Theoretical Computer Science 621 (2016) 92–102

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Topology recognition and leader election in colored networks

Dariusz Dereniowski[a,*,1], Andrzej Pelc[b,2]

[a] Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Narutowicza 11/12, 80-233 Gdańsk, Poland
[b] Département d'informatique, Université du Québec en Outaouais, Gatineau, Québec J8X 3X7, Canada

ARTICLE INFO

Article history:
Received 27 October 2014
Received in revised form 22 January 2016
Accepted 27 January 2016
Available online 2 February 2016
Communicated by D. Peleg

Keywords:
Topology recognition
Leader election
Colored network
Local model

ABSTRACT

Topology recognition and leader election are fundamental tasks in distributed computing in networks. The first of them requires each node to find a labeled isomorphic copy of the network, while the result of the second one consists in a single node adopting the label 1 (leader), with all other nodes adopting the label 0 and learning a path to the leader. We consider both these problems in networks whose nodes are equipped with not necessarily distinct labels called *colors*, and ports at each node of degree d are arbitrarily numbered $0, 1, \dots, d-1$. Colored networks are generalizations both of labeled networks, in which nodes have distinct labels, and of anonymous networks, in which nodes do not have labels (all nodes have the same color).

In colored networks, topology recognition and leader election are not always feasible. Hence we study two more general problems. Consider a colored network and an input I given to its nodes. The aim of the problem TOP, for this colored network and for I, is to solve topology recognition in this network, if this is possible under input I, and to have all nodes answer "unsolvable" otherwise. Likewise, the aim of the problem LE is to solve leader election in this network, if this is possible under input I, and to have all nodes answer "unsolvable" otherwise.

We show that nodes of a network can solve problems TOP and LE, if they are given, as input I, an upper bound k on the number of nodes of a given color, called the *size* of this color. On the other hand we show that, if the nodes are given an input that does not bound the size of any color, then the answer to TOP and LE must be "unsolvable", even for the class of rings.

Under the assumption that nodes are given an upper bound k on the size of a given color, we study the time of solving problems TOP and LE in the $\mathcal{LOCAL}$ model in which, during each round, each node can exchange arbitrary messages with all its neighbors and perform arbitrary local computations. We give an algorithm to solve each of these problems in arbitrary networks in time $O(kD + D\log(n/D))$, where D is the diameter of the network and n is its size. We also show that this time is optimal, by exhibiting classes of networks in which every algorithm solving problems TOP or LE must use time $\Omega(kD + D\log(n/D))$.

© 2016 Elsevier B.V. All rights reserved.

* Corresponding author.
E-mail addresses: deren@eti.pg.gda.pl (D. Dereniowski), pelc@uqo.ca (A. Pelc).

1 Partially supported by National Science Centre grant DEC-2015/17/B/ST6/01887.

2 Partly supported by the 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.2016.01.037
0304-3975/© 2016 Elsevier B.V. All rights reserved.

1. Introduction

1.1. The model and the problem

Topology recognition and leader election are fundamental tasks in distributed computing in networks. The goal of topology recognition is for each node of the network to acquire a faithful map of it (an isomorphic copy of the underlying network with all nodes having distinct identifiers), with the position of the node marked in the map. If nodes can solve this problem, any other distributed task, such as leader election [27,38], minimum weight spanning tree construction [5], renaming [2], etc. can be performed by them using only local computations. Thus topology recognition converts all distributed problems to centralized ones, in the sense that nodes can solve any distributed problem simulating a central monitor.

Leader election, first stated in [32], is likewise of fundamental importance. Each node of the network has a Boolean variable initialized to 0 and, after the election, exactly one node, called the *leader*, should change this value to 1. All other nodes should know which one becomes the leader by discovering a path to it. Notice that the above two problems are equivalent: having a map of the network with distinct node labels, nodes can elect the node with the smallest label as the leader, and conversely, knowing a leader, nodes can construct a map of the network using the leader as a stationary token, cf. [11].

It should be noted that formulations of the leader election problem vary across the literature (cf. [33]). In a weak formulation, every node should only know if it is the leader or not. (In [34], this task was called selection, by contrast to election.) In a strong formulation, every node should moreover get to know who is the leader. We adopt the latter formulation of the leader election problem. When nodes have distinct identities, knowing who is the leader means outputting its identity. In our scenario distinct identities need not exist, hence knowing who is the leader means that every node outputs a path to the leader, coded as a sequence of ports. This formulation of leader election was used, e.g., in [24] for anonymous networks.

A network is modeled as a simple undirected connected graph. As commonly done in the literature, cf., e.g. [43] or KT_0 model in [6], we assume that ports at a node of degree d have arbitrary fixed labelings $0, \dots, d-1$. We do not assume any coherence between port labelings at various nodes. As for nodes, we assume that they are equipped with not necessarily distinct labels called *colors*. In applications, colors may be types of the devices interconnected by the network, such as workstations, servers, laptops, or mobile phones. Networks with colored nodes are generalizations both of labeled networks in which nodes have distinct labels, and of anonymous networks, in which nodes do not have labels (all nodes have the same color). Nodes communicate by exchanging arbitrary messages along links. A node sending a message through a given port appends the port number to the message, and a node receiving a message through a port is aware of the port number by which the message is received.

If nodes have distinct identities, both topology recognition and leader election are easily accomplished in any network. By contrast, it is well known that, in the absence of distinct node labels, these tasks are often impossible, if no additional information about the network is provided to nodes. In fact, even the less demanding task of reconstructing an unlabeled isomorphic copy of the network is sometimes impossible. For example, in an anonymous ring whose each edge has port numbers 0 and 1 at its endpoints, not only topology recognition and leader election cannot be achieved but even the size of the ring cannot be learned by nodes. Providing the size of the network as input is not a remedy either: the authors of [43] give examples of two (anonymous) non-isomorphic graphs of size 6 whose nodes cannot decide in which of these two graphs they are.

Due to these impossibilities, we consider two problems more general than topology recognition and leader election, respectively. Consider a colored network and an input I given to its nodes. The aim of the problem TOP, for this colored network and for I, is to solve topology recognition in this network, if this is possible under input I, and to have all nodes answer "unsolvable" otherwise. Likewise, the aim of the problem LE is to solve leader election in this network, if this is possible under input I, and to have all nodes answer "unsolvable" otherwise.

Our goal is to find out what type of input has to be given to the nodes of a colored network in order to enable them to solve problems TOP and LE, and what is the minimal time in which they can solve these problems, if this input is provided. To investigate time, we use the extensively studied $\mathcal{LOCAL}$ model [37]. In this model, communication proceeds in synchronous rounds and all nodes start simultaneously. In each round each node can exchange arbitrary messages with all its neighbors and perform arbitrary local computations. The time of completing a task is the number of rounds it takes.

1.2. Our results

We first show that nodes of a network can solve problems TOP and LE, if they are given an upper bound k on the number of nodes of a given color, called the *size* of this color. This means that, if such an upper bound is known to all nodes (even if they do not know any upper bound on the total number of nodes or on the number of colors), then they can correctly decide if leader election and topology recognition are feasible in the given network, and if so, they can perform these tasks. On the other hand, if the nodes are given an input that does not bound the size of any color, then the answer to TOP and LE must be "unsolvable", even for the class of rings.

