THE GENUS OF REGULAR LANGUAGES AND DIRECTED GRAPH EMULATORS

Guillaume Bonfante, Deloup Florian

THE GENUS OF REGULAR LANGUAGES AND DIRECTED GRAPH EMULATORS

GUILLAUME BONFANTE[1], FLORIAN DELOUP[2]

1- Université de Lorraine - LORIA - Nancy
2- Université Paul Sabatier - IMT - Toulouse

Abstract. The article continues our study of the genus of a regular language L, defined as the minimal genus among all genera of all finite deterministic automata recognizing L. Here we define and study two closely related tools on a directed graph: directed emulators and automatic relations. A directed emulator morphism essentially encapsulates at the graph-theoretic level an epimorphism onto the minimal deterministic automaton. An automatic relation is the graph-theoretic version of the Myhill-Nerode relation. We show that an automatic relation determines a directed emulator morphism and respectively, a directed emulator morphism determines an automatic relation up to isomorphism. Consider the set S of all directed emulators of the underlying directed graph of the minimal deterministic automaton for L. We prove that the genus of L is $\min_{G \in S} g(G)$. We also consider the more restrictive notion of directed cover and prove that the genus of L is reached in the class of directed covers of the underlying directed graph of the minimal deterministic automaton for L. This stands in sharp contrast to undirected emulators and undirected covers which we also consider. Finally we prove that if the problem of determining the minimal genus of a directed emulator of a directed graph has a solution then the problem of determining the minimal genus of an undirected emulator of an undirected graph has a solution.

Contents

1. Introduction

The genus of a regular language L is the minimal genus of a finite deterministic automaton recognizing the language L. In particular, a planar language is a language with genus 0. We defined the genus in [2] and showed that this notion defines a proper hierarchy of (regular) languages. The decidability of the planarity of a regular language is a question asked by R. V. Book and A. K. Chandra [5] in 1976.

It should be noted that the genus of a regular language behaves quite differently from the its set-theoretical size. First, as we showed in [2], a minimal genus deterministic automaton recognizing a given language has not minimal size in general, thus differs from the minimal automaton in the sense of Myhill-Nerode. Secondly, in a sequel [3] to the first article, we proved that the size of the minimal automaton recognizing a language L with minimal genus may be exponentially larger than the minimal automaton itself. The existence of an upper bound remains open. This suggests that the computation of the genus of a regular language should have a high complexity.

We also proved that under a fairly generic hypothesis[1], the genus of a regular language is computable. In particular, under this hypothesis, the planarity of a regular language is decidable. One should point out that the proof is entirely constructive and yields an implementable algorithm. We conjecture that the genus of a regular language is computable in general.

In [2], we showed that a minimal genus deterministic automaton recognizing a given language has not minimal size in general, thus differs from the minimal automaton in the sense of Myhill-Nerode.

[1]The minimal automaton must not contain small cycles, the size of which depends on the size of the alphabet. Moreover, automata are supposed to be complete.

In a sequel to the first article ([3]), we showed that the size of the minimal automaton recognizing a language L with minimal genus may be exponentially larger than the minimal automaton itself. The existence of an upper bound remains open. Anyway, this shows that the witness automaton of minimal genus can be very large with respect to the size of the minimal automaton that serves as reference. That is an insight that computing the genus of a language should have a high complexity. In this third opus of the series, we look at the problem with a graph-theoretical approach. The graph-theoretical substance of the relation of the minimal automaton to the genus minimal automaton is the notion of *minimal directed emulator* (§4). A particular case is the notion of *directed cover*. Most of this paper is devoted to the systematic study of directed emulators.

The notion of directed emulator is the natural refinement of the notion of (undirected) graph emulator, introduced by R. M. Fellows in 1985. In order to study the properties of emulators, he also used the notion of graph cover and conjectured that a connected finite graph has a finite planar emulator if and only if it has a finite planar cover. It took more than twenty years before Y. Rieck and Y. Yamashita found a counterexample [17] to the conjecture. However, P. Hliněnỳ had found ten years earlier [12] an example of a graph having an emulator of genus strictly less than the minimal genus of any of its covers. On the one hand, we show that the situation for directed graphs is much simpler: a directed graph has a directed emulator of genus g if and only if it has a directed cover of genus g (Proposition 2). On the other hand, it follows from our constructions (Corollary 5.10) that a solution to the directed emulation genus problem implies a solution to the undirected emulation genus problem, a well-known difficult problem even in genus 0.

A major result of this paper is that a regular language has genus less than or equal to g if and only if a directed cover of the underlying graph of its minimal deterministic automaton has genus less than or equal to g (Theorem 5). As a consequence, we show that the original problem of determining the genus of a regular language is equivalent to the problem of determining the minimal genus of a directed cover of the underlying directed graph (as opposed to the undirected version). In particular, the problem of determining the planarity of a regular language is equivalent to the problem of determining the planarity of a directed cover of the underlying directed graph. The latter equivalence was originally proved by D. Kuperberg in [15] via a different approach.

Another important result of the paper is the correspondence between directed emulators and automatic relations on directed graphs as stated in Theorem 1. Roughly speaking, an automatic relation on a directed graph is a device that mimicks the Myhill-Nerode equivalence relation at the level of automata. There is no privileged subset of edges (sets of initial and final states respectively) on a directed graph G but only a class of complete final systems. It is proved here (Theorem 2) that a relation in a directed graph is automatic if and only if it stems from a MN-recursive relation, reminiscent

of the recursive description of the original Myhill-Nerode relation. In the particular (important) case when the directed graph has a reachable vertex, we show that an automatic relation is induced by a Myhill-Nerode relation with respect to a single distinguished subset of vertices (Corollary 2.1).

Four main mathematical structures arise. First, (finite state) automata are used to describe regular languages. Forgetting input states and final states (albeit not completely, cf. the rôle of complete final systems in Theorem 2 and Corollary 2.1) yields the notion of semi-automata. Forgetting letters on edges leads to the notion of directed graphs. Finally, forgetting the direction on edges leads to undirected graphs. These three forgetful operations are performed when the genus is computed. Our main contribution is to delineate the relationships between these four structures under the lens of the genus problem.

As a whole, our approach aims at extracting the minimal structure required to compute the genus of a regular language. We use categories which we think render correctly the structural framework of the problem. Many authors have developped a categorical approach to closely related structures. For instance, J. Adámek and V. Trnková [1] present a categorical view of automata. T. Colcombet and D. Petrişan in [8] develop a nice theory in which automata are seen as functors. They provide sufficient conditions to ensure the existence of minimal automata. We should also mention the work by A. Joyal, M. Nielsen and G. Winskell about concurrency modelling in [13], where their transition systems suppose input states (actually a unique one). This is an example of structure one naturally meets in our setting. Several obstacles lie in our way to build a full fledged categorical theory. First, there are two sorts of morphisms, emulators and covers. Secondly, to each directed emulator, one may associate a directed cover with the same genus (one should keep in mind that this is not true for the undirected graphs) but this construction is not natural. As another example, removing parallel edges can be done with emulators, not with covers. On the other hand, a directed cover lends itself to a description in terms of a universal property, see for instance Proposition 2. The subtle interplay between the two lies at the heart of the complexity analysis of the genus of a regular language.

Acknowlegments. F.D. wishes to thank Philippe Balbiani for an illuminating discussion about bisimulation at the Institut de recherche en informatique de Toulouse (IRIT) in 2014. G.B. thanks François Lamarche for some technical hints. Both authors would like to thank Denis Kuperberg for preliminary work in 2015 and for pointing out an error in an earlier version of this manuscript.

2. Outline

Given an automaton $\mathtt{A}$, let $L(\mathtt{A})$ denote the language recognized by $\mathtt{A}$ and let $\mathrm{G}_{\mathtt{A}}$ denote the underlying simple directed graph. The *genus* $g(L)$ of a regular language L is the minimal genus of a deterministic automaton

recognizing the language:

$$g(L) = \min \{g(\mathrm{G}_{\mathtt{A}}) \mid L(\mathtt{A}) = L, \mathtt{A} \text{ deterministic}\}.$$

A directed emulator morphism between directed graphs is a graph morphism π that sends the outgoing edges at each vertex w surjectively onto the outgoing edges at $\pi(w)$. More precisely, a directed graph G' is a *directed emulator* of a directed graph G if there is a directed graph epimorphism $G' \to G$ such that for any vertex v of G, any outgoing edge e starting at v and any vertex v' of G' in the preimage of v, there is an outgoing edge e' starting at v'. A more restrictive notion is that of *directed cover* where the outgoing edges at w are sent bijectively onto the outgoing edges at the image of w.

Let L be a regular language, let $\mathtt{A}_{\min}(L)$ be the Myhill-Nerode minimal automaton recognizing L and let $G(L) = \mathrm{G}_{\mathtt{A}_{\min}}(L)$ be the underlying directed graph.

In a first step, we give alternative descriptions of emulators and covers in terms of automatic relations on a directed graph. These will be Theorem 1 and 2 for emulators and Proposition 8 for covers.

From there, we describe the lattice structure underlying emulators which will lead to minimal objects (in the sense of their size) in the spirit of [8].

Putting these together, we prove Theorem 5 that states that the three following statements are equivalent:

(i) the regular language L has genus g
(ii) the directed graph $G(L)$ has a directed emulator of genus g
(iii) the directed graph $G(L)$ has a directed cover of genus g.

This statement can be reformulated as an equivalence of the decidability of the genus problem for regular languages and directed graphs.

Throughout the paper, some statements are asserted without proof. We do it only when proofs are immediate consequences of the definitions. Details are provided whenever a more elaborate proof is required or a specific construction.

3. Preliminary material

3.1. **Graphs.** Throughout this paper, V and E are finite sets. Given a set (resp. a category) A, we denote by 1_A the identity (resp. the identity functor) on A. Given a set A, we usually denote by $\sim$ an equivalence relation on A. If $\sim'$ is another equivalence relation, we write $\sim \subseteq \sim'$ if $x \sim y$ implies $x \sim' y$.

A *directed graph* (sometimes shortened to *digraph*) G consists of a set V of *vertices* and a set E of *edges* and two maps $s, t : E \rightrightarrows V$ (resp. "source" and "target").

Given an edge e, the notation $x \xrightarrow{e} y$ states that $s(e) = x$ and $t(e) = y$. An edge $x \xrightarrow{e} x$ is a *loop* at vertex x. Given a graph G, we denote by V_G its

vertices, by E_G its edges and by s_G, t_G its corresponding source and target functions. We shall drop subscripts if the context is clear.

A *walk* of length $k \geq 1$ in a graph G starting at $v \in V$ and ending at $w \in V$ is a sequence $e_1, \ldots, e_k$ of edges with $s(e_{i+1}) = t(e_i)$ for $i \leq k-1$ such that $v = s(e_1)$ and $w = t(e_k)$. It will be convenient to regard a single vertex $v \in V$ as a walk of length 0. A vertex $w \in V$ is *reachable from a vertex* $v \in V$ is there is a walk $e_1, \ldots, e_k$ in G such that $s_G(e_1) = v$ and $t_G(e_k) = w$. In particular the target vertex is reachable from the source vertex. The relation $v \leq w$ if and only if w is reachable from v is a preorder on the set V of vertices. A vertex v is *reachable* if v is reachable from any vertex in V. A directed graph G is *reachable* if there is at least one reachable vertex in G. A vertex v is *co-reachable* if any vertex in V is reachable from v. A directed graph G is *co-reachable* if there is at least one co-reachable vertex in G.

Given a directed graph G, the *ordered boundary map* is the map $\Delta_G : E_G \to V_G \times V_G$ defined by $\Delta_G(e) = (s_G(e), t_G(e))$. An edge $x \xrightarrow{e} y \in E_G$ is *simple* if there is no other edge e' such that $\Delta_G(e) = \Delta_G(e')$. The graph G is *simple* if any of its edges is simple.

A *morphism* $G \to H$ *between directed graphs* G and H is a pair (p, q) where $p : V_G \to V_H$ and $q : E_G \to E_H$ satisfy the relations:

$$p \circ s_G = s_H \circ q \quad \text{and} \quad p \circ t_G = t_H \circ q. \tag{1}$$

Setting $p^{\times 2} : V_G \times V_G \to V_H \times V_H,\ (x, y) \mapsto (p(x), p(y))$, the previous equation is equivalent to: $p^{\times 2} \circ \Delta_G = \Delta_H \circ q$. This relation is referred as the *adjacency relation*.

The identity $(1_{V_G}, 1_{E_G}) : G \to G$ is a morphism and morphisms compose.

Lemma 1. *Directed graphs and morphisms between them form a category denoted $\mathscr{D}$. Directed simple graphs and morphisms between them form a full subcategory $\mathscr{D}_S$ of $\mathscr{D}$.*

As defined, the category of graphs is equivalent to the homset category $\mathrm{Hom}_{\mathrm{Fset}}(\cdot \overset{\mathrm{s}}{\underset{\mathrm{t}}{\rightrightarrows}} \cdot, -)$ where Fset is the category of finite sets and maps between them. We do not use it explicitly but the equivalence occurs between the lines in the sequel.

A *graph epimorphism* (resp. *monomorphism*, *isomorphism*) is a graph morphism (p, q) such that both maps p and q are surjective (resp. injective, bijective).

A subset $W \subseteq V$ of vertices of a graph $G = (V, E, s, t)$ determines the graph $G_{|W} = (W, E_W, s|_{E_W}, t|_{E_W})$ where $E_W = \{e \in E \mid \Delta(e) \in W \times W\}$. Similarly, a subset $F \subseteq E$ of edges determines the graph $G_{|F} = (V_G, F, s|_F, t|_F)$. A directed *subgraph* of G is a graph H such that there is a set of vertices W and a set of edges F such that $H = (G_{|W})_{|F}$. We also say that G contains H. If $\phi : G \to H$ is a monomorphism, then G is isomorphic to some directed subgraph of H.

Given a morphism $(p, q) : G \to H$, we define its image graph to be $(p, q)(G) = (p(V_G), q(E_G), s, t)$ with $s(q(e)) = p(s_G(e))$ and $t(q(e)) = p(t_G(e))$ for all $e \in E_G$. It is clear that $(p, q)(G)$ is a subgraph of H and that $(p, q) : G \to (p, q)(G)$ is an isomorphism.

Given a directed graph G, consider the directed graph defined $R(G) = (V_G, \Delta_G(E_G), \pi_1, \pi_2)$ with $\pi_1(x, y) = x$ and $\pi_2(x, y) = y$.

Lemma 2. $\Delta_{R(G)} = 1_{E_{R(G)}}$.

Corollary 2.1. *The graph $R(G)$ is simple.*

Corollary 2.2. *The assignment $G \mapsto R(G)$ extends to a functor $\mathscr{D} \to \mathscr{D}_S$ that assigns to a morphism $(p, q) : G \to G'$ the morphism $R(p, q) : R(G) \to R(G')$ defined by $R(p, q) = (p, p^{\times 2})$.*

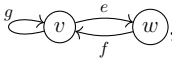
Proof. An edge $\Delta_G(e) \in E_{R(G)}$ is mapped to $p^{\times 2} \circ \Delta_G(e)$. This is a morphism since the adjacency relation is satisfied, for $\Delta_{R(G')} \circ p^{\times 2} = p^{\times 2} = p^{\times 2} \circ \Delta_{R(G)}$ according to Lemma 2. □

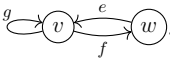
Lemma 3. *The pair $\rho_G = (1_{V_G}, \Delta_G) : G \to R(G)$ is an epimorphism. It is an isomorphism when restricted to simple graphs. Moreover, $G \mapsto \rho_G$ defines a natural transformation $1_{\mathscr{D}} \to I \circ R$ where I denotes the inclusion functor $\mathscr{D}_S \to \mathscr{D}$:*

$$\begin{array}{ccc} G' & \xrightarrow{\rho_{G'}} & R(G') \\ {\scriptstyle\psi}\downarrow & & \downarrow{\scriptstyle R(\psi)} \\ G & \xrightarrow{\rho_G} & R(G) \end{array}$$

Lemma 4. $R \circ I \circ R = R$.

There is yet another endofunctor we shall need. Let $G = (V, E, s, t)$ be a directed graph. Then $G^{\mathrm{op}} = (V, E, t, s)$ is another directed graph. A morphism $(f, g) : G \to G'$ is sent to $(f, g)^{\mathrm{op}} = (f, g) : G^{\mathrm{op}} \to G'^{\mathrm{op}}$.

Example 1. Let $G =$ 

, then $G^{\mathrm{op}} =$ 

. In the drawings, $\{v, w\}$ are the vertices, $\{e, f, g\}$ the edges and arrows describe sources and targets.

We record a few properties of the graphs that are "invariant" under op.

Lemma 5. *The following relations hold:*

$$(G^{\mathrm{op}})^{\mathrm{op}} = G. \tag{2}$$

$$R(G^{\mathrm{op}}) = R(G)^{\mathrm{op}}. \tag{3}$$

We record the following fact which we shall refine later.

