.org/10.1016/j.dam.2018.01.018.