Labelled (Hyper)Graphs, Negotiations and the Naming Problem*

Jérémie Chalopin[1], Antoni Mazurkiewicz[2], and Yves Métivier[3]

[1] LIF, Aix-Marseille Université
39 rue Joliot-Curie
13453 Marseille France
jeremie.chalopin@lif.univ-mrs.fr

[2] Institue of Computer Science of PAS
Ordona 21, PL-01-237
Warsaw, Poland
amaz@ipipan.waw.pl

[3] Université de Bordeaux, LaBRI
351 Cours de la Libération
33405 Talence France
metivier@labri.fr

Abstract. We consider four different models of process interactions that unify and generalise models introduced and studied by Angluin et al. [AAD$^+$04] and models introduced and studied by Mazurkiewicz [Maz04a,Maz04b]. We encode these models by labelled (hyper)graphs and relabelling rules on this labelled (hyper)graphs called negotiations. Then for these models, we give complete characterisations of labelled graphs in which the naming problem can be solved. Our characterizations are expressed in terms of locally constrained homomorphisms that are generalisations of known graph homomorphisms.

1 Introduction

Three major process influence (interaction) models in distributed systems have principally been considered: the message passing model, the shared memory model, the local computation model. In the three models the processes are represented by vertices of a graph and the interactions are represented by edges or hyperedges. In the message passing model processes interact by messages: they can be sent along edges. In the shared memory model atomic read/write operations can be performed on registers associated with hyperedges. In the local computation model, interactions are defined by labelled graph relabelling rules; supports of rules (graphs used for the description of labellings) are edges

* This work was supported by grant No ANR-06-SETI-015-03 awarded by Agence Natioanle de la Recherche

or stars. These models (and their sub-models) reflect different system architectures, different levels of synchronization and different levels of abstraction. The structure of the communication or of the interaction subsystem is represented as a graph. In general this graph is static: it means that it remains fixed during the distributed computation. Some works consider dynamic graphs: some links may fail and recover and some nodes may fail and recover.

In this paper we consider local computations on another kind of dynamic distributed system: processes are mobile and they interact when they are sufficiently close to each other or when some localisation conditions are verified. In [AAD$^+$04], Angluin et al. consider a distributed system where a set of moving sensors can interact when they are sufficiently close to each other. They assume that every pair of sensors repeatedly come sufficiently close to each other for being able to communicate, i.e., the interaction graph is the complete graph. In their work, they consider finite-state sensors (each sensor has a constant number of bits of memory) and they study the computational power of such a system.

In [Maz04a,Maz04b] the distributed system is presented in the following way. There is a number of individuals, each of them brings an integer as a label. They are grouped into associations: within an association they can communicate, exchange information, and modify their labels; there is no possibility of direct communication between individuals that do not belong to the same association. However, since some individuals can be affiliated to more than one association, indirect communication between remote individuals is possible using individuals with multiple affiliations as go between. Such systems are called communication structures. Associations act by their assemblies that take place from time to time; an association is active during its assembly, and passive out of it. The purpose of an assembly is to exchange information among participants and to update the states of the participants.

In this work, we consider a system where the processes have an unbounded number of states and where a computation step can involve an arbitrary number of processes. Moreover, we do not assume that each process can interact with any other process: we just assume that the communication structure is connected. We study the computational power of such systems through the naming problem.

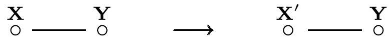
The naming problem. We focus on the naming problem, that is a classical problem to highlight differences between different models of distributed computing. A distributed algorithm $\mathcal{A}$ is a naming algorithm if each execution of $\mathcal{A}$ terminates and leads to a final configuration where all processes have unique identities. Being able to give dynamically and in a distributed way unique identities to all processes is very important since many distributed algorithms work correctly only under the assumption that all processes can be unambiguously identified. In this paper naming is done using a distributed enumeration algorithm. A distributed enumeration algorithm assigns to each network vertex a unique integer ; in such a way we obtain a bijection between the set $V(G)$ of vertices and $\{1, 2, \ldots, |V(G)|\}$.

The study of the naming problem makes it possible to highlight combinatorial tools useful for other problems like termination detection or recognition (see [YK96a,YK96b,BCG$^+$96,BV01]).

Formal Models. A *communication structure* is defined by a set of *individuals* that belong to different *associations.* Some individuals can be affiliated to more than one association. A communication structure is represented as an undirected hypergraph: vertices represent individuals and hyperedges define associations. In the particular case where all associations have exactly two members, a communication structure can be seen as a simple graph. Labels (states) are attached to individuals and associations thus we consider labelled (hyper)graphs which are defined by a labelling function λ which associates to a vertex or an (hyper)edge a label. In the more general model of computation called *labelled negotiations*, a computation step enables to modify the states of the vertices of a hyperedge and the label of the hyperedge itself according only to their previous states. In other words, in one computation step, the members of an association synchronize, exchange their labels and modify them. We consider communication structures where hyperedges cannot be labelled and then vertices cannot always distinguish the different hyperedges they belong to. This restriction leads us to study *unlabelled negotiations* We also consider models of computations where in one computation step, one vertex observes the states of the vertices of a hyperedge it belongs to and the state of the hyperedge (if available) and modifies only its state and the state of the hyperedge (if this one is available). Such a model of computation will be called *cellular.* Thus, we study *cellular (un)labelled negotiations.*

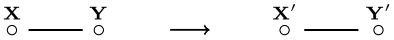
Our results. We characterize labelled (hyper)graphs where the naming problem can be solved in the four different models we consider. We first show that cellular labelled negotiations have the same computational power as labelled negotiations (Proposition 1). To give our characterization, we generalize locally constrained graph homomorphisms to hypergraphs (Section 2). This enables us to formulate conveniently necessary conditions (Lemma 1) inspired by Angluin's lifting lemma [Ang80]. It turns out that the necessary conditions are also sufficient. Then we present algorithms that solve the naming problem (Theorems 1, 2 and 3) (Theorem 1 is another formulation of a result presented in [Maz04a]).

