Learning Substitutable Binary Plane Graph Grammars

Rémi Eyraud, Jean-Christophe Janodet, Tim Oates

JMLR: Workshop and Conference Proceedings 21:1–15, 2012 The 11th International Conference on GI

Learning Substitutable Binary Plane Graph Grammars *

Rémi Eyraud REMI.EYRAUD@LIF.UNIV-MRS.FR
QARMA team, Laboratoire d'Informatique Fondamentale de Marseille, France

Jean-Christophe Janodet JANODET@IBISC.UNIV-EVRY.FR
IBISC lab, University of Evry Universud Paris, France

Tim Oates OATES@CS.UMBC.EDU
CoRaL lab, CSEE department, University of Maryland, Baltimore County, USA

Editors: Jeffrey Heinz, Colin de la Higuera, and Tim Oates

Abstract

While some heuristics exist for the learning of graph grammars, few has been done on the theoretical side. Due to complexity issues, the class of graphs has to be restricted: this paper deals with the subclass of plane graphs, which correspond to drawings of planar graphs. This allows us to introduce a new kind of graph grammars, using a face-replacement mechanism. To learn them, we extend recent successful techniques developed for string grammars, and based on a property on target languages: the substitutability property. We show how this property can be extended to plane graph languages and finally state the first identification in the limit result for a class of graph grammars, as far as we know.

1. Introduction

Although Graph Grammars have been defined and studied for 3 decades from a language-theoretical standpoint (see [Rozenberg and Ehrig, 1997] for an overview), the learning of graph grammars is a difficult problem that was poorly investigated in the literature yet. As shown by Tim Oates in the tutorial he gave during ICGI'10, most contributions concern heuristics tailored for graphs involved in restricted application domains. This is the case of both most famous algorithms, *Subdue* [Cook and Holder, 2000] and *FFSM* [Huan et al., 2003], and their extensions (see [Matsuda et al., 2002; Jonyer et al., 2003; Kukluk et al., 2008] for the main ones). On the theoretical side, it appears that learnability results are even more rare and often provide us with preliminary results, rather than effective learning algorithms. *E.g.*, Jeltsch and Kreowski [1991] give an algorithm that generates the set of grammars consistent with a given set of graphs. Brijder and Blockeel [2011] investigate the inference of a grammar consisting of one production rule only, given a graph and a distinguished pattern with many occurrences. Few necessary conditions for the learning of graph grammars have also been studied under unrestricted Gold's paradigm [Costa-Florencio, 2009]. The situation is a bit different in other branches of Machine Learning: many techniques have appeared to tackle problems related to, *e.g.*, Social or Biological Networks. However, they generally hide the complexity of the graph structures into abstract numerical structures (see graph kernels [Vishwanathan et al., 2010] for instance).

* This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

There are many reasons for this, going from the profusion of incomparable graph grammar formalisms to the hardness of the model itself. Concerning the latter, many basic problems, such as the search for a subgraph in a graph, and thus the possibility to parse a graph with a grammar, are generally $\mathcal{NP}$-complete. Nevertheless, the main reason for the absence of positive learning result for graph grammars is probably that no kind of graph grammars was designed with the aim of learning. Indeed, one way to tackle the difficulties induced by generative devices (grammars) consists in restricting the languages. That is, the successful approach in Grammatical Inference is often to determine features that are learnable, which usually correspond to observable properties in any set of examples, and then to focus only on the languages that share these characteristics.

Hence, in the case of graph languages, we should first determine which kind of graphs are likely to be learnable, and then choose the kind of grammars to use. For reasons that will be developed in this paper, a promising candidate is the class of *plane graphs*, that is, planar graphs embedded in the plane (see Fig. 1). Note that a *planar* graph has a set X of vertices and a set E of edges as usual, but also a set F of faces, as soon as this graph is embedded in the plane. A planar graph may have several incomparable drawings, so we define a plane graph by fixing the embedding. More formally, a plane graph stands for an isotopy class of planar embeddings for a given planar graph [Fusy, 2007]. A plane graph is thus a planar graph that is embedded in the plane without edge-crossing and up to continuous deformations. Given a planar embedding of a planar graph, Fáry [1948] proved that it is always possible to move the vertices, within the same isotopy class, so that the edges are drawn with straight-line segments. We shall use such straight-line drawings in the following.

Now, as no graph grammar formalism captures the specificities of such plane graphs, we choose not to use existing general graph-grammar formalisms, but propose a new one, the *binary plane graph grammars*. These grammars can be seen as face-replacement grammars, thus constitute an interesting alternative to standard node-replacement or hyperedge-replacement grammars. Indeed, their rules replace one face by a new plane graph, that is sewn in mother graph using a syntactic gluing law. Of course, it is beyond the scope of this paper to provide a complete analysis of this formalism and thus some important features will not be studied below. For instance, the production rules that we consider have binary right hand-sides and thus are similar to those of the Chomsky normal forms for context-free string grammars. Nevertheless, we do not study the generative capacities of our rules, thus do not claim that any possible definition of a plane graph grammar would be equivalent to a grammar made of our rules only.

On the other hand, we investigate below the learnability of this new type of graph grammars, and prove that one can get formal learnability results in this setting. We believe that this is quite an interesting improvement *w.r.t.* the state of the art. Concerning the difficulties, note that when one is trying to learn from graphs, negative data are usually not available. We know since the work by Gold [1967] that it is not possible to identify in the limit any superfinite class of languages from positive data, and thus we need to restrict ourselves to a subclass of plane graph languages. The recent successes of distributional learning for string grammars [Clark, 2010] and tree grammars [Kasprzik and Yoshinaka, 2011] motivate us to define an analogue of substitutable context-free languages [Clark and Eyraud, 2007] for plane graph languages.

Preliminaries about plane graphs are given in Section 2. The substitutability property is adapted and described in Section 3. In Section 4, we define the binary plane graph grammars as well as the generation process. Section 5 is devoted to our learning procedure and the proof of its convergence. We conclude the paper with a short discussion in Section 6.

2. On Plane Graphs

We have introduced the plane graphs using the notion of embeddings, *i.e.*, functions that map vertices to points, and edges to curves. However, this mathematical approach is quite unsuitable for designing algorithms. As the set of faces is the corner stone to describe plane graphs, we introduce *plane graph systems* below, which allow us to describe any *connected* plane graph through its faces.

Let $X \subset \mathbb{N}$ be the alphabet of vertices. Let X^* be the set of all *strings* over X, and ϵ the empty string. Given a string $x = a_1 \dots a_n$, we denote $|x| = n$ its length and x^R the reverse string of x, that is to say $x^R = a_n \dots a_1$.

A *circular string* is intuitively a string in which the last symbol is followed by the first: there is no first symbol but just a mapping associating to each symbol the next one. We denote a circular string by $[u]$, with the convention that if u and v are two strings, then $[uv] = [vu]$. The set of all circular strings over X is denoted by $X^\top$. We set $[x]^R = [x^R]$.

Now consider the plane graph of Figure 1. The outer face is f_1 and bounded (inner) faces are f_2 and f_3. Each face has only one boundary since the graph is connected. Such a boundary can be described by a circular string of vertices in which two consecutive vertices and the last and first vertices are linked by an edge. Conventionally, we follow a boundary by leaving it to the right. In other words, the bounded face is on the left of the walk. Hence, the boundary of face f_3 is $[53634]$, or equivalently $[63453]$, by circular permutation.