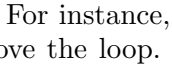
Lemma 6. *The functor $R : \mathscr{D} \to \mathscr{D}_S$ is essentially surjective and full. The endofunctor $(-)^{\mathrm{op}}$ is both full and faithful.*

Proof. R is essentially surjective since $\rho_G : G \to R(G)$ is an isomorphism for simple graphs. Fullness is a direct consequence of Lemma 4. The second statement follows from the identity (2). □

Excising loops in a directed graph consists in removing all loops from the set of edges while keeping other edges and the set of vertices.

Definition 1. Let G be a directed graph. Let $L = \{e \in E_G \mid s_G(e) = t_G(e)\}$ be the subset of loops. The *excision* of G is the graph G with all loops removed:

$$\mathrm{Exc}(G) = G_{|E_G - L}.$$

Remark 1. The excision is not functorial since it cannot be extended to *graph morphisms*. For instance, there is a graph morphism  but not if we remove the loop. We shall see later that in the appropriate category, Exc becomes a functor.

3.2. **Undirected graphs.** An *undirected graph* $G = (V, E, \partial)$ consists of a set V of *vertices*, a set E of *edges* and a map $\partial : E \to \mathcal{P}_2(V)$ from the edges to the set of unordered pairs of vertices[2]. A *morphism* $G \to H$ *between undirected graphs* is a pair of maps $p : V_G \to V_H$ (between vertices) and $q : E_G \to E_H$ (between edges) such that:

$$\partial_H \circ q = p^{\otimes 2} \circ \partial_G \tag{4}$$

with $p^{\otimes 2}(\{x, y\}) = \{p(x), p(y)\}$.

Undirected graphs and morphisms between them form a category denoted by Gr. The canonical projection $pr_V : V \times V \to \mathcal{P}_2(V)$ mapping $(v, w) \mapsto \{v, w\}$ induces a forgetful functor $U : \mathscr{D} \to \mathrm{Gr}$. It maps objects $G \mapsto (V_G, E_G, pr_{V_G} \circ \Delta_G)$ and it acts as the identity on morphisms.

Lemma 7. $U(G^{\mathrm{op}}) = U(G)$.

3.3. **Semi-automata (labelled graphs).** We consider here an enrichment of graphs where one associates a label to each edge. We call these semi-automata. They are sometimes called labelled graphs or transition systems (see for instance [13]). For us, the structure is in-between graphs and automata, thus their denomination. For basic definitions about automata, we refer to [9] and [19].

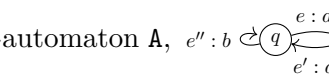
Definition 2. A *semi-automaton* $\mathtt{A}$ is a 3-tuple $\mathtt{A} = (G, \Sigma, \ell)$ made of

- a directed graph G whose vertices and edges are respectively called *states* and *transitions*,
- a fixed set Σ, the *alphabet*, together with a surjective map $\ell : E \to \Sigma$ (labelling of the edges in A).

[2]We define $\mathcal{P}_2(V) = \{\{v, w\} \mid v, w \in V\}$. So, formally, $X \in \mathcal{P}_2(V)$ is either a singleton or a pair.

Again, given a semi-automaton $\mathtt{A}$, we write $G_{\mathtt{A}}$ its underlying graph, an operation that will turn out to be functorial. We use freely $V_{\mathtt{A}}$ for states, $E_{\mathtt{A}}$ for transitions, etc.

Remark 2. Some authors do not assume the labelling ℓ to be onto. In this case, we say the alphabet is *underused*. In this paper, we assume that all semi-automata have no underused alphabet. The hypothesis is required for Lemma 9.

Example 2. In the drawing of a semi-automaton $\mathtt{A}$, 

, e stands for the edge while a denotes its corresponding label.

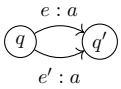
Let $\mathtt{A}$ and $\mathtt{B}$ be two semi-automata. A *morphism* $(f, g, \alpha) : \mathtt{A} \to \mathtt{B}$ is a directed graph morphism $(f, g) : G_{\mathtt{A}} \to G_{\mathtt{B}}$ together with $\alpha : \Sigma_{\mathtt{A}} \to \Sigma_{\mathtt{B}}$ between alphabets such that

$$\alpha \circ \ell_{\mathtt{A}} = \ell_{\mathtt{B}} \circ g. \tag{5}$$

If α is the identity, we say that the morphism is *strict*. If f and g are identities, the morphism is called a *relabelling*.

The identity $(1_{V_{\mathtt{A}}}, 1_{E_{\mathtt{A}}}, 1_{\Sigma_{\mathtt{A}}}) : \mathtt{A} \to \mathtt{A}$ is a (strict) morphism. And it is easily seen that the composition of two morphisms (resp. strict morphisms) is a morphism (resp. a strict morphism). We denote by **Semi** the category of semi-automata and their morphisms and by $\mathbf{Semi}^0$ the category of semi-automata with their strict morphisms. The category $\mathbf{Semi}^0$ is a faithful subcategory of **Semi**.

We defined semi-automata in a slightly unorthodox way because of the tropism towards graphs in this paper. Indeed, the following example



is a semi-automaton for which there are two transitions labelled a between q and q'. This is not compatible with the standard definition of the transitions as a function $\delta : V \times \Sigma \to 2^V$ with V the set of states and Σ the alphabet. But actually, when we will restrict to deterministic automata, such a case will simply vanish.

Let $\mathtt{A}$ be a semi-automaton, W a subset of $V_{\mathtt{A}}$ and $E \subseteq E_{\mathtt{A}}$. Let $(\mathtt{A}_{|W})_{|E} = (G_{|W})_{|E}, \ell(\{e \in \tilde{E}\}), \ell_{e \in \tilde{E}})$ with $\tilde{E} = E_{(G_{|W})_{|E}}$. A semi-automaton produced in this fashion will be called a *sub-semi-automaton* of $\mathtt{A}$.

Given a semi-automaton morphism $(f, g, \alpha) : \mathtt{A} \to \mathtt{B}$, the semi-automaton $(f, g, \alpha)(\mathtt{A}) = ((f, g)(G_{\mathtt{A}}), \alpha(\Sigma_{\mathtt{A}}), \ell_{\mathtt{B}|\alpha(\Sigma_{\mathtt{A}})})$ is a sub-automaton of $\mathtt{B}$.

Lemma 8. *Given a function $\alpha : \Sigma \to \Delta$ and a semi-automaton $\mathtt{A} = (G, \Sigma, \ell)$, the triple $\mathtt{A}_{\backslash\alpha} = (G, \alpha(\Sigma), \alpha \circ \ell)$ is a semi-automaton and furthermore, $(1_{V_{\mathtt{A}}}, 1_{E_{\mathtt{A}}}, \alpha) : \mathtt{A} \to \mathtt{A}_{\backslash\alpha}$ is a relabelling morphism.*

Proof. Let $E = E_{\mathtt{A}_{\setminus\alpha}} = E_{\mathtt{A}}$. We have $\alpha \circ \ell(E) = \alpha(A)$ since ℓ is surjective. So, $\mathtt{A}_{\setminus\alpha}$ is a semi-automaton. Second, we have $(\alpha \circ \ell) \circ 1_{E_{\mathtt{A}_{\setminus\alpha}}} = \alpha \circ \ell$, it is a morphism. □

Proposition 1. *For any morphism ϕ, there is a strict morphism π and a relabelling morphism λ such that $\phi = \lambda \circ \pi$ and a strict morphism ψ and a relabelling μ such that $\phi = \psi \circ \mu$.*

Proof. Given $(f, g, \alpha) : \mathtt{A} \to \mathtt{B}$, one of the decomposition is: $(f, g, \alpha) = (f, g, 1_\Sigma) \circ (1_V, 1_E, \alpha)$ where $V = V_{\mathtt{A}}$, $E = E_{\mathtt{A}}$ and $\Sigma = \Sigma_{\mathtt{B}}$.

The other decomposition is: $(f, g, \alpha) = (1_{V_{\mathtt{B}}}, 1_{E_{\mathtt{B}}}, \alpha) \circ (f, g, \ell/g)$ where $\ell/g(g(e)) = \ell(e)$ for all $e \in E_{\mathtt{A}}$. The triple $(f, g, \ell/g)$ is a proper morphism. Indeed, for all $g(e) \in E_{(f,g,\ell/g)(\mathtt{A})}$, we have $\alpha(\ell/g(g(e))) = \alpha(\ell_{\mathtt{A}}(e)) = \ell_{\mathtt{B}}(g(e)) = \ell_{\mathtt{B}}(1_{E_{\mathtt{B}}}(g(e)))$. Thus, Equation 5 holds. □

By definition, a semi-automaton $\mathtt{A}$ is a directed graph with extra structure. Forgetting this extra structure yields the *underlying* directed graph $G_{\mathtt{A}}$ of the automaton.

Lemma 9. *The assignment $\mathtt{A} \mapsto \mathrm{G}_{\mathtt{A}}$ yields a forgetful faithful functor $\mathrm{G}_{(\_)} :$ **Semi** $\to \mathscr{D}$.*

Proof. Let $\mathtt{A}$ and $\mathtt{B}$ be two semi-automata. Consider the induced map $\mathrm{Hom}_{\mathbf{Semi}}(\mathtt{A}, \mathtt{B}) \to \mathrm{Hom}_{\mathscr{D}}(\mathrm{G}_{\mathtt{A}}, \mathrm{G}_{\mathtt{B}})$. Suppose two morphisms $(f_i, g_i, \alpha_i) : \mathtt{A} \rightrightarrows \mathtt{B}$, $i = 0, 1$, yield the same graph morphism $(f, g) : \mathrm{G}_{\mathtt{A}} \to \mathrm{G}_{\mathtt{B}}$: they coincide at the level of the underlying graph hence the maps on the set V of vertices in $\mathtt{A}$ and the set E of edges in $\mathtt{A}$ coincide: $f_0 = f_1 = f$ and $g_0 = g_1 = g$. Let $\alpha_i : \Sigma_{\mathtt{A}} \to \Sigma_{\mathtt{B}}$, $i = 0, 1$, be the respective maps between the alphabets. Both maps satisfy the relation $\alpha_i \circ \ell_{\mathtt{A}} = \ell_{\mathtt{B}} \circ g$, $i = 0, 1$. Therefore $\alpha_0|_{\ell_{\mathtt{A}}(E)} = \alpha_1|_{\ell_{\mathtt{A}}(E)}$. Since $\mathtt{A}_0$ and $\mathtt{A}_1$ have no underused alphabet, $\alpha_0 = \alpha_1$. □

Remark 3. The functor $\mathrm{G}_{(\_)}$ is not full. As an example, the induced map $\mathrm{Hom}_{\mathbf{Semi}}\left(\begin{array}{c}\scriptstyle e:a\\ \scriptstyle u\\ \scriptstyle v\\ \scriptstyle w\\ \scriptstyle e':a\end{array}, \begin{array}{c}\scriptstyle e:a\\ \scriptstyle u\\ \scriptstyle v\\ \scriptstyle w\\ \scriptstyle e':b\end{array}\right) \to \mathrm{Hom}_{\mathscr{D}}\left(\begin{array}{c}\scriptstyle e\\ \scriptstyle u\\ \scriptstyle v\\ \scriptstyle w\\ \scriptstyle e'\end{array}, \begin{array}{c}\scriptstyle e\\ \scriptstyle u\\ \scriptstyle v\\ \scriptstyle w\\ \scriptstyle e'\end{array}\right)$ is not onto. For instance, the identity on the directed graph is not the image of a semi-automaton morphism.

Remark 4. The functor $\mathrm{G}_{(\_)}$ preserves surjectivity (resp. injectivity).

3.4. **The genus of a graph.** We recall a few definitions. The *genus* $g(\Sigma)$ *of a closed oriented surface* Σ is half the dimension of the first real homology vector space $H_1(\Sigma; \mathbb{R})$. Alternatively, it is the maximum number of mutually disjoint simple closed topologically nontrivial curves $C_1, \ldots, C_g$ such that the complement $\Sigma - (C_1 \cup \cdots \cup C_g)$ remains connected. This yields a natural notion of genus of a graph.

Definition 3 (Genus of a graph)**.** A graph has *genus* n if its geometrical realization is embeddable in a surface of genus n but cannot be embedded in a surface of strictly smaller genus. We note $g(G)$ the genus of a graph G.

The definition makes sense for directed and undirected graphs alike.

Lemma 10. *For any directed graph,* $g(G) = g(U(G))$.

Proof. The geometric realization only depends on the underlying undirected graph. □

All the proofs dealing with the genus of some graph will rely on one of the following observations.

Lemma 11. *If* G *is isomorphic to* H*, then* $g(G) = g(H)$.

Lemma 12. *The functor* $(-)^{\mathrm{op}}$ *preserves the genus.*

Proof. For a directed graph G, $g(G^{\mathrm{op}}) = g(U(G^{\mathrm{op}}))$ (lemma 10) $= g(U(G))$ (lemma 7) $= g(G)$ (lemma 10). □

Lemma 13. *Removing an edge from a graph does not increase the genus: for any graph* G *and* $e \in E_G$*,* $g(G - e) \leq g(G)$.

Lemma 14. *The functor* R *preserves the genus.*

Proof. Induction from the previous lemma. □

Lemma 15. *The excision operation preserves the genus:* $g(\mathrm{Exc}(G)) = g(G)$.

Proof. By the previous lemma, $g(\mathrm{Exc}(G)) \leq g(G)$. On the other hand, (the geometric realization of) a loop at a given vertex embeds as the boundary of a small disc, hence if $\mathrm{Exc}(G)$ embeds in a surface Σ, then G also embeds there, so $g(G) \leq g(\mathrm{Exc}(G))$. □

4. Directed emulators

In this section, we define directed emulators and study their properties. The basic material is introduced in §4.1. Categorical and closure properties are presented in §4.2. Finally we briefly discuss the relation to undirected emulators in §4.3.

4.1. Basic definitions.

The following definition is the main object of this section.

Definition 4. Let $\phi = (p, q) : G \to H$ be a directed graph morphism. It is a *directed emulator morphism* when:

(i) p is surjective and
(ii) ϕ verifies the *edge outgoing lifting property*. That is, for any edge $e \in E_H$ and any vertex $x' \in V_G$ such that $p(x') = s_H(e)$, there is an edge $e' \in E_G$ such that $q(e') = e$ and $s_G(e') = x'$.

We say that the directed emulator morphism ϕ is a *directed cover morphism* whenever in clause (ii), the edge e' is unique.

If $\phi : G \to H$ is a directed emulator morphism (or sometimes shorter: a directed emulator), we say that G is a directed emulator of H and that H is a directed amalgamation of G. Finally, we say that the edge e' in the definition emulates the edge e.

Example 3. The identity $1_G : G \to G$ is a directed emulator morphism. More generally, an isomorphism $\psi : G \to H$ is a directed emulator morphism. It is also a directed cover morphism.

Remark 5. A directed emulator shall not be a directed cover as shown by the morphism: 

.

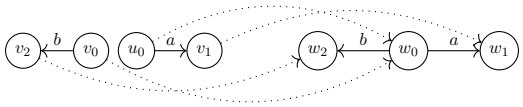
Example 4. For any directed graph G, the map $\rho_G : G \to R(G)$ is a directed emulator morphism.

The following observation is a direct consequence of the definition.

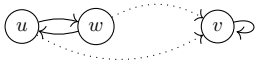
Lemma 16. *A directed emulator morphism is a directed graph epimorphism.*

Proof. By definition, the vertex map is surjective. Given an edge e in the base graph, since p is surjective, there is a vertex $x \in p^{-1}(s_G(e))$ and then, by the edge outgoing lifting property, there is an edge e' such that $q(e') = e$. □

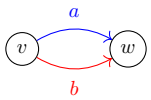
Remark 6. A directed graph epimorphism is not necessarily a directed emulator morphism. For instance, in the epimorphism below, the central node w_0 has two outgoing edges while each element in its preimage (u_0, v_0) has only one.



Example 5. Directed amalgamation can "create" loops:



Remark 7. There are in general several directed emulator morphisms between a digraph and a directed amalgamation of it. For instance, the digraph depicted below is a directed emulator of itself in two ways:



The first one is the identity and the other one swaps the two edges. In both cases, the map on the vertices is the identity.

Definition 5. Let G be a directed graph and $x \in V_G$ a vertex. The *centripetal star* of x is the set $\mathrm{outE}_G(x)$ *of outgoing edges from* x, that is $\mathrm{outE}_G(x) = \{e \in E_G \mid x \xrightarrow{e} y \text{ for some } y \in V_G\}$.

