Acta Informatica 41, 293–314 (2005)
Digital Object Identifier (DOI) 10.1007/s00236-004-0160-0

Families of automata characterizing context-sensitive languages

Christophe Morvan, Chloé Rispal

Institut Gaspard Monge, Université de Marne-la-Vallée, 5, bd Descartes, 77454 Marne-la-Vallée cedex 2, France
(e-mail: {christophe.morvan|chloe,rispal}@univ-mlv.fr)

Received: 17 December 2003 / 23 November 2004
Published online: 27 December 2004 – © Springer-Verlag 2004

Abstract. In the hierarchy of infinite graph families, rational graphs are defined by rational transducers with labelled final states. This paper proves that their traces are precisely context-sensitive languages and that this result remains true for synchronized rational graphs.

1 Introduction

During the last fifteen years, there has been a great deal of interest around families of infinite graphs. The decidable properties of these families provide a nice framework for validation and verification. Muller and Schupp introduced in [16] the transition graphs of *pushdown automata* and proved that their monadic second order theory was decidable. A few years later, Courcelle extended this result to *regular graphs* generated by deterministic graph grammars, [7]. In 1996 Caucal used inverse rational substitution (followed by a rational restriction) to define the *prefix-recognizable graphs*; they have a decidable second order monadic theory [4]. The automatic graphs form a more general family of graphs. They are automatic structures, defined in 2000 by Blumensath and Grädel [1], and have, thus, a decidable first order theory. Very recently Colcombet considered an interesting extension of prefix-recognizable graphs, namely the *VRP-graphs* (vertex replacement with product) [6]. They are obtained using vertex replacement systems and a graph product. Their first-order theory with accessibility is decidable.

The study of infinite graph families is also naturally linked to language theory. Precisely, the transition graphs of pushdown automata and prefix-recognizable graphs are defined from language theory. Recently, Urvoy extended the work of Ginsburg and Greibach [20] to define abstract families of graphs [22]. The connection between families of graphs and language theory is even deeper: they constitute an elegant characterization of families of languages. If we consider the *trace* of a graph as the language of path labels leading from an *initial* set of vertices to a *final* set of vertices, then traces form one of the most important link between graphs and languages. For example, it is well known that the traces of finite graphs are regular languages [11]. By construction, the traces of the transition graphs of push-down automata are the context-free languages. These languages are also the traces of prefix-recognizable graphs [4]. At this time, the languages corresponding to the VRP-graphs is still unknown. In 2001 Caucal used Turing machines to define a class of graphs whose traces are recursively enumerable languages [3].

In this paper we establish a new correspondence between the Chomsky hierarchy [5] and families of graphs. We prove that the traces of rational graphs (generated by labelled rational transducers [14]), are context-sensitive languages. We show that this result remains true if we restrict to synchronized graphs [18]. In those cases the traces correspond to path labels between finite sets. Extending initial and final sets to rational sets, letter-to-letter rational graphs also trace context-sensitive languages.

This article is organized in three sections. The first one uses *finite transducers*, that is finite automata labelled with pairs, to define the *rational graphs*. Some basic results and definitions about context-sensitive languages are also recalled. The second section proves that the trace of any rational graph can be recognized using a linear bounded Turing machine, and is therefore a context-sensitive languages. Finally, the third section uses the Penttonen normal form [17] to prove that any context-sensitive language is the trace of a rational graph. Indeed, it proves that the synchronized rational graphs, which is a proper subclass of rational graphs [1], [21], are sufficient to obtain any context-sensitive language.

2 Preliminary definitions

In this section, we recall basic definitions concerning infinite graphs and context-sensitive languages. In the first part, rational graphs, synchronized graphs and letter-to-letter graphs are defined from transducers. Then, context-sensitive languages are characterized both from Turing machines and from Penttonen's rewriting systems in the second part.

2.1 Graphs and transducers

Let $\mathcal{A}$ be a finite set of labels. A simple arc labelled *graph* is a subset of $V \times \mathcal{A} \times V$ where V is an arbitrary set of *vertices*.
We denote by $s \xrightarrow[G]{a} t$ the existence of the arc (s, a, t) in the graph G or simply

by $s \xrightarrow{a} t$ when there is no ambiguity.

