Computing on Anonymous Networks with Sense of Direction*

Paola Flocchini† Alessandro Roncato‡ Nicola Santoro§

Abstract

Sense of direction refers to a set of global consistency constraints of the local labeling of the edges of a network. Sense of direction has a large impact on the communication complexity of many distributed problems. In this paper, we study the impact that sense of direction has on *computability* and we focus on anonymous networks. We establish several results. In particular, we prove that with weak sense of direction, the intuitive *knowledge-computability hierarchy* between levels of a priori structural knowledge collapses. A powerful implication is the formal proof that shortest path routing is possible in anonymous networks with sense of direction. We prove that weak sense of direction is computationally stronger than topological awareness. We also consider several fundamental problems; for each, we provide a complete characterization of the anonymous networks on which it is computable with sense of direction.

1 Introduction

A *distributed system* is a collection of autonomous entities communicating by the exchange of finite sequence of bits (called *messages*). The communication topology of the system can be described as a labeled graph (G, λ) where nodes correspond to the system entities, edges represent pairs of neighboring entities (i.e., entities which can

*Research supported in part by a grant (#A2415) from NSERC, a NATO Advance Research Fellowship, and an international scholarship by CNR. A preliminary version of this paper will appear in the Proceedings of the *3rd International Colloquium on Structural Information and Communication Complexity.*

†Département d' Informatique et de Recherche Opérationnelle, Université de Montréal, Montréal, Quebec, H3C 3J7, Canada. (flocchin@iro.umontreal.ca)

‡Facoltà di Scienze Matematiche Fisische e Naturali, Corso di Laurea in Informatica, Via Torino 155, 30173 Mestre, Italia. (roncato@dsi.unive.it)

§Center for Parallel and Distributed Computing, School of Computer Science, Carleton University, Ottawa K1S 5B6, Canada. (santoro@scs.carleton.ca)

communicate directly), and the label associated by a node to an incident edges represent the the corresponding port number. If the entities of a system (G, λ) do not have globally unique identifiers, the system is said to be *anonymous*.

A central problem in distributed computing is undoubtely the study of the interplay between computability (i.e., class of solvable problems) and system structure (i.e., specific assumptions or limitations of the distributed system in which the computation has to take place).

A large amount of research has been devoted to the study of *computability* in anonymous systems. Clearly, which problems can be solved in anonymous systems depend on many factors including the structural properties of G as well as on the amount of a priori knowledge about G available to the entities. Most of the investigations have focused on systems with a specific topology: rings (e.g., [2, 5, 20, 23]), meshes and tori [3, 19], hypercubes [13], Cayley graphs (e.g., [14]).

More recently, Yamashita and Kameda have assumed a different viewpoint [29]: they considered four problems (leader election, edge-election, spanning-tree construction, topology recognition) and four types of structural information (topological awareness, size of graph, upperbound on size, no information); then, for each problem and type of information, they studied the class of graphs on which that problem was solvable with that type of a priori knowledge. Their characterization did not make any assumption on the labeling λ other than it is a *local orientation*, i.e., a node assigns distinct labels to the edges incident to it. On the other hand, it is well known that, if λ satisfies the set of consistency constraints called *sense of direction* [9], the communication complexity of several distributed problem is drastically reduced (e.g., see [7, 12, 16, 17, 18, 25]). Because of its impact on complexity, a great deal of research has been devoted to the analysis of properties of $\mathcal{SD}$ (e.g., see [4, 11, 24, 27]).

Little is known on the impact of sense of direction in anonymous networks. Most of the results are implicit and follow from observing that the existing work on computability in *specific* anonymous networks consider systems with sense of direction (e.g., hypercubes with the traditional "dimensional" labeling, rings with the "left-right" orientation, etc.). More recently, it has been shown that, in anonymous networks, the presence of sense of direction reduces the message complexity of the broadcast and depth-first traversal in *any* network [8]. In spite of its theoretical importance and possible practical implications, the fundamental issue of computability in anonymous networks with sense of direction has never been investigated before. Neither it has been studied the relationship in anonymous systems between sense of direction and other forms of structural information (e.g, topological awareness, metric information, etc.).

In this paper we start to fill this gap. We study the problem of computing on anonymous systems in presence of (weak) *sense of direction*.

We first focus on the levels of a priori structural knowledge studied in [29]: No Information, Upper Bound on Network Size, Knowledge of Network Size, and Topological Awareness, as well as a higher level called Complete Topological Knowledge. With respects to those levels, there exists an intuitive *knowledge-computability hierarchy*: the class of graphs on which a problem is solved with a given level of knowledge includes those on which the same problem is solved with a *lower* level of topological knowledge. Using both the notion of *view* (introduced in [29]), as well as the novel notion of *surrounding*, we prove that

> *the knowledge-computability hierarchy collapses in anonymous systems with weak sense of direction.*

In fact, we prove the stronger result that for *any* problem P and *any* structural knowledge $\mathcal{K}$, the class $W_{\mathcal{K}}(P)$ of anonymous graphs in which with any weak sense of direction P is always solvable with knowledge $\mathcal{K}$ *includes* the class $W_{complete}(P)$.
We then prove that

> *weak sense of direction has exactly the same computational power than complete topological knowledge.*

In fact we show that, for any problem P, the class of graphs $D_{complete}(P)$ on which P can be solved with complete topological knowledge *coincides* with the class of graphs $W(P)$ on which P is solvable with weak sense of direction (and no other a priori structural knowledge).

Besides their theoretical relevance for the understanding of the interplay between knowledge and computability, the above two results have many important implications. For example, they formally prove that "with sense of direction you don't need names for routing":

> *in any anonymous network with any weak sense of direction it is possible to do shortest-path routing.*

We then focus on the computability relationship in anonymous networks between sense of direction and topological awareness. We consider the problems studied in [29]: Leader Election, Edge Election, Spanning Tree Construction, Topology Recognition, as well as the more complex problem of Complete Topology Recognition. First, we characterize the classes of graphs on which it is possible to solve them with sense of direction. Using these problems as test cases, we then investigate the relationship between computability with sense of direction and computability with topological awareness. We prove that

> *sense of direction is strictly more powerful than topological awareness.*

More precisely, for the problem of leader election, edge election and spanning-tree construction we prove that there exist graphs in which these problems can be solved with *any* weak sense of direction; without weak sense of direction, none of these problems is solvable even in presence of topological awareness. For the Election problem, we provide a complete characterization of this class of graphs in terms of symmetry and singularity.

A similar result is shown to hold for the complete topology recognition problem; also in this case provide a characterization of the class of graphs in which topological awareness is sufficient to solve the problem.

The paper is organized as follows. In the next section, we give the basic definitions. In Section 3, we introduce the notion of view and surrounding, and we give some properties. In Section 4, we analyze the relationship between view and surrounding and we study the computational power of sense of direction. In Section 5, we analyze the computational relations between the topological awareness and sense of direction proving that sense of direction is strictly more powerful than topological awareness. Finally, in Section 6, we discuss some open problems.

2 Framework and Basic Properties

2.1 Labeled Graphs

Let $G = (V, E)$ be a graph where nodes correspond to entities and edges correspond to direct bidirectional communication links between entities. Let $E(x)$ denote the set of edges incident to node x. Let $\mathcal{G}$ denote the set of all graphs.

A *path* π in G is a non empty sequence of edges $[\langle x_0, x_1\rangle, \langle x_1, x_2\rangle, \ldots, \langle x_{m-1}, x_m\rangle]$, $\langle x_i, x_{i+1}\rangle \in E(x_i)$, in which the endpoint of one edge is the starting point of the next edge. A path is a *cycle* if the starting point x_0 coincides with the ending point x_m; a path is *simple* if it does not contain any cycle. The *reverse path* of π, denoted with $\overline{\pi}$ is the path belonging to $P[x_{m+1}, x_1]$ s.t. $\overline{\pi} = [\langle x_{m+1}, x_m\rangle, \ldots, \langle x_3, x_2\rangle\langle x_2, x_1\rangle]$. For every $x, v \in V$, let $d_G(x, v)$ denote the *distance* between x and v, and analogously for each $x \in V$ and $e \in E$, let $d_G(x, e)$ denote the distance between the x and e. Let $P[x]$ denote the set of all the paths with $x \in V$ as a starting point, let $P[x, y]$ denote the set of paths starting from node $x \in V$ and ending in node $y \in V$, and let $P^d[x, y]$ denote the set of paths of length at most d, that belong to $P[x, y]$.

Given a graph $G = (V, E)$ and a set Σ of labels, a *local edge-labeling* (or labeling) function for $x \in V$ is any function $\lambda_x : E(x) \to \Sigma$ which associates a label $l \in \Sigma$ to each edge $e \in E(x)$. The *labeling* λ of G is the set of local labeling functions, that is $\lambda = \{\lambda_x : x \in V\}$. By (G, λ) we shall denote a *labeled graph*, that is a graph G on which it is defined a labeling λ. A labeling λ has a *symmetric* function ψ if and only if for each $\langle x, y\rangle \in E$, $\lambda_y(\langle x, y\rangle) = \psi(\lambda_x(\langle x, y\rangle))$.

Given a labeling λ and a node $x \in V$, let $\Lambda_x : P[x] \to \Sigma^+$ be the *path-labeling function* defined as follows: for every path $\pi \in P[x_1]$ starting from x_1, $\Lambda_{x_1}(\pi) = [\lambda_{x_1}(\langle x_1, x_2\rangle), \ldots, \lambda_{x_m}(\langle x_m, x_{m+1}\rangle)]$ where $\pi = [\langle x_1, x_2\rangle, \ldots, \langle x_m, x_{m+1}\rangle]$. Let $\Lambda_{(G,\lambda)}[x] = \{\Lambda_x(\pi) : \pi \in P[x]\}$, and $\Lambda_{(G,\lambda)}[x, y] = \{\Lambda_x(\pi) : \pi \in P[x, y]\}$. When (G, λ) is clear from the context, we will omit the subscript (G, λ) from the notation.

Definition 1 $\mathcal{LO}$ - Local Orientation
A labeled graph (G, λ) *has* local orientation *iff* $\forall x \in V, \forall e_1, e_2 \in E(x),\ \lambda_x(e_1) = \lambda_x(e_2) \Leftrightarrow e_1 = e_2$.

That is, a labeling is a local orientation when each node can distinguish among its incident edges.

By definition of local orientation, two different edges outgoing from x have two different labels. We can extend this property to paths of arbitrary length as follows:

Property 1 *If* λ *is a local orientation, then for each* $\pi_1, \pi_2 \in P[x]$*:*
$\pi_1 = \pi_2 \Leftrightarrow \Lambda_x(\pi_1) = \Lambda_x(\pi_2)$

Proof From the definition of Λ_x, it immediately follows that $\pi_1 = \pi_2 \Rightarrow \Lambda_x(\pi_1) = \Lambda_x(\pi_2)$. By contradiction, suppose that there exists $\pi_1, \pi_2 \in P[x]$ such that $\pi_1 \neq \pi_2$ and $\Lambda_x(\pi_1) = \Lambda_x(\pi_2)$. Let y the last node common to both paths (note that, possibly $x = y$). It follows that there are two different edges outgoing from y with the same label; this contradicts the assumption of local orientation. $\square$

That is, in systems with local orientation, to different paths starting from the same node correspond different strings of labels.