A directed graph morphism $(p, q) : G \to H$ induces, for each vertex $x \in V_G$, a map

$$q_x : \mathrm{outE}_G(x) \to \mathrm{outE}_H(p(x)),\ e \mapsto q(e).$$

Definition 6. If at each $x \in V_G$, q_x is injective (resp. surjective), then we say that $(p, q) : G \to H$ is a directed immersion (resp. a directed submersion).

A directed emulator morphism $(p, q) : G' \to G$ lifts any outgoing edge $e \in E_G$ to an outgoing edge e' from any specified vertex in the preimage $p^{-1}(s_G(e)) \in V_{G'}$. This implies the following observation.

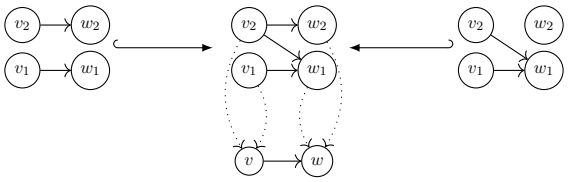
Lemma 17. *A directed graph epimorphism is a directed emulator morphism if and only if it is a directed submersion. A directed cover is both a submersion and an immersion.*

Lemma 18 (Extraction of a directed cover)**.** *Let G' be a directed emulator of a directed graph G. Then there is a directed subgraph G'' of G with the following properties:*

(1) G'' is a directed cover of G;
(2) $V_{G'} = V_{G''}$.

Proof. Let $p(x') = x$. By Lemma 17, the emulator morphism induces a surjection $\mathrm{OutE}(x') \to \mathrm{OutE}(x)$. Remove if necessary some outgoing edges from x' so that a bijection $\mathrm{OutE}(x') \to \mathrm{OutE}(x)$ is obtained. Proceed thus on each vertex in the fibre $p^{-1}(x)$ for each vertex $x \in G$. The restriction of the directed emulator morphism is then a directed cover morphism. □

The construction is not universal. Here are two non isomorphic subgraphs verifying the conditions above:



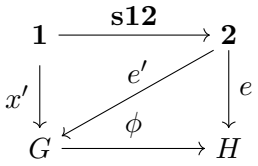
Proposition 2. *Let $g \geq 0$. A directed graph has a directed emulator of genus g if and only if it has a directed cover of genus g.*

Proof. Lemma 18 shows that if a genus g directed graph G' emulates G, then it contains a directed subgraph G'' that covers G. Therefore $g(G'') \leq g(G') = g$. Therefore the class of directed emulators of G contains a directed cover of minimal genus. □

We set $\mathbf{1} = \textcircled{v}$, $\mathbf{2} = \textcircled{v} \longrightarrow \textcircled{w}$ and the morphism $\mathbf{s12} : \mathbf{1} \to \mathbf{2} = [v \mapsto v]$. Definition 4 can be reformulated as follows:

Proposition 3. *A epimorphism $\phi : G \to H$ is a directed emulator if and only if for all square diagram as below there is a morphism e' making the*

two triangles commute. The morphism is a cover if and only if there is at most one morphism e' as a solution.

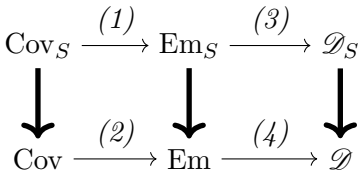


4.2. **The category of emulators.**

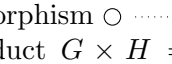
Proposition 4. *The composition of two directed emulators (resp. directed covering) morphisms is a directed emulator (resp. directed covering) morphism.*

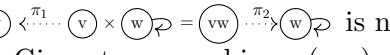
The identities being directed emulators (resp. covers), see Example 3, directed graphs and directed emulator morphisms between them form a subcategory Em of $\mathscr{D}$. Simple directed graphs and directed simple emulator morphisms between them form a subcategory of $\mathscr{D}_S$. The category of covers Cov comes with its subcategory Cov_S of covers over simple graphs.

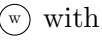
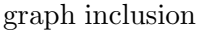
Lemma 19. *To sum up, we have the inclusions:*



Thick arrows correspond to full subcategory relationships whereas thin ones correspond to simple inclusions. For (1) and (2), this is justified by Remark 5. For (3) and (4), we use Remark 6.

The graph  is a terminal object for $\mathscr{D}$, but neither for Em nor Cov since the graph morphism  is neither an emulator nor a cover.

The graph product $G \times H = (V_G \times V_H, E_G \times E_H, s_G \times s_H, t_G \times t_H)$ does not restricts to Em nor Cov. Indeed, the projection morphism π_2 in  is not an emulator.

Given two morphisms $(p, q) : G \to H$ and $(p', q') : G \to H$, the graph inclusion $G_{|v \in V_G : p(v) = p'(v) | \{e \in E_G : q(e) = q'(e)\}} \to G$ is an equalizer for the two morphisms in $\mathscr{D}$. But some pairs of morphisms have no equalizers in Em. Consider for instance the graph $G =$  with its two morphisms 1_G and $Swap = [a \mapsto b, b \mapsto a]$. The graph inclusion  is an equalizer in $\mathscr{D}$ but not in Em nor Cov.

Since directed emulators are epimorphisms, we have:

Proposition 5. *Any morphism in* Em *is right-cancellative:* $\phi_1 \circ \psi = \phi_2 \circ \psi \Rightarrow \phi_1 = \phi_2$.

Lemma 20. *Given a morphism $\phi : G \to K$ in $\mathscr{D}$ and a directed emulator $\phi' : H \to K$, in the pull back diagram:*

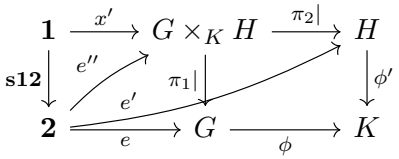
$$\begin{array}{ccc} G \times_K H & \xrightarrow{\pi_2|} & H \\ {\scriptstyle \pi_1|}\downarrow & & \downarrow{\scriptstyle \phi'} \\ G & \xrightarrow{\phi} & K \end{array}$$

the morphism $\pi_1|$ is a directed emulator. The property holds for covers.

Proof. For a direct calculation, we recall that $L = G \times_K H$ defined by

$$V_L = \{(u, v) \in V_G \times V_H \mid p(u) = p'(v)\},$$
$$E_L = \{(e, f) \in E_G \times E_H \mid q(e) = q'(f)\},$$
$$\Delta_L(e, f) = ((s_G(e), s_H(f)), (t_G(e), t_H(f)))$$

is a pullback. But, let us consider the diagram below. We suppose the left square commutes (the right one also commutes being a pullback).



Then, since ϕ' is a directed emulator, there is an edge e' such that $\phi' \circ e' = \phi \circ e$. But, then, due to the pullback, given e and e', there is an edge e'' as shown. Thus, $\pi_1|$ is a directed emulator (Proposition 3). Thanks to the pull-back, it preserves the unicity, thus e'' is unique if ϕ' is a cover. □

Corollary 20.1. *Given a subgraph G of K and a directed emulator morphism $H \to K$, there is a directed morphism $\pi : H' \to G$ with H' a subgraph of H. The property holds for covers.*

Proof. Since pullbacks preserve monomorphisms, the morphism $\pi_{2|}$ above is an monomorphism. Thus, $G \times_K H$ is isomorphic to a subgraph H' of H. □

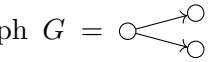
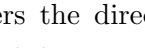
Lemma 21. *We have $R(G \times_K H) \simeq R(G) \times_{R(K)} R(H)$ for all graphs G, K, H.*

Proof. On vertices, since R acts as the identity, the result is immediate. On edges, the result follows from the definitions. □

Proposition 6. *The forgetful functor R preserves directed emulators.*

Proof. Let $(p, q) : G' \to G$ be a directed emulator. Let us verify that $R(p, q) = (p, p^{\times 2}) : R(G') \to R(G)$ is a directed emulator. Let $\Delta_G(e)$ be an edge in $R(G)$ and let $x' \in R(G')$ such that $p(x') = s_{R(G)}(\Delta_G(e)) = s_G(e)$. Since (p, q) is an emulator, there is $e' \in G'$ such that $q(e') = e$ and $s_{G'}(e') = x'$. By definition, $s_{R(G')}(\Delta_{G'}(e')) = s_{G'}(e') = x'$ and $p^{\times 2}(\Delta_{G'}(e')) = \Delta_G(q(e')) = \Delta_G(e)$.

□

Remark 8. The functor R does not preserve directed covers. For instance, the directed graph $G =$  covers the directed graph $H =$  . However, $R(G) = G$ only emulates (and does not cover) $R(H) =$.

That can be summed up by the diagram:

$$\begin{array}{ccccc} \mathrm{Cov}_S & \longrightarrow & \mathrm{Em}_S & \longrightarrow & \mathscr{D}_S \\ \downarrow & & \downarrow\uparrow \mathrm{R} & & \downarrow\uparrow \mathrm{R} \\ \mathrm{Cov} & \longrightarrow & \mathrm{Em} & \longrightarrow & \mathscr{D} \end{array}$$

The removal of loops (Exc) does not determine a functor in the category of directed graphs (Remark 1). The next observation is that Exc becomes one in the category Em^0 of directed graphs with directed emulators maps as morphisms.

Definition 7. A *directed emulator map* from a directed graph G to an other directed graph H is a pair (p, q) made of a surjective map $p : V_G \to V_H$ and a *partial* function $q : E_G \to E_H$ that is compatible with the adjacency relation and such that the pair (p, q) has the edge outgoing lifting property.

Restricted to simple graphs, this yields a full subcategory Em^0_S of Em^0. A directed emulator morphism is a directed emulator map whose edge function is total. Thus, the category Em is a subcategory of Em^0.

Given a morphism $(p, q) : G \to H$, we define $\mathrm{Exc}(p, q) = (p', q') : \mathrm{Exc}(G) \to \mathrm{Exc}(H)$ with $p' = p$ and $q' = q_{|\{e \in E_G : p(s_G(e)) \neq p(t_G(e))\}}$. To justify that the definition is correct, we check three facts. First, the image of q' stays within edges in $\mathrm{Exc}(H)$. Indeed, if $p(s_G(e)) \neq p(t_G(e))$, then $s_H(q(e)) \neq t_H(q(e))$ so that it is not a loop, thus in $\mathrm{Exc}(H)$. Second, being a restriction of q by definition, q' respect the adjacency relation. Third, if e is an edge in $\mathrm{Exc}(H)$ and $x' \in \mathrm{Exc}(G)$ verifies $p(x') = s_H(e)$, there is an edge e' in G such that $q(e') = e$ and $s_G(e') = x'$. Since e' is not a loop (otherwise $q(e')$ would be itself a loop), it is an edge within $\mathrm{Exc}(G)$.

Proposition 7. Exc *is a functor* $\mathrm{Em} \to \mathrm{Em}^0$. *The excision* Exc *is a endofunctor of* Em^0 *that restricts to an endofunctor of* Em^0_S. *Furthermore,*

(1) $\mathrm{Exc} \circ \mathrm{Exc} = \mathrm{Exc}$*;*
(2) $\mathrm{Exc} \circ R = R \circ \mathrm{Exc}$.

Proof. Exc acts on a graph G by removal of all loops; for any directed graphs G, G', Exc acts on the set $\mathrm{Hom}_{\mathrm{Em}^0}(G', G)$ of directed emulator maps as the identity. The results follow. □

Lemma 22 (Extension of a directed cover over an excized graph)**.** *Suppose that $\psi : G' \to G$ is a directed cover morphism between directed graphs. Furthermore, assume that $G = \mathrm{Exc}(H)$. Then there exists a directed graph H' and a directed cover morphism $\varphi : H' \to H$ such that*

(1) $\mathrm{Exc}(H') = G'$*;*

(2) *The directed cover morphism φ extends the directed cover morphism ψ, that is $\varphi|_{G'} = \psi$.*

$$\begin{array}{ccc} H' & \xrightarrow{\text{Exc}} & G' \\ {\scriptstyle\varphi}\downarrow & & \downarrow{\scriptstyle\psi} \\ H & \xrightarrow{\text{Exc}} & G. \end{array}$$

The statement holds replacing covers by emulators.

Proof. The construction is explicit. Let $\psi = (p, q) : G' \to G$ and let E^0 be the set of loops of H. For each loop $e \in E^0$, create a loop $e_{x'}$ in G' at each preimage $x' \in p^{-1}(s(e))$ and set $s(e_{x'}) = t(e_{x'}) = x'$. Set

$$H' = \left(V_{G'}, E_{G'} \bigcup_{e \in E^0} \bigcup_{v' \in p^{-1}(s(e))} e_{v'}, s_{G'} \cup s, t_{G'} \cup t \right).$$

Extend the directed cover morphism ψ to H' by sending each edge $e_{x'}$ to e. Rename the extended direct cover morphism φ. By construction, $\text{Exc}(H') = G'$ and $\varphi|_{G'} = \psi$. This proves (1) and (2). The construction is valid for directed emulators. □

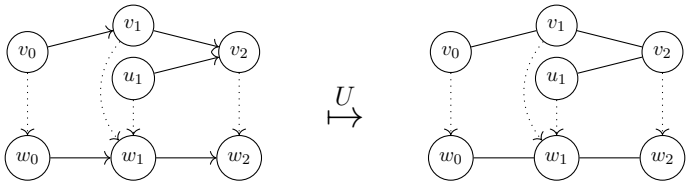
There is a dual notion to the definition of a directed emulator. The original definition distinguishes the outgoing edges. One could distinguish the incoming edges instead. Hence, parallel to Lemma 17, we consider the set $\text{InE}(y) = \{e \in E \mid t(e) = y\}$ of incoming edges at e and require that the map $\text{InE}(y') \to \text{InE}(y)$ induced by the epimorphism be surjective. This defines a new notion of directed emulator, which we call *incoming directed emulator*, while the original notion is renamed an *outgoing directed emulator*. If this does not seem to cause confusion, we shall keep the terminology of directed emulator for an outgoing directed emulator. In contexts where we need to be more precise, we shall use the full terminology. A *bidirected emulator* is a directed graph morphism that is both an incoming directed emulator and an outgoing directed emulator.

Lemma 23. *A directed graph morphism $\phi : G \to H$ is an incoming directed emulator if and only if $\phi^{op} : H^{\text{op}} \to G^{\text{op}}$ is a outgoing directed emulator.*

4.3. **The case of undirected graphs.** In this section, we discuss the relationship between directed and undirected graph emulators. First, we recall the original definition of an undirected graph emulator introduced by M. R. Fellows in his PhD thesis in 1985 (see [10]).

Fellows' definition. Let G be an undirected graph. We say that an undirected graph G' is an *emulator* of G if there is a graph epimorphism $(p, q) : G' \to G$ such that for any edge $e \in E_G$ with $\partial(e) = \{x, y\}$ and any $x' \in V_{G'}$ such that $p(x') = x$, there is an edge $e' \in E_{G'}$ such that $q(e') = e$. Again, we say that it is a cover when the edge e' is uniquely defined.

Note that the functor U does not send directed emulators to emulators (and neither directed covers to covers). For instance, on the left, we have a directed emulator. But, on the right, node u_1 has only one adjacent edge where w_1 has two.



Definition 8. The *bidirection* of an undirected graph G is the directed graph $\overleftrightarrow{G} = (V, E, s, t)$ defined by $V = V_G$ and

$$E = \{(e, x, y) \mid e \in E_G \text{ and } \partial_G(e) = \{x, y\}\},$$
$$s(e, x, y) = x \text{ and } t(e, x, y) = y.$$

Notice that nonloop edges in G are duplicated in $\overleftrightarrow{G}$ whereas the set $\{\partial e = \{x\} \mid e \in E_G\}$ of loops in G are in one-one correspondence with the set $\{(e, x, x) \mid e \in E_G,\ \partial_G(e) = \{x\}\}$ of loops in $\overleftrightarrow{G}$.

Let $\phi = (p, q) : G \to H$ be a graph morphism. We define $\overleftrightarrow{\phi} = (p, q')$ with $q'((e, x, y)) = (q(e), p(x), p(y))$. It is clear that $\Delta_{\overleftrightarrow{H}}(q'(e, x, y)) = \Delta_{\overleftrightarrow{H}}(q(e), p(x), p(y)) = (p(x), p(y)) = p^{\times 2}(\Delta_{\overleftrightarrow{G}}(e))$.