A *path* $s \underset{G}{\overset{u}{\Longrightarrow}} t$ of a graph G leading from a vertex s to a vertex t and of label u is a finite sequence $(s_0, a_1, s_1)...(s_{n-1}, a_n, s_n)$ of arcs of G such that $u = a_1...a_n$, $s = s_0$ and $t = s_n$.
A *trace* of a graph G is the language $L(G, I, F)$ of path labels leading from a set I of initial vertices to a set F of final vertices:

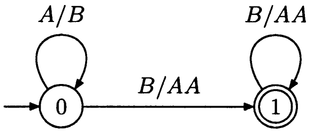
$$L(G,I,F) \;=\; \{\, u \mid \exists\, s \in I\; \exists\, t \in F,\; s \underset{G}{\overset{u}{\Longrightarrow}} t \,\}$$

An *automaton* A is a triple (G, I, F) where G is a finite graph and I and F are initial and final sets of states. The language, $L(A)$, recognized by A is the trace $L(G, I, F)$.
Let Σ be a finite alphabet. We denote by Σ^* the set of finite words over letters of Σ, and we write ε for the empty word.
A *transducer* T is a finite automaton where labels have been modified to recognize relations instead of sets of words. It is defined by a finite subset of $Q \times \Sigma^* \times \Sigma^* \times Q$ of labelled arcs, where Q is a finite set of states, by a set $I \subseteq Q$ of initial states, and by a set $F \subseteq Q$ of final states. So a transducer is labelled by pairs of words. Any transition (p, u, v, q) of a transducer T is denoted by $p \underset{T}{\xrightarrow{u/v}} q$ or by $p \xrightarrow{u/v} q$ when T is understood.

A path $p_0 \xrightarrow{u_1/v_1} p_1 \ldots p_{n-1} \xrightarrow{u_n/v_n} p_n$ with $u = u_1...u_n$ and $v = v_1...v_n$ is labelled u/v and is denoted by $p_0 \underset{T}{\overset{u/v}{\Longrightarrow}} p_n$. A path is *successful* if it leads from an initial state to a final one. A pair $(u, v) \in \Sigma^* \times \Sigma^*$ is *recognized* by a transducer if there exists a successful path labelled u/v.

Definition 2.1. *A relation is rational if it is recognized by a transducer.*

We denote by $Rat(\Sigma^* \times \Sigma^*)$ the set of binary rational relations. The following transducer, with initial state 0 (marked by an incoming arrow) and final state 1 (marked by a double circle) recognizes the rational relation $\{\, (A^n B^m\, ,\, B^n A^{2m}) \mid n \geq 0, m > 0 \,\}$.

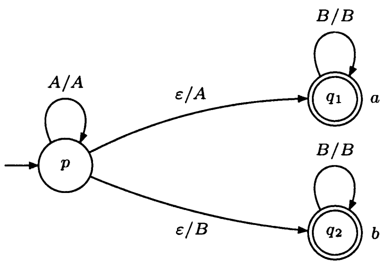


2.2 Hierarchy of rational graphs

Using words as vertices, infinite graphs can be defined by the relations between the extremities of its arcs. Given any graph $G \subseteq \Sigma^* \times \mathcal{A} \times \Sigma^*$, we

denote by $\xrightarrow[G]{a}$ the relation $\{\ (s,t) \mid (s,a,t) \in G\ \}$. The graph G is *rational* if for each $a \in \mathcal{A}$, the relation $\xrightarrow[G]{a}$ is rational.
For instance, the following graph, on the left, called the grid is a rational graph since it is defined by the following transducer, on the right.

$$\begin{tikzcd}
\varepsilon \arrow[r, "a"] \arrow[d, "b"] & A \arrow[r, "a"] \arrow[d, "b"] & A^2 \arrow[r, dash, dashed] \arrow[d, "b"] & {} \\
B \arrow[r, "a"] \arrow[d, "b"] & AB \arrow[r, "a"] \arrow[d, "b"] & A^2B \arrow[d, "b"] & \\
B^2 \arrow[r, "a"] \arrow[d, dash, dashed] & AB^2 \arrow[r, "a"] & A^2B^2 \arrow[dr, dash, dashed] & \\
{} & & & {}
\end{tikzcd}$$



Subfamilies of rational graphs are defined from subsets of rational relations.
If a transducer has labels over $\Sigma \times \Sigma$ it is called a *letter-to-letter transducer*: it is a transducer labelled by pairs of letters instead of pairs of words.

