Abstract

This paper investigates the power of local computations on graphs considering the classical recognition problem. The recognition problem asks for computing some topology information on a network of anonymous processes, assuming some initial knowledge about the underlying graph.

We have introduced in [GMM00] a definition of recognition with initial knowledge by means of local relations and we have given a necessary condition for a class of graphs to be recognizable. In this paper we prove that this condition is sufficient. Some applications of this complete characterization are presented.

1 Introduction

Local computations on graphs which are described by graph rewriting [LMS99] are an adequate model for distributed computing. Rewriting provide a general tool for describing and analyzing distributed algorithms proving their correctness. In this paper we consider local computations with some initial knowledge of the network.

We focus on the classical recognition problem. We consider graphs are uniformly labelled by some initial label (which may encode some knowledge on the graph) and the presence or the absence of certain final labels determine whether the graph is accepted or not. The recognition problem asks for properties like acyclicity, planarity etc. Several basic properties like reachability, completeness, or acyclicity can be recognized by local computations with initial knowledge. On the other hand, we cannot determine whether a graph...
by local computations, provided that the given graph is labelled in an
distinctive way without initial knowledge [LMZ95, CM94]. However, the presence of a
distinguished vertex allows to gather information. In particular, it has been
demonstrated that it is possible to detect a given minor in a graph with a distinguished
vertex [BM98], hence also to determine whether the graph is planar. A natural question is whether some additional information encoded in the initial uniform labelling
of a graph can help for deciding for example planarity.

The classical proof techniques used for showing the non-existence of
solutions are based on coverings [Ang80, FLM86], which is a notion known from
algebraic topology. Coverings have been also used for simulation [BL86]. A graph
$G$ is a covering of a graph $H$ if there is a surjective morphism from $G$ to $H$ which
is locally bijective. The general idea is as follows. If $G$ and $H$ are two graphs
such that $G$ covers $H$, then every local computation on $H$ induces a local
computation on $G$. As a consequence, every class of graphs which is recognizable
by local computations (without any knowledge of the graph) must be closed
under coverings. Using this fact it has been proved that the class of planar graphs is recognizable by local computations [LMZ95]. More generally, it has been
shown that up to some few exceptions, every minor-closed class of graphs (the
class of graphs characterized by a finite set of forbidden minors by the
Kuratowski Theorem) is not closed under coverings, implying that it is not recognizable
by local computations [CM94]. In particular, one cannot decide by local
computations whether a given graph is included as a minor in an arbitrary
graph.

A graph $G$ is called covering-minimal if every covering from $G$ to some
other graph is a bijection. The class of covering-minimal graphs plays an important role
in the study of local computations: knowing the size, there exists an election algorithm
for the class of covering-minimal graphs [Maz97]. It is easy to verify using
some properties of rings, that the property of being covering-minimal is not recognizable
by local computations without any initial knowledge of the graph. Using [Maz97] we note that this property is recognizable if we have as initial knowledge the size of the graph. Having an odd number of vertices or having exactly one vertex with a certain label are examples of properties which are not recognizable without initial knowledge, but become recognizable if the graph size is known. Thus, recognizability under
the assumption that the size is known to the algorithm is significantly more
powerful than recognizability without initial knowledge.

To study recognizable class of graphs with some additional knowledge,
we cannot apply the covering argument directly. In [GMM00] we have introduced
a new construction. Let $\iota$ be a function which codes some knowledge on graphs.
Consider that two graphs $G$ and $G'$ are in relation if $\iota(G) = \iota(G')$ and there
exists a graph $H$ such that $G$ and $G'$ are coverings of $H$. Let $\sim$ denote the transitive
relation of this relation. We have proved in [GMM00] that a recognizable class of graphs
must be closed under this relation. In this paper we prove the converse and
we obtain a characterization (Theorem 2) of classes of graphs recognizable
under initial knowledge by local computations: a class of graphs $\mathcal{F}$ is locally recognized
able with some additional knowledge if and only if it is recursive and stable with respect to the relation \( \sim \). We give various applications of this theorem, in particular, we deduce immediately that classes of graphs that are recognizable stable by union, intersection and complement.