Related Work. In [AAD$^+$04,AAER05,AAE06] Angluin et al. study the power of models of computation by pairwise interactions of identical finite-state agents. The general question is of characterising what computations are possible. They prove in particular that all predicates stably computable are semilinear, in the model in which finite state agents compute a predicate of their inputs via two-way interactions in the all-pairs family of communication networks [AAE06]. This kind of computations may be encoded by local computations on edges of labelled graphs. The case of one-way communication between two agents corresponds to cellular local computations on edges, thus of the form:



where X, Y and X' are labels (states) attached to vertices, $X' = f(X, Y)$ and f is a transition function. In [CMZ06], a complete characterization of labelled graphs for which enumeration and election are possible is presented.

The case of two-way communication corresponds to local computations on edges of labelled graphs, thus of the form:



where X, Y, X' and Y' are labels (states) attached to vertices, $X' = f_1(X, Y)$, $Y' = f_2(Y, X)$ and f_1, f_2 are transition functions. Graphs for which the naming problem is solvable are characterized in [Cha05].

All-pairs family of communication network is captured by our model by considering the case where each association has no name and has exactly two members, and the set of associations defines the complete graph. The two-way interaction model of [AAE06] corresponds to our general model of computation. The one-way population protocol defined in [AAER05] corresponds to the cellular computation model. In [Maz04a,Maz04b] associations are labelled and cellular relabellings are not considered.

Overview. The structure of this paper is as follows. Section 2 reviews basic definitions of communication structures and negotiations. In Section 3 first we prove that labelled negotiations can be simulated by cellular labelled negotiations, then we present characterisations of communication structures which admit a naming algorithm using (cellular) labelled negotiations. Section 4 presents characterisations of communication structures which admit a naming algorithm using (cellular) unlabelled negotiations. Section 5 presents final remarks.

2 Preliminaries

2.1 Communication Structures and Labelled Graphs

A *communication structure* C is defined by a set $B(C)$ of *individuals* and a set $A(C)$ of *associations*: each association is a set of individuals. Each individual $b \in B(C)$ belongs to one or more associations $a \in A(C)$ and it will be denoted by $b \in a$; one will say that b is a *member* of a and that a *contains* b. Each association may contain an arbitrary number of elements and two distinct associations can have the same members. A communication structure is connected if for any associations $a, a' \in A(C)$, there exists a sequence $a_0, a_1, \ldots, a_n$ such that $a_0 = a$, $a_n = a'$ and for any $i \in [1, n]$, there exists an individual that belongs to a_{i-1} and a_i. In the following, we will only consider connected communication structures.

A communication structure C is bilateral if each association contains exactly two elements. In this case C can be represented by a graph where vertices are individual and edges are associations.

A communication structure C may be viewed as a hypergraph where vertices denote individuals and hyperedge denote associations. It will be represented by a simple bipartite graph G_C that is a classical representation of hypergraphs. The set of vertices $V(G_C)$ contains two disjoint subsets $V_A(G_C)$ and $V_B(G_C)$.

Each association a (resp. individual b) of C corresponds to a vertex $v_a \in V_A(G_C)$ (resp. $v_b \in V_B(G_C)$). If an individual b belongs to an association a, then there is an edge $\{v_a, v_b\}$ in $E(G_C)$. Given a vertex $v \in V_A(G_C) \cup V_B(G_C)$, $N_{G_C}(v)$ denotes the set of neighbours of $v \in G_C$, i.e., the set $\{v' \mid \{v, v'\} \in E(G_C)\}$. A graph homomorphism φ from G to G' is a mapping from $V(G)$ to $V(G')$ such that if $\{v, w\} \in E(G)$, then $\{\varphi(v), \varphi(w)\} \in E(G')$.

We want to extend to communication structures the definitions of coverings, pseudo-coverings and submersions that are used in [Cha05,CM04,CMZ04,GMM04] for graphs, that are bilateral communication structures studied in [Maz04a]. We give a definition of homomorphism between communication structures: it is a generalization of graph homomorphisms.

Definition 1. *Given two communications structures C and C', a mapping φ from $B(C)$ to $B(C')$ and from $A(C)$ to $A(C')$ is a homomorphism from C to C' if it induces a graph homomorphism of G_C to $G_{C'}$ such that for each vertex $v_a \in V_A(G_C)$, the following holds: (1) $|N_{G_C}(v_a)| = |N_{G_{C'}}(\varphi(v_a))|$, and (2) $\varphi(N_{G_C}(v_a)) = N_{G_{C'}}(\varphi(v_a))$.*

Throughout the paper we will consider communication structures where individuals and associations are labelled with labels from a recursive label set L that admits a total order $<_L$. A labelled communication structure will be denoted by $\mathbf{C} = (C, \lambda)$ where C is the underlying unlabelled communication structure and where $\lambda : B(C) \cup A(C) \to L$ is a labelling function. A mapping φ from $\mathbf{C} = (C, \lambda)$ to $\mathbf{C}' = (C', \lambda')$ is a homomorphism if φ is a homomorphism from C to C' that preserves the labelling, i.e., for each $x \in B(C) \cup A(C), \lambda(x) = \lambda'(\varphi(x))$.

For any set S, $|S|$ denotes the cardinality of S while $\mathcal{P}_{\text{fin}}(S)$ is the set of finite subsets of S.

2.2 Locally constrained homomorphisms

We now define submersions, coverings and pseudo-coverings of communication structures that are just generalizations of existing definitions for graphs. A communication structure $\mathbf{C}$ is a submersion of $\mathbf{C}'$ if there exists a locally surjective homomorphism from $\mathbf{C}$ to $\mathbf{C}'$.

Definition 2. *Given two communication structures $\mathbf{C} = (C, \lambda)$ and $\mathbf{C}' = (C', \lambda')$, $\mathbf{C}$ is a* submersion *of $\mathbf{C}'$ via a homomorphism φ if for each vertex $v_b \in V_B(G_C)$, $\varphi(N_{G_C}(b)) = N_{G_{C'}}(\varphi(b))$. In this case, we say that φ is a locally surjective homomorphism from $\mathbf{C}$ to $\mathbf{C}'$.*

In other words, a homomorphism φ from $\mathbf{C}$ to $\mathbf{C}'$ is locally surjective if for each individual $b \in B(C)$, the associations that contain $\varphi(b)$ are the images of the associations that contain b. A communication structure $\mathbf{C}$ will be called *submersion-minimal* if for any $\mathbf{C}'$ with $|B(C')| < |B(C)|$, $\mathbf{C}$ is not a submersion.