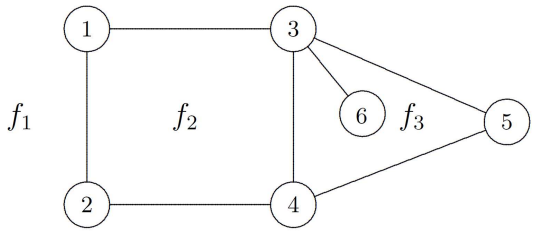

Figure 1: A plane graph with 3 faces.

We now introduce the following description system for connected plane graphs:

Definition 1 (Plane Graph System) *A* plane graph system *(PGS for short) is a tuple* $S = \langle X, E, F, o, \mathcal{D} \rangle$ *such that (1)* $\langle X, E \rangle$ *is a connected planar simple graph [Gibbons, 1985], (2)* F *is a finite nonempty set of symbols called the* faces*, (3)* $o \in F$ *is a distinguished symbol called the* outer face *and (4)* $\mathcal{D} : F \to X^\top$ *is a function, called the* boundary descriptor*, that maps any face to its boundary. For sake of simplicity, we shall confuse any face* f *with the description of its boundary* $\mathcal{D}(f)$*. Hence, function* $\mathcal{D}$ *will be kept implicit and we simply denote by* $\langle X, E, F, o \rangle$ *the plane graph system* S*. At last, let* $\mathbb{G}$ *be the set of all plane graph (systems). Any subset of* $\mathbb{G}$ *is called a* plane graph language*.*

The *size* of a PGS $G = \langle X, E, F, o\rangle$ is $|G| = \sum_{f\in F} |f|$. Given any edge e, we denote by faces(e) the set of faces incident to edge e. Notice that faces(e) can contain either 1 or 2 faces (only one in the case of a *pendant edge*). We use nodes(f) and edges(f) for the set of vertices and edges along the boundary of face f, respectively.

For instance, consider the plane graph of Fig. 1. The corresponding PGS is $S = \langle X, E, F, o\rangle$ with $X = \{1, 2, 3, 4, 5, 6\}$, $E = \{\{1, 2\}, \{1, 3\}, \{2, 4\}, \{3, 4\}, \{3, 5\}, \{3, 6\}, \{4, 5\}\}$, $F = \{f_1, f_2, f_3\}$, $o = f_1$ and $f_1 = [13542]$, $f_2 = [1243]$, $f_3 = [34536]$. Moreover, we have faces($\{3, 4\}$) $= \{f_2, f_3\}$, faces($\{3, 6\}$) $= \{f_3\}$, nodes(f_3) $= \{3, 4, 5, 6\}$ and edges(f_3) $= \{\{3, 4\}, \{4, 5\}, \{5, 3\}, \{3, 6\}\}$. Note that every plane graph can be described with a plane graph system, but the converse does not hold in general. *E.g.*, if two faces of a PGS are declared to have [123] and [124] as respective boundaries, then no plane graph exists with such faces. Less straightforward examples exist. In order to get round these technical difficulties, it seems nevertheless possible to add further syntactic conditions in Def. 1. This work is still in progress.

Definition 2 (Set of contiguous faces) *Let $S = \langle X, E, F, o\rangle$ be a PGS. Two distinct faces $f, f' \in F$ are* adjacent *if $\exists e \in E$: faces(e) $= \{f, f'\}$. The faces of a subset $K \subseteq F$ are* contiguous *if $\forall f, f' \in K$, a sequence $f = f_0, f_1, \dots f_n = f'$ of faces in K exists such that $\forall i \in \{0, 1, \dots, n-1\}$, f_i and f_{i+1} are adjacent.*

Given a subset $K \subseteq F$ of contiguous faces. We denote by $outer(K)$ the (boundary of the) outer face of that set. For instance, on the PGS of Fig. 1, $outer(\{f_2, f_3\}) = f_1 = outer(\{f_2, f_3, f_1\})$ and $outer(\{f_3\}) = [354]$. Although beyond the scope of this paper, note that face $outer(K)$ can be computed in polynomial time using the *normalization* procedure introduced in [de la Higuera et al., 2012].

We also need a concatenation operation to glue together 2 distinct plane graphs whose outer faces share edges.

Definition 3 (Concatenation) *Let $G_1 = \langle X_1, E_1, F_1, o_2\rangle$ and $G_2 = \langle X_2, E_2, F_2, o_2\rangle$ be two PGS such that $F_1 \cap F_2 = \emptyset$ and $X_1 \cap X_2 = \{a_1, \dots a_k\}$, $k > 1$, and $o_1 = [a_1 \dots a_k y]$ and $o_2 = [a_k \dots a_1 z]$ with $y \in X_1^*, z \in X_2^*$ and $|y| \geq 2, |z| \geq 2$. The* concatenation *of G_1 and G_2, written $G_1 \diamond G_2$, is the PGS $G = \langle X_1 \cup X_2, E_1 \cup E_2, (F_1 \setminus \{o_1\}) \cup (F_2 \setminus \{o_2\}) \cup \{o\}, o\rangle$ with $o = [yz]$.*

One can easily prove that the definition of $G_1 \diamond G_2$ corresponds to the one of a PGS.

We finally need a way to compare two PGS:

Definition 4 (Plane isomorphism [de la Higuera et al., 2012]) *Let $G_1 = \langle X_1, E_1, F_1, o_1\rangle$ and $G_2 = \langle X_2, E_2, F_2, o_2\rangle$ be two PGS. We say that G_1 and G_2 are* plane-isomorphic*, written $G_1 \cong_p G_2$, if there exist a 1-to-1 mapping $\chi : X_1 \to X_2$ over the vertices and a 1-to-1 mapping $\xi : F_1 \to F_2$ over the faces such that (1) the outer face is preserved: $\xi(o_1) = o_2$ and (2) the boundaries are preserved: $\forall f_1 \in F_1, f_2 \in F_2$, if $\xi(f_1) = f_2$ and $f_1 = [a_1 \dots a_n]$ then $f_2 = [\chi(a_1) \dots \chi(a_n)]$.*

Plane-isomorphism is decidable in $\mathcal{O}(|E_1| \cdot |E_2|)$ time [de la Higuera et al., 2012]. Moreover, notice that if $G_1 \cong_p G'_1$ and $G_2 \cong_p G'_2$, and both $G_1 \diamond G_2$ and $G'_1 \diamond G'_2$ are defined, then $G_1 \diamond G_2 \cong_p G'_1 \diamond G'_2$.

3. The Substitutability Property for Plane Graph Languages

Definition 5 (Plane context) *A plane context is a tuple $C = \langle X, E, F, h, o\rangle$ such that (1) $\langle X, E, F, o\rangle$ is a plane graph system and (2) $h \in F \setminus \{o\}$ is a distinguished face called the* hole *of context C and (3) h has no pendant edge.*

The plane-isomorphism relation is extended to contexts in the obvious way: two contexts $C = \langle X, E, F, h, o\rangle$ and $C' = \langle X', E', F', h', o'\rangle$ are plane-isomorphic if $\langle X, E, F, o\rangle \cong_p \langle X', E', F', o'\rangle$ and the image of h by the bijection on the faces is h', i.e. $\xi(h) = h'$.