Related work. Among models related to our model there are local computation systems as defined by Rosenstiehl et al. [RFH72], Angluin [Ang80], Yama- mashita and Kameda [YK96]. In [RFH72] a synchronous model is considered, where vertices represent (identical) deterministic finite automata. The basic computation step is to compute the next state of each processor according to the states of its neighbours. In [Ang80] an asynchronous model is considered. A basic computation step means that two adjacent vertices exchange their states and then compute new ones. In [YK96] an asynchronous model is studied, a basic computation step means that a processor either changes its state and sends a message or it receives a message.

Please, note that this paper is an extended abstract of [GM01].

2 Basic Notions and Notation

2.1 Labelled Graphs

The notation used here is essentially standard [Ros00]. We only consider undirected and connected graphs without multiple edges and self-loops.

Let \( k \) be an integer. We say that a graph \( G \) is a \( k \)-covering of a graph \( H \) if there exists a surjective homomorphism \( \gamma \) from \( G \) onto \( H \) such that for every vertex \( v \) in \( G \), the restriction of \( \gamma \) to \( B_G(v,k) \) is a bijection onto \( B_H(\gamma(v),k) \). The covering is proper if \( G \) and \( H \) are not isomorphic. It is called connected if \( G \) (and thus \( H \)) is connected. A graph \( G \) is called \( k \)-covering-minimal if every \( k \)-covering \( \tilde{G} \) from \( G \) to some \( H \) is a bijection. Moreover, for any \( k < k' \), a \( k' \)-covering is also a \( k \)-covering. Note that a 1-covering is exactly a covering in the classical sense.

Throughout the paper we will consider only connected graphs where vertices and edges are labelled with labels from a possibly infinite alphabet \( L \). A labelled graph \( (G,\lambda) \) is denoted by \( G \), where \( G \) is a graph and \( \lambda : E(G) \rightarrow L \) is the labelling function (in most cases we use only a vertex-function). The graph \( G \) is called the underlying graph and the mapping \( \lambda \) is the labelling of \( G \). The class of labelled graphs over some fixed alphabet \( L \) denoted by \( \mathcal{G} \).

Let \( (G,\lambda) \) and \( (G',\lambda') \) be two labelled graphs. Then \( (G,\lambda) \) is a subgraph \( (G',\lambda') \), denoted by \( (G,\lambda) \subseteq (G',\lambda') \), if \( G \) is a subgraph of \( G' \) and \( \lambda \) is the restriction of the labelling \( \lambda' \) to \( V(G) \cup E(G) \).

A mapping \( \phi : V(G) \cup E(G) \rightarrow V(G') \cup E(G') \) is a homomorphism from \( G \) to \( G' \) if \( \phi \) is a graph homomorphism from \( G \) to \( G' \) which preserves the labelling, i.e. such that \( \lambda'(\phi(x)) = \lambda(x) \) holds for every \( x \in V(G) \cup E(G) \).
Definition 2 Let $\varphi$ be a relabelling relation and $k > 0$ be an integer. The next definition states that a local relabelling relation $\mathcal{R}$ is $k$-locally generated if the following is satisfied: For every labelled graph $(G, \lambda')$, $(H, \eta')$, $(H, \eta')$ and every vertices $v \in V(G)$, $w \in V(H)$ such that $B_G(v, k)$ and $B_H(v, k)$ are isomorphic via $\varphi$: $V(B_G(v, k)) \rightarrow V(B_H(v, k))$, $\varphi(v) = w$, the following three conditions

1. If $\mathcal{R}$ is a relabelling relation if whenever two labelled graphs are in $\mathcal{R}$, then the underlying graphs are equal (we say equal, not only isomorphic).

2. $\mathcal{R}$ is $k$-local if only labels of a ball of radius $k$ may be changed by $\mathcal{R}$.

3. The relation $\mathcal{R}$ is $k$-irreducible if $\mathcal{R}$ is $k$-local for some $k > 0$.
1. \( \lambda(x) = \eta(\phi(x)) \) and \( \lambda'(x) = \eta'((\phi(x)) \) for all \( x \in V(B_G(v,k)) \cup E(B_G(v,k)) \)

2. \( \lambda(x) = \lambda'(x), \) for all \( x \notin V(B_G(v,k)) \cup E(B_G(v,k)) \)

3. \( \eta(x) = \eta'(x), \) for all \( x \notin V(B_H(w,k)) \cup E(B_H(w,k)) \)

imply that \( (G,\lambda)^R (G,\lambda') \) if and only if \( (H,\eta)^R (H,\eta') \).