Let $\mathcal{O}$ denote the set of all labeled graphs with local orientation. Given $(G, \lambda) \in \mathcal{O}$, let $\underset{(G,\lambda)}{\rightarrow}: V \times \Sigma^* \rightarrow V$ be the partial function defined as follows: $\underset{(G,\lambda)}{\rightarrow}(v, \alpha) = w \Leftrightarrow \exists \pi \in P[v, w] \wedge \Lambda_v(\pi) = \alpha$. Note that, by Property 1, $\underset{(G,\lambda)}{\rightarrow}$ is well defined. In the following, the symbol $\underset{(G,\lambda)}{\rightarrow}$ will be also used in infix notation (i.e. $v \underset{(G,\lambda)}{\rightarrow} \alpha$ shall denote $\underset{(G,\lambda)}{\rightarrow}(v, \alpha)$). Moreover, when (G, λ) is clear from the context, we shall denote $\underset{(G,\lambda)}{\rightarrow}$ with $\rightarrow$.

Property 2 *For each* $u \in V$*,* $\alpha \in \Lambda[u] \Leftrightarrow (u \rightarrow \alpha) \in V$*.*

Proof By definition of $\Lambda[u]$, $\alpha \in \Lambda[u]$ if and only if there exists $\pi \in P[u, v]$ for some $v \in V$ and $\Lambda_u(\pi) = \alpha$. By definition of $\rightarrow$, $\pi \in P[u, v]$ and $\Lambda_x(\pi) = \alpha$ if and only if $u \rightarrow \alpha = v$. $\square$

2.2 Sense of Direction

Definition 2 $=$ *Given* (G, λ)*, a* consistent coding function *c for* λ *is any function with domain* Σ^+*, such that* $\forall x, y, z \in V$*,* $\forall \pi_1 \in P[x, y]$*,* $\pi_2 \in P[x, z]$

$$c(\Lambda_x(\pi_1)) = c(\Lambda_x(\pi_2)) \quad \Leftrightarrow \quad y = z.$$

In other words, a consistent coding function maps paths originating from the same node to the same value (called *local name*) if and only if they end in the same node. We have that:

Property 3 *If c is consistent in (G, λ), then $(G, \lambda) \in \mathcal{O}$.*

Proof Let c be consistent in (G, λ). For any node u and any two different arcs $e_1 = \langle u, v \rangle$, $e_2 = \langle u, z \rangle$ incident on the u, by definition of Λ_u and of c we have that $c(\lambda_u(e_1)) = c(\Lambda_u(e_1)) = c(\Lambda_u(e_2)) = c(\lambda_u(e_2))$ if and only if $e_1 = e_2$. $\square$

Property 4 *c is consistent in (G, λ) if and only if:*
$\forall x \in V\ \forall \alpha, \beta \in \Lambda[x]$: $x \to \alpha = x \to \beta \Leftrightarrow c(\alpha) = c(\beta)$. *(*)*

Proof
($\Rightarrow$) By contradiction, let c be consistent and there exist a node x and two strings α and β, such that either
1) $x \to \alpha = x \to \beta$ and $c(\alpha) \neq c(\beta)$, or
2) $x \to \alpha \neq x \to \beta$ and $c(\alpha) = c(\beta)$.
Case 1): $\alpha, \beta \in \Lambda[x]$ implies that there exist two paths π_1, $\pi_2 \in P[x, x \to \alpha]$ where $\Lambda_x(\pi_1) = \alpha$, $\Lambda_x(\pi_2) = \beta$. Thus, $c(\Lambda_x(\pi_1)) = c(\alpha) \neq c(\beta) = c(\Lambda_x(\pi_2))$; by consistency of c, $x \to \alpha \neq x \to \beta$ which yields a contradiction.
Case 2): $\alpha, \beta \in \Lambda[x]$ implies that there exist two paths $\pi_1 \in P[x, x \to \alpha]$ and $\pi_2 \in p[x, x \to \beta]$ such that $\Lambda_x(\pi_1) = \alpha$ and $\Lambda_x(\pi_2) = \beta$. By consistency of c, $c(\alpha) = c(\beta)$ implies $x \to \alpha = x \to \beta$ which yields a contradiction.
($\Leftarrow$) By contradiction suppose that the condition $(*)$ holds and c is not consistent; that is, there are three nodes x, y, z and two paths $\pi_1 \in P[x, y]$, $\pi_2 \in P[x, z]$ such that either
1) $c(\Lambda_x(\pi_1)) = c(\Lambda_x(\pi_2))$ and $y \neq z$, or
2) $c(\Lambda_x(\pi_1)) \neq c(\Lambda_x(\pi_2))$ and $y = z$.
Case 1): $x \to \Lambda_x(\pi_1) = y \neq z = x \to \Lambda_x(\pi_2)$. Noting that $\Lambda_x(\pi_1), \Lambda_x(\pi_2) \in \Lambda[x]$, the condition $(*)$ implies $c(\Lambda_x(\pi_1)) \neq c(\Lambda_x(\pi_2))$ which yields a contradiction.
Case 2): $x \to \Lambda_x(\pi_1) = y = z = x \to \Lambda_x(\pi_2)$. Noting that $\Lambda_x(\pi_1), \Lambda_x(\pi_2) \in \Lambda[x]$, the condition $(*)$ implies $c(\Lambda_x(\pi_1)) = c(\Lambda_x(\pi 2))$ which yields a contradiction. $\square$

Definition 3 $\mathcal{WSD}$ - Weak Sense of Direction
Given a labeled graph (G, λ) and a coding function c for λ, (G, λ) has Weak Sense of Direction *c iff c is consistent for λ. Alternatively, we shall say that c is a $\mathcal{WSD}$ in (G, λ).*

2.3 Structural Knowledge

We shall adopt the terminology and definitions of [29]. An algorithm A for a problem P must work on any network G using the available information about G, decide whether it can solve P for G, and solve P correctly if it can. If A determines that it cannot solve P for G, then it must report this fact.

For a problem P and an algorithm A for P, let $S(P, A)$ denote the set of graphs for which A can solve P, no matter what the particular instance of the knowledge, the communication timing among the processors, the edge labeling, and the particular consistent coding function are.

Given a structural information $\mathcal{K}$, Let $ALG_{\mathcal{K}}$ be the set of all algorithms for systems with local orientation that use the information $\mathcal{K}$, and let $ALG^{W}_{\mathcal{K}}$ be the set of all algorithms for systems with weak sense of direction that use the knowledge $\mathcal{K}$ and a consistent coding function. We define the set of graphs in which we can correctly solve a problem P as:

$$D_{\mathcal{K}}(P) = \{S \in S(P, A) : A \in ALG_{\mathcal{K}}\}$$
$$W_{\mathcal{K}}(P) = \{S \in S(P, A) : A \in ALG^{W}_{\mathcal{K}}\}$$

By definition, we have:

Property 5 *For any problem P and any knowledge $\mathcal{K}$,*

$$D_{\mathcal{K}}(P) \subseteq W_{\mathcal{K}}(P).$$

We investigate the computability in anonymous networks with weak sense of direction with respect to the four levels of structural knowledge studied in [29]:

- *No Information* (*noinfo*): No network attribute information is available.

- *Upper Bound on Networks Size* (*upbound*): A constant upper bound on the number of nodes in the network is available.

- *Network Size* (*size*): The exact number of nodes in the network is available.

- *Topological Awareness* (*topology*): The topology (i.e., the adjacences matrix) of (G, λ) is available.

We also consider a higher from of knowledge. To formally introduce it, we need the following definition.

Definition 4 lg-isomorphism
Given two labeled graphs $(G = (V, E), \lambda)$ and $(G' = (V', E'), \lambda')$, a bijective function $\chi : V \to V'$ is a labeled graph isomorphism *for (G, G') iff:*
1) $\langle u, v\rangle \in E \Leftrightarrow \langle \chi(u), \chi(v)\rangle \in E'$;
2) $\lambda(\langle u, v\rangle) = \lambda'(\chi(u), \chi(v)\rangle)$.

We can now formally define the last form of knowledge considered here:

- *Complete Network Topology* (*complete*): Each node knows a labeled graph, the same for all nodes, which is lg-isomorphic to G as well as its own isomorphic image in that graph.

Then, by definition, we have the following properties.

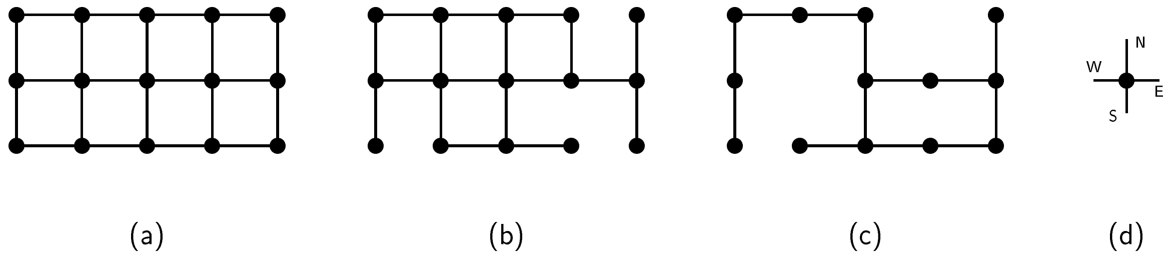


Figure 1: (a) (b) (c) Different graphs with the same consistent coding function c. (d) edge labeling of the three graphs.

Property 6 *[29]* $\mathcal{LO}$ Knowledge-Computation Hierarchy
For any problem P

$$D_{noinfo}(P) \subseteq D_{upbound}(P) \subseteq D_{size}(P) \subseteq D_{topology}(P) \subseteq D_{complete}(P)$$

Property 7 $\mathcal{WSD}$ Knowledge-Computation Hierarchy
For any problem P

$$W_{noinfo}(P) \subseteq W_{upbound}(P) \subseteq W_{size}(P) \subseteq W_{topology}(P) \subseteq W_{complete}(P)$$

It is interesting to note that the knowledge of a consistent coding function alone, without the exchange of messages, allows each node to locally derive an upper bound on the size of the network. In fact, the codomain of any consistent coding function c must contain at least n different symbols in order to distinguish between the nodes of the graph. Thus:

Property 8

$$W_{noinfo}(P) = W_{upbound}(P)$$

However, the knowledge of a consistent coding function is *not sufficient* for deriving locally (i.e., without exchanging messages) neither the topology, nor the exact size of the network as shown in the following.