Definition 2.2. *A relation is letter-to-letter if it is recognized by a letter-to-letter transducer.*

A graph G is a *letter-to-letter graph* if for each $a \in \mathcal{A}$, the relation G_a is letter-to-letter.
Another particular subset of rational relations called *left-synchronized* relations has been studied by Elgot and Mezei [8] and then by Frougny and Sakarovitch [10]. Those relations are recognized by letter-to-letter transducers with rational terminal functions completing one side of the recognized pairs. The terminal function associates a relation to each terminal state of the transducer. Then, the relation defined is the set of labels of path ending at a state q, concatenated with pairs of the terminal function's value in q. For example, a pair (u,v) belongs to a synchronized relation R, if there exists two pairs of words (u',v') and (u'',v'') such that $(u,v) = (u',v') \cdot (u'',v'')$ with the following condition: there exists a terminal state q, a path labelled (u',v') leading to q, and (u'',v'') belongs to the value of the terminal function in q. Formally:

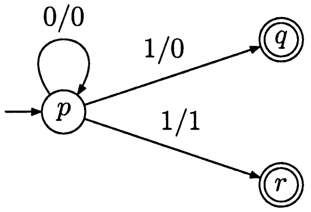
Definition 2.3. *A relation over $\Sigma^* \times \Sigma^*$ is left-synchronized if it is recognized by a letter-to-letter transducer T with terminal function f taking values in*

$$\mathit{Dif}_{Rat} \ = \ (\mathit{Rat}(\Sigma^*) \times \{\varepsilon\}) \cup (\{\varepsilon\} \times \mathit{Rat}(\Sigma^*))$$

That is, a left-synchronized relation is a finite union of elementary relations of the form $R.S$ where $R \in \mathit{Rat}((\Sigma \times \Sigma)^*)$ and $S \in \mathrm{Dif}_{Rat}$.

Right-synchronized relations are defined symmetrically with an initial rational function. A rational relation is *synchronized* if it is left-synchronized or right-synchronized.

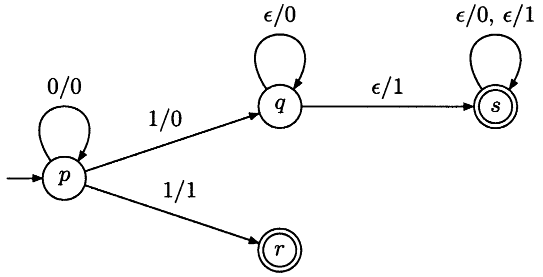
Example 2.4. Let us consider the relation $|_2$ defined by $x|_2y$ if x is a power of 2 dividing y. Provided integers are coded in base 2 (with lowest bits on the left), the relation $|_2$ is left-synchronized. This relation is recognized by the following letter-to-letter transducer with the terminal function f defined by $f(q) = (\varepsilon, 0)^*(\varepsilon, 1)\{(\varepsilon, 0), (\varepsilon, 1)\}^*$ and $f(r) = (\varepsilon, \varepsilon)$.



As the terminal function is rational, it can be introduced in the transducer adding states and transitions. A *left-synchronized transducer* is a transducer such that each path leading from an initial vertex to a final one can be divided into two parts: the first one only contains arcs of the form $\{p \xrightarrow{A/B} q | p, q \in Q \wedge A, B \in \Sigma\}$ while the second part contains either arcs of the form $\{p \xrightarrow{A/\epsilon} q | p, q \in Q \wedge A \in \Sigma\}$ or arcs of the form $\{p \xrightarrow{\epsilon/B} q | p, q \in Q \wedge B \in \Sigma\}$ (but not both).

Remark 2.5. Automatic structures, [1], or automatic groups, [9], are defined by automatic relations which are equivalent to synchronized relations.

Example 2.6. The following left-synchronized transducer recognizes the left-synchronized relation of Example 2.4.



A graph G is *left-synchronized* if for each $a \in \mathcal{A}$, the relation G_a is left-synchronized.

Subfamilies of rational relations are closed under union, intersection and complementation.

Theorem 2.7. **[8]** *The rational left-synchronized relations (respectively letter to letter relations) form a boolean algebra.*

A very important consequence of this result is the decidability of the first order theory of the graphs defined using synchronized relations.

We also use particular left-synchronized relations. A binary relation R is *recognizable* if it is a finite union of products $S \times T$ where $S, T \in Rat(\Sigma^*)$. A binary relation R over words is of *bounded length difference* if there exists an integer b such that $|\,|u| - |v|\,| \leq b$ for any $(u, v) \in R$.

Proposition 2.8. **[10]** *The family of synchronized relations contains the recognizable relations and the rational relations of bounded length difference.*

When working with rational or synchronized graphs, it is sufficient to consider the traces between singletons instead of rational sets.