Let $S = \langle X, E, F, o\rangle$ and $S' = \langle X', E', F', o'\rangle$ be two PGS such that $X \cap X' = \emptyset$. Let $f \in F$ and $f' \in F'$ be two faces. Every 1-to-1 mapping $\phi : \text{nodes}(f) \to \text{nodes}(f')$ can be extended to the set of all vertices X as follows: $\hat{\phi} : X \to \text{nodes}(f') \cup X$ such that $\hat{\phi}(a) = \phi(a)$ if $a \in \text{nodes}(f)$ and $\hat{\phi}(a) = a$ otherwise. This function is extended to faces: $\hat{\phi}([a_1 \dots a_n]) = [\hat{\phi}(a_1) \dots \hat{\phi}(a_n)]$, to set of faces: $\hat{\phi}(F) = \{\hat{\phi}(f) : f \in F\}$ and to set of edges: $\hat{\phi}(E) = \{\{\hat{\phi}(a), \hat{\phi}(b)\} : \{a, b\} \in E\}$. By extension, given a PGS $G = \langle X, E, F, o\rangle$, $\hat{\phi}(G) = \langle \hat{\phi}(X), \hat{\phi}(E), \hat{\phi}(F), \hat{\phi}(o)\rangle$.

We can now define the *gluing* or *wrapping* operation.

Definition 6 (Gluing) *Let $C = \langle X, E, F, h, o\rangle$ be a context and $S = \langle X', E', F', o'\rangle$ be a PGS such that $X \cap X' = \emptyset$. Let ϕ be a bijective function from* $\text{nodes}(o')$ *to* $\text{nodes}(h)$*. The* gluing *of S into C following gluing function ϕ, denoted $C \odot_\phi S$, is the PGS $G = \langle X_G, E_G, F_G, o_G\rangle$ such that (1) $X_G = X \cup X' \setminus \text{nodes}(o')$, (2) $E_G = E \cup \hat{\phi}(E')$, (3) $F_G = (F \setminus \{h\}) \cup \hat{\phi}(F' \setminus \{o'\})$ and (4) $o_G = o$. In this case, we say that S is a* subgraph *or a* pattern *of G.*

Note that a pattern is actually a subset of contiguous faces in a plane graph. More general notions of subgraphs exist (based on subsets of vertices and edges, independently on faces), but they generally induce intractable problems. *E.g.*, the search of a pattern in a plane graph is tractable, whereas the search of a general subgraph is $\mathcal{NP}$-complete for plane graphs [de la Higuera et al., 2012]. In the following, term subgraph exclusively means pattern.

Definition 7 (Substitutability) *Two PGS $G = \langle X, E, F, o\rangle$ and $G' = \langle X', E', F', o'\rangle$ are* substitutable w.r.t. *a plane graph language L if whenever there exist two contexts C and C', $C \cong_p C'$, and two gluing functions ϕ and ϕ' such that $C \odot_\phi G$ is in L and $C' \odot_{\phi'} G'$ is in L, then for all contexts C'',*

$$\left(\exists \phi_1 :\ C'' \odot_{\phi_1} G \in L\right) \iff \left(\exists \phi_2 :\ C'' \odot_{\phi_2} G' \in L\right)$$

where for all $a \in \text{nodes}(o)$, for all $b \in \text{nodes}(o')$, if $\phi(a) = \phi'(b)$ then $\phi_1(a) = \phi_2(b)$.

If G and G' are substitutable, we will note $G \equiv_S^L G'$, or $G \equiv_S G'$ when there is no ambiguity.

Lemma 8 *Let G, G' and G'' be PGS. If $G \cong_p G'$ and $G' \equiv_S G''$ then $G \equiv_S G''$.*

Proof [Hint] Let χ be the 1-to-1 function that maps the vertices of G onto those of G' (as in the definition of plane isomorphism). Let C be a context such that there exists a gluing functions ϕ such that $C \odot_\phi G''$ in L. As $G' \equiv_S G''$ there exists ϕ' such that $C \odot_{\phi'} G'$ in L. By construction we have $C \odot_{\phi' o \chi} G = C \odot_{\phi'} G'$ and thus $C \odot_{\phi' o \chi} G$ is in L. ■

4. The Design of Binary Plane Graph Grammars

Definition 9 (Plane non-terminal) *A* plane non-terminal *is a couple* (N, r) *where* N *is a symbol and* r *an integer greater than* 2 *called the rank of* N*. In the following, we assume that any symbol has a unique rank, that is, if* (N, r_1) *and* (N, r_2) *are plane non-terminals, then* $r_1 = r_2$*.*

A non-terminal is just the shape of a plane graph. That is, it can be viewed as a PGS with a unique inner face made of r vertices and no pendant edge.

We now define the two types of grammar rules used in our grammars.

Definition 10 (Plane graph lexical rule) *A* plane graph lexical rule *is a pair* (A_x, G_*)*, written* $A_x \to G_*$*, where (1)* x *is a string, (2)* $(A, |x|)$ *forms a plane non-terminal and (3)* $G_* = \langle X, E, F, o\rangle$ *is a plane graph system such that* $|F| = 2$ *and* $o = [x^R]$*.*

Notice that the (unique) inner face of G_* does not have to be $[x]$: there may be pendant edges.

Definition 11 (Plane graph binary production) *A* plane graph binary production *is a tuple written* $A_x \to B_y C_z$*, where (1)* $(A, |x|)$*,* $(B, |y|)$ *and* $(C, |z|)$ *are plane non-terminals, and (2)* x, y, z *are strings with no repeated symbols such that* $\langle X, E, \{[y], [z], [x^R]\}, [x^R]\rangle$ *is a PGS, with* $X = \text{nodes}([y]) \cup \text{nodes}([z])$ *and* $E = \text{edges}([y]) \cup \text{edges}([z])$*.*

A production can be seen as the development of a face A made with $rank(A)$ vertices into 2 adjacent faces whose overall shape is the same as A. Fig. 2 gives a graphical representation of an example of a plane graph binary production.

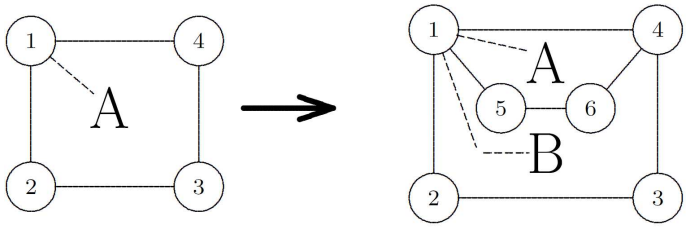

Figure 2: A graphical representation of the (recursive) plane graph binary production $A_{1234} \to A_{1564}\ B_{123465}$. The dashed lines attach each non-terminal to the vertex appearing at the head of the string corresponding to that face in the rule.