A communication structure $\mathbf{C}$ is a covering of $\mathbf{C}'$ if there exists a locally bijective homomorphism from $\mathbf{C}$ to $\mathbf{C}'$.

Definition 3. *Given two communication structures* $\mathbf{C}$ *and* $\mathbf{C}'$*,* $\mathbf{C}$ *is a* covering *of* $\mathbf{C}'$ *via a homomorphism* φ *if for each vertex* $v_b \in V_B(G_C)$*,* $|N_{G_C}(v_b)| = |N_{G_{C'}}(\varphi(v_b))|$ *and* $\varphi(N_{G_C}(v_b)) = N_{G_{C'}}(\varphi(v_b))$*. In this case, we say that* φ *is a locally bijective homomorphism from* C *to* C'*.*

In other words, a homomorphism φ from $\mathbf{C}$ to $\mathbf{C}'$ is locally bijective if for each individual $b \in B(C)$ φ induces a bijection between the associations that contain b and the associations that contain $\varphi(b)$. A communication structure $\mathbf{C}$ will be called *covering-minimal* if $\mathbf{C}$ is not a covering of any $\mathbf{C}'$ such that $|B(C')| < |B(C)|$.

We now define pseudo-coverings that generalize pseudo-coverings of graphs introduced in [Cha05].

Definition 4. *Given two communication structures* $\mathbf{C} = (C, \lambda)$ *and* $\mathbf{C}' = (C', \lambda')$*,* $\mathbf{C}$ *is a* pseudo-covering *of* $\mathbf{C}'$ *via a homomorphism* φ *if there exists a subset* A_0 *of* $A(C)$ *such that the communication structure* $\mathbf{C}_0 = (C_0, \lambda_0)$ *defined by* $B(C_0) = B(C)$*,* $A(C_0) = A_0$ *and for each* $x \in B(C_0) \cup A(C_0), \lambda_0(x) = \lambda(x)$ *is a covering of* $\mathbf{C}'$ *via the restriction of* φ *to* C_0*.*

A communication structure $\mathbf{C}$ will be called *pseudo-covering-minimal* if $\mathbf{C}$ is not a pseudo-covering of any $\mathbf{C}'$ such that $|B(C')| < |B(C)|$.

Obviously, if a communication structure $\mathbf{C}$ is a covering of $\mathbf{C}'$, then $\mathbf{C}$ is a pseudo-covering of $\mathbf{C}'$ and if $\mathbf{C}$ is a pseudo-covering of $\mathbf{C}'$, then $\mathbf{C}$ is a submersion of $\mathbf{C}'$.

2.3 Negotiations and Relabelling Rules

In our models, in one computation step the states of an association and its members are modified according only to their previous states. An algorithm can then be described by a set $\mathcal{R}$ of relabelling rules $r = (\lambda_r, \lambda'_r)$ where λ_r and λ'_r are two labellings of an association. A computation step is then an application of a rule to some association of the communication structure. We will note $\mathbf{C}\mathcal{R}\mathbf{C}'$ if $\mathbf{C}'$ can be obtained from $\mathbf{C}$ by applying a rule of $\mathcal{R}$ to some association of $\mathbf{C}$. Obviously, $\mathbf{C}$ and $\mathbf{C}'$ have the same underlying communication structure C, only the labelling of the active association is modified. Thus, slightly abusing the notation, $\mathcal{R}$ will stand both for a set of rules and the induced relabelling relation over labelled communication structure. The transitive closure of such a relabelling relation is noted $\mathcal{R}^*$. Computations using uniquely this type of relabelling rules are called in our paper *negotiations*.

The relation $\mathcal{R}$ is called *noetherian* on a communication structure $\mathbf{C}$ if there is no infinite relabelling sequence $\mathbf{C}_0\mathcal{R}\mathbf{C}_1\mathcal{R}\dots$ with $\mathbf{C}_0 = \mathbf{C}$. The relation $\mathcal{R}$ is noetherian if it is noetherian on each communication structure. Clearly, noetherian relations code always terminating algorithms.

An algorithm encoded with such computation rules is a distributed algorithm in the sense that two computation steps can be applied simultaneously to two distinct associations, provided that no individual belongs to both of associations.

In the following, we will consider four different models of negotiations. The most general model described above is called *labelled negotiations*. We will also deal with communication structures where the associations cannot be labelled, this model will be called *unlabelled negotiations*. We will also consider models where in one computation step, the label of at most one member can be modified, i.e., in one computation step, one member modifies its label and the label of an association it belongs to (if associations can be labelled) according to the labels of all the members of this association and to the label of the association (if the associations can be labelled). When the associations can be labelled, the model corresponding to this kind of computation steps will be called *cellular labelled negotiations* and when the associations cannot be labelled, it will be called *cellular unlabelled negotiations*.

Given a terminating algorithm $\mathcal{A}$ using labels in a set L, one will say that an algorithm $\mathcal{A}'$ using labels in a set L' simulates $\mathcal{A}$ if there exists a mapping $\pi : L' \to L$ such that for any communication structure $\mathbf{C}$, each execution of $\mathcal{A}'$ on $\mathbf{C}$ terminates and for each execution of $\mathcal{A}'$ on $\mathbf{C}$ with a final labelling λ', there exists an execution of $\mathcal{A}$ on $\mathbf{C}$ with a final labelling λ on $\mathbf{C}$, such that for each $x \in B(C) \cup A(C)$, $\pi(\lambda'(x)) = \lambda(x)$. It is obvious that any algorithm using cellular labelled negotiations is an algorithm that uses labelled negotiations; in fact, cellular labelled negotiations have the same computational power as labelled negotiations:

Proposition 1. *Any algorithm $\mathcal{A}$ using labelled negotiations can be simulated by an algorithm $\mathcal{A}'$ that uses cellular labelled negotiations.*

In the following, we say that an algorithm $\mathcal{A}$ has the termination detection property if for any execution of $\mathcal{A}$ on $\mathbf{C}$, there exists an individual $b \in B(C)$ that can detect locally (according only to its state) that the computation is over, i.e., that each individual has computed its final value.

