Theoretical Computer Science 608 (2015) 178–189

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Strategies for parallel unaware cleaners

Christian Ortolf*, Christian Schindelhauer

University of Freiburg, Department of Computer Science, Computer Networks, Germany

ARTICLE INFO

Article history:
Available online 30 September 2015

Keywords:
Visit time
Competitive analysis
Mobile agent
Robot
Multi-robot graph exploration

ABSTRACT

We investigate the parallel traversal of a graph with multiple robots unaware of each other. All robots traverse the graph in parallel forever and the goal is to minimize the time needed until the last node is visited (*first visit time*) and the time between revisits of a node (*revisit time*). We also want to minimize the *visit time*, i.e. the maximum of the first visit time and the time between revisits of a node. We present randomized algorithms for uncoordinated robots, which can compete with the optimal coordinated traversal by a small factor, the so-called *competitive ratio*.

For any number of robots ring and path graph simple traversal strategies allow constant competitive factors even in the worst case. For grid and torus graphs with n nodes and any number of robots there is an $\mathcal{O}(\log n)$-competitive algorithm for both visit problems succeeding with high probability, i.e. with probability $1 - n^{-\mathcal{O}(1)}$. For general graphs we present an $\mathcal{O}(\log^2 n)$-competitive algorithm for the first visit problem, while for the visit problem we show an $\mathcal{O}(\log^3 n)$-competitive algorithm both succeeding with high probability.

1. Introduction

Today, we are used to robotic lawn mowers and robotic vacuum cleaning. The current best-selling technology relies on robots which have no communication features and in some cases use maps of the environment. If we model the environment as an undirected graph, then its traversal by a single robot is an NP-hard minimum Traveling Salesman problem, for which efficient constant factor approximation algorithms are known [3]. Now, the robot owner deploys additional robots. How well do these robots perform? Can we guarantee that two parallel unaware lawn mowers will cut all grass better than one? And how do they compare to a couple of perfectly choreographed mowers? What about more robots, where each robot has no clue how many co-working devices exist nor where they are?

Here, we investigate these questions. We model the cleaning area by a graph with identifiable nodes and edges. All robots know only their own position and the graph. They will never learn how many robots are involved, nor any other robots' positioning data. So, we assume that robots pass each other on the same node without noticing. We are looking for a traversal strategy of the graph which is self-compatible, since we assume that all robots are clones performing the same strategy.

It seems apparent that such a strategy must be probabilistic, since robots starting from the same node would otherwise follow identical routes, which would not allow for any speedup. However, we will see that this is not the case for cycle and path graphs.

* Corresponding author.
E-mail addresses: ortolf@informatik.uni-freiburg.de (C. Ortolf), schindel@informatik.uni-freiburg.de (C. Schindelhauer).

http://dx.doi.org/10.1016/j.tcs.2015.09.026
0304-3975/

Related work To our knowledge this unaware parallel cleaning model is new, therefore we will point out similarities to other problems.

The parallel unaware cleaning can be seen as a variation of the multi-robot exploration [4,6,8,5,16]. The goal of the online multi-robot exploration is to steer a group of robots to visit every node of an unknown graph. The term *unknown* means that the exploring algorithm knows only edges adjacent to formerly visited nodes. The performance of such online algorithms is usually provided by a competitive analysis comparing the online solution to the optimal offline strategy, where an algorithm is given knowledge of the whole graph beforehand. This model is close to our first visit time model with two important differences: In parallel unaware cleaning each robot knows the full graph, while multi-robot exploration robots know only the explored graph. In our model there is no communication, while in robot exploration robots exchange their graph information.

It was recently shown that if more than dn robots are used in the multi-robot exploration problem, where d is the diameter of the graph and n the number of nodes, then one can achieve a constant competitive factor for multi-robot exploration [4]. The competing offline exploration can explore a graph in time $\Theta(\frac{n}{k}+d)$, therefore an exploration using $k=\frac{n}{d}$ robots is of special interest, because it allows the offline algorithm to make full use of all its robots.