Definition 12 (Binary plane graph grammar) *A* binary plane graph grammar *(or simply* plane graph grammar*)* $\mathcal{G}$ *is a tuple* $\langle \mathcal{N}, P_L, P_B, \mathcal{A}\rangle$ *such that* $\mathcal{N}$ *is a set of plane non-terminals,* P_L *is a set of plane graph lexical rules,* P_B *is a set of plane graph binary productions and* $\mathcal{A} \subseteq \mathcal{N}$ *is the set of axioms.*

Definition 13 (Plane sentential form) *Let $\mathcal{G} = \langle \mathcal{N}, P_L, P_B, \mathcal{A} \rangle$ be a plane graph grammar. A* plane sentential form *is a couple $\langle G, \mathcal{L} \rangle$ where $G = \langle X, E, F, o \rangle$ is a PGS and $\mathcal{L} : F \to \mathcal{N} \times X$ is a partial function such that $\mathcal{L}(f) = (N, a)$ implies that $|\text{nodes}(f)| = rank(N)$ and $a \in \text{nodes}(f)$.*

Function $\mathcal{L}$ labels some faces with non-terminal symbols. It more precisely attaches the label to one distinguished vertex a of the face. This allows us to introduce some control during the application of a rule. Indeed, this trick is used to avoid all possible rotations of the right hand-side of the rule when this right hand-side is glued in mother graph. We formally detail this below.

4.1. Apply a lexical rule

A lexical rule $R : A_{a_1 \dots a_m} \to G_*$, with $G_* = \langle X_*, E_*, \{f_*, o_*\}, o_* \rangle$, is applicable to a sentential form $S = \langle G, \mathcal{L} \rangle$, with $G = \langle X, E, F, o \rangle$, if there exists a face $f = [a'_1 \dots a'_m] \in F$ such that $\mathcal{L}(f) = (A, a'_1)$. The resulting sentential form corresponds to S where $\mathcal{L}(f)$ is not defined anymore and f is replaced by the graph G_*, whose vertices are consistently renamed. More formally, *applying R to S following f*, consists in creating the sentential form $S' = \langle G', \mathcal{L}' \rangle$ with $G' = \langle X', E', F', o' \rangle$ such that

- $F' = F \setminus \{f\} \cup \hat{\phi}(f_*)$
- $\forall f' \in F, f' \neq f$: if $\mathcal{L}(f')$ is defined then $\mathcal{L}'(f') = \mathcal{L}(f')$
- $X' = X \cup \hat{\phi}(X_*)$
- $E' = E \cup \hat{\phi}(E_*)$

where ϕ is a injection defined as follow: $\forall i, 1 \leq i \leq m$, $\phi(a_i) = a'_i$; otherwise $\phi(c) \in \mathbb{N} \setminus X$.

4.2. Apply a binary production rule

A production rule $R : A_{a_1 \dots a_m} \to B_{b_1 \dots b_n}\ C_{c_1 \dots c_p}$ is applicable to a sentential form $S = \langle G, \mathcal{L} \rangle$, where $G = \langle X, E, F, o \rangle$, if there exists a face $f = [a'_1 \dots a'_m] \in F$ such that $\mathcal{L}(f) = (A, a'_1)$. *Applying R to S following f*, consists in creating the sentential form $S' = \langle G', \mathcal{L}' \rangle$ with $G' = \langle X', E', F', o' \rangle$ s.t.

- $F' = F \setminus \{f\} \cup \{\hat{\phi}([b_1 \dots b_n]), \hat{\phi}([c_1 \dots c_p])\}$
- $\forall f' \in F, f' \neq f$: if $\mathcal{L}(f')$ is defined then $\mathcal{L}'(f') = \mathcal{L}(f')$
- $\mathcal{L}'(\hat{\phi}([b_1 \dots b_n])) = (B, \phi(b_1))$, $\mathcal{L}'(\hat{\phi}([c_1 \dots c_p]) = (C, \phi(c_1))$
- $X' = X \cup \text{nodes}(\hat{\phi}([b_1 \dots b_n])) \cup \text{nodes}(\hat{\phi}([c_1 \dots c_p]))$
- $E' = E \cup \text{edges}(\hat{\phi}([b_1 \dots b_n])) \cup \text{edges}(\hat{\phi}([c_1 \dots c_p]))$

where ϕ is a injection defined as follow: $\forall i, 1 \leq i \leq m$, $\phi(a_i) = a'_i$; otherwise $\phi(c) \in \mathbb{N} \setminus X$.

4.3. Representable languages

Given a plane graph grammar $\mathcal{G} = \langle \mathcal{N}, P_L, P_B, \mathcal{A} \rangle$, we say that a plane graph $G = \langle X, E, F, o \rangle$ is derivable with $\mathcal{G}$, or that G belongs to the language of $\mathcal{G}$, if there exists a sequence of sentential forms $S_1, \ldots, S_n$ such that

- $S_1 = \langle G_1, \mathcal{L}_1 \rangle$ is an *initial sentential form*, that is, $G_1 = \langle X_1, E_1, F_1, o_1 \rangle$ with $F_1 = \{o^R, o\}$ and $o_1 = o$ and $\mathcal{L}_1$ is only defined for o^R: $\mathcal{L}_1(o^R) = (N, a)$, with $N \in \mathcal{A}$ and $a \in nodes(o^R)$,
- $\forall i, 1 \leq i < n$: S_{i+1} is obtained from S_i by applying a rule of $\mathcal{G}$,
- $S_n = \langle G_n, \mathcal{L}_n \rangle$ with $G_n \cong_p G$ and $\forall f, \mathcal{L}_n(f)$ is not defined.

n is said to be the *length of the derivation*. An example of a derivation is given in Fig. 3.

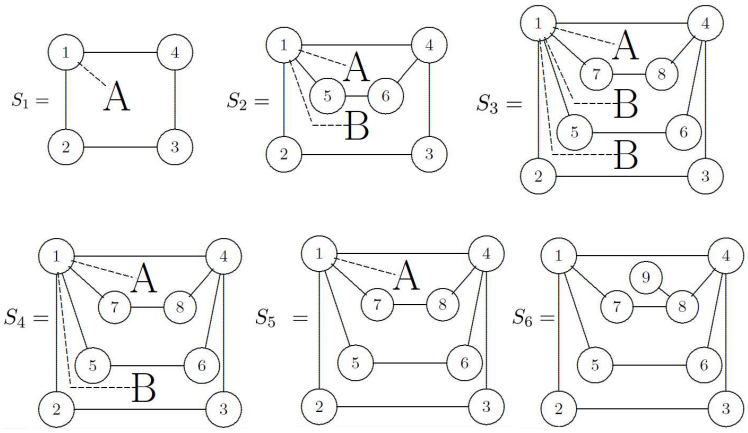

Figure 3: A graphical representation of an example of a derivation in the grammar $\mathcal{G} = \langle \{(A, 4), (B, 6)\}, P_L, P_B, \{(A, 4)\} \rangle$, with $P_L = \{A_{1234} \to [123534], B_{123465} \to [123465]\}$ (for sake of simplicity, inner faces are given instead of PGS) and $P_B = \{A_{1234} \to A_{1564}B_{123465}\}$. This derivation corresponds to the sequence of sentential forms $S_1, S_2, S_3, S_4, S_5, S_6$. The functions $\mathcal{L}_i$, $1 \leq i \leq 6$, are represented via dash lines.

We will write $N \Rightarrow^*_{\mathcal{G}} G$ when $G = \langle X, E, F, o \rangle$ is derivable with $\mathcal{G}$ using an initial sentential form S_1 where $\mathcal{L}_1$ is defined only on o^R and $\mathcal{L}_1(o^R) = (N, a)$, for some $a \in$ nodes(o^R). The language represented by $\mathcal{G}$ is $L(\mathcal{G}) = \{G : \exists G', \exists N \in \mathcal{A}\ s.t.\ N \Rightarrow^*_{\mathcal{G}} G' \wedge G' \cong_p G\}$. Notice that this definition implies that the language is closed under plane isomorphism.