For any subgraph $G = (V', E')$ of the labeled graph (G, λ), let $\lambda|_{G'}$ be the restriction of λ on G', that is $\lambda|_{G'} = \{\lambda'_x : x \in V \wedge \lambda'_x(e) = \lambda_x(e) \forall e \in E\}$. By definition of $\mathcal{WSD}$, we can easily prove that:

Property 9 *Let (G, λ) be a labeled graph with $\mathcal{WSD}$ c. For any subgraph G' of G, $(G', \lambda|_{G'})$ is a labeled graph with $\mathcal{WSD}$ c.*

By property 9, the three different labeled graphs of Figure 1 have the same $\mathcal{WSD}$ c. This means that the availability of a consistent coding function, without the exchange of messages, is not sufficient to derive the size of the network (see graphs a) and c)). It also means that availability of both consistent coding function and exact knowledge of n, without the exchange of messages, is not sufficient to derive the topology of the network (graphs a) and b)).

The properties of this section can be summarized as follows:

$$
\begin{array}{ccccccccc}
D_{noinfo}(P) & \subseteq & D_{upbound}(P) & \subseteq & D_{size}(P) & \subseteq & D_{topology}(P) & \subseteq & D_{complete}(P) \\
 & & \cap| & & \cap| & & \cap| & & \cap| \\
W_{noinfo}(P) & = & W_{upbound}(P) & \subseteq & W_{size}(P) & \subseteq & W_{topology}(P) & \subseteq & W_{complete}(P)
\end{array}
$$

3 View and Surrounding

3.1 View

A crucial concept when computing on anonymous networks is the one of *view*, introduced in [29]. The view $T_{(G,\lambda)}(v)$ of a node v in a labeled graph (G, λ) is an infinite, labeled, rooted tree, defined recursively as follows. $T_{(G,\lambda)}(v)$ has the root x_0 corresponding to v. For each vertex v_i adjacent to v in G, $T_{(G,\lambda)}$ has a node x_i and an edge from x_0 to x_i with labels $\lambda_v(\langle v, v_i\rangle)$ and $\lambda_{v_i}(\langle v, v_i\rangle)$ at its x_0's and x_i's ends respectively. Node x_i is now the root of $T_{(G,\lambda)}(v_i)$ from v_i.

In other words, the view $T_{(G,\lambda)}(v)$ of a node v is a rooted subgraph of the universal cover U induced by the collection of paths $P[v]$ in U.

The formal introduction of the concept of view and the characterization of its link with computability in anonymous systems is due to Yamashita and Kameda [28, 29]; in particular, they showed that an entity's view represent all the information about the system it can learn by exchanging messages [29]. The relationship between view and universal cover (a notion originating from algebraic topology) has been made explicit by Norris [22]. Both notions are being used (sometimes under different names) in the distributed computing literature (e.g., see [1, 6, 2, 15, 22, 21, 26]).

In the following, we shall refer to a node of a view by using the sequence of labels in the shortest path (in the view) from the root to that node. Since a view is a tree, such a naming in not ambiguous, and shall be called *canonical*. Thus, in a canonical naming, node x in view T is $x = \alpha$, where α is the (unique) sequence of edge labels in the shortest path in T from the root to x. Throughout the paper we shall exclusively use canonical naming of the nodes.

Let Σ be a (finite) set. Let Σ^* be the set of strings of element of Σ including the *empty* string ϵ, let $\Sigma^d \subseteq \Sigma^*$ be the set of strings of length at most d, and let $\Lambda^d_{(G,\lambda)}[x, y] = \Lambda_{(G,\lambda)}[x, y] \cap \Sigma^d$, and $\Lambda^d_{(G,\lambda)}[x] = \bigcup_{y \in V} \Lambda^d[x, y]$. Given $\alpha, \beta \in \Sigma^*$, let $\alpha \cdot \beta \in \Sigma^*$ be the *concatenation* of the α and β; given a set of strings A let $\alpha \cdot A = \{\alpha \cdot w : w \in A\}$. Given a pair of strings $\langle\alpha, \beta\rangle$ and a string γ, let $\gamma \cdot \langle\alpha, \beta\rangle = \langle\gamma \cdot \alpha, \gamma \cdot \beta\rangle$; given a set B of pair of strings, let $\gamma \cdot B = \{\gamma \cdot \langle\alpha, \beta\rangle : \langle\alpha, \beta\rangle \in B\}$.

When no ambiguity arises, we shall denote a view $T_{(G,\lambda)}(v)$ simply by $T(v)$. For any integer $d \geq 0$, let $T^d(v)$ denote the d-view of node v, i.e., $T(v)$ truncated to distance d. Given a labeled graph X, let $\mathcal{V}(X)$, $\mathcal{E}(X)$ and $\mathcal{L}(X)$ denote the vertices, the edges and the labeling of X, respectively.

By definition of view and of canonical naming, the following properties immediately

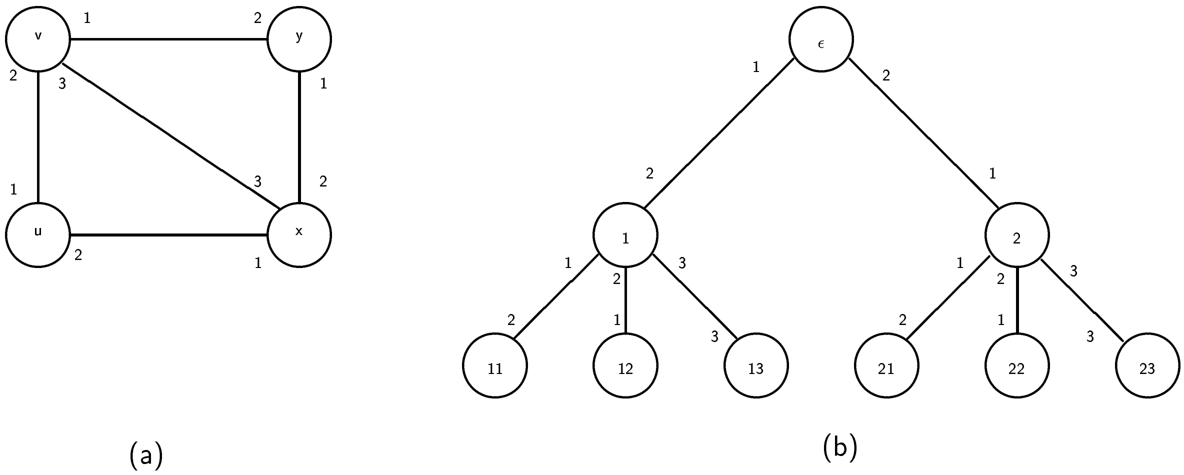


Figure 2: A labeled graph and its 2-view from node u.

follow.

Property 10 *Given $(G = (V, E), \lambda)$ and $u \in V$:*

1. $T^0(u) = ((\{\epsilon\}, \emptyset), \emptyset)$;

2. $\mathcal{V}(T^{i+1}(u)) = \{\epsilon\} \cup \bigcup_{v:\langle u,v\rangle \in E} \Big(\lambda_u(\langle u, v\rangle) \cdot \mathcal{V}(T^i(v))\Big)$;

3. $\mathcal{E}(T^{i+1}(u)) = \{\langle \epsilon, \lambda_u(\langle u, v\rangle)\rangle : \langle u, v\rangle \in E\} \cup \bigcup_{v:\langle u,v\rangle \in E} \Big(\lambda_u(\langle u, v\rangle) \cdot \mathcal{E}(T^i(v))\Big)$;

4. $\mathcal{L}(T^{i+1}(u))_\alpha(\langle \alpha, \beta\rangle) = \begin{cases} l & \text{if } \alpha = \epsilon \wedge \beta = l \\ \lambda_{u \to l}(\langle u \to l, u\rangle) & \text{if } \alpha = l \wedge \beta = \epsilon \\ \mathcal{L}(T^i(u \to l))_{\alpha'}(\langle \alpha', \beta'\rangle) & \text{if } \alpha = l \cdot \alpha' \wedge \beta = l \cdot \beta' \end{cases}$

Example 1 *A labeled graph G and its 2-view from node u are shown in Figure 1; each node in this view is uniquely identified by the sequence of labels corresponding to the shortest path from u in the tree.*

Note that the labeling of a view is not a local orientation. The following properties of the views express some simple relationships between nodes, edges and paths in a canonical view (for the proof, see the Appendix).

Property 11 *Let $\alpha, \beta \in V_T$, and let $\to$ denote $\underset{(G,\lambda)}{\to}$.*

1. $\langle \alpha, \beta\rangle \in \mathcal{E}(T^d(u))$ *iff* $\langle u \to \alpha, u \to \beta\rangle \in E$ *and* $\beta = \alpha \cdot \lambda_{u \to \alpha}(\langle u \to \alpha, u \to \beta\rangle)$.

2. $\mathcal{L}(T^d(v))_\alpha(\langle \alpha, \alpha \cdot l\rangle) = l$.

3. $\alpha \in \Lambda^d_{(G,\lambda)}[u, u \rightarrow \alpha] \Leftrightarrow \alpha \in \mathcal{V}(T^d(u))$.

4. $\alpha \in \Lambda^d_G[u] \Leftrightarrow \alpha \in \mathcal{V}(T^d(u))$.

Property 12 *[22] Let $|V| = n$.*

$$T^{n-1}(u) = T^{n-1}(v) \Leftrightarrow \forall d \geq 0 : T^d(u) = T^d(v)$$

Property 13 *[29] Let $|V| = n$.*

1. *The cardinality $\sigma_{(G,\lambda)}$ of the set $\{v : T^{n-1}(v) = T^{n-1}(u)\}$ is the same for all u.*

2. *$\sigma_{(G,\lambda)}$ divides n.*

3. *For each d, $\sigma_{(G,\lambda)}$ divides $\#_d(G)$, where $\#_d(G)$ is the number of vertices in G with degree d.*

The value $\sigma_{(G,\lambda)}$ is called *symmetricity* of the labeled graph (G, λ) in [29].

3.2 Surrounding

We now introduce the concept of *surrounding* of a node in a labeled graph; this notion is stronger than the one of view described before. For each $\alpha \in \Lambda^d_{(G,\lambda)}[u]$, let $[\![\alpha]\!]^d_u = \Lambda^d_{(G,\lambda)}[u, u \underset{(G,\lambda)}{\rightarrow} \alpha]$ (or $[\![\alpha]\!]$ when u and d are given).