\( R \) is locally generated if it is \( k \)-locally generated for some \( k > 0 \).

We give now a fundamental lemma which connects \( k \)-coverings and \( k \)-generated relabelling relations. It states that whenever \( G \) is a \( k \)-covering of \( G' \), every \( k \)-local computation in \( G' \) can be lifted to a \( k \)-local computation in \( G \) which is compatible with the \( k \)-covering relation. This is expressed in the following lemma:

**Lemma 1** Let \( R \) be a \( k \)-locally generated relabelling relation and let \( (G',\lambda'_1)^R (G',\lambda'_2) \) be a \( k \)-covering of \( (G',\lambda'_1) \) via \( \gamma \). Moreover, let \( (G',\lambda'_1)^R (G',\lambda'_2) \). Then a relabelling \( \lambda_2 \) of \( G \) exists such that \( (G,\lambda_1)^R (G,\lambda_2) \) and \( (G,\lambda_2) \) is a \( k \)-covering of \( G' \).

### 2.3 Distributed Computations of Local Computations

The notion of relabelling sequence defined above, obviously corresponds to the notion of *sequential* computation. Let us also note that a \( k \)-locally generated relabelling relation also allows parallel rewritings, since non-overlapping \( k \)-balls may be relabelled independently. Thus we can define a distributed way of computing by saying that two consecutive relabelling steps concerning non-overlapping \( k \)-balls may be applied in any order. We say that such relabelling steps commute and they may be applied concurrently. More generally, any two relabelling sequences such that the latter one may be obtained from the former one by successions of such commutations lead to the same resulting labelled graph. One of our notion of relabelling sequence may be regarded as a *serialization* [Maz87].

Our model is clearly asynchronous: several relabelling steps may be done at the same time but we do not require that all steps have to be performed. In the sequel we will essentially deal with sequential relabelling sequences but the reader should keep in mind that such sequences may be done in a distributed way.

### 3 Graph Recognizers and Initial Knowledge

Let \( L \) be a set of labels. The problem addressed in this section can be informally described as follows. Let \( \mathcal{F} \) be some class of (labelled) graphs. We will say that \( \mathcal{F} \) can be *locally recognized* if there exists some graph relabelling system (or, generally, some locally generated graph relabelling relation) such that from any uniformly labelled graph \( (G,\lambda_0) \) some final labelling can be computed that allows to decide whether \( G \) belongs to the class \( \mathcal{F} \) or not.
Definition 3 A labelled graph recognizer is a pair \((R, K)\) where \(R\) is a relabelling relation and \(K\) is a class of labelled graphs.
A graph \((G, \lambda)\) is recognized if \(\text{Irred}_R(G, \lambda) \cap K \neq \emptyset\).

We are interested in recognizing graphs which have a certain initial knowledge encoded in the initial labelling. Let \(G\) be a graph and \(\alpha\) a label in \(L\). Then the uniform labelling on \(G\) with \(\alpha\), that is every vertex is labelled by \(\alpha\).

Definition 4 A graph recognizer with initial knowledge is a triple \((R, K, \iota)\) where \(R\) is a labelled graph recognizer, and \(\iota\) is a function which associates each graph \(G\) a label \(\iota(G) \in L\). The set of graphs recognized by \((R, K, \iota)\) as \(\{G \mid (G, \Lambda_{\iota(G)})\) is recognized by \((R, K)\}\).

A recognizer \((R, K, \iota)\) is said to be deterministic if, restricted to inputs \((G, \Lambda_{\iota(G)})\), we have the following two properties:
- \(R\) is noetherian.
- Either \(\text{Irred}(G, \Lambda_{\iota(G)}) \cap K = \emptyset\) or \(\text{Irred}(G, \Lambda_{\iota(G)}) \subseteq K\).

We can now define recognizable classes of graphs.

Definition 5 A class \(F\) of graphs is said to be (deterministically) recognizable with initial knowledge \(\iota\) if there exists a locally generated (deterministic) recognizer \((R, K, \iota)\) recognizing exactly \(F\).