5. Learning Binary Plane Graph Grammars

5.1. The Learner

As we are interested in the class of substitutable plane graph languages, the learning algorithm has to deal with the distribution of contexts between subgraphs. The following

definition is adapted from the work on substitutable string languages [Clark and Eyraud, 2007]:

Definition 14 (Substitution graph) *Given a finite set of plane graphs LS, the substitution graph $SG = \langle V, E\rangle$ corresponding to LS is defined as followed : The set of vertices V is composed of all subgraphs extractable from the elements of LS. There is an edge between two nodes G and G' if their outer face contains the same number of vertices and there exist two contexts C and C', $C \cong_p C'$, and functions ϕ, ϕ' such that $C \odot_\phi G \in LS$ and $C' \odot_{\phi'} G' \in LS$.*

The aim of that graph is to help the inference of the reductions rules: all subgraphs in the same connected component are substitutable, and thus will be derived by the same non-terminal.

Given a set of contiguous inner faces F, we define $split(F)$ to be the set of couples (F_1, F_2) such that $F_1 \cup F_2 = F$ and $G_{F_1} \diamond G_{F_2}$ is defined, where for $i \in \{1, 2\}$, $G_{F_i} = \langle \text{nodes}(F_i), \text{edges}(F_i), F_i \cup \{outer(F_i)\}, outer(F_i)\rangle$. The function $number\_nodes(C)$ returns the number of nodes of the outer face of the graphs in the connected component C of a substitution graph.

Our learning algorithm is described in Algorithm 1.

Algorithm 1: SGL for graphs

Input: A learning set of plane graph systems $LS = \{G_i\}_{i=1}^n$
Output: A plane graph grammar $\langle \mathcal{N}, P_L, P_B, \mathcal{A}\rangle$
$SG = \langle V, E\rangle \leftarrow create\_substitution\_graph(LS)$;
$\mathcal{N} \leftarrow \emptyset$; $P_L \leftarrow \emptyset$; $P_B \leftarrow \emptyset$; $\mathcal{A} \leftarrow \emptyset$;
foreach *Connected component C_i of SG* **do**
  $\mathcal{N} \leftarrow \mathcal{N} \cup \{(N^i, number\_nodes(C_i))\}$;
  foreach *G in C_i* **do**
    $\mathcal{N}^t(G) \leftarrow N^i$;
    if $G \in LS$ **then**
      $\mathcal{A} \leftarrow \mathcal{A} \cup \{N^i\}$
foreach $G = \langle X_G, E_G, F_G, [(a_1 \dots a_n)^R]\rangle$ *in V* **do**
  if $|F_G| = 2$ **then**
    $P_L \leftarrow P_L \cup \{\mathcal{N}^t(G)_{a_1 \dots a_n} \to G\}$;
  else
    foreach $(F_1, F_2) \in split(F_G \setminus \{o_G\})$ **do**
      $P_B \leftarrow P_B \cup \{\mathcal{N}^t(G_*)_{a_1 \dots a_m} \to \mathcal{N}^t(G_1)_{b_1 \dots b_n}\, \mathcal{N}^t(G_2)_{c_1 \dots c_p}\}$ where
      $[b_1 \dots b_n] = outer(F_1)$, $[c_1 \dots c_p] = outer(F_2)$, $b_1 = b$, $c_1 = c$ and
      $\forall i : G_i = \langle X_i, E_i, F_i \cup \{outer(F_i)\}, outer(F_i)\rangle$;
return $\langle N, P_L, P_B, \mathcal{A}\rangle$

5.2. Time complexity

The number of contexts that can be generated from a given PGS can be exponential in the size of that PGS (it is the case for instance of the plane graph corresponding to a grid, like

a chest board). So the size of the substitution graph is in general exponential in the size of the learning sample. This is a well-known problem while using graph grammar formalisms as it has to be tackled to obtain an efficient parsing algorithm. Thus, different restrictions on graphs have been designed that allow a polynomial number of subgraphs (and therefore of contexts). For instance, while one is interested by hyper-edge replacement grammars, requirement of a logarithmic k-separability [Lautemann, 1989] is a usual way to tackle this problem. This property can be adapted to plane graphs in a quite straightforward way that the lack of place unfortunately does not allow us to detail here.

To create the substitution graph, we first need to compare all pairs of contexts to decide if they are plane isomorphic. This can be done in polynomial time in the size of the contexts [de la Higuera et al., 2012]. For the same reason, testing if two PGS are plane isomorphic can be done in polynomial time and so is the construction of the substitution graph.

All other steps of Algorithm 1 are polynomial in the size of the substitution graph.

5.3. Identification in the limit

We now define our learning criterion. This is identification in the limit from positive text [Gold, 1967], with polynomial bounds on data and computation [de la Higuera, 1997]. Note that the size of a set of plane graphs LS is defined as $|LS| = \sum_{G \in LS} |G|$.

Definition 15 (Polynomial identification in the limit) *A representation class $\mathbb{R}$ is identifiable in the limit from positive data with polynomial time and data iff there exist two polynomials $p(), q()$ and an algorithm $\mathcal{A}$ such that: (1) Given a positive sample LS of size m $\mathcal{A}$ returns a representation $R \in \mathbb{R}$ in time $p(m)$; (2) For each representation R of size n there exists a characteristic set CS of size less than $q(n)$ such that if $CS \subseteq LS$, $\mathcal{A}$ returns a representation R' such that $L(R) = L(R')$.*

However, this definition initially designed for the learning of regular string languages, is already unsuitable as a model for context free string grammars [de la Higuera, 1997], so one cannot expect all requirements to be fulfilled in the case of graph grammars. As it is beyond the scope of this paper to attempt to resolve this difficulty, we shall thus adopt this approach in this paper.

5.3.1. Proof the hypothesis is not too large

We first need the following technical lemma, that states that substitutability is a congruence with respect to concatenation:

Lemma 16 *Let G_1, G_2, G'_1, G'_2 be 4 PGS s.t. $G_1 \diamond G_2$ and $G'_1 \diamond G'_2$ are defined. If $G_1 \equiv_S G_2$ and $G'_1 \equiv_S G'_2$ then $G_1 \diamond G_2 \equiv_S G'_1 \diamond G'_2$*

Proof [Sketch] We index each element of the definition of a graph by its name. For instance X_{G_1} are the vertices of G_1, $F_{G_1 \diamond G_2}$ are the faces of $G_1 \diamond G_2$, $o_{G'_1 \diamond G'_2}$ is the outer face of $G'_1 \diamond G'_2$. Let $C = \langle X_C, E_C, F_C, h, o_C \rangle$ be a plane context such that there exists a bijection ϕ: $C \odot_\phi G_1 \diamond G_2$ is in L. Let $C_{C \diamond \hat{\phi}(G_1)} = \langle X_C \cup \hat{\phi}(X_{G_1}) \cup \text{nodes}(\hat{\phi}(o_{G_2})), E_C \cup \hat{\phi}(E_{G_1}) \cup \text{edges}(\hat{\phi}(o_{G_2})), (F_C \setminus h) \cup \hat{\phi}(F_{G_1} \setminus o_{G_1}) \cup \hat{\phi}(o_{G_2}), \hat{\phi}(o_{G_2}), o_C \rangle$. Notice that $C_{C \diamond \hat{\phi}(G_1)}$ is correctly

defined as its faces are contiguous since, by definition of the concatenation, there exists $e \in E_{G_1}$, $e \in \text{edges}(o_{G_1 \diamond G_2})$ and $\phi : \text{nodes}(o_{G_1 \diamond G_2}) \to \text{nodes}(h)$. It is easy to verify by using the definitions that $C_{C \diamond \hat{\phi}(G_1)} \odot_\phi G_2 = C \odot_\phi G_1 \diamond G_2$.