An algorithm will be described by a recursive set of rules of the form (λ_r, λ'_r). We can see each rule r as a couple of two multisets :

$$(\{\{\lambda_{r,0}, \lambda_{r,1}, \ldots, \lambda_{r,k}\}\}, \{\{\lambda'_{r,0}, \lambda'_{r,1}, \ldots, \lambda'_{r,k}\}\}).$$

We can apply r to an association a if a has k members, if the label of a (if available) is $\lambda_{r,0}$ and if the multisets of labels $\{\{\lambda(b) \mid b \in a\}\}$ is equal to $\{\{\lambda_{r,1}, \ldots, \lambda_{r,k}\}\}$. In this case, the label of a becomes $\lambda'_{r,0}$ and the label of each member of a labelled by $\lambda_{r,i}$ becomes $\lambda'_{r,i}$. In the case of cellular negotiations, for each $i > 1$, we should have $\lambda'_{r,i} = \lambda_{r,i}$.

When we want to describe a set of rules that do not depend on the size of the association a, we will write the precondition as a logical formula that the labels of a and its members must satisfy to apply the rule. Then we describe the new labels of a and of each member of a. This description enables to encode an infinite number of relabelling rules (one for each size of association) in a finite way.

2.4 Impossibility Result

The following lemma exhibits a strong link between homomorphisms and negotiations. This is a counterpart of the lifting lemma of Angluin [Ang80] adapted to communication structure homomorphisms.

Lemma 1 (Lifting Lemma). *Let $\mathcal{R}$ be a relabelling relation corresponding to an algorithm using labelled negotiations (resp. unlabelled negotiations, cellular unlabelled negotiations) and let $\mathbf{C}_1$ be a covering (resp. pseudo-covering, submersion) of $\mathbf{C}_2$. If $\mathbf{C}_2\mathcal{R}^*\mathbf{C}'_2$, then there exists $\mathbf{C}'_1$ such that $\mathbf{C}_1\mathcal{R}^*\mathbf{C}'_1$ and $\mathbf{C}'_1$ is a covering (resp. pseudo-covering, submersion) of $\mathbf{C}'_2$.*

Consequently, there cannot exist a naming algorithm $\mathcal{A}$ using labelled negotiations (resp. unlabelled negotiations, cellular unlabelled negotiations) on a communication structure $\mathbf{C}_1$ that is not covering-minimal (resp. pseudo-covering-minimal, submersion-minimal). Indeed, if $\mathbf{C}_1$ is a covering (resp. pseudo-covering, submersion) of $\mathbf{C}_2$ with $|B(C_2)| < |B(C_1)|$, consider a terminating execution ρ of $\mathcal{A}$ on $\mathbf{C}_2$ that leads to a final configuration $\mathbf{C}'_2$. From Lemma 1, one can construct a terminating execution on $\mathbf{C}_1$ from ρ that leads to a final configuration $\mathbf{C}'_1$ that is a covering (resp. pseudo-covering, submersion) of $\mathbf{C}'_2$. Consequently, there exists an individual in $\mathbf{C}'_2$ whose label appears at least twice in $\mathbf{C}'_1$: individuals do not have unique identities in $\mathbf{C}'_1$.

3 (Cellular) Labelled Negotiations

In this section, we give a characterization of communication structures where we can solve the naming problem using cellular labelled negotiations or labelled negotiations. We give a naming algorithm using labelled negotiations that solves the enumeration problem on any communication structure $\mathbf{C}$ that is covering-minimal.

Each individual b (resp. association a) attempts to get a number between 1 and $|B(C)|$ (resp. $|A(C)|$). Each individual (resp. association) chooses a number and collects the numbers of the associations it belongs to (resp. the numbers of its members) to construct its *local view*. Then, each individual and each association broadcasts its number with its label and its local view. If some individual b (resp. an association a) detects that there exists another individual b' (resp. another association a') with the same number, then it compares its label and its local view with the label and the local view of its opponent. If the label or the local view of b is "weaker", then b chooses a new number and broadcasts it again. At the end of the computation, each individual and each association will have a unique number if the communication structure is covering-minimal.

Labels. Consider a communication structure $\mathbf{C} = (C, \lambda)$ with an initial labelling $\lambda\colon B(C) \cup A(C) \to L$. During the computation each individual $b \in B(C)$ will acquire new labels of the form $(\lambda(b), n(b), N(b), M(b), S(b))$ and each association $a \in A(C)$ will get labels of the form $(\lambda(a), m(a), P(a))$ where:

- the first component $\lambda(b)$ (resp. $\lambda(a)$) is just the initial label (and thus remains fixed during the computation),
- $n(b) \in \mathbb{N}$ (resp. $m(a) \in \mathbb{N}$) is the current *identity number* of b (resp. a) computed by the algorithm,
- $N(b) \in \mathcal{P}_{\text{fin}}(\mathbb{N})$ (resp. $P(a) \in \mathcal{P}_{\text{fin}}(\mathbb{N})$) is the *local view* of b (resp. a). Intuitively, the algorithm will try to update the current view in such a way that $N(b)$ (resp. $P(a)$) will consist of the current identities of the associations

that contains b (resp. of the members of a). Therefore $N(b)$ (resp. $P(a)$) will be always a finite (possibly empty) set of integers,

- $M(b) \subseteq \mathbb{N} \times L \times \mathcal{P}_{\mathrm{fin}}(\mathbb{N})$ is the current *individual-mailbox* of b. It contains the whole information about individuals received by b during the computation.
- $S(b) \subseteq \mathbb{N} \times L \times \mathcal{P}_{\mathrm{fin}}(\mathbb{N})$ is the current *association-mailbox* of b. It contains the whole information about associations received by b during the computation.