Lemma 24. *The assignment* $\overleftrightarrow{(-)}$ *yields a functor* $\mathrm{Gr} \to \mathscr{D}$ *that is right adjoint to the functor* U*.*

Proof. For the statement about adjoints, let G be a directed graph and let H be an undirected graph. We define a map $\Phi_{G,H} : \mathrm{Hom}_{\mathscr{D}}(G, \overleftrightarrow{H}) \to \mathrm{Hom}_{\mathrm{Gr}}(U(G), H)$ as follows. Let $(p, q) : G \to \overleftrightarrow{H}$ be a directed graph morphism. Let $\pi_3^1 : E_{\overleftrightarrow{H}} \to E_H$, $\pi_3^1(e, x, y) = e$. Then set

$$\Phi_{G,H}(p, q) = (p, \pi_3^1 \circ q) : U(G) \to H.$$

Conversely, given a morphism $(p', q') : U(G) \to H$, define a map $\Psi_{G,H}(p', q') : G \to \overleftrightarrow{H}$ by

$$\Psi_{G,H}(p', q') = (p', q''), \text{ with } q''(e) = (q'(e), p'(s_G(e)), p'(t_G(e))).$$

The maps $\Phi_{G,H}$ and $\Psi_{G,H}$ are inverse of each other, hence $\Phi_{G,H} : \mathrm{Hom}_{\mathscr{D}}(G, \overleftrightarrow{H}) \to \mathrm{Hom}_{\mathrm{Gr}}(U(G), H)$ is bijective. Naturality of the isomorphism Φ should be clear. □

Lemma 25. *The morphism* $\phi = (p, q) : G \to H$ *is an emulator if and only if* $\overleftrightarrow{\phi} : \overleftrightarrow{G} \to \overleftrightarrow{H}$ *is a directed emulator.*

Remark 9. Note that $\overleftrightarrow{\phi} : \overleftrightarrow{G} \to \overleftrightarrow{H}$ is a directed emulator if and only if it is a bidirected emulator. (Proof: use the natural isomorphism $(\overleftrightarrow{G})^{\mathrm{op}} \simeq \overleftrightarrow{G}$ for any G and Lemma 23.)

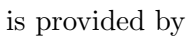
Proof. Suppose ϕ is an emulator. Since p is surjective, we only have to verify the outgoing edge lifting property. Suppose now $(e, x, y) \in \overleftrightarrow{H}$ and $p(x') = x$ with $x' \in \overleftrightarrow{G}$. Since ϕ is an emulator, there is an edge $e' \in E_G$ such that $q(e') = e$ and $\partial_G(e') = \{x', y'\}$ for some $y' \in V_G$. We have $q'(e', x', y') = (q(e'), p(x'), p(y')) = (e, x, p(y'))$. Do we have $p(y') = y$? Since $\Delta_H(q(e)) = p^{\otimes 2}(e) = \{p(x'), p(y')\} = \{x, y\}$, either $x = p(x') = p(y') = y$ or $p(x') \neq p(y')$ and since $p(x') = x$, we have $p(y') = y$.

Suppose now that $\overleftrightarrow{\phi} = (p, q')$ is a directed emulator. Again, since p is surjective, we only have to verify the edge lifting property. Let $e \in E_H$ with $\partial_H(e) = \{x, y\}$ and $x' \in V_G$ with $p(x') = x$. Then, $(e, x, y) \in E_{\overleftrightarrow{H}}$ and $x' \in V_G$ leads to an edge $(e', x', y') \in E_{\overleftrightarrow{G}}$ with $q'(e', x', y') = (e, x, y)$. By definition of q', that means that $q(e') = e$. □

Lemma 26. *Suppose that $\phi : G \to \overleftrightarrow{H}$ is a directed emulator, then, there is an emulator $\phi' : G' \to H$ with $g(G') = g(G)$.*

Proof. Since U is left adjoint to $\overleftrightarrow{(-)}$, Lemma 24 provides an undirected graph morphism $\phi' : U(G) \to H$. It is clear that $g(\overleftrightarrow{U(G)}) = g(G)$. It remains to check that ϕ' is an emulator morphism. For this, let $\phi = (p, q)$ so that $\phi' = (p, \pi_3^1 \circ q)$ with $\pi_3^1(e, x, y) = e$. Let $e \in E_H$ such that $\partial_H(e) = \{x, y\}$ and $x' \in V_{U(G)} = V_G$ such that $p(x') = x$. Then, $x \xrightarrow{(e,x,y)} y \in \overleftrightarrow{H}$ and $p(x') = x$ implies the existence of $e' \in E_G$ such that $q(e') = (e, x, y)$ with $s_G(e') = x'$. We have $\pi_1^3 \circ q(e') = e$ and $p(e') = x'$ as expected. □

Remark 10. The converse of Lemma 26 does not hold: the directed graph morphism $G \to \overleftrightarrow{H}$ induced by an emulator morphism $U(G) \to H$ is not a directed emulator morphism in general. A counterexample is provided by $U(G) = H =$.

Remark 11. The isomorphism $\mathrm{Hom}_{\mathscr{D}}(G, \overleftrightarrow{H}) \to \mathrm{Hom}_{\mathrm{Gr}}(U(G), H)$ provided by Lemma 24 is shown in the proof of Lemma 26 to restrict to an isomorphism from directed emulators to undirected emulators. However, it does not restrict to an isomorphism from *directed covers* to *undirected covers*. A counterexample is provided by $G =$  and $H =$  .

A (complete) *direction* $\vec{G}$ of an undirected graph G is a subgraph H of $\overleftrightarrow{G}$ such that $U(H)$ is isomorphic to G. It is easy to see that given a (complete) direction $\vec{G}$ of G and a graph morphism $G' \to G$, there exists a (complete) direction $\vec{G'}$ of G' inducing a directed graph morphism $\vec{G'} \to \vec{G}$. This observation does not hold in the categories of emulators and directed emulators. For instance, the picture opposite represents an undirected emulator (actually even an undirected cover). However, there is no choice of directions that turns it into a directed emulator: no matter which directions are

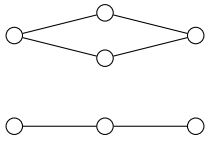
chosen, one of the two vertices in the preimage of the vertex
fails to satisfy the lifting outgoing edge property.

Lemma 27 (Lifting lemma)**.** *Let $\phi : G' \to G$ be an undirected graph emulator morphism onto a loopless graph G. Given a direction $\vec{G}$ of G, there is a direction $\vec{G'}$ of G' and a directed emulator $\phi' : \vec{G'} \to \vec{G}$.*

Proof. By Lemma 25, $\overleftrightarrow{\phi} : \overleftrightarrow{G'} \to \overleftrightarrow{G}$ is a directed emulator morphism. Then, apply Corollary 20.1 to the subgraph $\vec{G}$ of $\overleftrightarrow{G}$: there is a subgraph $H = \overleftrightarrow{\phi}^{-1}(\vec{G})$ of $\overleftrightarrow{G'}$ such that $\overleftrightarrow{\phi}$ restricts to a directed emulator $H \to \vec{G}$. If G is loopless, then $H = \overleftrightarrow{\phi}^{-1}(\vec{G})$ is a direction of G'. □

Remark 12. If G has loops then H may not be a direction of G, as the example in the previous paragraph shows.

The undirected version of the extraction of a directed cover from a directed emulator (Lemma 18) fails to hold. The following picture depicts a 4-vertex undirected emulator of a simple undirected 3-vertex graph that does not contain a 4-vertex cover as a subgraph.



It is left to the reader to verify on this example that any direction of the bottom graph yields a directed emulator on the top by lifting directions (Lemma 27) and that from it a directed cover with all the original vertices can be extracted (Lemma 18).

In other words, roughly speaking, it is much easier to find directed emulators compared to emulators (or bidirected emulators).

For the remainder of the paragraph, let us define, for a directed (resp. undirected) graph G,

$$g_{\text{cover}}(G) = \min\{g(G') \mid G' \text{ covers } G\}, \;\; g_{\text{em}}(G) = \min\{g(G') \mid G' \text{ emulates } G\}.$$

By Proposition 2 above, $g_{\text{cover}}(G) = g_{\text{em}}(G)$ for any *directed* graph G. One can ask whether this equality also holds for undirected graphs. In 1999, P. Hliněnỳ [12] found an emulator such that

$$g_{\text{em}}(G) \leq 3 < g_{\text{cover}}(G).$$

In particular, there exist undirected emulators of G that do not contain covers of G.

M. Fellows proposed the following conjecture for undirected graphs in 1985.

Conjecture 1. *A connected graph has a finite planar emulator if and only if it has a finite planar cover.*

The conjecture was proved false in 2009 by Y. Rieck and Y. Yamashita who found a counterexample [17]. Therefore, there are graphs G such that $g_{\mathrm{em}}(G) = 0$ and $g_{\mathrm{cover}}(G) \geq 1$.

The following conjecture is still open.

Conjecture 2 (Negami)**.** *A connected graph has finite planar cover if and only if its embeds into the projective plane.*

5. An automatic description of directed emulators and covers

5.1. **Automatic relations.** In this paragraph we give an alternative description of directed emulators and covers. Let $G = (V, E, s, t)$ be a directed graph.

Definition 9. A pair of equivalence relations $(\sim_V, \sim_E)$ on respectively V and E is said to be *automatic* when for all edges e, e', and any vertex x':

(i) $e \sim_E e' \implies s(e) \sim_V s(e') \ \wedge \ t(e) \sim_V t(e')$;
(ii) $x' \sim_V s(e) \implies \exists e' \in E : e' \sim_E e \ \wedge \ s(e') = x'$.

Clause (i) is next called *compatibility* (of $\sim_E$ with respect to $\sim_V$). Clause (ii) is the *bisimilarity* of $\sim_V$ with respect to $\sim_E$.

Given an equivalence relation $\sim_V$ on vertices, let $e \sim_E e' \iff (s(e) \sim_V s(e')) \wedge (t(e) \sim_V t(e'))$. Note that $\sim_E$ is compatible with $\sim_V$. If $\sim_V$ is bisimilar with respect to $\sim_E$, the pair $(\sim_V, \sim_E)$ forms an automatic relation. Such a relation is said to be vertex-induced.

Given an equivalence relation $\sim$ on V, we denote $[x]_\sim = \{z \in V \mid z \sim x\}$ be the equivalence class of $x \in V$. If the context is clear, we write $[x]$ for $[x]_\sim$. We let $V/\sim \, = \{[v] \mid v \in V\}$. In order to avoid notation overload we remove subscripts from $\sim_V$ and $\sim_E$ whenever this does not seem to cause confusion.

Lemma 28. *An automatic relation $\sim$ on G induces a new directed graph $G/\sim \, = (V_G/\sim, E_G/\sim, s/\sim, t/\sim)$ together with a morphism $(x \mapsto [x]_{\sim_V}, e \mapsto [e]_{\sim_E}) : G \to G/\sim$ called the canonical morphism (with respect to $\sim$). It is a directed emulator.*

Proof. It is clear that $[-]_\sim$ is an epimorphim. Let us verify the outgoing edge lifting property. Let $v \in V$ and let $[v] \xrightarrow{[e]} [w]$ be an edge in $G/\sim$. Since $[-]_\sim$ is onto, there is an edge $x \xrightarrow{e} y \in E$ such that $v \sim x$. By bisimilarity, there is an edge $v \xrightarrow{e'} v'$ in G such that $e' \sim e$ so that $[e'] = [e]$. □

We say that $G/\sim$ is an *automatic quotient* of G.

Example 6. Given any directed graph G, the identity relation $\sim$ on both vertices and edges is automatic and $G/\sim \, \simeq G$.

Lemma 29. *If $\sim$ is vertex-induced automatic then $G/\sim$ is simple.*

Let 1_V be the equality on the set V of vertices of some graph G. We denote $\sim_G$ its vertex-induced automatic relation.

Corollary 0.1. $G/\sim_G \simeq R(G)$.

Proposition 8. *The morphism associated to a pair of automatic relations $(\sim_V, \sim_E)$ is a cover if and only if (iii): $(e \sim_E e' \wedge e \neq e') \implies s(e) \neq s(e')$.*

Proof. Clause (iii) is a reformulation of the injectivity of the projection. □

In other words, adding clause (iii) in Definition 9 lead to a description of covers.

Proposition 9. *Let $(p, q) : G \to H$ be a directed emulator morphism. The relation $\sim$ defined on G by $v \sim_V w$ if $p(v) = p(w)$ and by $e \sim_E e'$ if $q(e) = q(e')$ is an automatic equivalence relation on G. Furthermore, H is simple if and only if $\sim$ is vertex-induced.*

Proof. The relation $\sim$ is clearly an equivalence relation. Let $x \sim y$ be two equivalent vertices. Let $x \xrightarrow{e} x' \in E_G$ be an edge. Applying q yields an edge $p(x) \xrightarrow{q(e)} p(x') \in E_H$. By the outgoing edge lifting property, there is an edge $y \xrightarrow{e'} y'$. We have $q(e') = q(e)$ so $p(y') = p(x')$, i.e. $x' \sim_V y'$. Hence $\sim_V$ has the bisimilarity property with respect to $\sim_E$. Compatibility is a direct consequence of adjacency preservation of q with respect to p.

For the second statement, the quotient by a vertex-induced automatic relation is simple by Lemma 29. Conversely, suppose that H is simple. Consider a pair of edges $e, e' \in G$ such that $s(e) \sim_V s(e')$ and $t(e) \sim_V t(e')$, i.e. $\Delta(q(e)) = \Delta(q(e'))$. Since H is simple, $q(e) = q(e')$. Thus $e \sim_E e'$. The conclusion follows. □

Definition 10. The relation defined by Prop. 9 is the *canonical automatic relation* associated to the directed emulator morphism $G \to H$.

Proposition 10 (composition of automatic relations)**.** *Let $\sim_1$ be an automatic relation on a directed graph G and $\sim_2$ an automatic relation on $G/\sim_1$. Define the relation $\sim$ on G by $e \sim e'$ if $[e]_{\sim_1} \sim_2 [e]_{\sim_1}$ and by $v \sim v'$ if $[v]_{\sim_1} \sim_e [v']_{\sim_1}$. The relation $\sim$ is automatic and verifies $[-]_\sim = [-]_{\sim_2} \circ [-]_{\sim_1}$.*

Proof. By definition, $[e]_\sim = [[e]_{\sim_1}]_{\sim_2}$ and $[v]_\sim = [[v]_{\sim_1}]_{\sim_2}$ for any edge e and any vertex v in G. The fact that the relation is automatic follows from the definition. □

Remark 13. Recall that an isomorphism $\phi : G \to H$ of directed graphs is a directed cover (Example 3). The canonical automatic relation $\sim$ associated to any isomorphism ϕ is the identity relation (defined by the identity on the vertices and the identity on the edges).

Theorem 1. *Any directed emulator morphism $\phi : G \to H$ splits in a unique way as $\phi = \iota \circ [-]_\sim$ where $\sim$ an automatic relation on G and ι is an isomorphism.*

Proof. Let $\sim$ be the canonical automatic relation related to $\phi = (p, q)$. We define $\iota = (p', q')$ with $p' : [v]_\sim \mapsto p(v)$ and $q' : [e]_\sim \mapsto q(e)$. Let us verify that the definition is correct. For two edges e_1 and e_2, we have $[e_1]_\sim = [e_2]_\sim$ iff $e_1 \sim e_2$ iff $q(e_1) = q(e_2)$. Actually, this proves that q' is injective. Moreover q' is surjective since q is surjective itself. Similarly p' is bijective. Thus ι is an isomorphism. Finally, by construction, $\iota \circ [-]_\sim = \phi$.

Suppose there is an other decomposition $\phi = \iota' \circ [-]_{\sim'}$. Suppose that $[v]_\sim \neq [v]_{\sim'}$. Thus, either there is some $u \sim v$ and $u \not\sim' v$ or $u \not\sim v$ and $u \sim' v$. In the first case, $\iota'([u]_{\sim'}) = p(u) = p(v) = \iota'([v]_{\sim'})$ contradicts the fact that ι' is an isomorphism. Otherwise, $u \sim' v$ but not $u \sim v$. Thus, $p(v) \neq p(u)$. However, $u \sim' v$ means $[u]_{\sim'} = [v]_{\sim'}$ leading to $p(u) = \iota'([u]_{\sim'}) = \iota'([v]_{\sim'}) = p(v)$. Again, a contradiction. The same argument applies to edges. □

Example 7. In the particular case when $\phi : G \to H$ is an isomorphism of directed graphs, $[-]_\sim = 1_G$ and $\iota = \phi$.

As suggested by the previous example, directed graph isomorphisms and canonical epimorphisms associated to automatic relations appear to be "orthogonal" notions.

Proposition 11. *The pair $(\mathscr{A}, \mathscr{I})$, that consists of the class of canonical epimorphisms associated to automatic relations on one side and the class of directed graph isomorphims on the other, is a factorization system in the category* Em.

Proof. The class of canonical epimorphisms associated to automatic relations is closed under composition according to Proposition 10, as well as the class of directed graph isomorphisms. Theorem 1 provides a decomposition $\phi = \iota \circ [-]_\sim$ for any directed emulator morphism ϕ where $\iota \in \mathscr{I}$ and $[-]_\sim \in \mathscr{A}$. It is easy to check that the decomposition is functorial. □

We are on the way to Theorem 3 as this is done by T. Colcombet and D. Petrişan in [8].