Hence, providing all nodes with an upper bound on the size of some color is the weakest assumption under which problems TOP and LE can be meaningfully solved. This justifies the use of this assumption in our algorithms.

MOST WIEDZY Pobrano z mostwiedzy.pl

Next, assuming that all nodes have an upper bound k on the size of a given color, we study the time of solving problems TOP and LE in the $\mathcal{LOCAL}$ model. We give an algorithm to solve each of these problems in arbitrary networks in time $O(kD + D\log(n/D))$, where D is the diameter of the network and n is its size. We also show that this time is optimal, by exhibiting classes of networks in which every algorithm solving problems TOP or LE must use time $\Omega(kD + D\log(n/D))$.

1.3. Related work

Early studies of leader election in networks mostly concerned the scenario where all nodes have distinct labels. This task was first studied for rings. A synchronous algorithm, based on comparisons of labels, and using $O(n\log n)$ messages was given in [27]. It was proved in [20] that this complexity is optimal for comparison-based algorithms. On the other hand, the authors showed an algorithm using a linear number of messages but requiring very large running time. An asynchronous algorithm using $O(n\log n)$ messages was given, e.g., in [38] and the optimality of this message complexity was shown in [9]. Deterministic leader election in radio networks has been studied, e.g., in [28,29,35] and randomized leader election, e.g., in [40]. In [25] the leader election problem is approached in a model based on mobile agents for networks with labeled nodes.

Many authors [1,3,4,8,17,30,31,39,41,43] studied leader election and other computational problems in anonymous networks. In particular, [7,43] characterize message passing networks in which leader election can be achieved when nodes are anonymous. The authors assume that nodes know an upper bound on the size of the network. In [42] the authors study the problem of leader election in general networks, under the assumption that labels are not unique. They characterize networks in which this can be done and give an algorithm which performs election when it is feasible. They assume that the number of nodes of the network is known to all nodes. In [19] the authors study feasibility and message complexity of sorting and leader election in rings with nonunique labels, while in [18] the authors provide algorithms for the generalized leader election problem in rings with arbitrary labels, unknown (and arbitrary) size of the ring, and for both synchronous and asynchronous communication. Characterizations of feasible instances for leader election and naming problems have been provided in [10,12,13]. Memory needed for leader election in unlabeled networks has been studied in [21]. In [22], the authors investigated the time of leader election in anonymous networks by characterizing this time in terms of the size and diameter of the network, and of an additional parameter, called level of symmetry, which measures how deeply nodes have to inspect the network to notice differences in their views of it. In [16], the authors studied feasibility of leader election among anonymous agents that navigate in a network in an asynchronous way.

Feasibility of topology recognition for anonymous networks with adversarial port labelings was studied in [43], under the assumption that nodes know an upper bound on the size of the network. The problem of efficiency of map construction by a mobile agent, equipped with a token and exploring an anonymous network, has been studied in [11]. In [15], the authors investigated the minimum size of advice that has to be given to a mobile agent, in order to enable it to reconstruct the topology of an anonymous network or to construct its spanning tree. In [23], tradeoffs between time of topology recognition and the size of advice given to nodes were studied in the $\mathcal{LOCAL}$ communication model.

2. Preliminaries

In this section we introduce some basic terminology and provide preliminary results known from the literature. Let G be a simple connected undirected network with the set of nodes V, and let c be a positive integer. Consider any surjective function $f : V \longrightarrow \{1,\dots,c\}$. The couple (G, f) is called a *colored network*, the function f is called a *coloring* of this network, and $f(v)$ is called the *color* of node v.

We will use the following notion from [43]. Let G be a network and v a node of G. We first define, for any $l \geq 0$, the *truncated view* $\mathcal{V}^l(v)$ at depth l, by induction on l. $\mathcal{V}^0(v)$ is a tree consisting of a single node x_0. If $\mathcal{V}^l(u)$ is defined for any node u in the network, then $\mathcal{V}^{l+1}(v)$ is the port-labeled tree rooted at x_0 and defined as follows. For every node v_i, $i = 1,\dots,k$, adjacent to v, there is a child x_i of x_0 in $\mathcal{V}^{l+1}(v)$ such that the port number at v corresponding to edge $\{v, v_i\}$ is the same as the port number at x_0 corresponding to edge $\{x_0, x_i\}$, and the port number at v_i corresponding to edge $\{v, v_i\}$ is the same as the port number at x_i corresponding to edge $\{x_0, x_i\}$. We say that the node x_i *represents* node v_i. Now node x_i, for $i = 1,\dots,k$ becomes the root of the truncated view $\mathcal{V}^l(v_i)$. The *view* of v is the infinite rooted tree $\mathcal{V}(v)$ with labeled ports, such that $\mathcal{V}^l(v)$ is its truncation to depth l, for each $l \geq 0$.

We will also use a notion similar to that of the view but corresponding to colored networks (cf. [36]). Consider a colored network (G, f). Let v be any node of G. Let $f^* : \mathcal{V}(v) \longrightarrow \{1,\dots,c\}$ be the function defined as follows: $f^*(x) = f(v)$, where x is a node of $\mathcal{V}(v)$ representing node v. The couple $(\mathcal{V}(v), f^*)$ is called the *colored view* of node v. Thus, the colored view of a node additionally marks colors of nodes represented in it. The couple $(\mathcal{V}^l(v), f_l)$, where f_l is the truncation of f^* to $\mathcal{V}^l(v)$, is called a *truncated colored view* of node v.

For the (truncated) views of a node v we will often omit the node v in the notation, thus writing $\mathcal{V}$ instead of $\mathcal{V}(v)$ and $\mathcal{V}^l$ instead of $\mathcal{V}^l(v)$, if the node v, called the *root* of the view, is clear from the context. The same convention applies to colored (truncated) views. The *level* l of a view $\mathcal{V}$, denoted $\mathrm{lev}_l(\mathcal{V})$, is the set of all its nodes at distance l (in $\mathcal{V}$) from the root of the view. For nodes u and v in the truncated view $\mathcal{V}^l$, we denote by $P(\mathcal{V}^l, u, v)$ the unique path in $\mathcal{V}^l$ from u to v, defined as a sequence of nodes in this truncated view. We denote by $P(\mathcal{V}^l, v)$ the unique path from the root of the view to v. For such a path P, we denote by $|P|$ the length of this path, defined as the number of edges in it. For a truncated view $\mathcal{V}^l$ and a node z in this view, we denote by $\mathcal{V}^l[z]$ the subtree of $\mathcal{V}^l$ rooted at z.

MOST WIEDZY Pobrano z mostwiedzy.pl

The following proposition was proved in [26].

Proposition 2.1. *For a n-node network of diameter D, $\mathcal{V}(u) = \mathcal{V}(v)$, if and only if $\mathcal{V}^h(u) = \mathcal{V}^h(v)$, for some $h \in \Theta(D\log(n/D))$.* □

The following proposition that follows from [14] shows that the truncation level h from Proposition 2.1 is the smallest possible, up to constant factors.

Proposition 2.2. *For any integers $D \le n$, there exists a network G of size $\Theta(n)$ and diameter $\Theta(D)$, with nodes u and v, both with unique views, such that $\mathcal{V}(u) \neq \mathcal{V}(v)$ but $\mathcal{V}^{h'}(u) = \mathcal{V}^{h'}(v)$, for some $h' \in \Theta(D\log(n/D))$. Moreover, there exists a network G' having the same size and diameter as that of G, with a node u', such that $\mathcal{V}^{h'}(u') = \mathcal{V}^{h'}(u)$.* □

Propositions 2.1 and 2.2 remain valid, when views are replaced by colored views.