The fundamental property of the algorithm is based on a total order on the set $\mathcal{P}_{\mathrm{fin}}(\mathbb{N})$ of local views, as defined by Mazurkiewicz [Maz97]. Consider two sets N_1, N_2 of integers. Suppose that $N_1 \neq N_2$. Then $N_1 \prec_1 N_2$ if the maximal element of the symmetric difference $N_1 \triangle N_2 = (N_1 \setminus N_2) \cup (N_2 \setminus N_1)$ belongs to N_2. Note that in particular the empty set is minimal for $\prec_1$. If $N(b) \prec_1 N(b')$ then we say that the local view $N(b')$ of b' is stronger than the one of b (and $N(b)$ is weaker than $N(b')$). We extend $\prec_1$ to a total order on $L \times \mathcal{P}_{\mathrm{fin}}(\mathbb{N})$: $(\ell, N) \prec_1 (\ell', N')$ if either $\ell <_L \ell'$ or ($\ell = \ell'$ and $N \prec_1 N'$). We will also use the reflexive closure $\preceq_1$ of $\prec_1$.

Labelled negotiations rules. We describe here the relabelling rules that define the enumeration algorithm. First of all, to launch the algorithm there is a special initial rule $\mathcal{R}_0$ that just extends the initial label $\lambda(b)$ (resp. $\lambda(a)$) of each individual b (resp. association a) to $(\lambda(b), 0, \emptyset, \emptyset, \emptyset)$ (resp. $(\lambda(a), 0, \emptyset)$). The label of an association a (resp. a member b of a) obtained by the application of a rule to a is denoted $(\lambda(a), m'(a), P'(a))$ (resp. $(\lambda(b), n'(b), N'(b), M'(b), S'(b))$).

The first rule $\mathcal{R}_1$ enables to update the mailboxes of all the individuals that belongs to a same association.

```
$\mathcal{R}_1$ :
if $\exists b, b' \in a, M(b) \neq M(b')$ or $S(b) \neq S(b')$ then
    $\forall b \in a, M'(b) := \bigcup_{b \in a} M(b)$ and $S'(b) := \bigcup_{b \in a} S(b)$ ;
```

The second rule $\mathcal{R}_2$ does not involve any synchronisation. It enables an individual b to change its identity if its current identity number $n(b)$ is 0 or if it knows that there exists another individual with the same number and a stronger label or a stronger local view.

```
$\mathcal{R}_2$ :
if $n(b) = 0$ or $\exists (n(b), \ell, N) \in M(b)$ such that $(\lambda(b), N(b)) \prec_1 (\ell, N)$ then
    $n'(b) := 1 + \max\{n' \mid (n', \ell', N') \in M(b)\}$;
    $M'(b) := M(b) \cup \{(n'(b), \lambda(b), N(b))\}$;
```

The rules $\mathcal{R}_3, \mathcal{R}_4, \mathcal{R}_5$ are designed such that one can apply one of these rules to some association a only if one cannot apply the preceding ones to a. The third rule enables an individual b_0 to modify its identity if it belongs to some association a such that there exists another individual $b_1 \in a$ with the same number, the same label and the same local view.

```
$\mathcal{R}_3$ :
  if $\forall b, b' \in a, M(b) = M(b')$ and $S(b) = S(b')$ and $\forall b \in a, n(b) \neq 0$ and
  $\forall (n(b), \ell, N) \in M(b), (\ell, N) \preceq_1 (\lambda(b), N(b))$ and $\exists b_0, b_1 \in a$ such that $n(b_0) = n(b_1)$
  then
      $n'(b_0) := 1 + \max\{n' \mid (n', \ell', N') \in M(b_0)\}$;
      $M'(b_0) := M(b_0) \cup \{(n'(b_0), \lambda(b_0), N(b_0))\}$;
```

The fourth rule is the counterpart for associations of the second rule. It enables to modify the identity of an association a if the current identity number $m(a)$ is 0 or if there exists another association with the same number and a stronger label or a stronger local view. When this rule is applied to a, the local view of the members of a is updated.

```
$\mathcal{R}_4$ :
  if $\forall b, b' \in a, M(b) = M(b')$, $S(b) = S(b')$ and $n(b) \neq n(b')$,
  $\forall b \in a, n(b) \neq 0$ and $\forall (n(b), \ell, N) \in M(b), (\ell, N) \preceq_1 (\lambda(b), N(b))$
  and $m(a) = 0$ or $\exists (m(a), \ell, P) \in S(b)$ such that $(\lambda(a), P(a)) \prec_1 (\ell, P)$ then
      $m'(a) := 1 + \max\{m' \mid (m', \ell', P') \in M(b)\}$;
      $P'(a) := \{n(b) \mid b \in a\}$;
      $\forall b \in a, N'(b) := N(b) \setminus \{m(a)\} \cup \{m'(a)\}$;
      $\forall b \in a, M'(b) := M(b) \cup \{(n(b'), \lambda(b'), N'(b')) \mid b' \in a\}$;
      $\forall b \in a, S'(b) := S(b) \cup \{(m'(a), \lambda(a), P'(a))\}\}$;
```

The last rule enables to update the local view of an association.

```
$\mathcal{R}_5$ :
  if $\forall b, b' \in a, M(b) = M(b')$, $S(b) = S(b')$ and $n(b) \neq n(b')$
  and $\forall b \in a, n(b) \neq 0$ and $\forall (n(b), \ell, N) \in M(b), (\ell, N) \preceq_1 (\lambda(b), N(b))$
  and $p(a) \neq 0$ and $\forall (m(a), \ell, P) \in S(b), (\ell, P) \preceq_1 (\lambda(a), P(a))$
  and $\exists b_0 \in a$ such that $n(b_0) \notin P(a)$ then
      $P'(a) := \{n(b) \mid b \in a\}$;
      $\forall b \in a, S'(b) := S(b) \cup \{(m(a), \lambda(a), P(a))\}\}$;
```

Any execution of the algorithm satisfies monotonicity properties. Any run ρ of the algorithm on a labelled communication structure $\mathbf{C} = (C, \lambda)$ terminates and yields a final labelling $(\lambda, n_\rho, N_\rho, M_\rho, S_\rho)$ of individuals and a final labelling $(\lambda, m_\rho, P_\rho)$ of associations.