5.2. **MN-recursive relations.** In this paragraph, the definition of an automatic relation is refined by labelling the edges. The first observation is that a directed graph endowed with an automatic relation induces naturally a semi-automaton. We then describe automatic relations in terms of a certain kind of recursive relations reminiscent of the Myhill-Nerode relation.

Definition 11. A directed graph G endowed with an automatic relation $\sim$ induces a semi-automaton (G, Σ, ℓ) defined by

$$\Sigma = E_G/\!\sim_E \text{ and } \ell : E \to E/\!\sim_E, e \mapsto [e].$$

This transition system, denoted $\mathtt{A}_\sim$ is the *canonical semi-automaton associated to the automatic relation* $\sim$.

Remark 14. The canonical semi-automaton $\mathtt{A}_\sim$ depends only on the equivalence relation $\sim_E$ on the set E_G of edges. However we are mainly interested in refinements of automatic relations in semi-automata. See for instance Lemma 30 and Definition 14.

Remark 15. The (important) special case when the automatic relation is the identity on vertices and edges (tautological semi-automata) is studied in 6.1.

Definition 12. A pair of equivalence relations $\sim$ on a semi-automaton $\mathtt{A}$ is said to be *automatic* if it is an automatic relation on its underlying graph $G_{\mathtt{A}}$ and $e \sim_E e'$ implies $\ell(e) = \ell(e')$ for all edges e, e'.

The following observation should be clear.

Lemma 30. *A pair of relations $(\sim_E, \sim_V)$ on a directed graph G is automatic if and only if it is automatic for the canonical semi-automaton $\mathtt{A}_\sim$.*

Proof. Suppose that $(\sim_V, \sim_E)$ is automatic. Labelling the set E of edges by $\ell : E \to E/\!\sim_E,\ e \mapsto [e]$ makes the relation automatic for the semi-automaton. The converse is trivial. □

For the following definitions, we consider a fixed semi-automaton $\mathtt{A}$ and a family $\mathscr{F} = \{F_i\}_{i\in I}$ of non-empty pairwise disjoint subsets of the set $V_{\mathtt{A}}$ of vertices of $\mathtt{A}$.

Definition 13. We define on $V_{\mathtt{A}}$ the equivalence relation $x \overset{\mathscr{F}}{\underset{0}{\sim}} y \iff \exists i : (x \in F_i \Leftrightarrow y \in F_i)$. Given $n \geq 1$, let $\overset{\mathscr{F}}{\underset{n}{\sim}}$ be the least equivalence relation such that $x \overset{\mathscr{F}}{\underset{n}{\sim}} y$ holds whenever the two following conditions are satisfied:

(1) $x \overset{\mathscr{F}}{\underset{n-1}{\sim}} y$;

(2) for each edge $x \xrightarrow{e} x'$, there is an edge $y \xrightarrow{f} y'$ such that $\ell(f) = \ell(e)$ and $x' \overset{\mathscr{F}}{\underset{n-1}{\sim}} y'$.

It follows from the definition that there exists $n \leq |V_{\mathtt{A}}|$ such that $\overset{\mathscr{F}}{\underset{n}{\sim}} = \overset{\mathscr{F}}{\underset{n+1}{\sim}}$. It follows from that we can drop the underscript n from the notation as long as $n \geq |V_{\mathtt{A}}|$. We shall simply write $u \overset{\mathscr{F}}{\sim} v$ without further comment.

Remark 16. A family of disjoint non-empty subsets of V naturally induces a partition of V as follows. Let $\mathscr{F} = \{F_i,\ i \in I\}$ be any family of disjoint non-empty subsets of V. Let $\mathscr{F}'$ be the partition of V obtained from $\mathscr{F}$ by adding the complement in V of the union of all subsets F_i, i.e., let $\mathscr{F}' = \mathscr{F} \cup \{(\cup_{i\in I} F_i)^{\complement}\}$. Then $\overset{\mathscr{F}}{\sim} = \overset{\mathscr{F}'}{\sim}$. That is, both the family $\mathscr{F}$ and *the partition $\mathscr{F}'$ induced by $\mathscr{F}$* yield the same equivalence relation.

Example 8 (Myhill-Nerode equivalence relation). Suppose that $\mathscr{F}$ consists of one unique subset F of states (or if one thinks in terms of partition, of F and $F^{\complement}$): $\mathscr{F} = \{F\}$. In this case we denote the relation $\overset{\{F\}}{\sim}$ simply by $\overset{F}{\sim}$. Set F to be the subset of final states. The relation $\overset{\{F\}}{\sim}$ is nothing but the Myhill-Nerode equivalence relation on the set of states, expressed algorithmically in order to recursively build all classes of equivalent states.

Lemma 31. *If the partition $\mathscr{G}$ is finer than $\mathscr{F}$ then $\overset{\mathscr{G}}{\sim} \subseteq \overset{\mathscr{F}}{\sim}$.*

Definition 14 (MN-recursive relation in a semi-automaton)**.** We define a relation $MN(\mathtt{A}, \mathscr{F}) = (\overset{\mathscr{F}}{\sim}_V, \overset{\mathscr{F}}{\sim}_E)$ on $\mathtt{A}$ by setting

$$e \overset{\mathscr{F}}{\sim}_E e' \iff s(e) \overset{\mathscr{F}}{\sim}_V s(e') \ \wedge \ t(e) \overset{\mathscr{F}}{\sim}_V t(e') \ \wedge \ \ell(e) = \ell(e').$$

Such a relation is said to be *MN-recursive*.

Lemma 32. *If the partition $\mathscr{G}$ is finer than $\mathscr{F}$ then* $\mathrm{MN}(\mathtt{A}, \mathscr{G}) \subseteq \mathrm{MN}(\mathtt{A}, \mathscr{F})$.

Proof. Follows from Lemma 31 and the definition of an MN-recursive relation. □

Lemma 33. *An MN-recursive relation is automatic.*

Proof. By construction, $\sim_E$ is compatible with $\overset{\mathscr{F}}{\sim}$ and the labelling. Let us check $\overset{\mathscr{F}}{\sim}$ is bisimilar with respect to $\sim_E$. Suppose $v \overset{\mathscr{F}}{\sim} w$ and $v \xrightarrow{e} v' \in G_{\mathtt{A}}$. By definition, there is an edge $w \xrightarrow{f} w'$ such that $v' \overset{\mathscr{F}}{\sim} w'$ and $\ell(e) = \ell(f)$. But then, by definition, $e \sim_E f$ is as expected. □

The next two observations aim at identifying MN-recursive relations. The first observation says that in order to prove that an MN-recursive relation on a canonical semi-automaton coincides with an automatic relation, it suffices to show that they coincide on the set of vertices. The second observation gives a partial criterion for this.

Lemma 34. *Let $\sim = (\sim_V, \sim_E)$ be an automatic relation on a directed graph G and let $\mathscr{F}$ be any partition of V_G. Then,* $\mathrm{MN}(\mathtt{A}_\sim, \mathscr{F}) = \sim$ *if and only if* $\overset{\mathscr{F}}{\sim}_V = \sim_V$.

Proof. Trivially if the two relations coincide, they coincide on the set of vertices. Conversely suppose that $\sim_V = \overset{\mathscr{F}}{\sim}_V$. The equality of the relations on the edges $\overset{\mathscr{F}}{\sim}_E = \sim_E$ follows from the series of equivalences

$$\begin{aligned} e \overset{\mathscr{F}}{\sim}_E e' &\iff s(e) \overset{\mathscr{F}}{\sim}_V s(e') \ \wedge \ t(e) \overset{\mathscr{F}}{\sim}_V t(e') \ \wedge \ \ell(e) = \ell(e') \\ &\iff s(e) \sim_V s(e') \ \wedge \ t(e) \sim_V t(e') \ \wedge \ e \sim_E e' \\ &\iff e \sim_E e'. \end{aligned}$$

□

Remark 17. The lemma is true, more generally, if in the statement we replace the equality $=$ between the relations by $\subseteq$ or $\supseteq$. The proof is mutatis mutandis the same.

Lemma 35. *Let $\sim = (\sim_V, \sim_E)$ be an automatic relation on a directed graph G and let $\mathscr{F}$ be any partition of V_G. Then $\sim \subseteq \mathrm{MN}(\mathtt{A}_\sim, \mathscr{F})$ if and only if $\sim_V \subseteq \overset{\mathscr{F}}{\underset{0}{\sim}}_V$.*

Proof. According to Remark 17, $\sim \subseteq \overset{\mathscr{F}}{\sim}$ if and only if $\sim_V \subseteq \overset{\mathscr{F}}{\sim}_V$. Thus it is enough to prove that $\sim_V \subseteq \overset{\mathscr{F}}{\sim}_V$ if and only if $\sim_V \subseteq \overset{\mathscr{F}}{\underset{0}{\sim}}_V$. The direct implication is trivial. Let us prove the converse. For simplicity, we drop the subscript V from the notation whenever there should be no confusion. Suppose that $x \sim y$ implies $x \overset{\mathscr{F}}{\underset{0}{\sim}} y$ for any $x, y \in V_G$. Suppose that $x \sim y$. We shall prove that $x \overset{\mathscr{F}}{\underset{n}{\sim}} y$ for any n by induction on $n \geq 0$. By assumption, the base case $n = 0$ holds. Suppose (induction hypothesis) that we have proved that $x \sim y$ implies that $x \overset{\mathscr{F}}{\underset{n-1}{\sim}} y$. Note that this implies that condition (1) of Definition 13 holds. Let us verify that condition (2) holds as well. Given an edge $x \xrightarrow{e} x'$, the bisimilarity property for $\sim_V$ with respect to $\sim_E$ yields an edge $y \xrightarrow{f} y'$ such that $e \sim_E f$. The compatibility property for $\sim_V$ with respect to $\sim_E$ then yields $x' = t(e) \sim_V t(f) = y'$. By the induction hypothesis, this implies that $x' \overset{\mathscr{F}}{\underset{n-1}{\sim}} y'$. Hence $x \overset{\mathscr{F}}{\underset{n}{\sim}} y$ holds as claimed. □

It will be convenient to define, for any vertex $w \in V$, the subset

$$\mathrm{Pr}(w) = \{v \in V \mid \text{there is a walk starting at } v \text{ and ending at } w\}.$$

We accept length 0 walks so $w \in \mathrm{Pr}(w)$ for any vertex $w \in V_G$. The existence of a walk from v to w (w is reachable from v) implies $\mathrm{Pr}(v) \subseteq \mathrm{Pr}(w)$. A subset $\{v_i\}_{i \in I}$ of vertices in V_G such that $V = \cup_{i \in I} \mathrm{Pr}(w_i)$ will be called a *complete final system of the directed graph G*. A complete final system $\{v_i\}_{i \in I}$ of G is *minimal* if no strict subfamily $\{v_j\}_{j \in J}$, $J \subset I, J \neq I$, is a final system of G. Although we shall not use it in the sequel, note that the cardinality of a minimal complete final system is an invariant of the directed graph.

Lemma 36. *Minimal complete final systems of G have the same cardinality.*

Proof. Let S, T be two minimal complete final systems. Let $s \in S$. By completeness of T, there is a walk starting at s ending at some $t \in T$. By completeness of S, there is a walk starting at t ending at some $s' \in S$. Now minimality implies $s = s'$ for otherwise since $\mathrm{Pr}(s) \subseteq \mathrm{Pr}(s')$, the smaller system $S - \{s\}$ would still be complete. Thus $s \in S$ is paired to an element $t \in T$ such that $\mathrm{Pr}(s) = \mathrm{Pr}(t)$. Moreover there is no further walk from s to some $t' \in T$, $t' \neq t$, for otherwise the smaller system $T - \{t\}$ would still be

complete. Therefore, the assignment that sends $s \in S$ to the unique $t \in T$ such that $\mathrm{Pr}(t) = \mathrm{Pr}(s)$ is a well-defined map $S \to T$. The symmetry of S and T implies that it is bijective. □

We are now ready to prove a converse to Lemma 33.

Proposition 12 (Automatic relations are MN-recursive)**.** *If a pair of relations $\sim = (\sim_V, \sim_E)$ on a directed graph G is automatic, then $\sim = \mathrm{MN}(\mathtt{A}_\sim, \mathscr{F})$ for the canonical semi-automaton $\mathtt{A}_\sim$ associated to $\sim_E$ and for the family $\mathscr{F}$ that consists of the $\sim_V$-equivalence classes of the vertices of any complete final system of G.*

Proof. Let S be a minimal complete final system for G and let $\mathscr{F}$ denote the corresponding partition of V. In accordance with Remark 16, each element in $\mathscr{F}$ is $[s]_{\sim_V}$, $s \in S$, and possibly (if non empty) the subset $(\cup_{s\in S}[s]_{\sim_V})^{\complement}$. In accordance to Lemma 34, it suffices to prove that $\sim_V = \overset{\mathscr{F}}{\sim}_V$.

For simplicity, we drop the subscript V from the notation. Let us prove first that $\sim_V \subseteq \overset{\mathscr{F}}{\sim}_V$. Suppose that $x \sim y$. Either $x \in [s]$ for some $s \in S$ (and then x belongs to exactly one $[s] \in \mathscr{F}$) or $x \in (\cup_{s\in S}[s]_{\sim_V})^{\complement}$ (and then x also belongs to exactly one class in $\mathscr{F}$). Since $x \sim y$, it follows that y has the same property as x. Hence $x \overset{\mathscr{F}}{\underset{0}{\sim}} y$. Then by Lemma 35, $x \overset{\mathscr{F}}{\sim} y$.

Conversely, assume that $x \overset{\mathscr{F}}{\sim} y$. We have to prove that $x \sim y$. By assumption $x \overset{\mathscr{F}}{\underset{0}{\sim}} y$, that is, either there is $s \in S$ such that $[x] = [y] = [s]$ (and then $x \sim y$) or $x, y \in (\cup_{s\in S}[s])^{\complement}$. Given a vertex x in G, there is at least one walk from x to (one of the vertices in) $[s]$ for some $s \in S$. That justifies to proceed by induction on the length of such a walk (of minimal length). By such an induction, we prove that $x \overset{\mathscr{F}}{\sim} y$ implies $x \sim y$. The walk has length 0 if and only if $[x] = [s] = [y]$, as we've just already observed. Otherwise, there is an edge $x \xrightarrow{e} x'$, where the vertex x' is "closer" to $[s]$ for some $s \in S$. Since the relation $\overset{\mathscr{F}}{\sim}$ is automatic, there is an edge $y \xrightarrow{f} y'$ with $e \overset{\mathscr{F}}{\sim}_E f$. Hence $\ell(e) = \ell(f)$ for the label ℓ of the canonical semi-automaton $\mathtt{A}_\sim$, thus $e \sim_E f$. So $x = s(e) \sim_V s(f) = y$ by compatibility of $\sim_E$ with respect to $\sim_V$. □

Theorem 2. *A relation on a directed graph G is automatic if and only if it is MN-recursive with respect to some semi-automaton $\mathtt{A} = (G, \Sigma, \ell)$. More precisely, given a pair of relations $\sim = (\sim_V, \sim_E)$ on a directed graph G, the following assertions are equivalent:*

1. *The relation $\sim$ on G is automatic.*
2. *The relation $\sim$ on the canonical semi-automaton $\mathtt{A}_\sim$ is automatic.*
3. *$\sim = \mathrm{MN}(\mathtt{A}_\sim, \mathscr{F})$ where $\mathscr{F}$ is the family of $\sim_V$-equivalence classes of the vertices of any complete final system of G.*

(4) $\sim = \mathrm{MN}(\mathtt{A}_\sim, \mathscr{G})$ *where* $\mathscr{G}$ *is the partition of* V *into* $\sim_V$*-equivalence classes.*

Proof. (1) $\iff$ (2): Lemma 30. (2) $\implies$ (3): Prop. 12. To end the proof, we have to prove that $\mathrm{MN}(\mathtt{A}_\sim, \mathscr{G}) = \mathrm{MN}(\mathtt{A}_\sim, \mathscr{F})$. Since $\mathscr{G}$ is finer than (the partition induced by) $\mathscr{F}$, Lemma 32 ensures that $\mathrm{MN}(\mathtt{A}_\sim, \mathscr{G}) \subseteq \mathrm{MN}(\mathtt{A}_\sim, \mathscr{F})$. But $\mathrm{MN}(\mathtt{A}_\sim, \mathscr{F}) = \sim$ on the canonical semi-automaton. Hence it suffices to prove that $\sim \subseteq \mathrm{MN}(\mathtt{A}_\sim, \mathscr{G})$. By Lemma 34, it suffices to prove that $\sim_V \subseteq \overset{\mathscr{G}}{\underset{0}{\sim}}_V$. But by definition of $\mathscr{G}$, $x \sim_V y$ is equivalent to $x \overset{\mathscr{G}}{\underset{0}{\sim}}_V y$. □

Whenever a vertex is reachable, the previous result yields a rather appealing (and presumably, familiar) description of an automatic relation in terms of one single subset of vertices.