Define the following equivalence relations on the set of nodes of a colored network (G, f). Let $u \sim v$ if and only if $(\mathcal{V}(u), f^*) = (\mathcal{V}(v), f^*)$, and let also $u \sim_t v$ if and only if $(\mathcal{V}^t(u), f_t) = (\mathcal{V}^t(v), f_t)$. Let Π be the partition of all nodes into equivalence classes of $\sim$, and Π_t the corresponding partition for $\sim_t$. It follows from [43] that all equivalence classes in Π are of equal size σ. In view of Proposition 2.1 this is also the case for Π_t, for some $t \in \Theta(D\log(n/D))$. On the other hand, for smaller t, equivalence classes in Π_t may be of different sizes. Every equivalence class in Π_t is a union of some equivalence classes in $\Pi_{t'}$, for $t < t'$. The following result was proved in [36]. It says that if the sequence of partitions Π_t stops changing at some point, it will never change again.

Proposition 2.3. *If $\Pi_t = \Pi_{t+1}$, then $\Pi_t = \Pi$.* □

The following proposition, easily proved by induction on T, implies that if $\mathcal{V}^T(u) = \mathcal{V}^T(v)$ for some nodes u and v, and all nodes are given the same information about the network, then any algorithm solving TOP or LE in time at most T must give the same output, when executed by u and by v.

Proposition 2.4. *Let u be a node in a colored network G and u' a node in a colored network G'. Suppose that initially all nodes of G and G' have the same input. Let T be a positive integer and assume that $\mathcal{V}^T(u) = \mathcal{V}^T(u')$. For any $t \le T$, let M be the message received by u through port p in round t. Then, message M is received by u' through port p in round t.* □

Next, we define the notion of a *colored quotient graph*, which is a generalization of the notion of quotient graph introduced in [43]. Given a colored network (G, f), its colored quotient graph $(Q, \overline{f})$ is defined as follows. Nodes of Q are equivalence classes of the above defined relation $\sim$. If a and b are two such classes, there is an edge joining a and b in Q, with port number p at a and q at b, if and only if there is an edge joining nodes u and v in G, with port number p at u and q at v, where u belongs to the class a and v belongs to the class b. (Hence, unlike G, the graph Q can have self-loops and multiple edges.) The function $\overline{f}$ is defined on all nodes of Q by the formula $\overline{f}(a) = f(u)$, where u belongs to the class a.

Finally, we give the formal definitions of the main problems TOP and LE considered in this paper. Both these problems are to be solved in an unknown colored graph. For each of these problems, all nodes are given some common input I. In order to solve the problem LE, every node has to output a sequence of port numbers leading from this node to a single node, called the leader, if this task is possible to perform using input I; otherwise, all nodes must output the answer "unsolvable". In order to solve the problem TOP, every node v has to output an isomorphic copy C_v of the underlying graph, with all nodes labeled by distinct identifiers, and the node v correctly marked in C_v, if this task is possible to perform using input I; otherwise, all nodes must output the answer "unsolvable".

3. The algorithm and its analysis

In this section we describe and analyze an algorithm for solving the problems TOP and LE, assuming that some bound k on the size of one of the colors is known. This color will be denoted by α. Recall that, by definition, there exists at least one node with color α in the network.

The algorithm aims at computing the colored quotient graph. Once this is achieved, the solutions to TOP and LE will follow easily. The task of computing the colored quotient graph is divided into three procedures called `TestRepetition`, `ComputeView` and `ComputeQuotientGraph`. Procedure `ComputeView` computes the truncated colored view of the executing node up to a certain depth l. This depth l depends on the answers returned by several calls to `TestRepetition` in procedure `ComputeView`. Finally, procedure `ComputeQuotientGraph` uses the view and, by further extending it to an appropriate depth, obtains the colored quotient graph. We now give the detailed description of the above procedures. They are all formulated for an executing node w. We write $\mathcal{V}$ and $\mathcal{V}^l$ instead of $\mathcal{V}(w)$ and $\mathcal{V}^l(w)$, respectively.

Procedure `TestRepetition` uses the following notion of distance between colors and nodes. Given a truncated colored view $(\mathcal{V}^l, f)$ and a node v belonging to it, the *distance from v to color α* is the length of the shortest path in $\mathcal{V}^l[v]$ that connects v with some node with color α. Given a view $\mathcal{V}^l$ and a node v belonging to it, we say that v has a *copy in $\mathcal{V}^i$*, $i \le l$,

MOST WIEDZY Pobrano z mostwiedzy.pl

if there exists v' in $\mathcal{V}^i$ such that v' and v represent the same node of the network. Additionally, if $|P(\mathcal{V}^l, v')| < |P(\mathcal{V}^l, v)|$, then we say that v has a *high copy* in $\mathcal{V}^l$.

Before giving the pseudocode of procedure `TestRepetition`, we describe its high-level idea. Given a truncated colored view $(\mathcal{V}^l, f)$ and a node v in it as an input, the goal of this procedure is to return `true` if v has a high copy in $\mathcal{V}^l$. This is done by exploiting the only tool available to the algorithm: counting nodes in $\mathcal{V}^l$ of color α. Namely, for each node u in $P(\mathcal{V}^l, v)$, the procedure computes the distance from u to color α in $\mathcal{V}^l[u]$ and then it takes the maximum over all such distances, denoted by d'. If no node with the color α is observed in $\mathcal{V}^l[u]$, then d_u is set to ∞ which results in $d' = \infty$ and the procedure returns `false`. As proven below (cf. Lemma 3.1) checking whether the distance from the root to v in $\mathcal{V}^l$ is at least $2(k+1)(d'+1)$ is sufficient for procedure `TestRepetition` to return `true`. Note that there exist networks such that $|P(\mathcal{V}^l, v)|$ is smaller than $2(k+1)(d'+1)$ but v still has a high copy in $\mathcal{V}^l$. However, in such cases the procedure `TestRepetition` is unable to certify that and, thanks to later calls to `TestRepetition` in our algorithm, several descendants of v will be recognized to have high copies.

Procedure `TestRepetition`$(\mathcal{V}^l, f, v)$.

```
Input: Truncated colored view $(\mathcal{V}^l, f)$, $l \geq 1$, a node $v$ in $\mathcal{V}^l$.
Output: true or false.
  for all $u$ in $P(\mathcal{V}^l, v)$ do
    $d_u \leftarrow$ distance from $u$ to color $\alpha$ in $(\mathcal{V}^l[u], f)$;
    (Possibly $d_u = \infty$ if $\alpha$ does not appear in $(\mathcal{V}^l[u], f)$.)
  end for
  $d' \leftarrow \max\{d_u \mid u \text{ in } P(\mathcal{V}^l, v)\}$
  if $|P(\mathcal{V}^l, v)| \geq 2(k+1)(d'+1)$ then
    return true
  else
    return false
  end if
```

Lemma 3.1. *Let k be an upper bound on the size of the color α in the network and let v be a node in the network. Let d' be the maximum distance from a node u to color α in $\mathcal{V}^l(u)$ taken over all nodes u in $P(\mathcal{V}^l, v)$. If $|P(\mathcal{V}^l, v)| \geq 2(k+1)(d'+1)$, then v has a high copy in $\mathcal{V}^l$.*

Proof. Find an ancestor u of v in $\mathcal{V}^l$ such that the length of the path from the root to u is exactly $2(k+1)(d'+1)$. Let $u_0, \ldots, u_{k+1}$ be such nodes in the path $P(\mathcal{V}^l, u)$ that u_i is at distance $2i(d'+1)$ from the root. Note that $u_{k+1} = u$. For each $i \in \{1, \ldots, k+1\}$, let z_i be a node with color α at distance at most d' from u_i in $\mathcal{V}^l[u_i]$. Such a node z_i exists by definition of d' and by the fact that $d' \neq \infty$, for any $i \in \{1, \ldots, k+1\}$. The latter follows from $|P(\mathcal{V}^l, v)| \geq 2(k+1)(d'+1)$.