The mapping defined by n_ρ and m_ρ is a locally bijective homomorphism from $\mathbf{C}$ to $\mathbf{C}'_\rho$. Consequently, if $\mathbf{C}$ is covering-minimal, it implies that in the final configuration, $\{n_\rho(b) \mid b \in B(C)\} = [1, |B(C)|]$. From Lemma 1 and Proposition 1, we get the following theorem.

Theorem 1. *For every communication structure* $\mathbf{C}$*, there exists a naming algorithm for* $\mathbf{C}$ *using (cellular) labelled negotiations if and only if* $\mathbf{C}$ *is covering-minimal.*

Suppose that all the individuals initially know $|B(C)|$, then the termination detection of the algorithm is possible on a covering-minimal communication structure $\mathbf{C}$. Indeed, once an individual gets the identity number $|B(C)|$, it knows that all the individuals have different identity numbers that will not change any more.

4 (Cellular) Unlabelled Negotiations

We now consider unlabelled negotiations and cellular unlabelled negotiations. We give characterizations of communication structures where we can solve the naming problem using these two kinds of negotiations. A corollary of these characterizations is that unlabelled negotiations have a strictly greater computational power than cellular unlabelled negotiations.

4.1 Cellular Unlabelled Negotiations

The algorithm uses the same ideas as the algorithm of the previous section. The main difficulty is to achieve to update correctly the local view of the individuals.

Consider a communication structure $\mathbf{C} = (C, \lambda)$ with an initial labelling $\lambda \colon B(C) \to L$. During the computation each individual $b \in B(C)$ will acquire new labels of the form $(\lambda(b), n(b), N(b), M(b))$ where:

- $n(b) \in \mathbb{N}$ is the identity number of b.
- $N(b) \in \mathcal{P}_{\text{fin}}(\mathcal{P}_{\text{fin}}(\mathbb{N}))$ is the local view of b. Intuitively, the algorithm will try to update the local view of b such that $N(b)$ contains a set $\{n(b') \mid b' \in a\}$ for each association a that contains b.
- $M(b)$ is the mailbox of b and it contains the whole information the individual b has about the network.

We also need to define a total order on local views. We will just generalize the order defined above. Consider two sets N_1, N_2 whose elements are some sets of integers ordered by $\prec_1$. Suppose that $N_1 \neq N_2$. Then $N_1 \prec_2 N_2$ if the maximal element for $\prec_1$ of the symmetric difference $N_1 \triangle N_2 = (N_1 \backslash N_2) \cup (N_2 \backslash N_1)$ belongs to N_2. Again, we extend $\prec_2$ to a total order on $L \times \mathcal{P}_{\text{fin}}(\mathcal{P}_{\text{fin}}(\mathbb{N}))$: $(\ell, N) \prec_2 (\ell', N')$ if either $\ell <_L \ell'$ or ($\ell = \ell'$ and $N \prec_2 N'$).

Cellular unlabelled negotiations rules. The label of an individual b_0 after the application of a relabelling rule to an association a that modifies the state of b_0 is denoted by $(\lambda(b_0), n'(b_0), N'(b_0), M'(b_0))$. The three first rules have the same meaning as the three first rules of the algorithm described in the previous section.

```
$\mathcal{R}_1$ :
if $\exists b \in a, M(b) \setminus M(b_0) \neq \emptyset$ then
    $M'(b_0) := \bigcup_{b \in a} M(b);$

$\mathcal{R}_2$ :
if $n(b_0) = 0$ or $\exists (n(b_0), \ell, N) \in M(b_0)$ such that $(\lambda(b_0), N(b_0)) \prec_2 (\ell, N)$ then
    $n'(b_0) := 1 + \max\{n' \mid (n', \ell', N') \in M(b_0)\};$
    $M'(b_0) := M(b_0) \cup \{(n'(b_0), \lambda(b_0), N(b_0))\};$
```

```
$\mathcal{R}_3$ :
if $\forall b, b' \in a, M(b) = M(b')$,
$\forall b \in a, n(b) \neq 0$ and $\forall (n(b), \ell, N) \in M(b), (\ell, N) \preceq_2 (\lambda(b), N(b))$
and $\exists b \in a, b \neq b_0$ such that $n(b_0) = n(b)$ then
    $n'(b_0) := 1 + \max\{n' \mid (n', \ell', N') \in M(b_0)\}$;
    $M'(b_0) := M(b_0) \cup \{(n'(b_0), \lambda(b_0), N(b_0))\}$;
```

The fourth rule enables an individual b to add the set $S' = \{n(b') \mid b' \in a\}$ of the current identity numbers of the members of an association a it belongs to. In this case, all the sets S belonging to $N(b)$ such that $S \prec_1 S'$ are removed.

The intuitive justification for the deletion of all such S is the following. Suppose that there exists an association a that contains b such that the set $S' = \{n(b') \mid b' \in a\}$ does not belong to $N(b)$. Suppose that there is a computation step that enables to modify the state of b according to the states of the members of a. Then, since the very purpose of the view $N(b)$ is to stock the identity numbers of all the members of all the associations it belongs to, we should add S' to the view $N(b)$ of b.

If the state of b is modified according to a for the first time, then adding S' to $N(b)$ is sufficient. But, it can also be the case that b modified its state according to a in the past and in the meantime another member b' of a has modified its identity. Then b should not only add S' to $N(b)$ but it should remove the old set of identity numbers corresponding to a from its view. The problem is that b cannot know which set it should remove from its view. However, since our algorithm ensures that the identity numbers of individuals can only increase, we know that the eventual old set of numbers S is weaker for $\prec_1$ than S' and consequently, by removing all the $S \prec_1 S'$, we are sure to delete all invalid information. Of course, we can also delete legitimate informations from the local view of b. But in this case, b can recover this information by some new applications of $\mathcal{R}_4$ to the other associations that contain b.

```
$\mathcal{R}_4$ :
if $\forall b, b' \in a, M(b) = M(b')$ and $n(b) \neq n(b')$,
$\forall b \in a, n(b) \neq 0$ and $\forall (n(b), \ell, N) \in M(b), (\ell, N) \preceq_2 (\lambda(b), N(b))$
and $\{n(b') \mid b' \in a\} \notin N(b_0)$ then
    $S' := \{n(b') \mid b' \in a\}$;
    $N'(b_0) := N(b_0) \setminus \{S \mid S \prec_1 S'\} \cup \{S'\}$;
    $M'(b_0) := M(b_0) \cup \{(n(b_0), \lambda(b_0), N'(b_0))\}$;
```

Properties. The algorithm we described has the same interesting properties as the one described in Section 3. And from Lemma 1, we get:

Theorem 2. *For every communication structure* $\mathbf{C}$*, there exists a naming algorithm for* $\mathbf{C}$ *using cellular unlabelled negotiations if and only if* $\mathbf{C}$ *is submersion-minimal.*

Again, if the individuals initially know $|B(C)|$, then the termination detection of the algorithm is possible in a submersion-minimal communication structure: once an individual gets the number $|B(C)|$, it knows that each individual has a unique number that will not change any more.