In this paper, we deal with graph recognizers where the relation \(R\) is generated. Moreover, the set \(K\) is defined by means of a so-called final condition which represents a logical formula inductively defined as follows: (i) for label \(\ell \in L\), \(\ell\) is a formula and (ii) if \(\varphi\) and \(\psi\) are formulas then so are \(\neg\varphi\) and \(\varphi \land \psi\). Now, for \(\ell \in L\), a labelled graph satisfies the formula \(\ell\) if it contains at least one \(\ell\)-labelled vertex or edge, and by induction, it satisfies \(\varphi \lor \psi\) if it satisfies \(\varphi\) or \(\psi\) and so on in the usual way. Thus, such final conditions work only on the set of labels appearing in a given final labelling, or in other words, they only verify the presence or the absence of some specific labels but not to count the number of labels. We will denote by \(K(\varphi)\) the set of labelled graphs which satisfy the formula \(\varphi\) (some examples are given in [LMZ95]).

A simple example of recognition system is given below.

Example 1 Tree recognition
The following relabelling system recognizes trees.
The set of labels is \(\{\varepsilon, F, T\}\). As there is no initial information, \(\varepsilon\) is the label on all vertices.

- First rule: If a node has exactly one \(\varepsilon\)-labelled neighbour then it changes its label to \(F\).
Second Rule: If a node has no $\varepsilon$-labelled neighbour then it changes to $T$.

The final condition is the presence of the label $T$.

Remark: In $\iota$, we can encode some information about the underlying graph as an upper bound on the number of vertices, a tight bound (that is a bound that for all graphs $G$, $|V(G)| \leq b(G) < 2|V(G)|$), the exact number of vertices, even the topology.

Remark: The knowledge of the topology does not solve all problems; in particular it does not enable to solve the election problem [GMM00]. For the recognition problem it remains to compute the correspondence between vertices of the graph and vertices of the abstract graph given by the initial knowledge.

We give now a simple relation between graph recognizers and $k$-coverings from which we obtain new results on graph recognizers working on graphs with some initial knowledge.

For $k > 0$, we define a relation $\sigma^k_1$ by letting $G \sigma^k_1 G'$ if:

- $\iota(G) = \iota(G')$
- There exists a graph $H$ such that $G$ and $G'$ are $k$-coverings of $H$.

Let $\sim^1_k$ denote the reflexive, transitive closure of $\sigma^1_k$. A class of graphs be said to be closed under $\sim^1_k$ if for any graphs $G$ and $G'$ such that $G \sigma^1_k G'$ in $\mathcal{F}$ if and only if $G'$ is in $\mathcal{F}$. We obtain the following necessary condition for recognizability:

**Proposition 1** [GMM00] Let $\mathcal{F}$ be a class of graphs which is deterministically recognizable using the initial knowledge $\iota$. Then $\mathcal{F}$ is closed under $\sim^1_k$.

### 4 Enumeration and Recognition

To prove the converse of Proposition 1, we describe an algorithm that recognizes a given $\sim^1_k$-closed, recursive class of graphs, using the initial knowledge. The results will be shown using the algorithm given by Mazurkiewicz in [Maz97]. It is worth noting that despite its simplicity, this algorithm provides all the underlying graph information that a distributed algorithm can be required to compute.

#### 4.1 Mazurkiewicz Algorithm: $M_k$

We give here a general description of the algorithm $M_k$. Initially all vertices have the same label. Every vertex attempts to get its own name, which shall be a number between 1 and $|V|$. A vertex chooses a name and broadcasts it together with its labelled neighbourhood all over the network. If a vertex $u$ discovers the code...
of another vertex \( v \) with the same name, then it compares its *neighbourhood view*, i.e., its labelled \( k \)-ball, with the neighbourhood view of its rival \( v \). If the neighbourhood view of \( v \) is “stronger”, then \( u \) chooses another name. Each new name broadcasted again over the network. At the end of the computation it is guaranteed that every node has a unique name, unless the graph is covering-

However, all nodes with the same name will have the same *neighbourhood view*, i.e., isomorphic labelled \( k \)-balls.