Corollary 2.1. *Let* G *be a labelled directed graph with at least one reachable vertex* v*. A relation* $\sim$ *on* G *is automatic if and only if* $\sim = \mathrm{MN}(\mathtt{A}_\sim, \{[v]_{\sim_V}\})$.

Proof. By assumption, $S = \{v\}$ is a minimal complete final system. Let $\mathscr{F}$ be the partition induced by the single class $[v]_{\sim_V}$. Theorem 2 applies. □

This corollary shows that at least in the case of a reachable vertex, an automatic relation can be regarded as the graph-theoretic version of a Myhill-Nerode relation with respect to one distinguished subset of vertices (final states).

5.3. **A partial order on automatic relations.** There is a natural partial order on automatic relations. We define it as follows: $(\sim_v, \sim_E) \leq (\sim'_v, \sim'_e)$ whenever $\sim_v \subseteq \sim'_v$ and $\sim_E \subseteq \sim'_e$.

Proposition 13. *Suppose that* $\sim \leq \sim'$*, then there is a directed emulator* $\phi_{\sim,\sim'} : G/\sim \to G/\sim'$ *with* $\phi_{\sim,\sim'} \circ [-]_{\sim'} = [-]_\sim$.

Proof. If $\sim \leq \sim'$, for all vertex $[x]_\sim \subseteq [x]_{\sim'}$ so that $[x]_\sim \mapsto [x]_{\sim'}$ is a function. The same holds for edges. Adjacency follows from the definitions. The equality $\phi_{\sim,\sim'} \circ [-]_{\sim'} = [-]_\sim$ is immediate. □

Lemma 37. *Automatic relations have least upper bounds.*

Proof. Given $(\sim_V, \sim_E)$ and $(\sim'_V, \sim'_E)$, let $\sim''_V$ and $\sim''_E$ be respectively the transitive closure of $\sim_V \cup \sim'_V$ and $\sim_E \cup \sim'_E$. Let us check that $(\sim''_V, \sim''_E)$ is an automatic relation. First, since $\sim_V$ and $\sim'_V$ are equivalence relations, the transitive closure of their union is an equivalence relation. The same argument holds for edges.

We argue by induction on the length of the transitive closure that the relation $\sim''_E$ is compatible with $\sim''_V$. The base case is immediate. For the inductive step, suppose that an edge $e \sim_E e' \sim''_E e''$ (the other case $e \sim'_E e' \sim''_E e''$ is similar). Then $s_G(e) \sim_V s_G(e')$. By induction, $s_G(e') \sim''_V s_G(e'')$. By transitivity, $s_G(e) \sim''_V s_G(e'')$. Similarly $t_G(e) \sim''_V t_G(e')$.

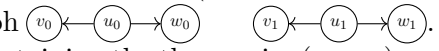
For bisimilarity, suppose that a vertex $x \sim''_V s_G(f)$ for some edge f. The base case is again immediate. For the inductive step, we suppose that we have $x \sim_V x' \sim''_V x'' = s_G(f)$ (the other case $x \sim'_V x' \sim''_V x''$ being symmetric). By induction, there is an edge $f' \sim''_E f$ with $s_G(f') = x'$. By bisimilarity, for $\sim$, there is an edge f'' such that $s_G(f'') = x$ and $f'' \sim_E f'$. By transitivity $f'' \sim''_E f$.

By a (tedious but not difficult) induction on the transitive closure, one verifies that $(\sim''_V, \sim''_E)$ is actually the least upper bound. □

The lower bound $(\sim''_V, \sim''_E)$ of two relations $(\sim_V, \sim_E)$ and $(\sim'_v, \sim'_E)$ is somewhat heavier to define. Set $f \sim''_E f'$ if $f \sim_E f'$ and $f \sim'_E f'$. The relation on vertices is defined by $x \sim''_V y$ if the following properties are satisfied:

(i) $x \sim_V y \wedge x \sim'_V y$;

(ii) if $x \xrightarrow{e} x'$ and $y \sim_V x$ then there is an edge $y \xrightarrow{f} y'$ such that $e \sim''_E f$.

Remark 18. Both conditions are necessary. If (ii) does not hold, the definition yields an equivalence relation that fails to be automatic (it fails to satisfy bisimilarity). For instance, consider the graph 

. Consider $\sim$ the vertex-induced automatic containing the three pairs (u_0, u_1), (v_0, v_1), (w_0, w_1) and an other vertex-induced automatic relation build on (u_0, u_1),(v_0, w_1),(w_0, v_1). We have $u_0 \sim u_1 \wedge u_0 \sim' u_1$ but this is not bisimilar with respect to $e \sim'' e' = e \sim e' \wedge e \sim' e'$ that is empty.

Lemma 38. *Automatic relations have greatest lower bounds.*

Proof. We prove similarly that the lower bound $(\sim''_V, \sim''_E)$ is an automatic relation. It follows from the definition that it is an equivalence relation. The compatibility of $\sim''_E$ with respect to $\sim''_V$ immediately follows from the compatibility of $\sim_E$ (resp. $\sim'_E$) with respect to $\sim_V$ (resp. $\sim'_V$).Bisimilarity is a direct consequence of (ii).

Let us verify it is the greatest upper bound. Take $\sim'''\leq\sim$ and $\sim'''\leq\sim'$. By definition, $e \sim''' e' \Rightarrow e \sim e' \wedge e \sim' e'$. Thus $e \sim'' e'$. For the same reason, $x \sim''' x' \Rightarrow x \sim x' \wedge x \sim' x'$ so that (i) holds. Suppose that (ii) does not hold. Then, there is an edge $f \in \mathrm{outE}(x)$ such that for any edge $g \in \mathrm{outE}(y)$ either $f \not\sim g$ or $f \not\sim' g$. In both case, $f \not\sim''' g$. But that means that $\sim'''$ has not the bisimilarity property for vertex x with respect to f. □

Corollary 2.2. *Automatic relations on a graph G form a lattice.*

Since the lattice is finite, there is a maximum element. The minimum is $(1_{V_G}, 1_{E_G})$.

Looking at the lattice as a category, Proposition 13 shows that $\sim \mapsto [-]_\sim : G \to G/\sim$ is a functor from the lattice to the coslice category, $\hom_{\mathrm{Em}}(G, -)$. Actually, we have:

Theorem 3. *In the coslice category,* $\hom_{\mathrm{Em}}(G, -)$*, the directed emulator* $[-]_\sim : G \to G/\sim$ *with* $\sim$ *the maximum element of the lattice is a terminal object.*

Proof. Suppose that $\phi : G \to H$ is a directed emulator. Then, there is an automatic relation $\sim'$ such that $\phi = \iota \circ [-]_{\sim'}$. Since $\sim$ is a maximum, by Proposition 13, $[-]_{\sim} = \phi_{\sim',\sim} \circ [-]_{\sim'} = \phi_{\sim',\sim} \circ \iota^{-1} \circ \phi$. Uniqueness is a direct consequence of Proposition 5. □

6. From graphs to regular languages

6.1. **Graphs as Semi-automata.**

Definition 15. Let G be a directed graph. The canonical semi-automaton $A_G = (G, E_G, 1_{E_G})$ associated to G with respect to the identity automatic relation ($\sim_V = 1_V$ and $\sim_E = 1_E$) is called the *tautological semi-automaton*.

The morphism $(f, g) : G \to H$ between two directed graphs is sent to $\mathrm{A}_{(f,g)} = (f, g, g) : \mathrm{A}_G \to \mathrm{A}_H$. Indeed, we have $g \circ 1_{E_G} = 1_{E_H} \circ g$, thus is a morphism. So, the tautological assignement forms a functor

$$\mathrm{A}_{(\_)} : \mathscr{D} \to \mathbf{Semi}.$$

Proposition 14. *The functor $G \to \mathrm{A}_G$ is full and faithful.*

Proof. Consider two morphisms $\phi = (f, g) : G \to H$ and $\psi = (f', g') : G \to H$ such that $\mathrm{A}_\phi = (f, g, g) = (f', g', g') = \mathrm{A}_\psi$. Then, $f = f'$ and $g = g'$.

Suppose now $(f, g, \alpha) : \mathrm{A}_G \to \mathrm{A}_H$. It satisfies $\alpha \circ \mathrm{id}_E = \mathrm{id}_{E'} \circ g$, that is, $\alpha = g$ so that $(f, g, \alpha) = \mathrm{A}_{(f,g)}$. □

Lemma 39. *The following properties hold:*

(*i*) *The identity $1_{\mathscr{D}} : G \to G$ is a natural transformation $1_{\mathscr{D}} \to \mathrm{G}_{\mathrm{A}_{(\_)}}$.*
(*ii*) *There is a natural transformation $\epsilon : \mathrm{A}_{\mathrm{G}_{(\_)}} \to 1_{\mathbf{Semi}}$.*

Proof. (*i*) follows from the definition. For (ii), we define $\epsilon_{\mathtt{A}} = (1_{V_{\mathtt{A}}}, 1_{E_{\mathtt{A}}}, \ell_{\mathtt{A}})$. It is a (relabelling) morphism $\mathrm{A}_{\mathrm{G}_{\mathtt{A}}} \to \mathtt{A}$. Indeed, due to (i), the two automata share the same graph: $\mathrm{G}_{\mathrm{A}_{\mathrm{G}_{\mathtt{A}}}} = \mathrm{G}_{\mathtt{A}}$. Thus $(1_{V_{\mathtt{A}}}, 1_{E_{\mathtt{A}}})$ is a graph morphism. Second, we have $\ell_{\mathtt{A}_{\mathrm{G}_{\mathtt{A}}}} = 1_{E_{\mathrm{G}_{\mathtt{A}}}} = 1_{E_{\mathtt{A}}} : E_{\mathtt{A}} \to E_{\mathtt{A}}$, again due to (*i*). Thus, $\ell_{\mathtt{A}} \circ \ell_{\mathtt{A}_{\mathrm{G}_{\mathtt{A}}}} = \ell_{\mathtt{A}} \circ 1_{E_{\mathtt{A}}}$ that leads to Equation 5.

Let us verify naturality. Given $(f, g, \alpha) : \mathtt{A} \to \mathtt{B}$, by definition, $\mathrm{A}_{\mathrm{G}_{(\_)}}(f, g, \alpha) = (f, g, g)$. We have to verify that the diagram commute:

$$\begin{array}{ccc} \mathtt{A}_{\mathrm{G}_{\mathtt{A}}} & \xrightarrow{(1_{V_{\mathtt{A}}}, 1_{E_A}, \ell_{\mathtt{A}})} & \mathtt{A} \\ {\scriptstyle (f,g,g)}\downarrow & & \downarrow{\scriptstyle (f,g,\alpha)} \\ \mathtt{A}_{\mathrm{G}_{\mathtt{B}}} & \xrightarrow{(1_{V_{\mathtt{B}}}, 1_{E_B}, \ell_{\mathtt{B}})} & \mathtt{B} \end{array}$$

For vertices and edges, this is trivial. The last equation is read $\alpha \circ \ell_{\mathtt{A}} = \ell_{\mathtt{B}} \circ g$ which is Equation 5. □

Corollary 3.1. *The tautological assignment $G \mapsto \mathtt{A}_G$ is left-adjoint to the forgetful functor $\mathrm{G}_{(\_)}$: $\mathbf{Semi} \to \mathscr{D}$.*

A semi-automaton $\mathtt{A}$ is *complete* (resp. *deterministic*) if given any state $q \in V_{\mathtt{A}}$, the map $\ell_q : \mathrm{OutE}(q) \to \Sigma_{\mathtt{A}},\ e \mapsto \ell(e)$ is surjective (resp. injective[3]). The following observation is a direct consequence of the definitions.

Lemma 40. *For any directed graph G, the tautological semi-automaton A_G is deterministic.*

Lemma 41. *If $\mathtt{A}$ is complete and deterministic, then, for all $q \in V_{\mathtt{A}}$, $\mathrm{outE}(q) \simeq \Sigma_{\mathtt{A}}$.*

Proof. By definition, ℓ_q is both surjective and injective. □

Corollary 3.2. *Suppose that $\phi : \mathtt{A} \to \mathtt{B}$ is strict morphism, $\mathtt{A}$ is both complete and deterministic and G_ϕ is a directed emulator, then, it is a cover.*

Proof. Let $\phi = (f, g, \alpha)$. For all $q \in V_{\mathtt{A}}$, we have $\ell_{\mathtt{B}} \circ g_q = \ell_q$ that is an isomorphism. Thus g_q is an isomorphism. □

Lemma 42. *Suppose that $\phi : \mathtt{A} \to \mathtt{B}$ is a semi-automaton epimorphism. Assume the following conditions:*

1. *The source semi-automaton $\mathtt{A}$ is complete;*
2. *The target semi-automaton $\mathtt{B}$ is deterministic.*

Then the morphism G_ϕ is a directed emulator.

Proof. Suppose that $\phi = (f, g, \alpha)$ with (f, g) an epimorphism. Then, f is surjective. Let us check the outgoing edge lifting property. Let $q_0 \xrightarrow{e} q_0' \in E_{\mathtt{B}}$ be a transition in $\mathtt{B}$ and $p_0 \in f^{-1}(q_0)$. We look for a preimage of e starting at p_0. Since (f, g) is an epimorphism, there exists $p_1 \in f^{-1}(q_0)$ and $p_1 \xrightarrow{\tilde{e}} q_1' \in E_{\mathtt{A}}$ such that $g(\tilde{e}) = e$, $f(p_1) = q_0$ and $f(q_1') = q_0'$. Furthermore, $\alpha(\ell_A(\tilde{e})) = \ell_{\mathtt{B}}(e)$. If $p_0 = p_1$, we are done. Otherwise, since $\mathtt{A}$ is complete, there is some edge $p_0 \xrightarrow{\tilde{e}'} p_1 \in E_{\mathtt{A}}$ starting from p_0 with the same label $\ell_{\mathtt{A}}(\tilde{e})$. Consider its image $g(\tilde{e}')$. We have $s_{\mathtt{B}}(g(\tilde{e}')) = q_0$ and $\ell_{\mathtt{B}}(g(\tilde{e}')) = \alpha(\ell_{\mathtt{A}}(\tilde{e}')) = \alpha(\ell_{\mathtt{A}}(\tilde{e})) = \ell_{\mathtt{B}}(e)$. Therefore the edge $g(\tilde{e}')$ has same source and same label as the edge $g(\tilde{e})$. Since $\mathtt{B}$ is deterministic, this implies that $g(\tilde{e}') = g(\tilde{e}) = e$ and we are done. □

6.2. **Finite state automata.** An *automaton* $\mathtt{A}$ is a semi-automaton endowed with a set of states $I \subseteq V_G$ that is called the set of *initial states* and a set $F \subseteq V_{\mathtt{A}}$ of *final states*. A state $q \in V_G$ is *accessible* if q is reachable from an initial state, that is, if there is a state $q_0 \in I$ and a walk starting at q_0 and ending at q. A state $q \in V_G$ is *co-accessible* if q is co-reachable to a final state, that is, if there is a state $q_1 \in F$ and walk starting at q and ending at q_1. *We suppose in this paragraph that any state $q \in V_G$ is accessible.*

[3]We rule out multiple transitions with the same source, target and label for a deterministic semi-automaton. (Not only this is consistent with the traditional definition, but this is required for the next theorem to hold.)

The standard definition considers automata with a unique initial state. We call them *finite state automata*. To denote the full class, we speak about multi-input state automata to stress the fact that multiple inputs are allowed. **Auto** denotes the full class.

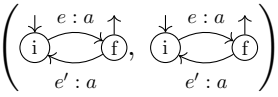
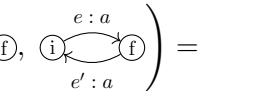
Definition 16. Let $\mathtt{A}$ and $\mathtt{B}$ be two automata. A *morphism between automata* $\mathtt{A} \to \mathtt{B}$ is a semi-automaton morphism (f, g, α) verifying

(1) $I_{\mathtt{A}} = f^{-1}(I_{\mathtt{B}})$ and
(2) $F_{\mathtt{A}} = f^{-1}(F_{\mathtt{B}})$.

In other words, $q \in I_{\mathtt{A}} \Leftrightarrow f(q) \in I_{\mathtt{B}}$ and $q \in F_{\mathtt{A}} \Leftrightarrow f(q) \in F_{\mathtt{B}}$.

The identity is an automaton morphism. Automata morphism compose componentwise. We denote **Auto** the category of automata and their morphisms.

Lemma 43. *The assignment that forgets the sets of initial and final states induces a forgetful faithful functor* $\mathbf{Auto} \to \mathbf{Semi}$.

Remark 19. The forgetful functor is not full. For instance, the induced map $H_1 = \mathrm{Hom}_{\mathbf{Auto}}\left(\right.$  $\left.\right) \to \mathrm{Hom}_{\mathbf{Semi}}\left(\right.$  $\left.\right) =$ H_2 is not onto. Indeed, H_1 contains only the identity while H_2 contains also the semi-automaton morphism induced by exchanging the two vertices.