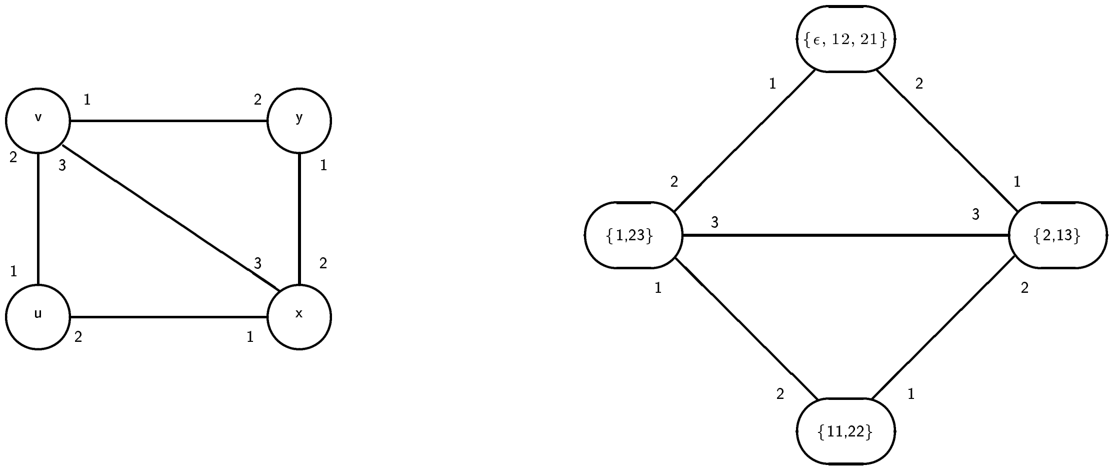
Given a labeled graph G, the d-surrounding of a node u is a labeled graph G' informally defined as follows. For each node v in G which is at distance at most d from u, there exists a node v' in G' denoted by the sequences of labels corresponding to the set of paths in G of length at most d starting from u and ending in v. There is an edge between two nodes v', w' in the surrounding iff the exists a sequence denoting w' that differs only in the last label l to one of the sequences denoting v'; in this case the edge in the surrounding is labeled by l (see Figure 1). Formally:

Definition 5 Surrounding
*Given a labeled graph $(G = (V, E), \lambda)$, an integer $d \geq 0$, and node $u \in V$, the d-*surrounding *of u is the labeled graph $N^d_{(G,\lambda)}(u)$ where*

1. $\mathcal{V}(N^d(u)) = \{[\![\alpha]\!]^d_u : \alpha \in \Lambda^d_{(G,\lambda)}[u]\}$;

2. *given α and β, $\langle [\![\alpha]\!]^d_u, [\![\beta]\!]^d_u\rangle \in \mathcal{E}(N(u))$ if and only if $e = \langle u \rightarrow \alpha, u \rightarrow \beta\rangle \in E$ and $d_G(u, e) \leq d$. The edge e will be called the* corresponding *edge of $\langle [\![\alpha]\!]^d_u, [\![\beta]\!]^d_u\rangle$.*

3. *$\mathcal{L}(N(u))_{[\![\alpha]\!]}(\langle [\![\alpha]\!], [\![\beta]\!]\rangle) = \lambda_{u\rightarrow\alpha}(\langle u \rightarrow \alpha, u \rightarrow \beta\rangle$ (i.e. the label of the corresponding edge).*



(a) (b)

Figure 3: A labeled graph and its 2-surrounding from node u.

Example 2 *A labeled graph G and its 2-surrounding from node u are shown in Figure 3. Notice the difference between the surrounding and the view (shown in Figure 1).*

The following properties on the surrounding of a node in a graph G describe some relations existing between paths in G and nodes and edges in the surrounding (for the proof, see the Appendix).

Property 14

1. *$\mathcal{V}(N^d_{(G,\lambda)}(u))$ is a partition of $\Lambda^d_{(G,\lambda)}[u]$.*
2. *$\langle [\![\alpha]\!], [\![\beta]\!] \rangle \in \mathcal{E}(N^d(u)) \Leftrightarrow \exists \alpha' \in [\![\alpha]\!], \beta' \in [\![\beta]\!] : \langle u \rightarrow \alpha', u \rightarrow \beta' \rangle \in E,\ \beta' = \alpha' \cdot \lambda_{u \rightarrow \alpha}(\langle u \rightarrow \alpha', u \rightarrow \beta' \rangle)$.*
3. *$|\mathcal{V}(N^d(u))| = |\{v \in V : d_G(u, v) \leq d\}|$.*
4. *$|\mathcal{E}(N^d(u))| = |\{e \in E : d_G(u, e) \leq d\}|$.*

For each $x \in V$, let $n_G(x) = \max_{v \in E} d(x, v)$ be the *node eccentricity* of x. For each $x \in V$, let $e_G(x) = \max_{e \in E} d(x, e)$ be the he *edge eccentricity* of x. The following are well known property of the node and edge eccentricity.

Property 15
1) for each $v \in V$, $n(v) \leq e(v) \leq n(v) + 1$.
2) for each $i < e(v)$, $\{e : d(v, e) \leq i\} \subset \{e : d(v, e) \leq i + 1\}$.

In the following we shall use the simpler notation $N(u)$ whenever referring to $N^{e(u)}(u)$.

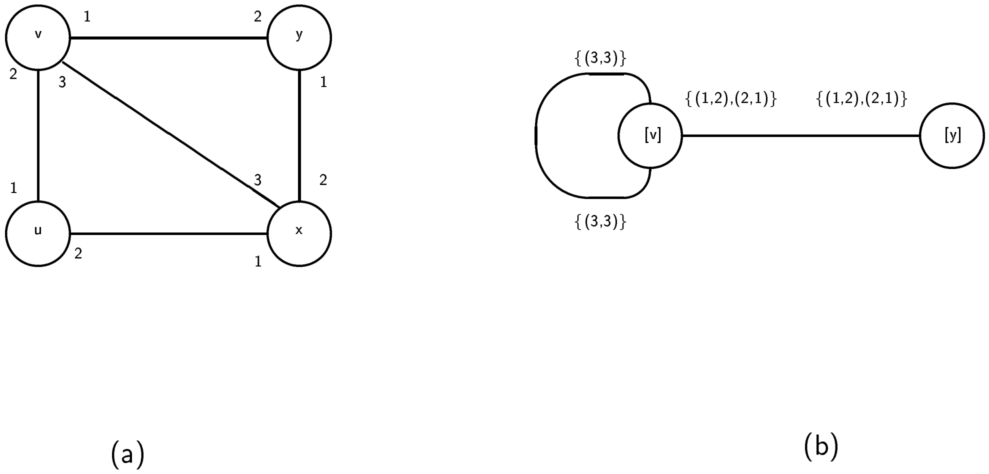


Figure 4: A labeled graph and its quotient.

4 Computability and Weak Sense of Direction

4.1 From Views to Surroundings

To analyze the relationship between view and surrounding, we need the notion of quotient graph induced by an equivalence relation on the nodes. Traditionally, the quotient graph $G_{//\equiv}$ of $(G = (V, E), \lambda)$ is defined as follows: the vertices of $G_{//\equiv}$ are the equivalence classes of $\equiv$, and there is and edge $\langle [u], [v] \rangle$ between the equivalence classes $[u]$ and $[v]$ (corresponding to nodes u and v) labeled l in $G_{//\equiv}$ if and only if there is and edge $\langle u', v' \rangle$ labeled l in G with $u' \in [u]$ and $v' \in [v]$. Notice that in general the quotient graph is a multigraph. We will transform the multigraph $G_{//\equiv}$ into a graph $G_{/\equiv}$ as follows. For each pair of adjacent nodes x and y in $G_{//\equiv}$ replace all the edges between them with a simple edge whose label is the set of all the labels of the replaced edges. In the particular case in which all labels in $G_{/\equiv}$ contain only one element, we shall further substitute each set by the element it contains. We shall call the resulting graphs $G_{/\equiv}$ the quotient graphs of G.

Definition 6 Quotient Graph
Let $(G = (V, E), \lambda)$ be a graph. The quotient graph $G/_{\equiv} = ((V', E'), \lambda')$ *of G induced by an equivalence relation $\equiv$ is defined as follows.*
$V' = \{[x]_\equiv : x \in V\}$, $\langle [x]_\equiv, [y]_\equiv \rangle \in E'$ if and only if there exists a corresponding edge *$\langle u, v \rangle \in E$ for some $u \in [x]_\equiv$ and $v \in [y]_\equiv$.*
Let $\lambda^*_{[x]_\equiv}(\langle [x]_\equiv, [y]_\equiv \rangle) = \{(\lambda_u(\langle u, v \rangle), \lambda_v(\langle u, v \rangle) : \langle u, v \rangle \in E \wedge u \in [x]_\equiv \wedge v \in [y]_\equiv\}$.
If for all $\langle [x]_\equiv, [y]_\equiv \rangle \in E'$, $\lambda^*_{[x]_\equiv}(\langle [x]_\equiv, [y]_\equiv \rangle) = \{(\lambda_u(\langle u, v \rangle), \lambda_v(\langle u, v \rangle)\}$ (i.e. there is only one pair of labels), then let $\lambda'_{[x]_\equiv}(\langle [x]_\equiv, [y]_\equiv \rangle) = \lambda_u(\langle u, v \rangle)$, and $\lambda'_{[y]_\equiv}(\langle [x]_\equiv, [y]_\equiv \rangle) = \lambda_v(\langle u, v \rangle)$; otherwise $\lambda' = \lambda^*$.

Example 3 *A labeled graph and its quotient are shown in Figure 3; note that $u \equiv y$ and $v \equiv x$. Notice the difference between the quotient and the surrounding (shown in*

Figure 3) of the same graph.

Definition 7 *Let f be a function defined over the set of nodes; the equivalence relation $\equiv_f$ between elements v and u of V is defined as follows:*

$$v \equiv_f u \Leftrightarrow f(v) = f(u).$$

The following lemma states an important relation between views and surroundings and will be used to understand the computational power of Sense of Direction. We show that, given a labeled graph (G, λ) with Sense of Direction, the quotient graph $T^d(u)/_{\equiv_c}$ of the view $T^d(u)$ induced by the consistent coding function c is equal to the d-surrounding of u.

Lemma 1 *If c is a consistent coding function for (G, λ), then $T^d(u)/_{\equiv_c} = N^d(u)$.*

Proof Let $T^d(u) = (T = (V_T, E_T), \overline{\lambda})$, $T^d(u)/_{\equiv_c} = (T/_c = (V_c, E_c), \lambda')$, and $N^d(u) = (N = (V_N, E_N), \lambda'')$.
Claim: For each $\alpha \in \Lambda^d_G[u]$, $[\alpha]_{\equiv_c} = [\![\alpha]\!]$.
$[\alpha]_{\equiv_c} = [\![\alpha]\!]$ if and only if for each β: $\beta \in [\alpha]_{\equiv_c} \Leftrightarrow \beta \in [\![\alpha]\!]$. By definition of quotient graph, $\beta \in [\alpha]_{\equiv_c} \Leftrightarrow \beta \in V_T \wedge \beta \equiv_c \alpha$. By definition of the equivalence relation $\equiv_c$, $\beta \equiv_c \alpha \Leftrightarrow c(\beta) = c(\alpha)$. Thus, $\beta \in [\alpha]_{\equiv_c} \Leftrightarrow \beta \in V_T \wedge c(\beta) = c(\alpha)$. By Property 11.3, $\alpha \in V_T \Leftrightarrow \alpha \in \Lambda^d[u, u \to \alpha]$ and $\beta \in V_T \Leftrightarrow \beta \in \Lambda^d[u, u \to \beta]$. By Property 4, $c(\beta) = c(\alpha) \Leftrightarrow u \to \beta = u \to \alpha$. Thus, $\beta \in V_T \wedge c(\beta) = c(\alpha) \Leftrightarrow \beta \in [\![\alpha]\!]$, concluding the proof of the claim.