A total order \( \prec \) is defined on the set of neighbourhood view \( \mathcal{B} \). A vertex has labels that are left unchanged. The rules are described below for a given centered \( k \)-ball \( B = B(v_0, k) \) with center \( v_0 \). The vertices \( v \) of \( B \) have labels \((n(v), N(v), M(v), d(v))\) representing the following information during the computation:

- \( n(v) \in \mathbb{N} \) is the name of the vertex \( v \),
- \( N(v) = (B, \lambda) \in \mathcal{L} \) is the neighbourhood view, i.e., \( B \) is a copy of \( B(v_0, k) \) for all \( u \in \mathcal{V}(B) \), \( \lambda(u) = n(u) \).
- \( M(v) \subset \mathbb{N} \times \mathcal{B} \) is the mailbox of \( v \) and contains the information read this step of the computation,
- \( f(v) \in \{ \text{UpToDate}, \text{Dated} \} \) is a flag that will be set to \text{Dated} when \( v \) vertex has to update its neighbourhood information.

The initial labelling of all vertices is \((0, \emptyset, \emptyset, \text{UpToDate})\).

The rules are described below for a given centered \( k \)-ball \( B = B(v_0, k) \) with center \( v_0 \). The vertices \( v \) of \( B \) have labels \((n(v), N(v), M(v), f(v))\). The labels obtained after applying a rule are \((n'(v), N'(v), M'(v), f'(v))\). To make the rules easier to read, we omit labels that are left unchanged.

1. **Diffusion rule:**
   Suppose that \( f(v) = \text{UpToDate} \) for every \( v \in B \). Moreover, the mailbox contents of \( B \) are inconsistent, i.e., \( M(v_0) \neq M(v) \), for some \( v \in B \).
   The diffusion rule yields: \( M'(v) = \bigcup_{w \in B} M(w) \), for all \( v \in B \).

2. **Renaming rule:**
   Suppose that \( f(v) = \text{UpToDate} \) for every \( v \in B \) and \( M(v) = M(v_0) \) for some \( v \in B \). Moreover, either \( n(v_0) = 0 \) or there is some \((n(v_0), N_1) \in M(v_0) \) such that \( N(v_0) \prec N_1 \), (i.e., there is a node with the same name and a “stronger” neighbourhood view). The renaming rule yields:
   \( n'(v_0) = 1 + \max \{ n \in \mathbb{N} \mid (n, N) \in M(v_0) \text{ for some } N \in \mathcal{L} \} \)
   (a) \( n'(v_0) = 1 + \max \{ n \in \mathbb{N} \mid (n, N) \in M(v_0) \text{ for some } N \in \mathcal{L} \} \)
   (b) The neighbourhood view \( N'(v_0) \) is obtained from \( N(v_0) \) by updating the name of \( v_0 \).
   (c) The mailbox contents changes to \( M'(v_0) = M(v_0) \cup \{ (n'(v_0), N'(v_0)) \} \)
   (d) For all \( v \in B(v_0, k) \setminus \{v_0\} \), let \( f'(v) = \text{Dated} \).
3. **Updating rule:**

Suppose that \( f(v_0) = \text{DATED} \). Then \( v_0 \) updates its neighbourhood \( N'(v_0) = (B(v_0, k), \lambda) \), where \( \lambda \) is given by \( \lambda(v) = n(v) \). The updating rule \( f' \) is given by setting \( f'(v_0) = \text{UP\textsc{To\textsc{Date}}} \).

Note that for \( k = 1 \) the updating rule is not needed, [Maz97]. The reason is that a neighbourhood of radius 1 is just a set, hence the neighbourhood of \( v_0 \) is the set itself, and neighbours can be updated within the renaming rule.