4.2 Unlabelled Negotiations

We add time-stamps to local views in order to obtain a pseudo-covering with the final labelling. Consider a communication structure $\mathbf{C} = (C, \lambda)$ with an initial labelling $\lambda \colon B(C) \to L$. Here again, each individual $b \in B(C)$ will acquire new labels of the form $(\lambda(b), n(b), N(b), M(b))$ where:

- $n(b) \in \mathbb{N}$ is the identity number of b.
- $N(b) \in \mathcal{P}_{\text{fin}}(\mathcal{P}_{\text{fin}}(\mathbb{N}) \times \mathbb{N})$ is the local view of b. Intuitively, the algorithm will try to update the local view of b such that $N(b)$ contains a set $\{n(b') \mid b' \in a\}$ for each association a that contains b. Moreover, a time-stamp o will be associated to each of these sets in order to enable an individual, when it is possible, to detect if it belongs to different associations whose members have the same numbers.
- $M(b)$ is the mailbox of b and it contains the whole information the individual b has about the network.

Again, we need a total order on local views. Consider two sets N_1, N_2 whose elements are some pairs (S, o) where $o \in \mathbb{N}$ and $S \in \mathcal{P}_{\text{fin}}(\mathbb{N})$. Given two elements (S, o) and (S', o'), one will generalize $\prec_1$ to say that $(S, o) \prec_1 (S', o')$ if $S \prec_1 S'$ or if $S =_1 S'$ and $o < o'$. We now define a new order $\prec_3$ for elements of $\mathcal{P}_{\text{fin}}(\mathcal{P}_{\text{fin}}(\mathbb{N}) \times \mathbb{N})$. We say that $N_1 \prec_3 N_2$ if the maximal element for $\prec_1$ of the symmetric difference $N_1 \bigtriangleup N_2 = (N_1 \setminus N_2) \cup (N_2 \setminus N_1)$ belongs to N_2. Again, we extend $\prec_3$ to a total order on $L \times \mathcal{P}_{\text{fin}}(\mathcal{P}_{\text{fin}}(\mathbb{N}))$: $(\ell, N) \prec_3 (\ell', N')$ if either $\ell <_L \ell'$ or ($\ell = \ell'$ and $N \prec_3 N'$).

Unlabelled negotiations rules. The label of an individual b after the application of a relabelling rule to an association a that contains b is denoted by $(\lambda(b), n'(b), N'(b), M'(b))$. The three first rules have the same meaning as the three first rules of the algorithms described above.

```
$\mathcal{R}_1$ :
if $\exists b, b' \in a, M(b) \neq M(b')$ then
    $\forall b \in a, M'(b) := \bigcup_{b \in a} M(b);$
```

```
$\mathcal{R}_2$ :
if $n(b) = 0$ or $\exists (n(b), \ell, N) \in M(b)$ such that $(\lambda(b), N(b)) \prec_3 (\ell, N)$ then
    $n'(b) := 1 + \max\{n' \mid (n', \ell', N') \in M(b)\};$
    $M'(b) := M(b) \cup \{(n'(b), \lambda(b), N(b))\};$
```

```
$\mathcal{R}_3$ :
if $\forall b, b' \in a, M(b) = M(b')$
and $\forall b \in a, n(b) \neq 0$ and $\forall (n(b), \ell, N) \in M(b), (\ell, N) \preceq_3 (\lambda(b), N(b))$
and $\exists b_0, b_1 \in a$ such that $n(b_0) = n(b_1)$ then
    $n'(b_0) := 1 + \max\{n' \mid (n', \ell', N') \in M(b_0)\};$
    $M'(b_0) := M(b_0) \cup \{(n'(b_0), \lambda(b_0), N(b_0))\};$
```

The fourth rule enables to update the local views of all the members of an association in one computation step. This rule can be applied to some association

a only if the preceding ones cannot be applied to a. One can apply this rule to a if there does not exists any time-stamp o such that for each $b \in a$, $(o, \{n(b') \mid b' \in a\})$ belongs to $N(b)$. When the rule is applied, a new time-stamp o' is generated and $(o', \{n(b') \mid b' \in a\})$ is added to $N(b)$ for each $b \in a$. For the same reasons as in Section 4.1, each time we add an element (S', o') in $N(b)$, we remove all the elements of $N(b)$ that are smaller than (S', o') for $\prec_1$.

```
$\mathcal{R}_4$ :
if $\forall b, b' \in a, M(b) = M(b')$ and $n(b) \neq n(b')$
and $\forall b \in a, n(b) \neq 0$ and $\forall (n(b), \ell, N) \in M(b), (\ell, N) \preceq_3 (\lambda(b), N(b))$
and $\nexists o$ such that $\forall b \in a, (\{n(b') \mid b' \in a\}, o) \in N(b)$ then
    $o' := 1 + \max\{o \mid \exists b \in a, \exists (S, o) \in N(b)\}$;
    $S' := \{n(b) \mid b \in a\}$;
    $\forall b \in a, N'(b) := N(b) \setminus \{(S, o) \mid (S, o) \prec_1 (S', o')\} \cup \{(S', o')\}$;
    $\forall b \in a, M'(b) := M(b) \cup \{(n(b'), \lambda(b'), N'(b')) \mid b' \in a\}$;
```

Properties. The algorithm we described has the same interesting properties as the ones described in Sections 3 and 4.1. Finally, we have:

Theorem 3. *For every communication structure* $\mathbf{C}$*, there exists a naming algorithm for* $\mathbf{C}$ *using unlabelled negotiations if and only if* $\mathbf{C}$ *is pseudo-covering-minimal.*

Again, if the individuals initially know $|B(C)|$, then the termination detection of the algorithm is possible in a pseudo-covering-minimal communication structure: once an individual gets the number $|B(C)|$, it knows that each individual has a unique number that will not change any more.