Since the size of color α is at most k, there exist $i, i' \in \{1, \ldots, k+1\}$, $i < i'$, such that z_i and $z_{i'}$ represent the same node of the network. Consider the sequence of ports that is a concatenation of the sequences of ports of the following paths: $P(\mathcal{V}^l, z_i)$ and $P(\mathcal{V}^l, z_{i'}, u_{k+1})$. This sequence of ports gives a valid path from the root to some node u' in the colored view $\mathcal{V}$, since z_i and $z_{i'}$ represent the same node of the network. We bound the length l' of $P(\mathcal{V}, u')$ as follows. Note that the first part of $P(\mathcal{V}, u')$ of length $|P(\mathcal{V}, z_i)|$ equals $P(\mathcal{V}, z_i)$. Thus, the length of this first part is at most

$$|P(\mathcal{V}^l, u_i)| + |P(\mathcal{V}^l, u_i, z_i)| \leq 2i(d'+1) + d'.$$

The length of the second part, i.e., the length of $P(\mathcal{V}^l, z_{i'}, u_{k+1})$ is at most

$$|P(\mathcal{V}^l, z_{i'}, u_{i'})| + |P(\mathcal{V}^l, u_{i'}, u_{k+1})| \leq d' + 2(k+1-i')(d'+1).$$

Therefore, since $i < i'$, we obtain

$$|P(\mathcal{V}, u')| \leq 2d' + 2i(d'+1) + 2(k+1-i')(d'+1) < 2(k+1)(d'+1).$$

Thus, u' belongs to $\mathcal{V}^{2(k+1)(d'+1)-1}$, or in other words, the node u that belongs to level $2(k+1)(d'+1)$ of $\mathcal{V}^l$ has a high copy in $\mathcal{V}^{2(k+1)(d'+1)-1}$. Since either $u = v$ or u is an ancestor of v, we obtain that v has a high copy in $\mathcal{V}^l$, as required. □

Corollary 3.1. *Consider a truncated colored view $(\mathcal{V}^l, f)$ and a node v in this view. If procedure* `TestRepetition` *returns* `true` *for input $\mathcal{V}^l$, f and v, then v has a high copy in $\mathcal{V}^l$.* □

We have proved that if procedure `TestRepetition` returns `true`, then this guarantees that the input node v has a high copy. However, for the correctness of our final algorithm we need to ensure that each infinite simple path in $\mathcal{V}$ originating from the root contains a node v that has a high copy, and that this fact will be detected by procedure `TestRepetition`. Moreover, in order to bound the time of our final algorithm, we need to estimate the distance from such v to the root, which is done in the next lemma.

MOST WIEDZY Pobrano z mostwiedzy.pl

Lemma 3.2. *Let $(\mathcal{V}^l, f)$ be a truncated colored view and let v be a node in it. If $l = 2(k+1)(D+1) + D$ and v belongs to level $2(k+1)(D+1)$ of $\mathcal{V}^l$, then procedure* `TestRepetition` *executed for $\mathcal{V}^l$, f and v returns* `true`*.*

Proof. For each node u of the path $P(\mathcal{V}^l, v)$, the node of the network represented by it is at distance at most D (in the network) from some node with color α. This implies that $d_u \le D$ for each such node u. Thus, $d' \le D$. We obtain that

$$|P(\mathcal{V}^l, v)| = 2(k+1)(D+1) \ge 2(k+1)(d'+1),$$

which implies that procedure `TestRepetition` returns true. □

Before describing the next procedure, we introduce some more notation. A truncated colored view $(\mathcal{V}^l, f)$ *covers the network* if, for each node x of the network, there exists a node u of $\mathcal{V}^l$ such that u represents x. We define procedure `communicate` which sends the currently acquired truncated colored view $(\mathcal{V}^t, f)$ to all neighbors and receives the messages containing currently acquired colored views of the same depth t from all neighbors. Note that after all nodes have performed `communicate` t times, each node can compute its truncated colored view $\mathcal{V}^t$.

We now describe procedure `ComputeView`. Again, we start with an informal description. In each iteration, the 'while' loop increments the depth of the view currently stored at the executing node. This is done by communicating with each neighbor. The crucial part is to decide when to stop. At some point, procedure `ComputeView` detects that the currently possessed truncated colored view $(\mathcal{V}^l, f)$ covers the network and this view is then returned. The above is achieved (see Lemma 3.3 below for a proof) by maintaining a set M, that is initially empty, consisting of nodes having high copies. Procedure `ComputeView` stops when each leaf of $\mathcal{V}^l$ is in M, which guarantees that $\mathcal{V}^l$ covers the network, as required.

Procedure `ComputeView`.

```
Input: None.
Output: Truncated colored view $(\mathcal{V}^l, f)$.
  $l \leftarrow 1$
  $M \leftarrow \emptyset$
  while there exists a leaf in $\mathcal{V}^l$ that is not in $M$ do
    for all $v$ in $\mathcal{V}^l$ do
      if TestRepetition($\mathcal{V}^l, v$) returns true then
        Add to $M$ the node $v$ and all its descendants in $\mathcal{V}^l$.
      end if
    end for
    communicate   {This extends $(\mathcal{V}^l, f)$ to $(\mathcal{V}^{l+1}, f)$.}
    $l \leftarrow l+1$
  end while
  return $(\mathcal{V}^l, f)$
```

Lemma 3.3. *Procedure* `ComputeView` *returns the truncated colored view $(\mathcal{V}^l, f)$ of the executing node, such that $\mathcal{V}^l$ covers the network and $l \le 2(k+1)(D+1) + D$.*

Proof. Lemma 3.2 implies that there exists $l \le 2(k+1)(D+1)+D$ such that all leaves of $\mathcal{V}^l$ are in M and hence the number of iterations of the 'while' loop of procedure `TestRepetition` is at most l. The latter relies on the fact that if a node having a high copy is detected by procedure `TestRepetition`, then procedure `ComputeView` adds to M this node with all its descendants in the current truncated view.