A (*length* n) *computation* in $\mathtt{A}$ starting at state q, ending at state q', is a walk $\pi = e_0 e_1 \dots e_n$ in $G_{\mathtt{A}}$. The *label* of the computation π is $\ell(\pi) = \ell(e_0)\ell(e_1)\cdots\ell(e_n) \in \Sigma_{\mathtt{A}}^*$. A computation in an automaton is *successful* if it starts at some initial state and ends at some final state.

Definition 17. Given an automaton $\mathtt{A}$ the subset $L(\mathtt{A}) \subseteq \Sigma_{\mathtt{A}}^*$ of the words w for which there is a a successful computation π such that $\ell(\pi) = w$ is called the language represented by the automaton.

From Kleene's Theorem, we know that the languages defined by (finite state and multi-input state) automata are the regular languages.

Lemma 44. *The image of a successful computation* π *in* $\mathtt{A}$ *by a semi-automaton morphism* $(f, g, \alpha) : \mathtt{A} \to \mathtt{B}$ *is a succesful computation* $\pi' = g(\pi)$ *in* $\mathtt{B}$. *Furthermore, the label of the image of the computation is the image of the label of the computation:* $\ell_{\mathtt{B}}(\pi') = \ell_{\mathtt{B}}(g(\pi)) = \alpha(\ell_A(\pi))$.

Proof. The function g preserving adjacency, $g(\pi)$ is a walk within $G_{\mathtt{B}}$. By Lemma 44 and items (1) and (2) of Definition 16, initial and final states are preserved. The equality of images is a direct consequence of Equation 5. □

We record a consequence:

Lemma 45. *If there is an automaton morphism* $(f, g, \alpha) : \mathtt{A} \to \mathtt{B}$ *then* $\alpha(L(\mathtt{A})) \subseteq L(\mathtt{B})$. *In particular, if there is a strict morphism* $\mathtt{A} \to \mathtt{B}$ *then* $L(\mathtt{A}) \subseteq L(\mathtt{B})$.

Proposition 15. *Suppose that $\phi : \mathtt{A} \to \mathtt{B}$ is a morphism between two automata $\mathtt{A}$ and G_ϕ is a directed emulator, then $\alpha(L(\mathtt{A})) = L(\mathtt{B})$ where $\phi = (f, g, \alpha)$. In particular, if ϕ is a strict morphism, $L(\mathtt{A}) = L(\mathtt{B})$.*

Proof. Any successful walk in $\mathtt{B}$ is the image of a walk in $\mathtt{A}$. The preimage in $\mathtt{A}$ of an initial (resp. final) state in $\mathtt{B}$ being itself initial (resp. final), the walk in $\mathtt{A}$ is successful. Thus, $\alpha(L(\mathtt{A})) \supseteq L(\mathtt{B})$. With the preceding Lemma, we conclude $\alpha(L(\mathtt{A})) = L(\mathtt{B})$. □

Remark 20. In Sakarovitch's book, a strict automaton morphism that induces a directed emulator morphism is called a totally surjective morphism [19, Chap. II, Def. 3.2].

Theorem 4. *Suppose that $\phi : \mathtt{A} \to \mathtt{B}$ is an automaton morphism. Assume the following conditions:*

(1) *The source automaton $\mathtt{A}$ is complete;*
(2) *The target automaton $\mathtt{B}$ is deterministic.*

Then the two assertions are equivalent:

(i) the morphism G_ϕ is an epimorphism,
(ii) G_ϕ is a directed emulator morphism.

Proof. Lemma 42 shows that $(i) \Rightarrow (ii)$. By Lemma 16, $(ii) \Rightarrow (i)$. □

Corollary 4.1. *If $\phi : \mathtt{A} \to \mathtt{B}$ is a strict epimorphism between complete and deterministic automata, then $G_\phi : G_\mathtt{A} \to G_\mathtt{B}$ is a directed covering map.*

Proof. First, by the preceding theorem, it is a directed emulator morphism. Then, by Corollary 3.2, it is a covering map. □

We recall that, given a deterministic finite state automaton $\mathtt{A}$, there is a minimal complete deterministic finite state automaton, denoted $\mathtt{A}_{\min}$, such that $L(\mathtt{A}_{\min}) = L(\mathtt{A})$ together with the canonical projection $\mathtt{A} \to \mathtt{A}_{\min}$ that sends equivalent states to their equivalence class; furthermore, $\mathtt{A}_{\min}$ is unique up to automaton (strict) isomorphism.

Corollary 4.2. *Let $\mathtt{A}$ be a deterministic automaton and let $\pi : \mathtt{A} \to \mathtt{A}_{\min}$ be the canonical epimorphism to the minimal automaton. There exists an automaton $\tilde{\mathtt{A}}$ such that the following properties are satisfied:*

(1) *$\tilde{\mathtt{A}}$ is complete and deterministic;*
(2) *$\tilde{\mathtt{A}}$ contains $\mathtt{A}$ as a subautomaton;*
(3) *$L(\tilde{\mathtt{A}}) = L(\mathtt{A})$;*
(4) *The underlying graph morphism $\mathrm{G}_{\tilde{\mathtt{A}}} \to \mathrm{G}_{\mathtt{A}_{\min}}$ is a covering morphism.*

Proof. Each time there is a state q in $\mathtt{A}$ and a letter $a \in \Sigma$ without edges $q \xrightarrow{e} q' \in E_\mathtt{A}$ with $\ell_\mathtt{A}(e) = a$, take $p = \pi(q)$. Since $\mathtt{A}_{\min}$ is complete, there is an outgoing edge $p \xrightarrow{e'} p'$ with $\ell_{\mathtt{A}_{\min}}(e') = a$. Take some node q' in $Q_\mathtt{A}$ in the fibre over p' (it exists since the morphism is onto). We add to $\mathtt{A}$ a new edge e'' between p and p' with label a and we set $\pi(e'') = e'$. After modification,

the automaton $\mathtt{A}$ still verify the hypotheses of the Corollary. We continue the process until A is complete, call the result $\tilde{\mathtt{A}}$. Since $\tilde{\mathtt{A}}$ is complete and deterministic, Corollary 4.1 leads to (4). □

We now seek a reconstruction of an automaton morphism from a directed emulator morphism. The following lemma is the first step.

Lemma 46. *Consider a directed graph* $\mathrm{G}_{\mathtt{A}}$ *induced by an automaton* $\mathtt{A}$*. Suppose that there is a directed emulator morphism* $\varphi : G' \to \mathrm{G}_{\mathtt{A}}$*. Then there exists an automaton* $\mathtt{A}'$ *and a strict automaton morphism* $\pi : \mathtt{A}' \to \mathtt{A}$ *such that*

(1) $G_\pi = \varphi$*;*
(2) $L(\mathtt{A}') = L(\mathtt{A})$*.*

Here, we do not ask the automaton $\mathtt{A}$ to be deterministic nor complete.

Proof. We begin with the commutative diagram:

$$\begin{array}{ccccc} \mathtt{A}_{G'} & \xrightarrow{\mathtt{A}_\varphi} & \mathtt{A}_{G_{\mathtt{A}}} & \xrightarrow{\epsilon_{\mathtt{A}}} & \mathtt{A} \\ \downarrow{\scriptstyle G} & & \downarrow{\scriptstyle G} & & \downarrow{\scriptstyle G} \\ G' & \xrightarrow{\varphi} & G_{\mathtt{A}} & \xrightarrow{1} & G_{\mathtt{A}} \end{array}$$

By Proposition 1, the morphism $\epsilon_{\mathtt{A}} \circ A_\varphi = \pi \circ \lambda$ for a strict morphism π and λ a relabelling. Then, set $\mathtt{A}' = \lambda(\mathtt{A}_{G'})$. We have $G_{\mathtt{A}'} = G_{A_{G'}} = G'$ and $G_\pi = \varphi$. However, as defined, $\mathtt{A}'$ is a semi-automaton. We define $I_{\mathtt{A}'} = \pi^{-1}(I_{\mathtt{B}})$ and $F_{\mathtt{A}'} = \pi^{-1}(F_{\mathtt{A}})$ so that π is actually an automaton morphism. We conclude with Lemma 15. □

Lemma 47. *Consider a directed graph* $G = \mathrm{G}_{\mathtt{A}}$ *induced by some deterministic automaton* $\mathtt{A}$*. Suppose that there is a directed emulator morphism* $G' \to G$*. Then there exists a deterministic automaton* $\mathtt{A}''$ *and a strict automaton epimorphism* $\pi : \mathtt{A}'' \to \mathtt{A}$ *such that*

(1) $G_\pi : \mathrm{G}_{\mathtt{A}''} \to \mathrm{G}_{\mathtt{A}}$ *is a directed covering morphism;*
(2) $\mathrm{G}_{\mathtt{A}''}$ *is a subgraph of* G'*;*
(3) $g(\mathtt{A}'') \leq g(G')$*.*

Proof. By the previous lemma, there is an automaton $\mathtt{B}$ together with a strict epimorphism $\mathtt{B} \to \mathtt{A}$ inducing the directed emulator morphism $G' \to G$. By Lemma 18, one can extract from the directed emulator G' a directed cover $G'' \subseteq G'$ over G. This determines a subautomaton $\mathtt{A}''$ of $\mathtt{B}$ with underlying directed graph $\mathrm{G}_{\mathtt{A}''} = G''$ that is a subgraph of G'. Therefore $g(\mathtt{A}'') = g(G'') \leq g(G')$. It remains to verify that $\mathtt{A}''$ is deterministic. Since the strict epimorphism $\mathtt{A}'' \to \mathtt{A}$ induces a covering morphism $G'' \to G$, for any state $q' \in Q_{\mathtt{A}'}$ and its image $q \in Q_{\mathtt{A}}$, the induced map $\mathrm{OutE}(q') \to \mathrm{OutE}(q)$ is a bijection between sets of labelled outgoing transitions. Since $\mathtt{A}$ is deterministic, $\ell_{\mathtt{A}}|_{\mathrm{OutE}(q)}$ is injective and so must be $\ell_{\mathtt{A}''}|_{\mathrm{OutE}(q')}$. Therefore $\mathtt{A}''$ is deterministic. □

We can get rid of the multiple edges in the previous lemma, i.e., a simple graph theoretical version of the previous lemma holds.

Lemma 48. *Consider a directed graph $G = \mathrm{G}_{\mathtt{A}}$ induced by some deterministic automaton $\mathtt{A}$. Suppose that there is a directed emulator morphism $\phi : H \to R(G)$. Then there exists a deterministic automaton $\mathtt{A}'$ and a strict automaton epimorphism $\mathtt{A}' \to \mathtt{A}$ such that*

(1) The induced directed emulator morphism $\mathrm{G}'_{\mathtt{A}} \to \mathrm{G}_{\mathtt{A}}$ is a directed covering morphism;
(2) $g(\mathtt{A}') \leq g(H)$.

Proof. The two morphisms $G \overset{\rho_G}{\to} R(G) \overset{\phi}{\leftarrow} H$ are directed emulators. Via Lemma 20, we have the diagram with $\pi_{1|}$ a directed emulator:

$$\begin{array}{ccc} G \times_{R(G)} H & \xrightarrow{\pi_{2|}} & H \\ {\scriptstyle \pi_{1|}}\downarrow & & \downarrow{\scriptstyle \phi} \\ G & \xrightarrow{\rho_G} & R(G) \end{array}$$

Observe that $R(G \times_{R(G)} H) \simeq R(G) \times_{R(R(G))} R(H) \simeq R(G) \times_{R(G)} R(H) \simeq R(H)$, the first equality being due to Lemma 21. Thus $g(G \times_{R(G)} H) = g(H)$ via Lemma 14. We conclude by Lemma 47. □

Finally we get rid of loops in the following sense.

Lemma 49. *Consider a directed graph $G = \mathrm{G}_{\mathtt{A}}$ induced by some deterministic automaton $\mathtt{A}$. Suppose that there is a simple directed cover morphism $H \to \mathrm{Exc}(R(G))$. Then there exists a deterministic automaton $\mathtt{A}'$ and a strict automaton epimorphism $\mathtt{A}' \to \mathtt{A}$ such that*

(1) The induced directed emulator morphism $\mathrm{G}'_{\mathtt{A}} \to \mathrm{G}_{\mathtt{A}}$ is a directed covering morphism;
(2) $g(\mathtt{A}') \leq g(H)$.

Proof. Apply Lemma 22 to obtain a directed cover H' over $R(G)$ such that $g(H') = g(H)$. Applying Lemma 48 yields the result. □

7. The genus of a regular language

In this section, we state and prove the two main results of the paper (Theorem 5 and Corollary 5.5). We also discuss the relationship with the undirected setting.

7.1. **Main results.** Let us recall the definition that we introduced in [2].

Definition 18. The *genus* $g(L)$ of a regular language L over alphabet A is the minimum of all genera of finite state deterministic automata computing L.

Before we proceed to the main result, we mention three important observations in the definition.

Remark 21 (Determinism). The word "deterministic" is essential in the definition of the genus, for it is known that any regular language has a genus 0 (planar) nondeterministic automaton that computes it, a nice result due to R. .V. Book and A. K. Chandra [5].

Remark 22 (Single-input versus multi-input). We would like to emphasize that the definition of genus here is confined to regular languages computed by automata with one single initial state (single-input automata), in accordance to our convention in this paper (cf. 6.2). Taking into account multi-input (deterministic) automata leads to a richer and more complex notion of genus, studied in [4].

Remark 23 (Completeness). It can be proved ([2]) that there always exists a *complete* deterministic automaton $\mathtt{A}$ such that $g(L) = g(\mathtt{A})$. Basically, given a deterministic (but not complete) automaton, for any missing transition, one may add a transition to a fresh trash state whose outgoing transitions are loops. That leaves the genus unchanged.

The directed graph $G(L)$ *associated to a regular language* L is the directed graph underlying the minimal automaton $A_{\min}(L)$ canonically associated to L: $G(L) = \mathrm{G}_{\mathtt{A}_{\min}(L)}$. For any regular language L, $g(L) \leq g(G(L))$.

Theorem 5. *Let L be a regular language. Let $n \in \mathbb{N}$. The following assertions are equivalent:*

1. *$g(L) \leq n$;*
2. *The directed graph $G(L)$ has a directed cover G of genus $g(G) \leq n$;*
3. *The directed graph $G(L)$ has a directed emulator G with $g(G) \leq n$;*
4. *The directed simple graph $R(G(L))$ has a directed simple cover G such that $g(G) \leq n$;*
5. *The directed simple graph $\mathrm{Exc}(RG(L))$ has a directed cover G such that $g(G) \leq n$.*

Proof. Suppose that L has genus $g(L) \leq n$. There is some finite deterministic complete automaton $\mathtt{A}$ computing L such that $g(\mathtt{A}) \leq n$. This automaton comes naturally with an automaton epimorphism $\mathtt{A} \to \mathtt{A}_{\min}(L)$. Applying the functor $\mathrm{G}_{(-)}$ yields (Theorem 4) a directed covering epimorphism $\mathrm{G}_{\mathtt{A}} \to G(L)$. Hence (1) $\Longrightarrow$ (2). The implication (2) $\Longrightarrow$ (3) is obvious. The functor R preserves directed emulation (Lemma 6) and genus (Lemma 14), thus (3) $\Longrightarrow$ (4). Excision also preserves the genus, thus (4) $\Longrightarrow$ (5). Suppose (5). Applying Lemma 49 provides a deterministic automaton $\mathtt{A}$ such that $L(\mathtt{A}) = L$ and $g(\mathtt{A}) \leq g(G) \leq n$. Therefore $g(L) \leq g(\mathtt{A}) \leq n$. This proves (1). □

Corollary 5.1. *If L, L' are two regular languages such that $\mathrm{Exc}(RG(L)) = \mathrm{Exc}(RG(L'))$, then $g(L) = g(L')$.*

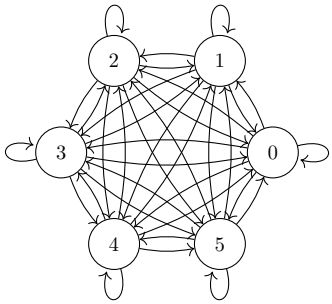
Corollary 5.2. *Let L, L' be regular languages such that $\mathrm{Exc}(RG(L'))$ is a subgraph of $\mathrm{Exc}(RG(L))$. Then $g(L') \leq g(L)$.*

Proof. A directed emulator H of $\mathrm{Exc}(RG(L))$ of minimal genus contains a directed emulator H' of $\mathrm{Exc}(RG(L'))$ as a subgraph. Hence $g(L) \geq g(H') \geq g(L')$. □

Corollary 5.2 can be used to bound genera of languages as well.