In order to prove the thesis, we must prove that: 1) $V_c = V_N$; 2) $E_c = E_N$; 3) $\lambda' = \lambda''$.
1) $V_c = V_N$ if and only if $A \in V_c \Leftrightarrow A \in V_N$. By definition of quotient graph, $A \in V_c \Leftrightarrow A = [\alpha]_{\equiv_c}$ for some $\alpha \in V_T$. By Property 11.4, $\alpha \in V_T \Leftrightarrow \alpha \in \Lambda^d[u]$. By the above claim, $[\alpha]_{\equiv_c} = [\![\alpha]\!]$. Thus, $A \in V_c \Leftrightarrow A = [\![\alpha]\!]$ for some $\alpha \in \Lambda^d[u]$. In other words, by definition of d-surrounding, $A \in V_c \Leftrightarrow A \in N_T$.
2.) $E_c = E_N$ if and only if $\langle A, B\rangle \in E_c \Leftrightarrow \langle A, B\rangle \in E_N$. By definition of quotient graph, $\langle A, B\rangle \in E_c \Leftrightarrow \exists \alpha \in A, \beta \in B, \langle \alpha, \beta\rangle \in E_T$. By Property 11.1, $\langle \alpha, \beta\rangle \in E_T \Leftrightarrow \langle u \to \alpha, u \to \beta\rangle = e \in E$, $\beta = \alpha \cdot \lambda_{u \to \alpha}(e)$ and $|\beta| \leq d$. By Property 14.2, $\exists \alpha \in A, \beta \in B : \langle u \to \alpha, u \to \beta\rangle = e \in E \wedge \beta = \alpha \cdot \lambda_{u\to\alpha}(e) \Leftrightarrow \langle A, B\rangle \in E_N$.
3.) Let $[\![\overline{\alpha}]\!] = A$ and $[\![\overline{\beta}]\!] = B$. By construction, $\lambda^*_A(\langle A, B\rangle) = \{(\overline{\lambda}_\alpha(\langle \alpha, \beta\rangle), \overline{\lambda}_\beta(\langle \alpha, \beta\rangle)) : \langle \alpha, \beta\rangle \in E_T \wedge \alpha \in A \wedge \beta \in B\}$. By Property 11.1, $\langle \alpha, \beta\rangle \in E_T \Leftrightarrow \langle u \to \alpha, u \to \beta\rangle \in E \wedge \beta = \alpha \cdot \lambda_{u\to\alpha}(\langle u \to \alpha, u \to \beta\rangle) \wedge |\beta| \leq d$ and by Property 11.2, $\overline{\lambda}_\alpha(\langle \alpha, \beta\rangle) = \lambda_{u\to\alpha}(\langle u \to \alpha, u \to \beta\rangle)$. Thus, $\lambda^*_A(\langle A, B\rangle) = \{(\lambda_{u\to\alpha}(\langle u \to \alpha, u \to \beta\rangle), \lambda_{u\to\beta}(\langle u \to \alpha, u \to \beta\rangle)) : \langle u \to \alpha, u \to \beta\rangle \in E \wedge \beta = \alpha \cdot \lambda_{u\to\alpha}(\langle u \to \alpha, u \to \beta\rangle) \wedge \alpha \in A \wedge \beta \in B\}$. By definition of quotient graph and of $\equiv_c$, $\lambda^*_A(\langle A, B\rangle) = \{(\lambda_{u\to\alpha}(\langle u \to \alpha, u \to \beta\rangle), \lambda_{u\to\beta}(\langle u \to \alpha, u \to \beta\rangle)) : \langle u \to \alpha, u \to \beta\rangle \in E \wedge \beta = \alpha \cdot \lambda_{u\to\alpha}(\langle u \to \alpha, u \to \beta\rangle) \wedge c(\alpha) = c(\overline{\alpha}) \wedge c(\beta) = c(\overline{\beta})\}$. By Property 4, $\lambda^*_A(\langle A, B\rangle) = \{(\lambda_{u\to\overline{\alpha}}(\langle u \to \overline{\alpha}, u \to \overline{\beta}\rangle), \lambda_{u\to\overline{\beta}}(\langle u \to \overline{\alpha}, u \to \overline{\beta}\rangle))\}$. By definition of λ', $\lambda'_A(\langle A, B\rangle) = \overline{\lambda}_{\overline{\alpha}}(\langle \overline{\alpha}, \overline{\beta}\rangle) = \lambda_{u\to\overline{\alpha}}(u \to \overline{\alpha}, u \to \overline{\beta})$, and $\lambda'_B(\langle A, B\rangle) = \overline{\lambda}_{\overline{\beta}}(\langle \overline{\alpha}, \overline{\beta}\rangle) = \lambda_{u\to\overline{\beta}}(u \to \overline{\alpha}, u \to \overline{\beta})$. By definition of λ'', $\lambda'_A(\langle A, B\rangle) = \lambda''_A(\langle A, B\rangle)$ and $\lambda''_B(\langle A, B\rangle) = \lambda''_B(\langle A, B\rangle)$. $\square$

An immediate consequence of the previous lemma is the following corollary.

Corollary 1 *For every labeled graph (G,λ) with $\mathcal{WSD}$,*

$$T^d(u) = T^d(v) \Rightarrow N^d(u) = N^d(v).$$

The following lemma characterizes the relationship between the surrounding of a node in a graph and the graph itself.

Lemma 2 *Let (G,λ) be a connected labeled graph.*
1) there exists a graph isomorphism χ for $(N^{e(u)},(G,\lambda))$ and $\chi([\![\epsilon]\!]) = u$
2) there exists a graph isomorphism for $(N^i(u), N^{i+1}(u))$ if and only if $i \geq e_G(u)$

Proof
1) Since G connected, we have that from node u it is possible to reach each other node x. Let $p_G(u,x)$ be an arbitrary shortest path in $P_G[u,x]$. Let $\chi : V \to V_N$ s.t. $\chi(x) = [\![\Lambda_u(p_G(u,x))]\!]$. First we shall prove that the function is well defined that is: there corresponds a $[\![\alpha]\!] \in V_N$ to each $x \in V$. By Property 14.3 and definition of $N(u)$, $p_G(u,x) \in \Lambda^{e(u)}[u]$. Thus, by definition of surrounding, $\chi(x) \in V_N$. Now we have to prove that χ is a labeling preserving isomorphism that is:
1.0.1) χ is one to one;
1.0.2) χ is onto;
1.0.3) χ^{-1} is s.t. $\chi^{-1}([\![\alpha]\!]) = u \rightharpoonup \alpha$;
1.1) $\langle x,y\rangle \in E \Leftrightarrow \langle \chi(x), \chi(y)\rangle \in E_N$
1.2) $\lambda_x(\langle x,y\rangle) = \lambda_{\chi(x)}(\langle \chi(x),\chi(y)\rangle)$
1.0.1) Let $\chi(x) = \chi(y)$, that means $[\![\Lambda_u(p(u,x))]\!] = [\![\Lambda_u(p(u,y))]\!]$. By definition of $[\![]\!]$, $\Lambda^{e(u)}[u, u \rightharpoonup \Lambda_u(p(u,x))] = \Lambda^{e(u)}[u, u \rightharpoonup \Lambda_u(p(u,y))]$. By Property 14.1, $\{\Lambda^{e(u)}[u,x] : x \in V\}$ is a partition of $\Lambda^{e(u)}[u]$ which implies $u \rightharpoonup \Lambda_u(p(u,x)) = u \rightharpoonup \Lambda_u(p(u,y))$. By definition of $\rightharpoonup$, we have that $x = y$.
1.0.2) Let $\alpha \in \Lambda^{e(u)}[u]$, this implies that $u \rightharpoonup \alpha \in V$. Thus, $\chi(u \rightharpoonup \alpha) = [\![\alpha]\!]$.
1.0.3) $\chi^{-1}([\![\alpha]\!]) = x$ such that $\chi(x) = [\![\alpha]\!])$. Let $x = u \rightharpoonup \alpha$, then $\chi(x) = [\![\Lambda_u(p(u, u \rightharpoonup \alpha))]\!] = [\![\overline{\alpha}]\!]$ s.t. $u \rightharpoonup \alpha = u \rightharpoonup \overline{\alpha}$. But by definition of $[\![]\!]$ follows that $[\![\alpha]\!] = [\![\overline{\alpha}]\!]$.
1.1) By definition of E_N, $\langle \chi(x), \chi(y)\rangle \in E_N$ if and only if $\exists \pi \in P_G^{e(u)}[u, u \rightharpoonup \Lambda_u(p(u,x))]$: $\pi\langle u \rightharpoonup \Lambda_u(p(u,x)), u \rightharpoonup \Lambda_u(p(u,y))\rangle \in P_G^d[u]$. By 1.0.3), $u \rightharpoonup \Lambda_u(p(u,x)) = \chi^{-1}([\![\Lambda_u(p(u,x))]\!]) = \chi^{-1}(\chi(x)) = x$ and $u \rightharpoonup \Lambda_u(p(u,y)) = y$. Thus, $\langle \chi(x), \chi(y)\rangle \in E_N$ if and only if $\exists \pi \in P_G^{e(u)}[u,x] : \pi\langle x,y\rangle \in P_G^d[u]$. By definition of $e(u)$, $\exists \pi \in P_G^{e(u)}[u,x] : \pi\langle x,y\rangle \in P_G^d[u]$ if and only if $\langle x,y\rangle \in E$.
1.2) By 1.1), the corresponding edge of $\langle \chi(x), \chi(y)\rangle$ is $\langle x,y\rangle$. Thus the label of the two edges are equal by definition of surrounding.
2) ($\Rightarrow$) Note that following the proof of point 1), nothing changes if we substitute $e(u)$ with $e(u)+1$. Thus $N^{e(u)+1}(u)$ is isomorphic to G, and G is isomorphic to $N^{e(u)}(u)$. By transitivity, we have that $N^{e(u)}(u)$ is isomorphic to $N^{e(u)+1}(u)$.
($\Leftarrow$) Let $M = \{e \in E : d_G(u,e) = n+1\}$. By Property 15.2, $|\{e : d(u,e) \leq i\}| < |\{e : d(u,e) \leq i+1\}|$. By Property 14.4, $|\mathcal{E}(N^i(u))| = |\{e : d(u,e) \leq i\}| < |\{e : d(u,e) \leq i+1\}| = |\mathcal{E}(N^{i+1}(u))|$. Thus, $N^i(u)$ is not isomorphic to $N^{i+1}(u)$. $\square$

An immediate consequence of the previous lemma and of Corollary 1 is the following corollary.

Corollary 2 *For any labeled graph* (G, λ) *with* $\mathcal{WSD}$*,*

$$T(u) = T(v) \Leftrightarrow N(u) = N(v).$$

4.2 Computational Power of $\mathcal{WSD}$

We now prove that, with weak sense of direction, the knowledge-computability hierarchy described in section 2.3 collapses.

Theorem 1 Hierarchy Collapse
For every $\mathcal{A}, \mathcal{B} \in \{$*noinfo, upbound, size, topology, complete*$\}$ *and every problem* P*,*

$$W_{\mathcal{A}}(P) = W_{\mathcal{B}}(P).$$

The theorem derives from the following more general result.