Consider the view $\mathcal{V}$ of the executing node, and let u be any node in $\mathcal{V}$. We argue that u has a copy in $\mathcal{V}^l$. Suppose for a contradiction that this is not the case and select u to be a node that does not have a copy in $\mathcal{V}^l$ and is closest to the root in $\mathcal{V}$. Let l' be the level of u in $\mathcal{V}$. (Clearly $l' > l$.) Since $\mathrm{lev}_l(\mathcal{V}^l) \subseteq M$, there exists an ancestor v of u such that $v \in \mathrm{lev}_i(\mathcal{V}^l)$, $i \le l$, and procedure `TestRepetition` returns `true` when executed for $\mathcal{V}^l$, f and v. By Corollary 3.1, v has a copy v' in $\mathcal{V}^{i-1}$. Thus, u has a copy u' in $\mathcal{V}^{l'-1}$. But then, by the minimality of l', u' has a copy in $\mathcal{V}^l$, which is also a copy of u. This is a contradiction because, by definition, u and u' represent the same node of the network. □

Procedure `ComputeQuotientGraph` computes the colored quotient graph $(\mathcal{Q}, g)$ of the network, provided that a colored view $(\mathcal{V}^l, f)$ that covers the network is given as an input. This is done by finding the minimum index i such that $\Pi_{i-1} = \Pi_i$. Note that this requires that each node learn its colored view till depth $l+i$, by exchanging messages with its neighbors.

We prove the following.

Lemma 3.4. *Let $(\mathcal{V}^l, f)$ be the truncated colored view computed by procedure* `ComputeView`*. Procedure* `ComputeQuotient-Graph` *called for $(\mathcal{V}^l, f)$ correctly computes the colored quotient graph of the network.*

MOST WIEDZY Pobrano z mostwiedzy.pl

Procedure `ComputeQuotientGraph`$(\mathcal{V}^l, f)$.

```
Input: Truncated colored view $(\mathcal{V}^l, f)$, $l \geq 1$.
Output: The colored quotient graph $(\mathcal{Q}, g)$.
  $\Pi_{-1} \leftarrow \emptyset$
  $\Pi_0 \leftarrow \{(\mathcal{V}^0(v), f) \mid v \in \mathcal{V}^l\}$
  $i \leftarrow 0$
  while $\Pi_i \neq \Pi_{i-1}$ do
    $i \leftarrow i + 1$
    communicate
    Compute $\Pi_i$
  end while
  Compute the labeled quotient graph $(\mathcal{Q}, g)$ using $\Pi_i$ and $\mathcal{V}^{l+i}$
  return $(\mathcal{Q}, g)$
```

Proof. By Lemma 3.3, $\mathcal{V}^l$ covers the network. Thus, Π_0 contains all nodes of the network. By Proposition 2.3, the partition Π_i obtained in the last iteration of the 'while' loop of procedure `ComputeQuotientGraph` equals Π. Hence, this partition Π_i is the set of all nodes of the colored quotient graph. This implies that the quotient graph can be computed on the basis of Π_i and $\mathcal{V}^l$ as follows. The color of a node of the quotient graph is set to the color of its elements in the truncated colored view. The edges and port numbers are added as in the definition of the quotient graph. □

In the formulation of our main algorithm we will use the following integer that each node can compute once it has the colored quotient graph $(\mathcal{Q}, g)$. For any node u of the quotient graph, let ξ_u be the sum of running times of procedures `ComputeView` and `ComputeQuotientGraph`$(\mathcal{V}^l, f)$, where $(\mathcal{V}^l, f)$ is computed by `ComputeView`. Let Ξ be the maximum of ξ_u over all nodes u of the quotient graph.

We are ready to state our main algorithm for solving the problems LE and TOP. We formulate it as a single procedure, since all steps leading to the computation of the colored quotient graph are identical in both cases.

Algorithm `Solve-LE-and-TOP`(k, α).

```
Input: An upper bound $k$ on the size of color $\alpha$.
Output: For LE — a sequence of port numbers leading from the executing node to the leader, or unsolvable, if leader election is impossible. For TOP —
  the topology of the network, or unsolvable, if topology recognition is impossible.
  $(\mathcal{V}^l, f) \leftarrow$ ComputeView
  $(\mathcal{Q}, g) \leftarrow$ ComputeQuotientGraph$(\mathcal{V}^l, f)$
  Perform communicate $\Xi$ times.
  if the number of nodes with color $\alpha$ in $(\mathcal{Q}, g)$ is at most $\lfloor k/2 \rfloor$ and $\mathcal{Q}$ is not a tree then
    return unsolvable
  else
    For LE — return a sequence of port numbers of the path $P(\mathcal{V}^l, v)$, where $v$ (which is the leader) corresponds to the node of $(\mathcal{Q}, g)$ whose colored
    view is lexicographically smallest (if the executing node is the leader, then the path is empty).
    For TOP — return $(\mathcal{Q}, g)$.
  end if
```

Theorem 3.1. *If a bound k on the size of a given color α is provided as an input, then Algorithm* `Solve-LE-and-TOP` *correctly solves problems* LE *and* TOP *in time $O(kD + D\log(n/D))$, where n is the size of the network and D is its diameter.*

Proof. First note that Algorithm `Solve-LE-and-TOP` correctly computes the colored quotient graph. Indeed, by Lemma 3.4, Algorithm `Solve-LE-and-TOP` obtains the colored quotient graph $(\mathcal{Q}, g)$ as a result of the call to procedure `ComputeQuotientGraph`. This is ensured by the fact that each node performs at least Ξ calls to `communicate` and therefore every node u can compute $\mathcal{V}^{\xi_u}$, which is enough to compute the colored quotient graph, by definition of ξ_u.

Once the colored quotient graph is computed by all nodes of the network, the correctness of Algorithm `Solve-LE-and-TOP` essentially follows from [43]. For completeness we include the short argument.

Assume that the colored quotient graph has at most $\lfloor k/2 \rfloor$ nodes with color α and it is not a tree (i.e., a graph that has cycles, multiple edges or self-loops). Then, topology recognition cannot be solved since there exist two non-isomorphic networks of size $2\lfloor k/2 \rfloor$ having $(\mathcal{Q}, g)$ as a quotient graph. Any potential topology recognition algorithm in these networks must have the same execution for each pair of nodes with the same colored views and thus such an algorithm must be incorrect. Hence, Algorithm `Solve-LE-and-TOP` correctly returns `unsolvable` for the problem TOP. For the problem of leader election, take any network of size $2\lfloor k/2 \rfloor$ having $(\mathcal{Q}, g)$ as a quotient graph. Two distinct nodes u and v in this network have the same colored view. Thus, any potential leader election algorithm incorrectly elects at least two leaders in this network. Hence, Algorithm `Solve-LE-and-TOP` correctly returns `unsolvable` for the problem LE as well.

Otherwise, i.e., if the colored quotient graph has more than $\lfloor k/2 \rfloor$ nodes with color α or it is a tree, then the network is isomorphic to $(\mathcal{Q}, g)$ and hence Algorithm `Solve-LE-and-TOP` gives a correct solution to the problem TOP. As for leader election, each node has a unique colored view under this assumption. Hence, the node with the lexicographically smallest colored view is unambiguously elected as the leader by each node.