For this scenario Dynia et al. [6] showed the online exploration of trees to be at best $\Omega(\frac{\log k}{\log\log k})$-competitive. If algorithms are restricted to greedy exploration an even stronger bound of $\Omega(k/\log k)$ is shown by Higashikawa et al. [11]. This bound matches the best known upper bound by Fraigniaud et al.'s greedy exploration algorithm in [8]. For further restricted graphs better bounds have been shown. An algorithm depending on a *density* parameter p was presented by Dynia et al. [5] with $\mathcal{O}(d^{1-1/p})$ competitiveness, e.g. $\mathcal{O}(d^{1/2})$ for trees embeddable in grids. For grids with convex obstacles, a polylogarithmic competitive bound of $\mathcal{O}(\log^2 n)$ was shown in [16], along with the lower bound of $\Omega(\frac{\log k}{\log\log k})$ matching the identical lower bound for trees.

Our problem also bears resemblance to the multi-traveling salesman problem (mTSP) [2,9], a generalization of the well-known traveling salesman problem (TSP) [13]. TSP is NP-hard even in the seemingly simpler Euclidean version [17], but can be efficiently approximated if it is allowed to visit nodes more than once [19].

The mTSP tries to cover the graph with a set of tours and minimize the length of the longest tour. This corresponds to the offline parallel cleaning problem, if we use the distance between nodes in the graph as cost measure between nodes in mTSP. Even if salesmen start at different nodes the problem can still be reduced to the regular mTSP [10].

A similar definition to our first visit time is the notion of *cover time* for random walks, likewise visit time can be compared to the *hitting time* $H(i,j)$, the expected time starting from node i to reach node j. Our robots are not forced to use random walks. So, the Lollipop graph, a lower bound construction for the cover time of $\Omega(n^3)$ [14] and obtained by joining a complete graph to a path graph with a bridge, can be cleaned quite efficiently by parallel unaware cleaners.

Patrolling algorithms [18] also require robots to repeatedly visit the same area. To the best of our knowledge no theoretical analysis exists with similarly restricted robots.

2. Model

In our model k robots are initially positioned on depot/starting nodes $S=(s_1,\dots,s_k)$ and their task is to visit all nodes V, with $|V|>1$, of an undirected connected graph $G=(V,E)$ and then repeat their visits as fast as possible. Nodes and edges can be identified and time is measured in rounds. An algorithm has to decide for each robot r in each round which edge to traverse to visit another node in the following round. This decision is based on the starting node s_r, the graph and the previous decisions of the robot. Each robot never learns the number and positions of other robots.

The **first visit time of a node** is the number of the round, when a robot visits this node for the first time. The **visit time of a node** is the supremum of all time intervals between any two visits of a node (revisit) including the time interval necessary for the first visit by any robot. The **long term visit time of a node** is the supremum of time intervals between any two visits of a node by any robot after an arbitrarily long time. The corresponding definitions for the full graph are given by the maximum (first/long term) visit time of all nodes. Note that the robots do not know and cannot compute the visit times. These times can only be known by an external observer.

The term **with high probability** refers to an event which occurs with probability $1-n^{-c}$ with constant $c\geq 1$. In all of our results, this constant c can be arbitrarily increased if one allows a larger constant factor for the run-time.

The term **distance** refers to the number of edges on a shortest path between two nodes.

The benchmark for our solution is the time of an algorithm with full knowledge, i.e. the number and positions of all robots. The quotient between the unaware visit time and the full knowledge visit time is our measure, also known as the **competitive factor** and sometimes referred to us as overhead. The worst case setting can be seen as an adversary placing the robots for a given algorithm.

3. Simple cleaning examples

As an illustration and starting example we show how differently a circle graph and a path graph behave in this setting, see Fig. 1 and Fig. 2. The simple algorithm sending robots in one direction in the circle, or just to one end on the line, then returning to the other end, performs quite differently for both graphs.