### 4.2 Properties of Mazurkiewicz Algorithm

Let \( G \) be a graph, then the labelling function obtained after a run \( \rho \) of Mazurkiewicz algorithm is noted \( \Pi_{\rho} \). If \( v \) is a vertex of \( G \), the 4-tuple \( \Pi_{\rho}(v) \) associated with \( v \) is denoted \((n_{\rho}(v), N_{\rho}(v), M_{\rho}(v), f_{\rho}(v))\). We obtain the following extension of Mazurkiewicz algorithm:

**Theorem 1** A run of Mazurkiewicz Enumeration Algorithm on a connected \( G \) terminates and yields a final labelling \( \Pi_{\rho} \) verifying the following conditions:

1. Let \( m \) be the maximal name in \( G \), \( m = \max_{v \in V(G)} n_{\rho}(v) \). Then for all \( 1 \leq l \leq m \) there is some \( v \in V(G) \) with \( l = n_{\rho}(v) \).
2. \( M_{\rho}(v) = M_{\rho}(v') \).
3. \((n_{\rho}(v), N_{\rho}(v)) \in M_{\rho}(v') \).
4. \( f_{\rho}(v) = \text{UP\textsc{To\textsc{Date}}} \).
5. Let \((n,N) \in M_{\rho}(v')\). Then \( n_{\rho}(v) = n \) and \( N_{\rho}(v) = N \) for some vertex \( v' \) only if there is no pair \((n,N') \in M_{\rho}(v')\) with \( N < N' \).
6. \( n_{\rho} \) induces a \( k \)-locally bijective labelling of \( G \).

We interpret the final labelling \( \Pi_{\rho} \) as a graph that each vertex could compute. For a mailbox \( M \), we define

\[
F(M) = \{(n,N) \in M \mid N' < N \text{ for all } (n,N') \in M\}.
\]

For a given \( M \) we define the graph \( G_M \) as the following graph:

\[
V(G_M) = \{n \mid \exists N, (n,N) \in F(M)\}
\]

\[
E(G_M) = \{\{n,n'\} \mid \exists N = (B,\lambda) \text{ such that } (n,N) \in F(M), \exists u,u' \in N, \lambda(u) = n, \lambda(u') = n' \text{ and } \{u,u'\} \in \lambda\}
\]

Let \( \rho \) be a run of \( M_0 \), then \( G_{\rho} = G_{M_{\rho}(u)} \) does not depend on \( u \), as stated in Theorem 1, we define \( G_{\rho} = G_{M_{\rho}(u)} \).
Proposition 2 Let $G$ be a graph.

1. For all runs $\rho$ of $M_k$, $G$ is a $k$–covering of $G_\rho$.

2. For all $H$ such that $G$ is a $k$–covering of $H$, there exists a run $\rho$ such that $\rho \simeq H \simeq G_\rho$.

4.3 Recognizing $\sim_k^1$–closed Graph Classes

In this section, we sketch a distributed algorithm for recognizing a given recursive and closed under $\sim_k^1$ closed class of graphs $\mathcal{F}$. Throughout the section we suppose that $\mathcal{F}$ is recursive and closed under $\sim_k^1$ without further mentioning it. We first need some notation. Let

$\mathcal{F}_k^1 = \{(H, \iota(G)) \mid G \in \mathcal{F} \text{ and } G \text{ is a } k\text{–covering of } H\}.$

We have this simple and fundamental lemma:

Lemma 2 Let $\mathcal{F}$ be a $\sim_k^1$–closed class of graphs. Let $G$ be a graph. Then any run $\rho$ of $M_k$:

$G \in \mathcal{F}$ if and only if $(G_\rho, \iota(G)) \in \mathcal{F}_k^1$.

We also remark that:

Lemma 3 For any $(H, \alpha)$, we have

1. $(H, \alpha) \in \mathcal{F}_k^1 \iff \exists K \in \mathcal{F} \text{ satisfying } K \text{ is a } k\text{–covering of } H \text{ and } \iota(K) = \iota(G)$.

2. $(H, \alpha) \notin \mathcal{F}_k^1 \iff \exists K \notin \mathcal{F} \text{ satisfying } K \text{ is a } k\text{–covering of } H \text{ and } \iota(K) = \iota(G)$.

We add some information to the 4–tuple computed by Mazurkiewicz algorithm. More precisely, we consider on each node $v$ two boolean variables, noted $b(v)$ and $r(v)$ whose initial values are false. When a node $v_0$ is relabeled by the Mazurkiewicz algorithm, $b(v)$ and $r(v)$ become false. If no rule of Mazurkiewicz algorithm may be applied on the node $v_0$ then $b(v_0)$ becomes true if all the nodes in the ball centered on $v_0$ can reconstruct the same graph. If $b(v)$ is true the node run the algorithm of test described below on $(G_\rho, \iota(G))$ setting the variable $r(v)$.