Theorem 2 *For every knowledge* $\mathcal{K}$ *and for every problem* P*,* $W_{\mathcal{K}}(P) \supseteq W_{complete}(P)$*.*

Proof We prove that $W_{noinfo} \supseteq W_{complete}$. By Lemma 4 in [29], each processor v can compute $T^d(v)$ for any nonnegative integer d. Note that no knowledge is required to compute $T^d(v)$. With c, each processor can compute $T^d(v)/_{\equiv_c}$ and stop when $T^{d-1}(v)/_{\equiv_c}$ is isomorphic to $T^d(v)/_{\equiv_c}$. By Lemma 2.2, each processor stops when $T^d(v)/_{\equiv_c}$ is equal to $N(v)$. Moreover, the processors selects $[\![\epsilon]\!]$ that by Lemma 2.1, is isomorphic to itself. □

Corollary 3 *For each* $\mathcal{K} \in \{$*noinfo, upbound, size, topology* $\}$ *and each problem* P*,*

$$W_{\mathcal{K}}(P) = W_{complete}(P).$$

Proof By Property 7 and Theorem 2. □

By Theorem 1, it follows that for all $\mathcal{K} \in \{$noinfo, upbound, size, topology, complete$\}$, all $W_{\mathcal{K}}(P)$ coincide; let $W(P)$ denote such a set.

We can now prove that complete knowledge of the topology and weak sense of direction have the same computational power.

Theorem 3 *For every problem* P*,*

$$D_{complete}(P) = W(P)$$

Proof Let A be an algorithm that solves P in any labeled graph $\in W(P)$. We will prove that it is possible to simulate the behavior of the algorithm A in $(G = (V, E), \lambda)$ doing only internal computations. Note that not all the labeling λ of a network admit a consistent coding function. Thus, in order to show the existence of such a simulation, we have to provide: 1) a labeling λ' 2) a consistent coding function c for λ'. Each processor v can locally compute its surrounding $N(v)$ and the surrounding of all the other processors. Let $\lambda'_x(e) = (N(x), \lambda_x(e))$ and $c(\Lambda'_x(\pi)) = c((N(x), l_1) \cdot (N(y), l_2) \cdot \ldots \cdot (N(z), l_k)) = [\![l_1 \cdot l_2 \cdot \ldots \cdot l_k]\!]_{N(x)}$. We will prove that c is a consistent coding function, that is for every $\pi_1 \in P[x, y]$ and $\pi_2 \in P[x, z]$: $c(\Lambda_x(\pi_1)) = c(\Lambda'_x(\pi_2)) \Leftrightarrow y = z$. Let $\lambda_x(\pi_i) = \alpha_i$ for $i = 1, 2$. By definition of surrounding, $[\![\alpha_1]\!] = [\![\alpha_2]\!] \Leftrightarrow x \rightarrow \alpha_1 = x \rightarrow \alpha_2 \Leftrightarrow y = z$. Then algorithm A must solve the problem P for (G, λ') using the function c. Thus, without exchanging any message, each processor can simulate the behavior of A in the local copy (G, λ') of the network it is running on. Note that the labeling of the graph remains the same, in fact λ' is the labeling of a local representation of the network. $\square$

As a consequence, we have that:

Corollary 4 *For every problem P,*

$$D_{noinfo}(P) \subseteq D_{upbound}(P) \subseteq D_{size}(P) \subseteq D_{topology}(P) \subseteq D_{complete}(P) = W(P)$$

The previous results have many important implications. For example, they formally prove that in any anonymous network with any weak sense of direction it is possible to do shortest-path routing. Let $\mathcal{G}$ denote the set of all graphs.

Corollary 5 $W(\textit{SHORTEST-PATH}) = \mathcal{G}$

4.3 Graph Characterization

In this section, we characterize the graphs on which, with sense of direction, it is possible to solve the problems studied in [29]: Leader Election, Edge Election, Spanning Tree Construction, Topology Recognition, as well as the more complex problem of Complete Topology recognition.

We recall the problems studied in [29]:

Leader Election Problem (ELECT-LEADER): Elect a processor as the leader in the sense that the elected processor knows that it has been elected and the other processors knows that they have not.

Edge Election Problem (ELECT-EDGE): Select a link $e = \langle u, v \rangle$ in the sense that each processors u and v know which port correspond to e and the other processors know that they are not incident with e.

Spanning Tree Construction Problem (SPANNING-TREE): Compute a spanning tree T of the network in the sense that each processor can tell which links incident to it are tree edges.

Topology Recognition Problem (FIND-TOPOLOGY): Compute on each processor a graph isomorphic to the network it is running on.

We also consider the following problem:

Complete Topology Recognition Problem (FIND-COMPLETE): Compute on each processor a labeled graph lg-isomorphic to the network it is running on; moreover, each processor select a node in the graph G that is isomorphic to itself.

By definition, the following property trivially holds.

Property 16
1) $D_{topology}(\textit{FIND-TOPOLOGY}) = W_{topology}(\textit{FIND-TOPOLOGY}) = \mathcal{G}$
2) $D_{complete}(\textit{FIND-COMPLETE}) = W_{complete}(\textit{FIND-COMPLETE}) = \mathcal{G}$

We first consider the *leader election* problem and recall the following result of [29].

Theorem 4 *(Theorem 1 in [29])*
$D_{topology}(\textit{ELECT-LEADER}) = \{G : \forall\lambda, (G, \lambda) \in \mathcal{O} \Rightarrow \sigma_{(G,\lambda)} = 1\}$.

Let $\mathcal{S}$ (Singular) be the set $\{G : \forall\lambda, (G, \lambda) \in \mathcal{W} \Rightarrow \sigma_{(G,\lambda)} = 1\}$. In other words, $\mathcal{S}$ is the set of graphs such that their symmetricity is one, for any labeling with $\mathcal{WSD}$.

We can now characterize the class of graphs with sense of direction on which the leader election problem is solvable.

Theorem 5 $W(\textit{ELECT-LEADER}) = \mathcal{S}$

Proof It follows from Theorem 4 and Corollary 2. □

Analogously, for *edge election* problem we recall the following result.

Theorem 6 *(Theorem 2 in [29])*
$D_{topology}(\textit{ELECT-EDGE}) = \{G : \forall\lambda, (G, \lambda) \in \mathcal{O} \Rightarrow (\sigma_{(G,\lambda)} \leq 2 \wedge \sigma_{(G,\lambda)} = 2 \Rightarrow \exists u, v \in V : N_{(G,\lambda)}(u) = N_{(G,\lambda)}(v) \wedge \langle u, v\rangle \in E(G))\}$.

Let $\mathcal{E}$ (Edge Singular) be the set $\{G : \forall\lambda, (G, \lambda) \in \mathcal{W} \Rightarrow (\sigma_{(G,\lambda)} \leq 2 \wedge \sigma_{(G,\lambda)} = 2 \Rightarrow \exists u, v \in V : N_{(G,\lambda)}(u) = N_{(G,\lambda)}(v) \wedge \langle u, v\rangle \in E(G))\}$. In other words, $\mathcal{E}$ is the set of graphs which have, for any labeling with $\mathcal{WSD}$, symmetricity less or equal to two; moreover, if their symmetricity is exactly two, there are at least two adjacent nodes with the same surrounding. We have that.

Theorem 7 $W(\textit{ELECT-EDGE}) = \mathcal{E}$

Proof It follows from Theorem 7 and from Corollary 2. $\square$

We now consider the *spanning tree* construction problem. From Theorem 3 in [29], we have that.

Theorem 8 *For any (G, λ) and any knowledge $\mathcal{K}$, there is an algorithm A to solve* ELECT-EDGE *iff there is an algorithm B to solve* SPANNING-TREE.

As a consequence of the previous result, we have that.

Theorem 9 $W(\textit{SPANNING-TREE}) = W(\textit{ELECT-EDGE})$.

The *topology construction* and the *complete topology* construction problems can be characterized as follows.

Theorem 10 $W(\textit{FIND-TOPOLOGY}) = W(\textit{FIND-COMPLETE}) = \mathcal{G}$

Proof It follows from Corollary 3 and Property 16.2. $\square$

The following result summarizes the relationship between the above classes of graphs; moreover, it proves the proper inclusions.
Let $\mathcal{T}$ denote the set of all tree networks.

Theorem 11
$W(\textit{ELECT-LEADER}) \subset W(\textit{ELECT-LEADER}) \cup \mathcal{T} \subset W(\textit{SPANNING-TREE}) = W(\textit{ELECT-EDGE}) \subset W(\textit{FIND-TOPOLOGY}) = W(\textit{FIND-COMPLETE}) = \mathcal{G}$.

Proof By the previous theorems all the equalities are proved. Moreover, it is easy to see that TREE$\subseteq W($SPANNING-TREE$)$. Thus we need only to provide graphs G_1, G_2 and G_3 such that:
1) $G_1 \in$TREE$-W($ELECT-EDGE$)$;
2) $G_2 \in W($SPANNING-TREE$)-$TREE;
3) $G_3 \in \mathcal{G} - W($SPANNING-TREE$)$.
1) Let $G_1 = (\{u, v\}, \{\langle u, v\rangle\})$ with labeling λ s.t. $\lambda_u = \lambda_v$ any constant function. It is easy to see that (G_1, λ) has sense of direction and $\sigma_{(G,\lambda)} = 2$. Thus, by Theorem 5, $G_1 \notin W($ELECT-EDGE$)$, but $G_1 \in$TREE.
2) Let G_2 be the graph shown in Figure 5. It is easy to see that $\sigma_{(G,\lambda)} = 2$ and that the labeled graph has Sense of Direction. Thus, $G_2 \notin W($ELECT-LEADER$)$. On the other hand, by Theorem 3 in [29] and by Corollary 3 it follows $G_2 \in W($ELECT-EDGE$)$.
3) Finally, it is clear that $\mathcal{G}$ properly contains $W($SPANNING-TREE$)$, since the Spanning Tree Problem is unsolvable for rings with sense of direction. $\square$

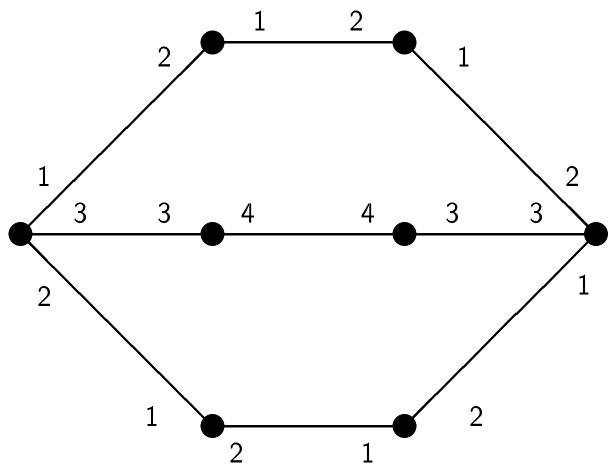


Figure 5: The graph G_2 in the proof of Theorem 11.