5 Final Remarks

The homomorphisms we introduced generalize locally constrained graph homomorphisms. These graph homomorphisms have already been studied in the literature [BV02,Rei32] and one can wonder how the combinatorial properties satisfied by graph homomorphisms can be generalized to homomorphisms of communication structures. Locally constrained graph homomorphisms have also been studied from the complexity point of view [FP05,KPT98]. In particular, it has been shown in [CP06] that it is co-NP-complete to decide whether a graph admits a naming algorithm in the models studied in [Cha05,CM04,CMZ04,Maz04a]. An interesting corollary of this result is the following : The problems that ask whether a given communication structure $\mathbf{C}$ admits a naming algorithm using labelled negotiations, cellular labelled negotiations, unlabelled negotiations, cellular unlabelled negotiations respectively, are co-NP-complete.

References

[AAD$^+$04] D. Angluin, J. Aspnes, Z. Diamadi, M.J. Fischer, and R. Peralta. Computation in networks of passively mobile finite-state sensors. In *Proc. of the 23rd Annual ACM Symposium on Principles of Distributed Computing (PODC 2004)*, pages 290–299. ACM press, 2004.

[AAE06] D. Angluin, J. Aspnes, and D. Eisenstat. Stably computable predicates are semilinear. In *Proc. of the 25th annual ACM symposium on Principles of distributed computing (PODC 2006)*, pages 292–299. ACM Press, 2006.

[AAER05] D. Angluin, J. Aspnes, D. Eisenstat, and E. Ruppert. On the power of anonymous one-way communication. In *Proc. of the 9th International Conference on Principles of Distributed Systems (OPODIS 2005)*, pages 307–318, 2005.

[Ang80] D. Angluin. Local and global properties in networks of processors. In *Proc. of the 12th Symposium on Theory of Computing (STOC 1980)*, pages 82–93, 1980.

[BCG$^+$96] P. Boldi, B. Codenotti, P. Gemmell, S. Shammah, J. Simon, and S. Vigna. Symmetry breaking in anonymous networks: characterizations. In *Proc. of the 4th Israeli Symposium on Theory of Computing and Systems (ISTCS 1996)*, pages 16–26. IEEE Press, 1996.

[BV01] P. Boldi and S. Vigna. An effective characterization of computability in anonymous networks. In *Proc. of Distributed Computing, 15th International Conference (DISC 2001)*, volume 2180 of *Lecture Notes in Computer Science*, pages 33–47. Springer-Verlag, 2001.

[BV02] P. Boldi and S. Vigna. Fibrations of graphs. *Discrete Mathematics*, 243(1-3):21–66, 2002.

[Cha05] J. Chalopin. Local computations on closed unlabelled edges: the election problem and the naming problem. In *Proc. of the 31st Conference on Current Trends in Theory and Practice of Informatics (SOFSEM 2005)*, volume 3381 of *Lecture Notes in Computer Science*, pages 81–90. Springer-Verlag, 2005.

[CM04] J. Chalopin and Y. Métivier. Election and local computations on edges. In *Proc. of Foundations of Software Science and Computation Structures, 7th International Conference (FOSSACS 2004)*, volume 2987 of *Lecture Notes in Computer Science*, pages 90–104. Springer-Verlag, 2004.

[CMZ04] J. Chalopin, Y. Métivier, and W. Zielonka. Election, naming and cellular edge local computations. In *Proc. of the 2nd International Conference on Graph Transformations (ICGT 2004)*, volume 3256 of *Lecture Notes in Computer Science*, pages 242–256. Springer-Verlag, 2004.

[CMZ06] J. Chalopin, Y. Métivier, and W. Zielonka. Local computations in graphs: the case of cellular edge local computations. *Fundamenta Informaticae*, 74(1):85–114, 2006.

[CP06] J. Chalopin and D. Paulusma. Graphs labelings derived from models in distributed computing. In *Proc. of Graph-Theoretic Concepts in Computer Science, 32nd International Workshop (WG 2006)*, volume 4271 of *Lecture Notes in Computer Science*, pages 301–312. Springer-Verlag, 2006.

[FP05] J. Fiala and D. Paulusma. A complete complexity classification of the role assignment problem. *Theoretical Computer Science*, 349(1):67–81, 2005.

[GMM04] E. Godard, Y. Métivier, and A. Muscholl. Characterization of classes of graphs recognizable by local computations. *Theory of Computing Systems*, 37(2):249–293, 2004.

[KPT98] J. Kratochvíl, A. Proskurowski, and J.A. Telle. Complexity of graph covering problems. *Nordic Journal of Computing*, 5(3):173–195, 1998.

[Maz97] A. Mazurkiewicz. Distributed enumeration. *Information Processing Letters*, 61(5):233–239, 1997.

[Maz04a] A. Mazurkiewicz. Bilateral ranking negotiations. *Fundamenta Informaticae*, 60(1-4):1–16, 2004.

[Maz04b] A. Mazurkiewicz. Multilateral ranking negotiations. *Fundamenta Informaticae*, 63(2-3):241–258, 2004.

[Rei32] K. Reidemeister. *Einführung in die kombinatorische topologie*. Vieweg, Brunswick, 1932.

[YK96a] M. Yamashita and T. Kameda. Computing on anonymous networks: Part i - characterizing the solvable cases. *IEEE Transactions on parallel and distributed systems*, 7(1):69–89, 1996.

[YK96b] M. Yamashita and T. Kameda. Computing on anonymous networks: Part ii - decision and membership problems. *IEEE Transactions on parallel and distributed systems*, 7(1):90–96, 1996.