We enumerate all the graph $K$. For each one we test if $K$ is a $k$–covering of $G_\rho$, and $\iota(K) = \iota(G)$. When we obtain a graph $K$ verifying these two conditions, we test if it belongs to $\mathcal{F}$ ($\mathcal{F}$ is recursive by hypothesis).

We also make some fairness assumptions on the execution on this algorithm: the execution is resetted whenever a $M_k$ rule is applied, such that there is never an infinite execution. Those assumptions can be omitted provided some technical improvements not detailed here. Finally we have:
Proposition 3 If for each node \( v \) of \( G \) \( b(v) = true \) then Mazurkiewicz algorithm has reached a terminal configuration. Conversely, when Mazurkiewicz algorithm has reached a terminal configuration then after a number of steps bounded by the size of \( G \) \( b(v) \) is true and remains true for all nodes of \( G \).

From Lemma 2 and Lemma 3, as \( \mathcal{F} \) is closed, we deduce whether \( G \) belongs to \( \mathcal{F} \) looking at the \( r(v) \) labels in the terminal configuration, i.e.:

Proposition 4 When Mazurkiewicz algorithm has reached a terminal configuration, we have:

\[
G \in \mathcal{F} \iff \forall v \quad r(v) = true.
\]

As a synthesis of the previous results we obtain:

Theorem 2 Let \( \mathcal{F} \) be a class of graphs and \( \iota \) an initial knowledge. The following statements are equivalent.

1. \( \mathcal{F} \) is \( k \)-locally recognizable with initial knowledge \( \iota \).
2. \( \mathcal{F} \) is closed under \( \sim_k \) and \( \mathcal{F} \) is a recursive set.

An interesting corollary is the stability by boolean operations.

Corollary 1 Classes of graphs that are \( k \)-locally recognizable with initial knowledge \( \iota \) are stable for union, intersection, and complement.

5 Applications: Particular Cases of Initial Knowledge

The results of the previous section state that the \( \sim_k \)–equivalence classes are a refinement of the atoms of recognizable classes of graphs. In this section we are investigating the structural properties of equivalence classes in some particular interesting cases.

5.1 No a priori Knowledge

Leighton [Lei82] gives a decidable criterium for two graphs admitting a common covering. He uses the so-called degree partition of a graph \( G \), i.e., the partition of the vertices of \( G \) into the minimal number of blocks \( B_0, B_1, \ldots, B_{t-1} \) for which there are constants \( r_{i,j} \), \( 0 \leq i, j < t \), such that every vertex \( v \) in \( B_i \) is incident to \( r_{i,j} \) edges linking \( v \) to vertices in \( B_j \). The degree refinement of \( G \) is then defined by the matrix \( R = (r_{i,j}) \). Two degree refinements \( R_1 \) and \( R_2 \) are considered to be the same if the two matrices are conjugated.

Theorem 3 (Leighton, 1982) Given any two finite connected graphs \( G \) and \( H \), \( G \) and \( H \) share a common finite covering if and only if they have the same degree refinement.
Concerning our relation \( \sim_k^\varepsilon \), we obtain for \( k = 1 \):

**Proposition 5** Let \( G \) and \( G' \) be two connected graphs. Then \( G \sim_1^\varepsilon G' \) if and only if \( G \) and \( G' \) have the same degree refinement.

**Remark:** We do not have such a characterization for \( k > 1 \).

As the degree refinement is computable on a graph we also obtain:

**Corollary 2** Let \( G \) be a graph. The equivalence class of \( G \) is \( 1 \)-recognizable with no initial knowledge.

In particular, \( 1 \)-recognizable graph classes with no initial knowledge can be seen as recursive sets of degree refinement.

### 5.2 Size Upbounds

We defined an upbound recognizable class to be a class that is recognizable by a relabelling system whatever the chosen upbounding function. It is then equivalent to say that an upbound recognizable class is a class that is closed for the \( \sim_k \) relation. So, if we define \( \sim_k^{\text{bound}} = \bigcup_b \sim_k^b \), where the union is taken over all upbounds \( b \) of the size of \( G \), we have

**Proposition 6** Let \( \mathcal{F} \) be a class of graphs. \( \mathcal{F} \) is upbound recognizable if and only if \( \mathcal{F} \) is closed for \( \sim_k^{\text{bound}} \).