5 Computational Relations between Sense of Direction and Topological Awareness

At the end of Section 4.2, the equality $D_{complete}(P) = W(P)$ was proven. In this section we investigate the relations between $D_{topology}$ and $W(P)$. In particular we consider the problems studied in Section 4.3 Using these problems as test cases, we investigate the relationship between computability with sense of direction and computability with topological awareness. With respect to all these problems, we prove that weak sense of direction is strictly more powerful than topological awareness

5.1 Elections and Spanning Tree Construction

With respect to the problems of leader election, edge election and spanning-tree construction, we prove that weak sense of direction is strictly more powerful than topological awareness. We prove that there exist graphs in which these problems can be solved with *any* weak sense of direction; without weak sense of direction, none of these problems is solvable even in presence of topological awareness.

Definition 8 *A labeled graph $(G = (V, E), \lambda)$ is* completely symmetric *if and only if all views are equal, that is $\sigma_{(G,\lambda)} = |V|$.*

The following lemma gives a necessary and sufficient condition for a graph to be completely symmetric.

Lemma 3 *[10] A labeled graph (G, λ) is completely symmetric if and only if G is d-regular for some d and λ is a symmetric local labeling using d labels.*

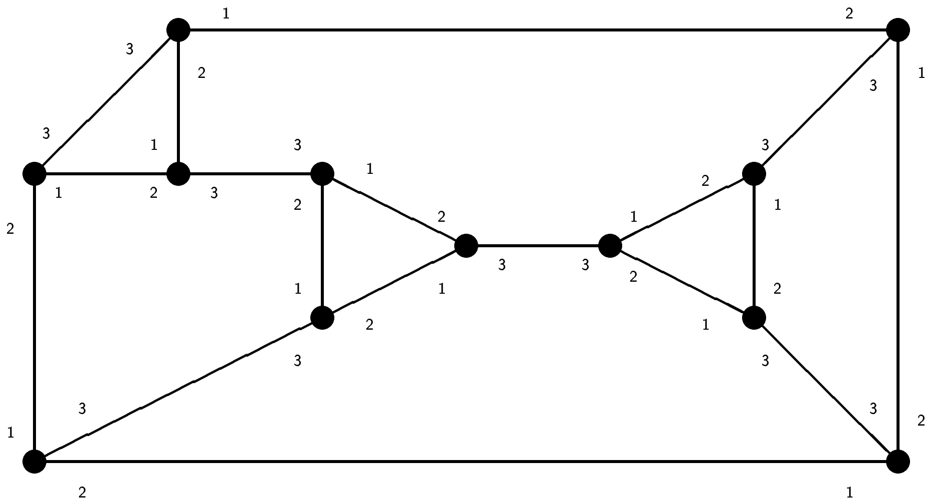


Figure 6: A symmetric labeling of the minimum regular identity graph.

Consider the labeled graph of Figure 6, we have that:

Property 17 *The labeled minimum identity graph (I, δ) shown in Figure 6 is completely symmetric.*

Proof The labeling δ is symmetric with the following function ψ: $\psi(1) = 2$, $\psi(2) = 1$ and $\psi(3) = 3$. Moreover, δ uses only 3 labels, namely 1, 2, and 3. By Lemma 3, we have that the graph is completely symmetric. □

The following theorem shows that the problem of leader election, edge election and spanning-tree construction are not solvable in (I, δ) without weak sense of direction, even in presence of topological awareness.

Theorem 12 *Let* $P \in \{$ELECT-EDGE, ELECT-LEADER, SPANNING-TREE$\}$.

$$I \notin D_{topology}(P)$$

Proof Let δ be the labeling of I defined in Property 17. By definition of σ, $\sigma(I) \geq \sigma_{(I,\delta)} = 12$. By Theorem 2 in [29], if $G \in D_{topology}(\text{ELECT-EDGE})$, then $\sigma(G) \leq 2$. Thus, $I \notin D_{topology}(\text{ELECT-EDGE})$.
(By Theorem 3 in [29], $D_{topology}(\text{ ELECT-LEADER}) \subset D_{topology}(\text{ELECT-EDGE})$. Thus, $I \notin D_{topology}(\text{ELECT-LEADER})$). □

However, as consequences of Theorem 5, these problems can be solved with *any* weak sense of direction.

Corollary 6 *Let* $P \in \{$*ELECT-EDGE, ELECT-LEADER, SPANNING-TREE*$\}$.

$$I \in W(P)$$

Proof We have to prove that for each labeling λ of I that is a sense of direction, the symmetry is 1. Note that, if a function is a lg-isomorphism, then it is also an isomorphism. In the identity graph there are no isomorphisms; hence, there are no lg-isomorphisms. By Lemma 2, two nodes with the same surrounding are lg-isomorphic. Thus, there are no nodes with the same surrounding. By Corollary 2, there are no nodes with the same view. Thus, by definition of symmetry, $\sigma_{(I,\lambda)} = 1$. By Theorems 5,7 and 11, the thesis follows. □

Corollary 7 *Let* $P \in \{$*ELECT-EDGE, ELECT-LEADER, SPANNING-TREE*$\}$.

$$I \in W(P) - D_{topology}(P)$$

We now consider the class of all the graphs W(ELECT-LEADER) $- D_{topology}$(ELECT-LEADER) and we provide a complete characterization of this class in terms of symmetry and singularity. Recall that $\mathcal{W}$ denote the set of all labeled graphs with weak sense of direction and $\mathcal{S} = \{G : \forall\lambda, (G,\lambda) \in \mathcal{W} \Rightarrow \sigma_{(G,\lambda)} = 1\}$. Let $\mathcal{B}$ be $\{G : \exists\lambda : (G,\lambda) \in \mathcal{O} \Rightarrow \sigma_{(G,\lambda)} \geq 2\})$.

Note that all identity graphs belong to $\mathcal{S}$.

Theorem 13

$$\mathcal{B} \cap \mathcal{S} = W(\textit{ ELECT-LEADER}) - D_{topology}(\textit{ ELECT-LEADER})$$

5.2 Complete Topology Reconstruction

In this section, we consider the Complete Topology Recognition problem. Also for this problem, we prove that sense of direction is strictly more powerful than topological awareness, and characterize the class of graphs in which topological awareness is sufficient to solve the problem.

Consider the labeled graph (H,γ) of Figure 7.
By Theorem 2, we have that the topology recognition problem can be solved in H with any weak sense of direction.

Theorem 14 $H \in W$*(FIND-COMPLETE)*

However, it cannot be solved in (H,γ) even in presence of topological awareness.
The following theorem shows a necessary and sufficient condition on the structure and the labeling of a graph so that topological awareness is sufficient to derive complete knowledge of the topology.

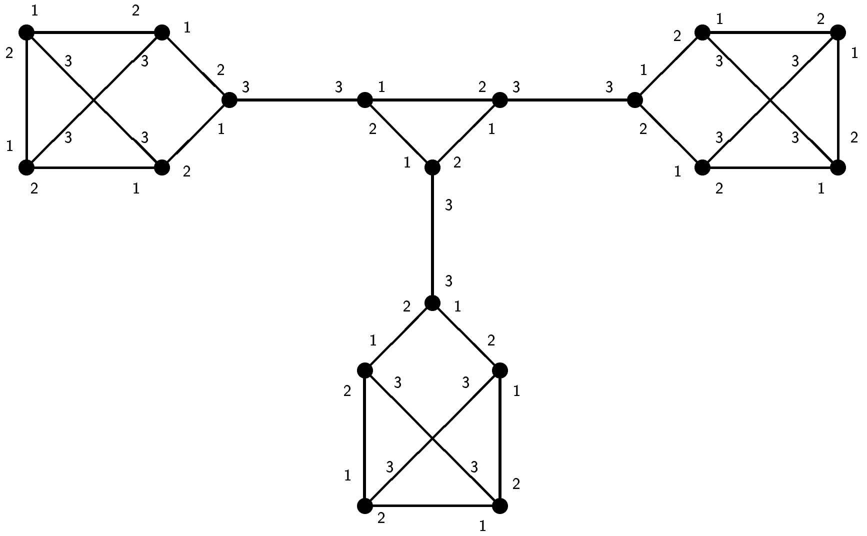


Figure 7: A completely symmetric labeled graph.

Theorem 15 *$G \in D_{topology}$(FIND-COMPLETE) if and only if for all labelings λ, λ', and all $u, v \in V$:*

$$T_{(G,\lambda)}(u) = T_{(G,\lambda')}(v) \Rightarrow N_{(G,\lambda)}(u) = N_{(G,\lambda')}(v).$$

Proof ($\Rightarrow$) By contradiction suppose that exists λ, λ', u and v such that $T_{(G,\lambda)}(u) = T_{(G,\lambda')}(v)$ and $N_{(G,\lambda)}(u) \neq N_{(G,\lambda')}(V)$. Processors with the same view must behave in the same way by Lemma 5 in [29], thus if u correctly compute $N_{(G,\lambda)}(u)$, then v wrongly compute as its neighborhood $N_{(G,\lambda)}(u)$.
($\Leftarrow$) The TOPOLOGY knowledge provides each processor with a graph G isomorphic to the network it is running on. Processor u uses the standard algorithm to compute $T_{(G,\lambda)}(u)$. Then, u labels the graph G with the labeling λ' in such a way that there is a node v' in (G, λ') for which $T_{(G,\lambda')}(v) = T_{(G,\lambda)}(u)$. By hypothesis, $N_{(G',\lambda')}(v) = N_{(G,\lambda)}(u)$ and processor u can locally compute $N_{(G,\lambda)}(u)$ looking at the labeled graph (G, λ'). $\Box$

We can show that:

Property 18 *The labeled graph (H, γ) shown in Figure 7 is completely symmetric.*

Thus, the following easily follows from the above two results.

Theorem 16 *$H \notin D_{topology}$(FIND-COMPLETE)*

Corollary 8 *$H \in W$(FIND-COMPLETE) $-$ $D_{topology}$(FIND-COMPLETE).*

6 Concluding Remarks

In this paper, we have studied the impact that sense of direction has on *computability* in anonymous networks and we have established several results about the impact that different levels of knowledge have on computability. Some of these results have powerful implications; one of such implication is, for example, the formal proof that, with sense of direction, shortest path routing is possible in anonymous networks. We have also considered several fundamental problems, and provided a complete characterization of the anonymous networks on which they are computable with sense of direction. Finally, we have shown that sense of direction is computationally stronger than topological awareness for a basic set of problems.

The characterizations, as well as the results on the relation between sense of direction and topological awareness, hold even if the coding function is unknown to the processors. In other words, the presence of sense of direction, without the knowledge of the specific coding function, would be sufficient for deriving the results of Sections 4.2 and 5. In fact, the processors need to know the coding function only when computing the surrounding to obtain complete knowledge of the topology.

It is apparent from the results presented here that, informally, the presence of sense of direction in a labeled graph lowers the symmetry of the network. The precise relationship between symmetries and sense of direction has been studied in details in [10].

Acknowledgements. We would like to thank Paolo Boldi, Sebastiano Vigna, and Masafumi Yamashita for their helpful comments.

References

[1] D. Angluin. Local and global properties in networks of processors. In *Proc. of 12th A.C.M. Symposium on Theory of Computing*, pages 82–93, 1980.

[2] H. Attiya, M. Snir, and M.K. Warmuth. Computing on an anonymous ring. *Journal of the A.C.M.*, 35(4):845–875, 1988.