As $G_2 \equiv_S G'_2$ there exists ϕ' such that $C_{C \diamond \hat{\phi}(G_1)} \odot_{\phi'} G'_2 \in L$. But $C_{C \diamond \hat{\phi}(G_1)} \odot_{\phi'} G'_2 = \langle X_C \cup \hat{\phi}(X_{G_1}) \cup \hat{\phi'}(X_{G'_2}), E_C \cup \hat{\phi}(E_{G_1}) \cup \hat{\phi'}(E_{G'_2}), (F_C \setminus h) \cup \hat{\phi}(F_{G_1} \setminus o_{G_1}) \cup \hat{\phi'}(F_{G'_2} \setminus o_{G_2}), o_C \rangle$ and thus it is equal to $C \odot_{id} \hat{\phi}(G_1) \diamond \hat{\phi'}(G'_2)$ where id is the identity function. Using the same kind of construction, we can define the context $C_{C \diamond \hat{\phi'}(G'_2)}$ such that $C_{C \diamond \hat{\phi'}(G'_2)} \odot_{id} \hat{\phi}(G_1) = C \odot_{id} \hat{\phi}(G_1) \diamond \hat{\phi'}(G'_2) \in L$. As $G_1 \equiv_S G'_1$ and $\hat{\phi}(G_1) \cong_p G_1$, there exists ϕ'' such that $C_{C \diamond \hat{\phi'}(G'_2)} \odot_{\phi''} G'_1 \in L$. Again, this is equivalent to write $C \odot_{id} \hat{\phi''}(G'_1) \diamond \hat{\phi'}(G'_2)$, and as $\hat{\phi''}(G'_1) \diamond \hat{\phi'}(G'_2) \cong_p G'_1 \diamond G'_2$ then there exists χ: $C \odot_\chi G'_1 \diamond G'_2 \in L$. ■

Lemma 17 *If $G = \langle X, E, F, o \rangle$ is a subgraph of a sample LS, then there exists a plane graph G' such that $\mathcal{N}^t(G) \Rightarrow^* G'$ and $G \cong_p G'$.*

Proof [Sketch] The proof can be done by induction on the number of faces of the graph. if $|F| = 2$, then by the construction of the grammar there is a lexical rule $\mathcal{N}^t(G)_{a_n \ldots a_1} \to G$ with $[a_1 \ldots a_n] = o$. Suppose the property holds for graphs with $|F| = k > 2$ faces. Let G be a graph such that $|F| = k + 1$. Let $G_1 = \langle \text{nodes}(F_1), \text{edges}(F_1), F_1 \cup \mathit{outer}(F_1), \mathit{outer}(F_1) \rangle$ and $G_2 = \langle \text{nodes}(F_2), \text{edges}(F_2), F_2 \cup \mathit{outer}(F_2), \mathit{outer}(F_2) \rangle$ such that $G_1 \diamond G_2$. G_1 and G_2 are also subgraphs of LS by definition and, by construction of the grammar, there exists a rule $\mathcal{N}^t(G)_{a_1 \ldots a_m} \to \mathcal{N}^t(G_1)_{b_1 \ldots b_n} \mathcal{N}^t(G_2)_{c_1 \ldots c_n}$ with $[a_1 \ldots a_m] = o$, $[b_1 \ldots b_n] = \mathit{outer}(F_1)$ and $[c_1 \ldots c_p] = \mathit{outer}(F_2)$.

This rule can be applied to the sentential form $\langle G', \mathcal{L}' \rangle$, with $G' = \langle X', E', \{o^R, o\}, o \rangle$, $\mathcal{L}'(o^R) = (\mathcal{N}^t(G), a_1)$. It gives the sentential form $\langle G'', \mathcal{L}'' \rangle$, with $G'' = \langle X'', E'', \{\hat{\phi}(\mathit{outer}(F_1)), \hat{\phi}(\mathit{outer}(F_2)), o\}, o \rangle$, $\mathcal{L}''(\mathit{outer}(F_1)) = (\mathcal{N}^t(G_1), b_1)$ and $\mathcal{L}''(\mathit{outer}(F_2)) = (\mathcal{N}^t(G_2), c_1)$ By the inductive hypothesis there exist G'_1 and G'_2 such that $\mathcal{N}^t(G_1) \Rightarrow^* G'_1$, $\mathcal{N}^t(G_2) \Rightarrow^* G'_2$, $G_1 \cong_p G'_1$ and $G_2 \cong_p G'_2$. Thus we have $\mathcal{N}^t(G) \Rightarrow^* G'_1 \diamond G'_2$ and $G \cong_p G'_1 \diamond G'_2$. ■

Lemma 18 *For all subgraphs G of a learning sample LS, for all PGS G', if $\mathcal{N}^t(G) \Rightarrow^* G'$ then G and G' are substitutable.*

Proof [Hint] Let $G = \langle X, E, F, o \rangle$. As the lemma holds for $G' \cong_p G$, we restrict ourselves to the case $G' \ncong_p G$. By induction on the length of the derivation k. If $k = 1$, then it means that a lexical production $\mathcal{N}^t(G)_{a_1 \ldots a_n} \to G''$ is applied and that $G'' \cong_p G'$. By the construction of the lexical rules, it means that G'' is a subgraph of LS that appears in the same component than G and thus G and G'' are substitutable. Lemma 8 implies that $G' \equiv_S G$.

Suppose this is true for all derivations of length less than k and let G' be a PGS obtained from $\mathcal{N}^t(G)$ using k derivation steps. It means that there exists a sequence of sentential form $S_1, \ldots, S_k$, $\forall i$, S_i is derived from S_{i-1}, $S_i = \langle G_i, \mathcal{L}_i \rangle$

such that $G_1 = \langle X_1, E_1, \{o^R, o\}, o\rangle$, $\mathcal{L}_1(o^R) = (\mathcal{N}^t(G), a)$ for some $a \in \text{nodes}(o)$, and $G_k \cong_p G'$, $\mathcal{L}_k$ being undefined for all faces of G_k. S_2 is obtained from S_1 applying a rule $\mathcal{N}^t(G)_{a1\dots a_m} \to \mathcal{N}^t(G_{F_1})_{b_1\dots b_n}\mathcal{N}^t(G_{F_2})_{c_1\dots b_p}$, where $outer(F_1) = [b_1 \dots b_n]$ and $outer(F_2) = [c_1 \dots c_p]$, $G_{F_i} = \langle X_{F_i}, E_{F_i}, F_i \cup outer(F_i), outer(F_i)\rangle$, for $i \in \{1, 2\}$. We have $G_{F_1} \diamond G_{F_2} \equiv_S G$ since they appear in the same connected component. There exist G'_{F_1} and G'_{F_2} such that $\mathcal{N}^t(G_{F_1}) \Rightarrow^* G'_{F_1}$, $\mathcal{N}^t(G_{F_2}) \Rightarrow^* G'_{F_2}$ and $G_k = G'_{F_1} \diamond G'_{F_2}$. By recursion, $G'_{F_1} \equiv_S G_{F_1}$ and $G'_{F_2} \equiv_S G_{F_2}$. A Lemma 16 holds, we have $G'_{F_1} \diamond G'_{F_2} \equiv_S G_{F_1} \diamond G_{F_2}$ and thus $G_k \equiv_S G$. ■