Corollary 5.3. *Any regular language L of size $|L|_{\mathrm{set}} \leq 6$ is planar.*

Proof. Consider the regular language Z_6 that consists of words $w = a_1 a_2 \cdots a_n$ on the alphabet $\mathbb{Z}/6\mathbb{Z}$ such that the sum $\sum_i a_i$ of its letters is 0 mod 6. The underlying graph of its minimal deterministic automaton is the complete simple directed graph of size 6:



This graph has a planar directed emulator (for this and other facts about Z_6, see [3, §2.1]). Let L be any regular language whose minimal automaton $\mathtt{A}$ has size at most 6. The simple graph $R(G_{\mathtt{A}}) = R(G(L))$ is then a subgraph of $R(G(Z_6))$, thus (Corollary 5.2) has a planar directed emulator. Hence by Theorem 5, L is planar.

□

Corollary 5.4. *Let L_1 and L_2 be two regular languages on disjoint alphabets. Then $g(L_1 \cup L_2) \geq \max(g(L_1), g(L_2))$.*

Proof. The minimal automaton $\mathtt{A}_{\min}(L_1 \cup L_2)$ for $L_1 \cup L_2$ contains both the minimal automaton $\mathtt{A}_{\min}(L_1)$ and the minimal automaton $\mathtt{A}_{\min}(L_2)$ as subgraphs. □

The *Language Genus Problem* is the following: given a regular language L and $n \in \mathbb{N}$, the answer is YES if $g(L) \leq n$, otherwise NO.

The *Directed Emulation Genus Problem* is: given a directed graph G and $n \in \mathbb{N}$, YES if there is a directed emulator G' of G such that $g(G') \leq n$, otherwise NO.

Theorem 5 allows to reduce the problem of the determining the genus of a language to a graph-theoretic problem in terms of directed emulators.

Corollary 5.5. *The* Language Genus Problem *has a solution if and only if the* Directed Emulation Genus Problem *restricted to co-reachable directed graphs has a solution.*

Proof. Theorem 5 shows a solution for the Directed Emulation Genus Problem implies a solution for the Language Genus Problem. We need to show the converse. Suppose that $G = (V, E)$ is a strongly connected directed graph. Consider its associated tautological semi-automaton $\mathtt{A}_G$ (see Definition 15). By assumption, there is a vertex $v_0 \in V$ such that all other vertices are reachable from it. We define v_0 to be the initial state and all vertices $v \in V$ to be final states. Let $\mathtt{A}$ be the resulting automaton. By Lemma 40, $\mathtt{A}$ is deterministic. Since v_0 is co-reachable, that every state is accessible (from v_0) in $\mathtt{A}$. Since every state is final, every state is trivially co-accessible (to itself). Furthermore, since every outgoing edge has a unique label, there cannot be two distinct and equivalent states, so $\mathtt{A}$ is minimal. We have $G = G(L(\mathtt{A}))$. By Theorem 5, $L(\mathtt{A})$ has genus g if and only if G has a directed emulator of genus g. □

The *planarity problem* is the genus problem (applied to directed graphs or regular languages) restricted to $n = 0$.

Corollary 5.6. *The planarity problems for co-reachable directed graphs and for regular languages respectively are equivalent.*

Remark 24. Corollary 5.6 was first proved by D. Kuperberg.

7.2. **Directed vs undirected.** We state in this paragraph other applications of Theorem 5 in relation to our study of the differences between directed and undirected emulators.

Our first observation is that Theorem 5 yields a bound for genus by means of *undirected* emulators.

Corollary 5.7. *Let L be a regular language and let $U_L = U(\mathrm{Exc}(RG(L)))$ be the undirected graph obtained from $G(L)$ by simplification and excision. Let $H \to U_L$ be any undirected emulator of U_L. Then $g(L) \leq g(H)$. In particular, if U_L has a planar emulator then L is planar.*

Proof. Suppose there is an emulator $\pi : H \to U_L$. Then, by Lemma 27, there is a directed emulator $\vec{\pi} : \vec{H} \to \mathrm{Exc(R(G(L))}$ (note that there is a direction such that $\vec{U}_L = \mathrm{Exc(R(G(L)))}$. Therefore by Theorem 5 $g(L) \leq g(\vec{H}) = g(H)$. □

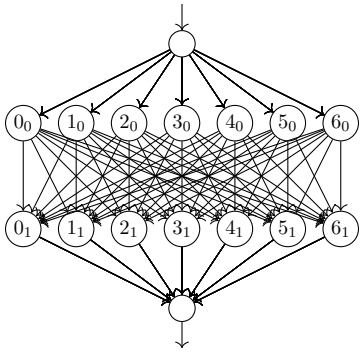
For the next corollary, for a directed graph G, denote $\mathrm{Em}_{\mathscr{D}}(G)$ the set of all finite directed emulators of G. For an undirected graph H, denote $\mathrm{Em}(H)$ the set of all finite emulators of H.

Corollary 5.8. *Let G be a simple loopless directed graph. Then*

$$\min_{\tilde{G} \in \mathrm{Em}_{\mathscr{D}}(G)} g(\tilde{G}) \leq \min_{H \in \mathrm{Em}(U(G))} g(H).$$

Remark 25. There are languages (resp. directed graphs) such that the inequality in Corollary 5.7 (resp. Corollary 5.8) is strict. In other words, the upper bound for the genus of directed emulators given by undirected emulators may be not reached. In particular, the converse of the last statement

in Corollary 5.7 does not hold. Consider the language L defined on the alphabet $\mathbb{Z}/7\mathbb{Z}$ as the set of three-letter words abc ($a, b, c \in \mathbb{Z}/7\mathbb{Z}$) such that $a+b+c = 0 \bmod 7$. The underlying graph $G(L)$ of the minimal deterministic automaton $\mathtt{A}_L$ for L is depicted below.



Corollary 5.9. *The directed graph $G(L)$ has a planar emulator and the undirected graph $U(G(L))$ has no planar emulator.*

Proof. As the language L is finite, there is a finite deterministic (planar) tree that recognizes it, so L is planar: $g(L) = 0$. By Theorem 5, the first assertion follows. For the second assertion, we need the following observations.

Proposition 16. *Let L be an m-letter regular language such that $m \geq 3$. Suppose that the outdegree at each vertex of $G(L)$ is m and that $U(G(L))$ has no cycles of length ≤ 2. Then $g(L) \geq 1$.*

Remark 26. This is a slight improvement of [3, Corollary 3.1].

Proof. This is a consequence of the genus formula [2, Theorem 5] recalled below. Consider a minimal embedding of a directed emulator H of $G(L)$ in a closed surface. Since there is no cycle of length 1 and 2, each face of the embedding has length at least 3 (the proof is the same as that of the claim in [2, Proof of Prop. 1]). Let f_i be the number of i-faces of the embedding. According to the genus formula,

$$\begin{aligned} g(L) = g(H) &= 1 - \frac{m+1}{m} f_1 - \frac{1}{2m} f_2 + \frac{m-3}{4m} f_3 + \frac{2m-4}{4m} f_4 + \cdots \\ &= 1 + \underbrace{\frac{m-3}{4m}}_{\geq 0} f_3 + \underbrace{\frac{2m-4}{4m}}_{\geq 0} f_4 + \cdots \\ &\geq 1. \end{aligned}$$

□

Lemma 50. *Let $Z_7^{1,2,3}$ be the language on the alphabet $\{1, 2, 3\} \subset \mathbb{Z}/7\mathbb{Z}$ that consists of words $a_1 a_2 \cdots a_n$ such that $\sum_i a_i = 0 \bmod 7$. Then $g(Z_7^{1,2,3}) \geq 1$.*

Proof. The minimal complete deterministic automaton $\mathtt{A}$ for $Z_7^{1,2,3}$ is seen to be defined by $Q = \mathbb{Z}/7\mathbb{Z}$, 0 being the initial and final state, and transitions

$i \xrightarrow{j} i+j$ for $i \in \mathbb{Z}/7\mathbb{Z}$ and $j \in \{1,2,3\} \subset \mathbb{Z}/7\mathbb{Z}$. The underlying graph $G(L)$ is easily seen to have outdegree m at each vertex and to have no cycles of length < 3. Therefore Lemma 16 applies. □

For the next lemma, we need to define the *directed cycle contraction* (see [14]). Let $G = (V, E)$ be a directed graph and $c = e_1 e_2 \cdots e_n$ a directed cycle in G (the map $\{1, \ldots, n\} \to E, i \mapsto e_i$ is injective and $s(e_1) = t(e_n)$). Let E_c denote the subset of edges e in E incident to c, that is, such that $s(e) = s(e_i)$ or $t(e) = t(e_i)$ for some $1 \leq i \leq n$. Let w be a new vertex. Define a bijective map by sending each edge $e \in E_c - \{e_1, \ldots, e_n\}$ to a new edge e' where the vertex in c incident to e is replaced by w. Denote the image E_w. The contraction of G along c is the new directed graph $G_c = (V', E')$ where $V' = (V - \{s(e_1), \ldots, s(e_n)\}) \cup \{w\}$ and $E' = (E - E_c) \cup E_w$.

Proposition 17. *If G has a directed emulator of genus $\leq g$, then G_c has a directed emulator of genus $\leq g$.*

Proof. [15, Theorem 12]. □

We now return to the proof of Corollary 5.9. By Lemma 25, $H \to U(G(L))$ is an (undirected) emulator morphism if and only if the induced map $\overleftrightarrow{H} \to \overleftrightarrow{U(G(L))}$ is a directed emulator morphism. In the directed graph $\overleftrightarrow{U(G(L))}$, for each $i \in \mathbb{Z}/7\mathbb{Z}$, contract the directed cycle $c_i = $ 

. (Note that all these cycles are disjoint in $\overleftrightarrow{U(G(L))}$.) Let $G = G_{c_1,\ldots,c_7}$ be the resulting directed graph and let $\tilde{G}$ be any directed emulator of G. Note that G contains $\mathtt{A}_{Z_7^{1,2,3}}$. Therefore

$$\begin{aligned} g(\overleftrightarrow{U(G(L))}) &\geq g(\tilde{G}) && \text{(Lemma 17)} \\ &\geq g(Z_7^{1,2,3}) && \text{(Corollary 5.2)} \\ &\geq 1 && \text{(Lemma 50).} \end{aligned}$$

□

We now consider the undirected version of the Emulation Genus Problem. The *Emulation Genus Problem* is: given a connected undirected graph G and $n \in \mathbb{N}$, answer YES if there is an emulator G' of G such that $g(G') \leq n$, otherwise NO.

Corollary 5.10. *The* Emulation Genus Problem *has a solution if the* Directed Emulation Genus Problem *has a solution.*

Proof. Let G be a graph. Assume that the Directed Emulation Genus Problem has a solution. It suffices, then, to prove that there is an emulator G' of G such that $g(G') \leq n$ if and only if there is a directed emulator H of $\overleftrightarrow{G}$ such that $g(H) \leq n$.

Suppose that $\phi : G' \to G$ is an emulator such that $g(G') \leq n$. By Lemma 25, $\overleftrightarrow{\phi} : \overleftrightarrow{G'} \to \overleftrightarrow{G}$ is a directed emulator. Furthermore, it is clear that $g(\overleftrightarrow{G'}) = g(G')$.

Conversely, suppose that $\phi : G' \to \overleftrightarrow{G}$ is a directed emulator with $g(G') \leq n$. According to Lemma 26, there is an emulator $G'' \to G$ with $g(G'') = g(G') \leq n$. □

Remark 27. Corollary 5.10 may be a little surprising at first. Indeed, the existence of a directed cover of genus g is equivalent to the existence of a directed emulator of genus g (Proposition 2), a fact that turns out to be wrong in the undirected setting (after the works of P. Hliněný [12], Y. Rieck and Y. Yamashita [17], respectively) as discussed above (§4.3). However, Corollary 5.10 (more precisely the key lemma used in the proof above) does not contradict this. Typically, the solution provided by Corollary 5.10 is an emulator and not a cover, even if one started from a directed cover $G' \to \overleftrightarrow{G}$. Indeed, the key lemma (Lemma 26) cannot be used in general to build an undirected cover from a directed cover. See our remarks there (Remark 11).

Remark 28. On the one hand, the existence of an emulator of an undirected graph of genus $\leq n$ is preserved under edge contraction, see [10, §2]. It follows that the Emulation Genus Problem is decidable (albeit not constructively). On the other hand, the existence of a directed emulator of genus $\leq n$ of a directed graph is not even preserved under edge contraction, see [15, §2].

8. Conclusion

We have shown that the Language Genus Problem is decidable if and only if the Directed Emulation Genus Problem is decidable. However, we do not have yet a complete proof of decidability. On another direction, we proved that the *Emulation Genus Problem* has a solution if the *Directed Emulation Genus Problem* has a solution. The former problem is known to have a theoretical solution (Remark 28). A general approach, suggested by a natural generalization of Corollary 5.2, consists in properly defining directed minors and proving a "directed graph minor" theorem analogous to the celebrated graph minor theorem of Robertson and Seymour [18, §10.5]. This is the approach aimed at in [15]. Suitable operations are defined at the level of the underlying graph $G(L)$ of the minimal deterministic automaton $\mathtt{A}_{\min}(L)$ that are non-increasing on the genus of the language L (hence, by Theorem 5, of any directed cover of $G(L)$). One should note, however, that the operations (and hence the "directed graph minors") are less elementary than the operations for undirected graphs. Even if this approach would be successful, one would furthermore need to find the minors of nonplanar emulable directed graphs. We would face the kind of issues that are discussed by M. Chimani, M. Derka, P. Hliněný and M. Klusáček in their article [7] (on undirected graphs). In any case, the complete relationship between

undirected and directed emulators, beyond Corollary 5.10, is intriguing and seems quite a challenging question.

Finally, we note that the fields of linear logic and automata theory are closely related. See for instance the early paper by R. Statman [20] who introduced the genus of a proof by means of a directed graph $G(\Pi)$ associated to a proof Π (see also [6]). We expect that the questions raised in this paper have applications or analogs in linear logic [11]. Furthermore, developments in linear logic include the assignment of more general objects than graphs to proof nets, empowering the full arsenal of categorical topological invariants to (suitable categories of) linear logic (see e.g., [16]).

References

[1] Jiří Adámek and Věra Trnkovà. *Automata and Algebra in Categories!!* Springer, 1990.

[2] Guillaume Bonfante and Florian L. Deloup. The genus of regular languages. *Math. Struct. Comput. Sci.*, 28(1):14–44, 2018.

[3] Guillaume Bonfante and Florian L. Deloup. Decidability of regular language genus computation. *Math. Struct. Comput. Sci.*, 29(9):1428–1443, 2019.

[4] Guillaume Bonfante and Florian L. Deloup. Multi-input genus and regular languages supported by a directed graph. *en présparation*, 2022.

[5] Ronald V. Book and Ashok K. Chandra. Inherently nonplanar automata. *Acta Informatica*, 6:8994, 1976.

[6] Alessandra Carbone. Logical structures and genus of proofs. *Annals of Pure and Applied Logic*, 161 (2):139–149, 2009.

[7] Markus Chimani, Martin Derka, Petr Hliněný, and Matěj Klusáček. How not to characterize planar-emulable graphs. *Advances in Applied Mathematics*, 50(1):46–68, 2013.

[8] Thomas Colcombet and Daniela Petrişan. Automata minimization: a functorial approach. *Log. Methods Comput. Sci.*, 16(1):32:1–32:28, 2020.

[9] Samuel Eilenberg. Automata, Languages and Machines. *Academic Press*, Vol A(New York), 1974.

[10] Michael R. Fellows and Michael A. Langston. Nonconstructive tools for proving polynomial-time decidability. *J. ACM*, 35(3):727–739, June 1988.

[11] Jean-Yves Girard. Linear logic. *J. Theoretical Computer Science*, 50 (1):1–102, 1987.

[12] Petr Hliněný. K4,4 -e has no finite planar cover. *Journal of Graph Theory*, 27:51–60, 1998.

[13] André Joyal, Mogens Nielsen, and Glynn Winskel. Bisimulation from open maps. *Inf. Comput.*, 127(2):164–185, 1996.

[14] Shiva Kintali and Qiuyi Zhang. Forbidden directed minors and kelly-width. *J. Theoretical Computer Science*, 662:40–47, 2017.

[15] Denis Kuperberg. Directed minors for minimal automata, 2017.

[16] Paul-André Melliès. Categorical semantics of linear logic, 2007.

[17] Yo'av Rieck and Yasushi Yamashita. Finite planar emulators for $K_{4,5} - 4K_2$ and $K_{1,2,2,2}$ and Fellows' conjecture. *European Journal of Combinatorics*, 31, 2010.

[18] Neil Roberston and Paul Seymour. Graph minors. xx. wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325–357, 2004.

[19] Jacques Sakarovitch. *Elements of Automata Theory*. Cambridge University Press, 2009.

[20] Richard Statman. *Structural Complexity of Proofs*. PhD thesis, Stanford University, Stanford, CA, 1974.