MOST WIEDZY Pobrano z mostwiedzy.pl

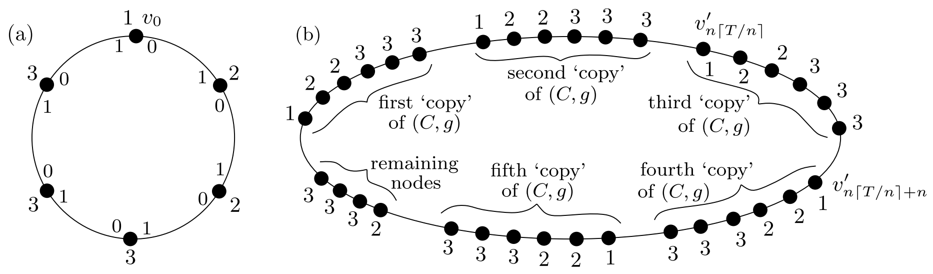

Fig. 1. (a) an example of a ring (C,g) obtained from $(x_1,x_2,x_3)=(1,2,3)$, $n=6$; (b) the corresponding ring (C',g') constructed for $T\in\{7,\dots,12\}$ and $(x'_1,x'_2,x'_3)=(5,11,18)$, $n'=34$.

It remains to bound the time of computation. It is at most $l+i+\Xi$, where l and i are the numbers of iterations of the 'while' loop of procedure `ComputeView` and `ComputeQuotientGraph`, respectively. By Lemma 3.2 and the formulation of procedure `ComputeView`, $l\in O(kD)$. By Proposition 2.1, $i\in O(D\log(n/D))$. Thus, by definition, $\Xi\in O(kD+D\log(n/D))$, which completes the proof. □

Corollary 3.2. *Let (G,f) be a colored network and let k be a bound on the size of color α. If all nodes of (G,f) have pairwise different colored views and the size of color α is strictly greater than $\lfloor k/2\rfloor$, then topology recognition and leader election are possible in G.* □

4. Negative results

In this section we present our negative results. The first of them is an impossibility result indicating that if no upper bound on the size of any color is given to nodes, then problems TOP and LE must have answer "unsolvable", even if other restrictions on the possible sizes of colors are known. In order to express this result in full generality, we formalize such possible restrictions as a set $R\subseteq\mathbb{N}^r$, where $\mathbb{N}$ denotes the set of positive integers and r is the number of colors. We translate the property that no upper bound on the size of any color is known, to the statement that for any point $(x_1,x_2,\dots,x_r)\in\mathbb{N}^r$ there exists a point $(y_1,y_2,\dots,y_r)$ in the restriction set R, such that $y_i\geq x_i$, for all $i\leq r$. Such a set R will be called *unbounded*. (An example of an input defining an unbounded restriction set is: there are three colors and the sizes of all of them are prime integers.) To make the impossibility result even stronger, we prove that it holds even for a very simple class of networks: on rings.

Proposition 4.1. *Let (C,g) be any colored ring. Consider an input I defining an unbounded restriction set R. Then, problems* TOP *and* LE *must have answer "unsolvable".*

Proof. Take any algorithm, call it A, that correctly solves problem TOP or problem LE in any colored ring under input I. Let r be the number of different colors that appear in the network, and let x_i be the size of color i, $i\in\{1,\dots,r\}$. Denote the nodes of C by $v_0,\dots,v_{n-1}$, where $n=x_1+\cdots+x_r$ and v_i is adjacent to $v_{(i+1)\bmod n}$, for each $i\in\{0,\dots,n-1\}$.

Let v_0 be any node of C. Let T be the time after which A produces the answer. Since the restriction set R is unbounded, there exists $(x'_1,\dots,x'_r)\in R$ such that $x'_i\geq 2x_i\lceil T/n\rceil+x_i$ for each $i\in\{1,\dots,r\}$. We construct a colored ring (C',g') on $n'=x'_1+\cdots+x'_r$ nodes $v'_0,\dots,v'_{n'-1}$ such that:

- v'_i is adjacent to $v'_{(i+1)\bmod n'}$, $i\in\{0,\dots,n'-1\}$,
- the port numbers of the edge $\{v'_i,v'_{i+1}\}$ at v'_i and v'_{i+1} are equal to the port numbers of the edge $\{v_{i\bmod n},v_{(i+1)\bmod n}\}$ at $v_{i\bmod n}$ and $v_{(i+1)\bmod n}$, respectively, $i\in\{0,\dots,2n\lceil T/n\rceil+n-1\}$, in C,
- the port numbers of the remaining edges are set arbitrarily in a way that guarantees proper port labeling.

Moreover, the colors are assigned to the nodes of C' as follows:

$$g'(v'_i)=g(v_{i\bmod n}),\quad i\in\{0,\dots,2n\lceil T/n\rceil+n-1\},$$

and the remaining nodes, i.e., the ones in $X=\{v'_{2n\lceil T/n\rceil+n},\dots,v'_{n'}\}$, receive colors in any way that ensures that the size of color $j\in\{1,\dots,r\}$ in C' is x'_j. See Fig. 1 for an example of the construction of (C',g'). The colored view of depth T of v_0 in (C,g) is the same as the colored views of depth T of $v'_{n\lceil T/n\rceil}$ and of $v'_{n\lceil T/n\rceil+n}$ in (C',g') because $T\leq n\lceil T/n\rceil$.

Consider the problem TOP. If the answer produced by A in (C,g) is "unsolvable", then the lemma follows. Hence, we may assume that A returns the topology of (C,g). By Proposition 2.4, the algorithm A executed by $v'_{n\lceil T/n\rceil}$ in (C',g') stops after time T and produces the same answer as for v_0 in (C,g). Thus, A must also return the topology of (C,g) when executed by $v'_{n\lceil T/n\rceil}$ in (C',g'). Since C and C' are of different sizes, we obtain a contradiction, as required.

MOST WIEDZY Pobrano z mostwiedzy.pl

Next consider the problem LE. If the answer produced by A in (C, g) is "unsolvable", then the lemma follows. Hence, we may assume that A elects a leader in (C, g). By Proposition 2.4, the algorithm A executed by $v'_{n\lceil T/n\rceil}$ and $v'_{n\lceil T/n\rceil+n}$ in (C', g') stops after time T and produces the same answer as in (C, g). Thus, these two nodes elect different leaders — a contradiction. □

We next turn attention to the issue of time needed to solve problems TOP and LE, assuming that an upper bound k on the size of some color is given to all nodes. We give a lower bound showing that the time $O(kD + D\log(n/D))$ of our algorithm is optimal. We first construct a class of networks, for which time $\Theta(kD)$ cannot be improved.

Proposition 4.2. *Let k, $D \le n$ be arbitrary positive integers. There exists a network of size $\Theta(n)$ and diameter $\Theta(D)$, whose nodes, if they are given as input an upper bound k on the size of one color α and have no other information on the network, need time $\Omega(kD)$ to solve the problems* TOP *and* LE*.*

Proof. For any n' and $d < n'/2$, we define a n'-node graph $G(n', d)$ (called a chordal ring). Denote by $v_0, \dots, v_{n'-1}$ the nodes of $G(n', d)$. For each $i \in \{0, \dots, n'-1\}$ and $j \in \{1, \dots, d\}$, let $\{v_i, v_{i+j}\}$ be an edge of $G(n', d)$, where the port number of this edge at v_i is $j-1$ and the port number at v_{i+j} is $d+j-1$. Note that the diameter of $G(n', d)$ is $\Theta(n'/d)$.