[3] P.W. Beame and H.L. Bodlaender. Distributed computing on transitive networks: the torus. In *Proc. of 6th Symposium on Theoretical Aspects of Computer Science*, pages 294–303, 1989.

[4] P. Boldi and S. Vigna. On the complexity of deciding sense of direction. In *Proc. of 2nd Colloquium on Structural Information and Communication Complexity*, International Informatic Series 2, pages 39–52, Olympia, 1995. Carleton University Press.

[5] P. Ferragina, A. Monti, and A. Roncato. Trade-off between computational power and common knowledge in anonymous rings. In *Proc. of 1st Colloquium on Structural Information and Communication Complexity*, International Informatic Series 1, pages 35–48, Ottawa, 1994. Carleton University Press.

[6] J. Fisher, N.A. Lynch, and Merritt. Easy impossibility proofs for distributed consensus. *Distributed Computing*, 1(1):26–39, 1986.

[7] P. Flocchini and B. Mans. Optimal election in labeled hypercubes. *Journal of Parallel and Distributed Computing*, 33(1):76–83, 1996.

[8] P. Flocchini, B. Mans, and N. Santoro. Distributed traversal and broadcasting in arbitrary network with distance sense of direction. In *Proc. of 9th International Symposium on Computer and Information Sciences*, pages 196–203, 1994.

[9] P. Flocchini, B. Mans, and N. Santoro. Sense of direction: formal definition and properties. In *Proc. of 1st Colloquium on Structural Information and Communication Complexity*, International Informatic Series 1, pages 9–34, Ottawa, Canada, 1994. Carleton University Press.

[10] P. Flocchini, A. Roncato, and N. Santoro. Complete symmetries and minimal sense of direction in labeled graphs. In *Proc. of 27th SE Conference on Combinatorics, Graph Theory and Computing*, Congressus Numerantium, 1996. to appear.

[11] P. Flocchini and N. Santoro. Topological constraints for sense of direction. In *Proc. of 2nd Colloquium on Structural Information and Communication Complexity*, International Informatic Series 2, pages 27–38, Olympia, 1995. Carleton University Press.

[12] T.Z. Kalamboukis and S.L. Mantzaris. Towards optimal distributed election on chordal rings. *Information Processing Letters*, 38:265–270, 1991.

[13] E. Kranakis and D. Krizanc. Distributed computing on anonymous hypercubes. In *Proc. of 3rd I.E.E.E. Symposium on Parallel and Distributed Processing*, pages 722–729, Dallas, 1991.

[14] E. Kranakis and D. Krizanc. Labeled versus unlabeled distributed Cayley networks. In *Proc. of 1st Colloquium on Structural Information and Communication Complexity*, International Informatic Series 1, pages 71–82, Ottawa, 1994. Carleton University Press.

[15] E. Kranakis, D. Krizanc, and J. van den Berg. Computing boolean functions on anonymous networks. In *Proc. of 17th International Colloquium on Automata, Languages, and Programming*, Lecture Notes in Computer Science 443, pages 254–267. Springer-Verlag, 1990.

[16] M.C. Loui, T.A. Matsushita, and D.B. West. Election in complete networks with a sense of direction. *Information Processing Letters*, 22:185–187, 1986. see also Information Processing Letters, vol.28, p.327, 1988.

[17] B. Mans and N. Santoro. On the impact of sense of direction in arbitrary networks. In *Proc. of 14th International Conference on Distributed Computing Systems*, pages 258–265, Poznan, 1994.

[18] T. Masuzawa, N. Nishikawa, K. Hagihara, and N. Tokura. Optimal fault-tolerant distributed algorithms for election in complete networks with a global sense of direction. In

Proc. of 3rd International Workshop on Distributed Algorithms, Lecture Notes in Computer Science 392, pages 171–182. Springer-Verlag, 1989.

[19] A. Monti and A. Roncato. A gap theorem for anonymous torus. *Information Processing Letters*, 1996. to appear.

[20] S. Moran and M.K. Warmuth. Gap theorems for distributed computation. *SIAM Journal on Computing*, 22(2):379–394, 1993.

[21] N. Norris. Computing functions on partially wireless networks. In *Proc. of 2nd Colloquium on Structural Information and Communication Complexity*, International Informatic Series 2, pages 53–64, Olympia, 1995. Carleton University Press.

[22] N. Norris. Universal covers of graphs: isomorphism to depth $n-1$ implies isomorphism to all depths. *Discrete Applied Mathematics*, 56:61–74, 1995.

[23] A. Roncato. Gap theorems for anonymous rings. In *Proc. of 2nd Colloquium on Structural Information and Communication Complexity*, International Informatic Series 2, pages 65–76, Olympia, 1995. Carleton University Press.

[24] N. Santoro, J. Urrutia, and S. Zaks. Sense of direction and communication complexity in distributed networks. In *Proc. of the 1st International Workshop on Distributed Algorithms*, pages 123–132, Ottawa, 1985. Carleton University Press.

[25] G. Singh. Leader election in complete networks. In *Proc. of 11th Symposium on Principles of Distributed Computing*, pages 179–190, 1992.

[26] I. Suzuki and M. Yamashita. Distributed anonymous mobile robots. In *Proc. of 3rd Colloquium on Structural Information and Communication Complexity*, International Informatic Series 6, Siena, 1996. Carleton University Press. to appear.

[27] G. Tel. Sense of direction in processor networks. In *Theory and Practice of Informatics*, Lecture Notes in Computer Science 1012, pages 50–82. Springer-Verlag, 1995.

[28] M. Yamashita and T. Kameda. Computing on anonymous networks. In *Proc. of 7th Symposium on Principles of Distributed Computing*, pages 117–130, 1988.

[29] M. Yamashita and T. Kameda. Computing on anonymous networks, part I: characterizing the solvable cases. *IEEE Transaction on Parallel and Distributed Computing*, 7(1):69–89, 1996.

Appendix

Proof (of Property 11)
(Point 1) By induction on d. The property trivially holds for $d = 0$. Let it hold for $d > 0$. Let $A = \{\langle \epsilon, \lambda_u(e)\rangle : e = \langle u, v\rangle \in E\}$, and $B = \bigcup_{v:\langle u,v\rangle\in E}(\lambda_u(\langle u, v\rangle) \cdot \mathcal{E}(T^d(v)))$. By Property 10.3, $\mathcal{E}(T^{d+1}(u)) = A \cup B$. Let $\langle \alpha, \beta\rangle = e \in \mathcal{E}(T^{d+1}(u))$. If $e \in A$, then the thesis trivially holds. If $e \in B$, then by inductive hypothesis the thesis holds for $\mathcal{E}(T^d(v))$. Thus, because of the definition of B, the thesis follows.
(Point 2). By induction on d. The property trivially holds when $d = 0$. Let the property hold for $d > 0$, and let the sets A and B be as in the proof of previous point. If $e \in A$, then the thesis follows by Property 10.3. If $e \in B$, then the thesis follows by induction.
(Point 3) By induction on d. The property holds when $d = 0$ since $\Lambda^0_{(G,\lambda)}[u, u] = \{\epsilon\} = \mathcal{V}(T^0(u))$. Let the property hold for $d > 0$: By Property 10.2, $\alpha \in \mathcal{V}((T^{d+1}(u))$ if and only if 1) $\alpha = \epsilon$, or 2) $\alpha = l \cdot \alpha'$ where $\alpha' \in \mathcal{V}(T^d(v))$ and $l = \lambda_u(\langle u, v\rangle)$. In the first case, $\alpha \in \Lambda^{d+1}[u, u \rightarrow \epsilon]$ which proves the thesis. In case 2, by inductive hypothesis, $\Lambda^d[v, v \rightarrow \alpha']$ if and only if $\alpha' \in \mathcal{V}(T^d(v))$. By definition of $\Lambda[v, v \rightarrow \alpha']$, there is a path starting from v and ending in $v \rightarrow \alpha'$ labeled with α'. But, l is the label of an arc from u to v; thus, $l \cdot \alpha'$ is a label in $\Lambda^{d+1}[u, v \rightarrow \alpha']$. Thus, noting that $u \rightarrow (l \cdot \alpha') = v \rightarrow \alpha'$, the thesis is proved.
(Point 4) This is a special case of the previous point, since $u \rightarrow \alpha$ is defined if and only if $\alpha \in \Lambda^d[u]$. □

Proof (of Propery 14)
(Point 1) We first prove that $[\![\alpha]\!] \cap [\![\beta]\!] \neq \emptyset \Rightarrow [\![\alpha]\!] = [\![\beta]\!]$. If $\gamma \in [\![\alpha]\!] \cap [\![\beta]\!]$, then there are two paths $\pi_1 \in P[u, u, \rightarrow \alpha]$ and $\pi_2 \in P[u, u \rightarrow \beta]$ both labeled with γ. By Property 1, we have that $\pi_1 = \pi_2$ which implies $u \rightarrow \alpha = u \rightarrow \beta$. Thus, by definition of $[\![]\!]$, $[\![\alpha]\!] = [\![\beta]\!]$. The thesis now immediately follows from the definition of $\Lambda^d[u]$.
(Point 2) By definition of surrounding, $\langle [\![\alpha]\!], [\![\beta]\!]\rangle \in \mathcal{E}(N^d(u)) \Leftrightarrow e = \langle u \rightarrow \alpha, u \rightarrow \beta\rangle \in E \wedge d_G(u, e) \leq d$. The distance between u and e is less or equal to d if and only if $\exists \pi \in P^{d-1}_G[u, u \rightarrow \alpha]$. Let $\alpha' = \Lambda_u(\pi)$ and $\beta' = \alpha' \cdot \lambda_{u\rightarrow\alpha}(e)$. The thesis follows noting that $u \rightarrow \alpha = u \rightarrow \alpha'$ and $u \rightarrow \beta = u \rightarrow \beta'$ because of the definition of $[\![\alpha]\!]$ and $[\![\beta]\!]$.
(Point 3) Let $f : \mathcal{V}(N^d(u)) \rightarrow \{v \in V : d_G(u, v) \leq d\}$ be the following function: $f([\![\alpha]\!]) = u \rightarrow \alpha$. In order to prove the thesis, we will prove that f is bijective. First of all we will prove that f is one-to-one. Let $f([\![\alpha]\!]) = f([\![\beta]\!])$, that is $u \rightarrow \alpha = u \rightarrow \beta$. By definition of $[\![\alpha]\!]$, $\beta \in [\![\alpha]\!]$ and by Point 1, $[\![\alpha]\!] = [\![\beta]\!]$. We will now prove that f is onto. Let $z \in \{v \in V : d_G(u, v) \leq d\}$. $d_G(u, z) \leq d$ implies that there exists $\pi \in P[u, z]$ and $|\pi| \leq d$. Thus, $\alpha = \Lambda_u(\pi) \in \Lambda^d[u]$ and from Point 1, it follows that $[\![\alpha]\!] \in V(N^d(u))$. Moreover, $f([\![\alpha]\!]) = u \rightarrow \alpha = z$.
(Point 4) The proof is similar to the one of Point 3. □