Theorem 19 *For all samples of a language L, the output $\mathcal{G}$ of Algorithm 1 is such that $L(\mathcal{G}) \subseteq L$.*

Proof Let $G \in L(\mathcal{G})$. Then there exists a plane graph G' in the learning sample and a plane graph G'' such that $\mathcal{N}^t(G') \Rightarrow^* G''$ and $G'' \cong_p G$. Lemma 18 states that G'' and G' are substitutable and thus $G \equiv_S G'$. As G' is an element of L, $G \in L$. ■

5.3.2. Proof the hypothesis is large enough

To prove that the hypothesis is large enough, we need to define a characteristic set, *i.e.* a subset of the target language L_* which ensures that the output $\mathcal{G}$ of the algorithm is such that $L(\mathcal{G}) = L_*$.

Construction of a characteristic sample. Let $\mathcal{G}_* = \langle \mathcal{N}_*, P_{L*}, P_{B*}, \mathcal{A}_*\rangle$ be a target grammar. We will assume without loss of generality, that $\mathcal{G}_*$ is reduced, that is to say for every non-terminal $N^k \in \mathcal{N}_*$, there is a sequence of sentential forms $\langle G_1, \mathcal{L}_1\rangle, \dots, \langle G_k, \mathcal{L}_k\rangle, \dots \langle G_n, \mathcal{L}_n\rangle$ such that $\langle G_1, \mathcal{L}_1\rangle$ is an initial sentential form and $\exists f_1 \in F_{G_1} : \mathcal{L}_1(f_1) = (N^j, a_j)$ with $N^j \in \mathcal{A}_*$, $\forall i, 1 \leq i < k$, $\langle G_{i+1}, \mathcal{L}_{i+1}\rangle$ is obtained from $\langle G_i, \mathcal{L}_i\rangle$ by applying a rule of $\mathcal{G}_*$, $\exists f_k \in F_{G_k}$: $\mathcal{L}_k(f_k) = (N^k, a_k)$, and $\forall f \in F_{G_n}$: $\mathcal{L}_n(f)$ is not defined. We are going to define a set $CS(\mathcal{G}_*)$ of plane graphs of L_*, such that Algorithm 1 will identify L_* from any superset of $CS(\mathcal{G}_*)$.

Given a non-terminal N^k, we define $C(N^k)$ to be one of the smallest context $\langle X_{G_k}, E_{G_k}, F_{G_k}, h_k, o_{G_k}\rangle$ such that there exists a sequence of sentential forms $\langle G_1, \mathcal{L}_1\rangle, \dots, \langle G_k, \mathcal{L}_k\rangle$ with $\langle G_1, \mathcal{L}_1\rangle$ being an initial sentential form and $F_{G_1} = [o_{G_k}, o^R_{G_k}]$, $\mathcal{L}_1(o_{G_k}) = (N^i, a_1)$, $N^i \in \mathcal{A}_*$, $\forall i, 1 \leq i < k$, $\langle G_{i+1}, \mathcal{L}_{i+1}\rangle$ is obtained from $\langle G_i, \mathcal{L}_i\rangle$ by applying a rule of $\mathcal{G}_*$, $\mathcal{L}_k(h_k) = (N^k, a_k)$ for some $a_k \in \text{nodes}(h_k)$, $\mathcal{L}_k$ undefined on other faces.

We also define $G(N^k)$ to be one of the smallest PGS such that $N^k \Rightarrow^*_{\mathcal{G}_*} G(N^k)$.

We can now define the characteristic set $CS(\mathcal{G}_*)$. For each production $N^i_x \to N^j_y N^k_z$ in P_{B*}, we add to $CS(\mathcal{G}_*)$ the PGS $C \odot_\phi \hat{\chi}(G_1) \diamond \hat{\chi}(G_2)$ where $\phi : \text{nodes}([x]) \to \text{nodes}(h)$ is a bijective function, $C = C(N^i), G_1 = G(N^j), G_2 = G(N^k)$ and $\chi : \text{nodes}(o_{G_1}) \cup \text{nodes}(o_{G_2}) \to \text{nodes}([y]) \cup \text{nodes}([z])$ is a bijective function such that $\hat{\chi}(o_{G_1}) = [y]$ and $\hat{\chi}(o_{G_2}) = [z]$. For each lexical rule $N^i_x \to G$ in P_{L*} we add to $CS(\mathcal{G}_*)$ the PGS $C \odot_\phi G$ where $\phi : \text{nodes}([x]) \to \text{nodes}(h)$ is a bijective function and $C = C(N^i)$.

The cardinality of this set is at most $|P_{B*}|+|P_{L*}|$ which is clearly polynomially bounded. In general the cardinality of the set will not polynomially bound the size of the sample, as it is already the case for string context-free grammars (see [Clark and Eyraud, 2007] for a detailed discussion). However, notice that if there exists a polynomial-sized structurally complete sample – that is to say a sample where there is at least a plane graph that uses each production rule [Dupont et al., 1994] – then the size of our characteristic set is polynomial.

Convergence. We must show that for any substitutable plane graph grammar $\mathcal{G}_*$, if the sample LS contains the characteristic sample $CS(\mathcal{G}_*)$, then $L(\mathcal{G}) = L(\mathcal{G}_*)$ where $\mathcal{G} = \langle \mathcal{N}, P_L, P_B, \mathcal{A} \rangle$ is the inferred grammar.

Lemma 20 *If $N \Rightarrow^*_{\mathcal{G}_*} G$ then there exists a subgraph G' of the learning sample and a plane graph G'' such that $N \Rightarrow^*_{\mathcal{G}_*} G'$, $\mathcal{N}^t(G') \Rightarrow^*_{\mathcal{G}} G''$ and $G'' \cong_p G$.*

Proof [Sketch] By recursion on the number of derivation steps k in $\mathcal{G}_*$. If $k = 1$ then there exists $N \to G'$ in P_{L*}, $G' \cong_p G$. By construction of the characteristic sample, G' is a subgraph of LS and thus $\mathcal{N}^t(G') \to G'$ is in P_L.

Suppose it is true for all derivations of size less than $k > 1$. There exist a sequence of sentential forms $\langle G_1, \mathcal{L}_1 \rangle, \ldots, \langle G_k, \mathcal{L}_k \rangle$ such that $\langle G_1, \mathcal{L}_1 \rangle$ is an initial sentential form with $\mathcal{L}(f_1) = (N, a)$, S_{i+1} is obtained from S_i by using a rule of $\mathcal{G}_*$, $G_k = G$ and $\mathcal{L}_k$ is not defined for any face. Let $N_x \to N^i_y N^j_z$ be the rule applied to S_1 to obtain S_2. By construction, there exist G_1 and G_2, $N^i \Rightarrow^*_{\mathcal{G}_*} G_1$, $N^j \Rightarrow^*_{\mathcal{G}_*} G_2$, and $G_1 \diamond G_2 = G$.