Since our result holds asymptotically, we may assume that $D \ge 3$ and $k \ge 12$. Consider an algorithm A for solving problem TOP or LE in any network. Let $d = \lfloor n/D \rfloor$. (Note that $D \ge 3$ implies $d < n/2$, as required in the construction of $G(n, d)$.) Let u be any node of $G(n, d)$. Let $(G(n, d), g)$ be the colored network in which $g(u) = \alpha$, and all other nodes of $G(n, d)$ have the same color α', different from α. Note that the diameter of $G(n, d)$ is $\Theta(D)$. We run the algorithm A in the colored network $(G(n, d), g)$. We argue that A should run for time $\Omega(kD)$. Suppose for a contradiction that A stops and produces an answer in round $T < \lfloor k/3 \rfloor D$.

Consider the network $G(kn, d)$ on the set of nodes $\{v'_0, \dots, v'_{kn-1}\}$. Construct a network G' by adding a pendant edge to the node v'_0 of $G(kn, d)$, i.e., add an extra node of degree 1 and attach it to v'_0. Let $u' = v'_{\lfloor k/2 \rfloor n}$. Let (G', g') be a colored network in which $g'(v'_{nj}) = \alpha$ for each $j \in \{0, \dots, k-1\}$, while all other nodes of this network have color α'. By construction of G', the distance between u' and v'_0 in G' is greater than T. Thus, the truncated colored view $\mathcal{V}^T(u')$ in (G', g') is the same as the truncated colored view $\mathcal{V}^T(u)$ in $(G(n, d), g)$. Now, we run A in the colored network (G', g').

Suppose that A is an algorithm solving the problem TOP. By Proposition 2.4, A executed on u and u' produces the same answer to problem TOP in $G(n, d)$ and G', respectively. If the answer on u is the topology of $G(n, d)$, then we immediately have a contradiction since the two networks are not isomorphic. On the other hand, if the answer is "unsolvable", then this answer is incorrect for G'. The latter is due to Corollary 3.2.

Let now A be an algorithm solving the problem LE. Let $v' = v'_{\lfloor kn/2 \rfloor + n}$. The distance from v' to v'_0 in G' is at least $(\lfloor k/2 \rfloor - 1)\lfloor n/d \rfloor \ge \lfloor n/d \rfloor \cdot k/3 \ge Dk/3 \ge T$ for $k \ge 12$. Thus, $\mathcal{V}^T(u') = \mathcal{V}^T(v')$. By Proposition 2.4, A produces the same output at u' and v' after time T. By Corollary 3.2, the algorithm A cannot output "unsolvable" because all nodes of G' have unique views and the size of color α is k in (G', g'). Thus, u' and v' elect different leaders — a contradiction. □

The other part of our lower bound follows from [14].

Proposition 4.3. *Let $D \le n$ be arbitrary positive integers. There exists a network of size $\Theta(n)$ and diameter $\Theta(D)$, whose nodes need time at least $\Omega(D\log(n/D))$ to solve problems* TOP *and* LE*, even if all nodes have the same color and they are given the size and the diameter of the network.*

Proof. By Proposition 2.2, there exists a network G of size $\Theta(n)$ and diameter $\Theta(D)$, with nodes u and v, both with unique views, such that $\mathcal{V}(u) \ne \mathcal{V}(v)$ but $\mathcal{V}^{h'}(u) = \mathcal{V}^{h'}(v)$, for some $h' \in \Theta(D\log(n/D))$. Thus, by Proposition 2.4, any algorithm A that stops after at most h' steps and produces an answer to problem LE, gives the same answer at u and v. Thus, this answer must be "unsolvable". (Otherwise, two distinct leaders would be elected.) However, since the nodes of G have pairwise different views and the size of the network is known, Corollary 3.2 implies that leader election is possible in this network. Thus, any algorithm solving problem LE needs time $\Omega(D\log(n/D))$ in G.

Now consider the problem TOP and let A be any algorithm solving this problem. Suppose for a contradiction that A stops after at most h' steps. By Proposition 2.2, there exists a network G', different than G, with the same size and diameter as G, with a node u', such that $\mathcal{V}^{h'}(u') = \mathcal{V}^{h'}(u)$. Thus, by Proposition 2.4, A returns the same answer at u and u'. Since G and G' are different, this answer must be "unsolvable". Since all nodes in G have pairwise different views, by Proposition 2.4 (where k is taken to be the size of G), TOP is possible in G — a contradiction. □

Theorem 3.1, together with Propositions 4.2 and 4.3, imply the following corollary showing that our algorithm is time-optimal.

Corollary 4.1. *The optimal time to solve problems* TOP *and* LE *on n-node networks with diameter D, assuming that nodes know only an upper bound k on the size of a given color, is $\Theta(kD + D\log(n/D))$.*

MOST WIEDZY Pobrano z mostwiedzy.pl

5. Conclusion

We showed that nodes of a colored network can solve problems TOP and LE, if they are given an upper bound on the number of nodes of a given color, and we studied the time of solving these problems in the $\mathcal{LOCAL}$ model, under this assumption.

Notice that the synchronous behavior of the $\mathcal{LOCAL}$ model can be easily reproduced in an asynchronous network, by defining, for each node u separately, an asynchronous round i consisting of the following actions of this node: node u performs local computations, then sends messages stamped with integer i to all its neighbors, and waits for messages stamped i from all neighbors. In order to implement this, every node must send a message with all consecutive stamps, until termination, some of the messages possibly empty. Our results concerning time of solving problems TOP and LE can be translated for asynchronous networks by replacing "the number of rounds" by "the maximum number of asynchronous rounds, over all nodes".

Let D be the diameter of the network. If nodes have distinct labels, then time $D+1$ in the $\mathcal{LOCAL}$ model is enough to solve any problem solvable on a given network, as after this time all nodes solve topology recognition. By contrast, in our scenario of colored nodes, time $D+1$ is often not enough, for example to elect a leader, or to perform topology recognition, even if these tasks are feasible. This is due to the fact that after time $t \leq D+1$ each node may learn only all colored paths of length t originating at it. Acquiring this information does not imply getting a picture of the radius t colored neighborhood of the node. This is because a node v may not know if two paths originating at it have the same other endpoint or not. We showed that these ambiguities may force time much larger than D to solve problems TOP and LE.

As it is always assumed in the $\mathcal{LOCAL}$ model, we allowed arbitrarily large messages to be sent in each round. Bounding the size of messages to logarithmic in the size of the network, as it is assumed in the alternative $\mathcal{CONGEST}$ model, would likely have an important impact on the time of solving problems TOP and LE. Hence an interesting open question is to establish the best time of solving these problems in the latter model.

