Theoretical Computer Science 608 (2015) 190–198

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Fast rendezvous with advice ☆

Avery Miller *, Andrzej Pelc **,[1]

Université du Québec en Outaouais, 101 rue Saint-Jean-Bosco, C.P. 1250, succ. Hull, Gatineau, Québec, J8X 3X7, Canada

ARTICLE INFO

Article history:
Available online 30 September 2015

Keywords:
Rendezvous
Advice
Deterministic distributed algorithm
Mobile agent
Time

ABSTRACT

Two mobile agents, starting from different nodes of an n-node network at possibly different times, have to meet at the same node. This problem is known as *rendezvous*. Agents move in synchronous rounds using a deterministic algorithm. In each round, an agent decides to either remain idle or to move to one of the adjacent nodes. Each agent has a distinct integer label from the set $\{1, \ldots, L\}$, which it can use in the execution of the algorithm, but it does not know the label of the other agent.

The main efficiency measure of a rendezvous algorithm's performance is its *time*, i.e., the number of rounds from the start of the earlier agent until the meeting. If D is the distance between the initial positions of the agents, then $\Omega(D)$ is an obvious lower bound on the time of rendezvous. However, if each agent has no initial knowledge other than its label, time $O(D)$ is usually impossible to achieve. We study the minimum amount of information that has to be available *a priori* to the agents to achieve rendezvous in optimal time $\Theta(D)$. Following the standard paradigm of *algorithms with advice*, this information is provided to the agents at the start by an oracle knowing the entire instance of the problem, i.e., the network, the starting positions of the agents, their wake-up rounds, and both of their labels. The oracle helps the agents by providing them with the *same* binary string called *advice*, which can be used by the agents during their navigation. The length of this string is called the *size of advice*. Our goal is to find the smallest size of advice which enables the agents to meet in time $\Theta(D)$. We show that this optimal size of advice is $\Theta(D \log(n/D) + \log\log L)$. The upper bound is proved by constructing an advice string of this size, and providing a natural rendezvous algorithm using this advice that works in time $\Theta(D)$ for all networks. The matching lower bound, which is the main contribution of this paper, is proved by exhibiting classes of networks for which it is impossible to achieve rendezvous in time $\Theta(D)$ with smaller advice.

☆ A preliminary version of this paper appeared in Proc. 10th International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics (ALGOSENSORS 2014).

* Principal corresponding author.
** Corresponding author.
E-mail addresses: avery@averymiller.ca (A. Miller), andrzej.pelc@uqo.ca (A. Pelc).

[1] Partially supported by NSERC discovery grant 8136 – 2013 and by the Research Chair in Distributed Computing at the Université du Québec en Outaouais.

http://dx.doi.org/10.1016/j.tcs.2015.09.025
0304-3975/

1. Introduction

1.1. Background

Two mobile agents, starting from different nodes of a network, have to meet at the same node in the same time. This distributed task is known as *rendezvous* and has received a lot of attention in the literature. Agents can be any mobile autonomous entities. They might represent human-made objects, such as software agents in computer networks or mobile robots navigating in a network of corridors in a building or a mine. They might also be natural, such as animals seeking a mate, or people who want to meet in an unknown city whose streets form a network. The purpose of meeting in the case of software agents or mobile robots might be the exchange of data previously collected by the agents or samples collected by the robots. It may also be the coordination of future network maintenance tasks, for example checking functionality of websites or of sensors forming a network, or decontaminating corridors of a mine.

1.2. Model and problem description

The network is modelled as an undirected connected graph with n unlabelled nodes. We seek deterministic rendezvous algorithms that do not rely on the agents perceiving node identifiers, and therefore can work in anonymous graphs as well (cf. [4]). The reason for designing such algorithms is that, even when nodes of the network have distinct identifiers, agents may be unable to perceive them because of limited sensory capabilities (e.g., a mobile robot may be unable to read signs at corridor crossings), or nodes may be unwilling to reveal their identifiers to software agents, e.g., due to security or privacy reasons. From a methodological point of view, if nodes had distinct identifiers visible to the agents, the agents could explore the graph and meet at the node with the smallest identifier. Hence, in this case, rendezvous reduces to graph exploration.

On the other hand, we assume that, at each node v, each edge incident to v has a distinct *port number* from the set $\{0, \ldots, d-1\}$, where d is the degree of v. These port numbers are fixed and visible to the agents. Port numbering is *local* to each node, i.e., we do not assume any relation between port numbers at the two endpoints of an edge. Note that in the absence of port numbers, edges incident to a node would be undistinguishable for agents and thus rendezvous would be often impossible, as an agent may always miss some particular edge incident to the current node, and this edge could be a bridge to the part of the graph where the other agent started. The previously mentioned security and privacy reasons for not revealing node identifiers to software agents are irrelevant in the case of port numbers. If the graph models a system of corridors of a mine or a building, port numbers can be made implicit, e.g., by marking one edge at each intersection (using a simple mark legible even by a mobile robot with very limited vision), considering it as corresponding to port 0, and all other port numbers increasing clockwise.

Agents are initially located at different nodes of the graph and traverse its edges in synchronous rounds. They cannot mark visited nodes or traversed edges in any way, and they cannot communicate before meeting. The adversary wakes up each of the agents, possibly in different rounds. Each agent starts executing the algorithm in the round of its wake-up. It has a clock that ticks at each round and starts at the wake-up round of the agent. In each round, each agent either remains at the current node, or chooses a port in order to move to one of the adjacent nodes. When an agent enters a node, it learns the node's degree and the port number by which it enters the node. When agents cross each other on an edge while traversing it simultaneously in different directions, they do not notice this fact.

Each agent has a distinct integer label from a fixed *label space* $\{1, \ldots, L\}$, which it can use as a parameter in the same deterministic algorithm that both agents execute. It does not know the label nor the starting round of the other agent. Since we study deterministic rendezvous, the absence of distinct labels would preclude the possibility of meeting in highly symmetric graphs, such as rings or tori, for which there exist non-trivial port-preserving automorphisms. Indeed, in such graphs, identical agents starting simultaneously and executing the same deterministic algorithm can never meet, since they will keep the same positive distance in every round. Hence, assigning different labels to agents is the only way to break symmetry, as is needed to meet in every graph using a deterministic algorithm. On the other hand, if agents knew each other's identities, then the smaller-labelled agent could stay inert, while the other agent would try to find it. In this case rendezvous reduces to graph exploration. Assuming such knowledge, however, is not realistic, as agents are often created independently in different parts of the graph, and they know nothing about each other prior to meeting.

The rendezvous is defined as both agents being at the same node in the same round. The main efficiency measure of a rendezvous algorithm's performance is its *time*, i.e., the number of rounds from the start of the earlier agent until the meeting. If D is the distance between the initial positions of the agents, then $\Omega(D)$ is an obvious lower bound on the time of rendezvous. However, if the agents have no additional knowledge, time $O(D)$ is usually impossible to achieve. This is due to two reasons. First, without any knowledge about the graph, even the easier task of *treasure hunt* [42], in which a single agent must find a target (treasure) hidden at an unknown node of the graph, takes asymptotically larger time in the worst case. Treasure hunt is equivalent to a special case of rendezvous where one of the agents is inert. In the worst case, this takes as much time as graph exploration, i.e., having a single agent visit all nodes. Second, even when the graph is so simple that navigation of the agents is not a problem, breaking symmetry between the agents, which is often necessary to achieve a meeting, may take time larger than D. Indeed, even in the two-node graph, where $D = 1$, rendezvous requires time $\Omega(\log L)$ [18].