By recursion, there exist two subgraphs of LS, G'_1 and G'_2, and two PGS G''_1 and G''_2 such that $N^i \Rightarrow^*_{\mathcal{G}_*} G'_1$, $N^j \Rightarrow^*_{\mathcal{G}_*} G'_2$, $\mathcal{N}^t(G'_1) \Rightarrow^*_{\mathcal{G}} G''_1$, $\mathcal{N}^t(G'_2) \Rightarrow^*_{\mathcal{G}} G''_2$ and $G''_1 \cong_p G_1$, $G''_2 \cong_p G_2$. Notice that this implies there exists a renaming function ϕ on the vertices of the external faces of G''_1 and G''_2 such that $\hat{\phi}(G''_1) \diamond \hat{\phi}(G''_2)$ is defined and $\hat{\phi}(G''_1) \diamond \hat{\phi}(G''_2) \cong_p G$.

By construction of the characteristic sample, there exist two subgraphs G'''_1 and G'''_2 of LS such that $G'''_1 \diamond G'''_2$ is a subgraph of LS, $G'''_1 \cong_p G(N^i)$ and $G'''_2 \cong_p G(N^j)$. As $L(\mathcal{G}_*)$ is a substitutable language, we have $G'''_1 \equiv_S G'_1$ and $G'''_2 \equiv_S G'_2$. Thus G'_1 and G'''_1 appear in the same component and thus correspond to the same non-terminal (and similarly for G'''_2 and G'_2). As there is a rule $\mathcal{N}^t(G'''_1 \diamond G'''_2)_x \to \mathcal{N}^t(G'''_1)_y \mathcal{N}^t(G'''_2)_z$ in P_B, we have $\mathcal{N}^t(G'''_1 \diamond G'''_2) \Rightarrow^*_{\mathcal{G}} \hat{\phi}(G''_1) \diamond \hat{\phi}(G''_2)$. ■

Theorem 21 *Let $\mathcal{G}_*$ be the target plane graph grammar corresponding to a substitutable plane graph language. Algorithm 1 returns a grammar $\mathcal{G}$ from any sample containing $CS(\mathcal{G}_*)$ such that $L(\mathcal{G}) = L(\mathcal{G}_*)$.*

Proof If $G \in L(\mathcal{G}_*)$ then there exists $N \in \mathcal{A}_*$ such that $N \Rightarrow^*_{\mathcal{G}_*} G$. By Lemma 20, it implies that there exists a subgraph G' of the learning sample and a plane graph G'' such that $G' \in L(\mathcal{G}_*)$, $\mathcal{N}^t(G') \Rightarrow^*_{\mathcal{G}} G''$ and $G'' \cong_p G$. By construction of the grammar, $\mathcal{N}^t(G') \in \mathcal{A}$ and thus $G \in L(\mathcal{G})$. Therefore $L(\mathcal{G}_*) \subseteq L(\mathcal{G})$. Due to Theorem 19, we have $L(\mathcal{G}) = L(\mathcal{G}_*)$. ■

6. Discussion

In addition to substitutability, other restrictions on the learned class have been done, explicitly or not. First, the grammar formalism implies that the number of nodes of the outer face of any generated PGS has to be bounded; otherwise an infinite number of axioms would be needed. In addition, the use of the concatenation operation in the definition of the split operations used by the learner also restricts the class. The requirement for a polynomial number of contexts is a strong restriction, but the condition we gave has to be refined and further analysis is needed.

Another important drawback of this work concerns the absence of a parsing algorithm for plane graph grammars. Clearly, we needed in this paper to define both a new graph grammar formalism and a learning algorithm, which is impossible due to the lack of space. Nevertheless, concerning this particular point, the results on the polynomiality of the search for patterns in plane graphs [de la Higuera et al., 2012] let us think that a CYK-like approach is likely to be tractable. This is a piece of work in progress.

Despite all these issues, this paper describes, to our knowledge, the first positive formal learning result for a class of graph grammars. Moreover, due to the interest of PGS in image processing [Samuel et al., 2010], we think that the learning of plane graph grammars, and more generally grammatical inference techniques, could be used to tackle image classification tasks.

References

R. Brijder and H. Blockeel. On the inference of non-confluent NLC graph grammars. *Journal of Logic and Computation*, 2011.

A. Clark. Towards general algorithms for grammatical inference. In *Proc. of ALT*, pages 11–30. Springer, 2010. Invited Paper.

A. Clark and R. Eyraud. Polynomial identification in the limit of substitutable context-free languages. *J. Machine Learning Research*, 8:1725–1745, 2007.

D. J. Cook and L. B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15: 32–41, 2000.

C. CostaFlorencio. Identification of hyperedge-replacement graph grammars. In *MLG'09, Poster Session*, Leuven, Belgium, 2009.

C. de la Higuera. Characteristic sets for polynomial grammatical inference. *Machine Learning Journal*, 27:125–138, 1997.

C. de la Higuera, J.-C. Janodet, E. Samuel, G. Damiand, and C. Solnon. Polynomial algorithms for open plane graph and subgraph isomorphisms. *Theoretical Computer Science*, 2012. to appear. Available here: http://labh-curien.univ-st-etienne.fr/~janodet/TCS.pdf.

P. Dupont, L. Miclet, and E. Vidal. What is the search space of the regular inference? In *Proc. of ICGI*, pages 25–37, 1994.

I. Fáry. On straight line representation of planar graphs. *Acta Univ Szeged. Sect. Sci. Math*, 11:229–233, 1948.

E. Fusy. *Combinatoire des graphes planaires et applications algorithmiques (in english).* PhD thesis, Ecole Polytechnique - ParisTech, 2007.

A. Gibbons. *Algorithmic graph theory.* Cambridge University Press, 1985.

E. Gold. Language identification in the limit. *Information and Control*, 10(5):447–474, 1967.

J. Huan, W. Wang, and J. Prins. Efficient mining of frequent subgraphs in the presence of isomorphism. In *ICDM*, pages 549–552, 2003.

E. Jeltsch and H.-K. Kreowski. Grammatical inference based on hyperedge replacement. In *Proc. Graph Grammars and their Application to Computer Science*, pages 461–474, 1991.

I. Jonyer, L. B. Holder, and D. J. Cook. Mdl-based context-free graph grammar induction. *International Journal of Artificial Intelligence Tools*, 13:65–79, 2003.

A. Kasprzik and R. Yoshinaka. Distributional learning of simple context-free tree grammars. In *ALT*, pages 398–412, 2011.

J. Kukluk, L. Holder, and D. Cook. Inference of edge replacement graph grammars. *International Journal on Artificial Intelligence Tools*, 17(3):539–554, 2008.

C. Lautemann. The complexity of graph languages generated by hyperedge replacement. *Acta Inf.*, 27(5):399–421, 1989.

T. Matsuda, H. Motoda, and T. Washio. Graph-based induction and its applications. *Advanced Engineering Informatics*, pages 135–143, 2002.

G. Rozenberg and H. Ehrig. *Handbook of Graph Grammars and Computing by Graph Transformation*, volume 1–3. World Scientific, 1997.

E. Samuel, C. de la Higuera, and J.-C. Janodet. Extracting plane graphs from images. In *SSPR/SPR*, pages 233–243, 2010.

S. Vishwanathan, N. Schraudolph, R. Imre Kondor, and K. Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242, 2010.