References

[1] D. Angluin, Local and global properties in networks of processors, in: Proc. 12th Annual ACM Symposium on Theory of Computing, STOC 1980, 1980, pp. 82–93.
[2] H. Attiya, A. Bar-Noy, D. Dolev, D. Koller, D. Peleg, R. Reischuk, Renaming in an asynchronous environment, J. ACM 37 (1990) 524–548.
[3] H. Attiya, M. Snir, M. Warmuth, Computing on an anonymous ring, J. ACM 35 (1988) 845–875.
[4] H. Attiya, M. Snir, Better computing on the anonymous ring, J. Algorithms 12 (1991) 204–238.
[5] B. Awerbuch, Optimal distributed algorithms for minimum weight spanning tree, counting, leader election and related problems, in: Proc. 19th Annual ACM Symposium on Theory of Computing, STOC 1987, 1987, pp. 230–240.
[6] B. Awerbuch, O. Goldreich, D. Peleg, R. Vainish, A trade-off between information and communication in broadcast protocols, J. ACM 37 (1990) 238–256.
[7] P. Boldi, S. Shammah, S. Vigna, B. Codenotti, P. Gemmell, J. Simon, Symmetry breaking in anonymous networks: characterizations, in: Proc. 4th Israel Symposium on Theory of Computing and Systems, ISTCS 1996, 1996, pp. 16–26.
[8] P. Boldi, S. Vigna, Computing anonymously with arbitrary knowledge, in: Proc. 18th ACM Symp. on Principles of Distributed Computing, PODC 1999, 1999, pp. 181–188.
[9] J.E. Burns, A formal model for message passing systems, Tech. Report TR-91, Computer Science Department, Indiana University, Bloomington, September 1980.
[10] J. Chalopin, Local computations on closed unlabelled edges: the election problem and the naming problem, in: Proc. 31st Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM 2005, 2005, pp. 82–91.
[11] J. Chalopin, S. Das, A. Kosowski, Constructing a map of an anonymous graph: applications of universal sequences, in: Proc. 14th International Conference on Principles of Distributed Systems, OPODIS 2010, 2010, pp. 119–134.
[12] J. Chalopin, A.W. Mazurkiewicz, Y. Métivier, Labelled (hyper)graphs, negotiations and the naming problem, in: Proc. 4th International Conference on Graph Transformations, ICGT 2008, 2008, pp. 54–68.
[13] J. Chalopin, Y. Métivier, Election and local computations on edges, in: Proc. Foundations of Software Science and Computation Structures, FoSSaCS 2004, 2004, pp. 90–104.
[14] D. Dereniowski, A. Kosowski, D. Pajak, Distinguishing views in symmetric networks: a tight lower bound, Theoret. Comput. Sci. 582 (2015) 27–34.
[15] D. Dereniowski, A. Pelc, Drawing maps with advice, J. Parallel Distrib. Comput. 72 (2012) 132–143.
[16] D. Dereniowski, A. Pelc, Leader election for anonymous asynchronous agents in arbitrary networks, Distrib. Comput. 27 (2014) 21–38.
[17] K. Diks, E. Kranakis, A. Malinowski, A. Pelc, Anonymous wireless rings, Theoret. Comput. Sci. 145 (1995) 95–109.
[18] S. Dobrev, A. Pelc, Leader election in rings with nonunique labels, Fund. Inform. 59 (2004) 333–347.
[19] P. Flocchini, E. Kranakis, D. Krizanc, F.L. Luccio, N. Santoro, Sorting and election in anonymous asynchronous rings, J. Parallel Distrib. Comput. 64 (2004) 254–265.
[20] G.N. Fredrickson, N.A. Lynch, Electing a leader in a synchronous ring, J. ACM 34 (1987) 98–115.
[21] E. Fusco, A. Pelc, How much memory is needed for leader election, Distrib. Comput. 24 (2011) 65–78.
[22] E. Fusco, A. Pelc, Knowledge, level of symmetry, and time of leader election, in: Proc. 20th Annual European Symposium on Algorithms, ESA 2012, in: Lecture Notes in Comput. Sci., vol. 7501, 2012, pp. 479–490.
[23] E. Fusco, A. Pelc, R. Petreschi, Use knowledge to learn faster: topology recognition with advice, in: Proc. 27th International Symposium on Distributed Computing, DISC 2013, in: Lecture Notes in Comput. Sci., vol. 8205, 2013, pp. 31–45.
[24] C. Glacet, A. Miller, A. Pelc, Time vs. information tradeoffs for leader election in anonymous trees, in: Proc. 27th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, 2016, pp. 600–609.
[25] M.A. Haddar, A.H. Kacem, Y. Métivier, M. Mosbah, M. Jmaiel, Electing a leader in the local computation model using mobile agents, in: Proc. 6th ACS/IEEE International Conference on Computer Systems and Applications, AICCSA 2008, 2008, pp. 473–480.
[26] J. Hendrickx, Views in a graph: to which depth must equality be checked?, IEEE Trans. Parallel Distrib. Syst. 25 (2014) 1907–1912.
[27] D.S. Hirschberg, J.B. Sinclair, Decentralized extrema-finding in circular configurations of processes, Commun. ACM 23 (1980) 627–628.
[28] T. Jurdzinski, M. Kutylowski, J. Zatopianski, Efficient algorithms for leader election in radio networks, in: Proc., 21st ACM Symp. on Principles of Distributed Computing, PODC 2002, 2002, pp. 51–57.

MOST WIEDZY Pobrano z mostwiedzy.pl

[29] D. Kowalski, A. Pelc, Leader election in ad hoc radio networks: a keen ear helps, in: Proc. 36th International Colloquium on Automata, Languages and Programming, ICALP 2009, in: Lecture Notes in Comput. Sci., vol. 5556, 2009, pp. 521–533.
[30] E. Kranakis, Symmetry and computability in anonymous networks: a brief survey, in: Proc. 3rd Int. Conf. on Structural Information and Communication Complexity, 1997, pp. 1–16.
[31] E. Kranakis, D. Krizanc, J. van der Berg, Computing Boolean functions on anonymous networks, Inform. and Comput. 114 (1994) 214–236.
[32] G. Le Lann, Distributed systems – towards a formal approach, in: Proc. IFIP Congress, North Holland, 1977, pp. 155–160.
[33] N.A. Lynch, Distributed Algorithms, Morgan Kaufmann Publ., Inc., 1996.
[34] A. Miller, A. Pelc, Election vs. selection: two ways of finding the largest node in a graph, CoRR, arXiv:1411.1319, 2014.
[35] K. Nakano, S. Olariu, Uniform leader election protocols for radio networks, IEEE Trans. Parallel Distrib. Syst. 13 (2002) 516–526.
[36] N. Norris, Universal covers of graphs: isomorphism to depth $N-1$ implies isomorphism to all depths, Discrete Appl. Math. 56 (1995) 61–74.
[37] D. Peleg, Distributed computing, a locality-sensitive approach, in: SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, 2000.
[38] G.L. Peterson, An $O(n \log n)$ unidirectional distributed algorithm for the circular extrema problem, ACM Trans. Program. Lang. Syst. 4 (1982) 758–762.
[39] N. Sakamoto, Comparison of initial conditions for distributed algorithms on anonymous networks, in: Proc. 18th ACM Symp. on Principles of Distributed Computing, PODC 1999, 1999, pp. 173–179.
[40] D.E. Willard, Log-logarithmic selection resolution protocols in a multiple access channel, SIAM J. Comput. 15 (1986) 468–477.
[41] M. Yamashita, T. Kameda, Computing on anonymous networks, in: Proc. 7th ACM Symp. on Principles of Distributed Computing, PODC 1988, 1988, pp. 117–130.
[42] M. Yamashita, T. Kameda, Electing a leader when processor identity numbers are not distinct, in: Proc. 3rd Workshop on Distributed Algorithms, WDAG 1989, in: Lecture Notes in Comput. Sci., vol. 392, 1989, pp. 303–314.
[43] M. Yamashita, T. Kameda, Computing on anonymous networks: part I – characterizing the solvable cases, IEEE Trans. Parallel Distrib. Syst. 7 (1996) 69–89.

MOST WIEDZY Pobrano z mostwiedzy.pl