We have

**Lemma 4** For all \( k \in \mathbb{N} \), \( \sim_k^\varepsilon = \sim_k^{\text{bound}} \).

**Proof.** For every \( k \in \mathbb{N} \), \( \sim_k^\varepsilon \supseteq \sim_k^{\text{bound}} \) is obvious. For the other inclusion, if we have a graph \( H \) that is a \( k \)-covering of \( G \) and \( G' \), then for an upper function such that \( b(G) = b(G') = \max(|G|,|G'|) \), we have \( G \sim_k^b G' \).

As an immediate corollary of the previous section, we obtain:

**Corollary 3** Let \( \mathcal{F} \) be a class of graphs that is \( k \)-recognizable having a bound on the size. Then \( \mathcal{F} \) is also \( k \)-recognizable with no initial information.

**Remark:** Having a bound on the number of vertices is the same as having no initial information in the recognition context. This, quite surprising, result is similar to one obtained in [YK96] (see Prop. 18). This is quite different in the termination detection context (see [MT00]).
5.3 Tight Upbound

We recall that a tight upbound is a bound $b$ such that for all graphs $G$, $b(G) < 2|V(G)|$.

**Proposition 7** The class of minimal graphs is recognizable knowing a tight bound.

**Proof.** Since a strict covering has at least two sheets, then the equivalence class $[G]_{TightBound}$ are singletons.

5.4 Knowing the Size

**Remark:** We can note that if $G$ and $H$ are both coverings of a graph $K$ and have the same number of vertices then they have the same number of edges and their cycle spaces have the same dimension.

That is to say that knowing the number of vertices is the same as knowing the number of edges.

5.5 Table

Let $k$ be any integer. We will summarize results in the following table where YES and NO answer the question whether the listed family is recognizable by a deterministic $k$-generated graphs recognizer with the corresponding knowledge. The number within parenthesis refers to the proof below.

<table>
<thead>
<tr>
<th>No Info Bound</th>
<th>Tight Bound</th>
<th>Number of vertices</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recursive union of trees</td>
<td>YES (1)</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>$k$-Minimal graphs</td>
<td>NO (2)</td>
<td>YES (3)</td>
<td>–</td>
</tr>
<tr>
<td>Family of graphs such that $V(G)$ is even</td>
<td>–</td>
<td>NO (4)</td>
<td>YES(5)</td>
</tr>
<tr>
<td>Planar graphs</td>
<td>–</td>
<td>–</td>
<td>NO (6)</td>
</tr>
<tr>
<td>Bipartite graphs</td>
<td>–</td>
<td>–</td>
<td>NO (7)</td>
</tr>
</tbody>
</table>

**Proof.**

1. By Theorem 2 and remarking that the $\sim_k^c$-class of a tree is a singleton.

2. The family of $k$-minimal graphs is not closed for $k$-covering. Let us consider $k$-rings. The minimal rings are the ones that have a prime number of rings.

3. See Prop. 7
4. if $p$ is an odd integer then consider the equivalence on the ring $R_{2p} \sigma_{\text{TightBound}} R_{3p}$ via the ring $R_p$ and corollary 1.

5. Trivial.

6. [GMM00].

7. [GMM00].

6 Conclusion

We present in this paper a characterization of classes of graphs that are recognizable by local computations with a given initial knowledge. Using a simple equivalence relation, we have established properties like the closure under operations, and we have investigated the equivalence classes in some particular cases. Thus, we have partially answered the question of determining what knowledge is necessary in order to recognize some properties of the underlying graph by local computations. Further work will consist in a full description of equivalence classes, especially with respect to quasi-coverings, which are used for termination detection [MT00].

References


Emmanuel Godard is with the LaBRI, Université Bordeaux I, 351, cours de la libération, 33405 Talence France. E-mail: godard@labri.u-bordeaux.fr

Yves Métivier is with the LaBRI, Université Bordeaux I - ENSEIRB, 351, cours de la libération, 33405 Talence France. E-mail: metivier@labri.u-bordeaux.fr