Lemma 2.9. *Let* $G \subseteq \Sigma^* {\times} \mathcal{A} {\times} \Sigma^*$ *be a left-synchronized graph.*
Let $I, F \in Rat(\Sigma^*)$ *and* $\$, \# \notin \Sigma$*.*
There exists a left-synchronized graph $H \subseteq (\Sigma^* \cup \{\$, \#\}) {\times} \mathcal{A} {\times} (\Sigma^* \cup \{\$, \#\})$ *such that*

$$L(G, I, F) \;=\; L(H, \{\$\}, \{\#\}).$$

Proof. **i)** For all $a \in \mathcal{A}$, we define:

$$F_a \;:=\; Dom(\xrightarrow[G]{a} \,\cap\, \Sigma^* {\times} F)$$

the set of vertices which are source of an arc leading to a final state. This set is rational being the domain of a rational relation. Then we create new arcs leading from those vertices to the vertex $\#$. More precisely, for all $a \in \mathcal{A}$, we define the arcs of the graph G' to be as follows:

$$\xrightarrow[G']{a} \;:=\; \xrightarrow[G]{a} \,\cup\, F_a {\times} a {\times} \{\#\}$$

This relation is left-synchronized as the union of a left-synchronized relation with a recognizable set. Moreover and by construction,

$$L(G, I, F) \;=\; L(G', I, \{\#\})$$

ii) By a symmetric argument, a graph G'' is defined such that,

$$L(G', I, \{\#\}) \;=\; L(G'', \{\$\}, \{\#\}). \qquad \square$$

2.3 Context-sensitive languages

In Chomsky's hierarchy of languages, the family of context-sensitive languages (Csl) is located between recursively enumerable and context-free languages. There are many different ways to characterize this family of languages. In the following, we recall two of those characterizations. The first one, due to Kuroda [12], defines context-sensitive languages as the languages recognized by Linearly Bounded Turing machines (LBM). The second characterization due to Penttonen [17], is based on a particular rewriting system.

Context-sensitive languages from Turing machines
A linearly bounded machine is a Turing machine such that the size of the tape is bounded, linearly, by the length of the input. These machines are a classical characterization of Csl.

Theorem 2.10. [12] *Context-sensitive languages are the set of languages recognized by linearly bounded Turing machines.*

Penttonen's characterization of Context-sensitive languages
A different characterization of Csl, due to Penttonen [17], is based on a rewriting system using particular rules.

Definition 2.11. *A rewriting system $\Gamma = \Gamma_1 \cup \Gamma_2$ is a* 2-system *if every rule of Γ_2 is of the form $AB \to AC$ with $B \neq C$ and every rule of Γ_1 is of the form $A \to a$ where A, B, C are letters of the non-terminal alphabet Σ and $a \in \mathcal{A}$.*

Context-sensitive languages are obtained by derivation of a 2-system from a linear language.

A language is *linear* if it can be generated by a grammar whose rules are of the form $Z \longrightarrow W$, where Z is a non-terminal, W is a word over terminal and non-terminal symbols, with at most one non-terminal.

Theorem 2.12. [17] *There exists a linear language L_{Lin} such that every context-sensitive language is $\{v \in \mathcal{A}^* \mid \exists\, u \in L_{Lin}\,,\ u \xrightarrow[\Gamma]{*} v\}$ for some 2-system Γ.*

3 From rational graphs to context-sensitive languages

In this section, we prove that the traces of rational graphs between initial and final rational sets of vertices are context-sensitive languages, this exposition is a detailed (and simplified) version of the first section of [15].

3.1 First approach

As we have seen in Section 2.3, a common characterization of Csl is given by LBM. The first idea is to simulate a rational graph with an LBM. Any vertex of the graph would be stored on the tape, and the machine would compute the next vertex.

This basic approach fails to recognize the traces of rational graphs: the length of the vertices may grow exponentially. Example 3.1 illustrates this situation.

Example 3.1. The transducer, having a single state p (initial and final labelled a) and a single transition $p \xrightarrow{A/AA} p$ defines the following graph:

$$\begin{tikzcd}
A \arrow[r, "a"] & A^2 \arrow[r, "a"] & A^4 \arrow[r, "a"] & \cdots \arrow[r, "a"] & A^{2^n} & - - - - \\
A^3 \arrow[r, "a"] & A^6 \arrow[r, "a"] & A^{12} \arrow[r, "a"] & \cdots \arrow[r, "a"] & A^{3.2^n} & - - - - \\
A^5 \arrow[r, "a"] & A^{10} \arrow[r, "a"] & A^{20} \arrow[r, "a"] & \cdots \arrow[r, "a"] & A^{5.2^n} & - - - - \\
\vdots & & \vdots & \vdots & &
\end{tikzcd}$$

The trace of this graph between A and A^* is obviously a^*. The problem is that the length of the vertices is exponential in the length of the recognized word. For example, the path recognizing a^3 is the following:

$$A \xrightarrow{a} AA \xrightarrow{a} A^4 \xrightarrow{a} A^8$$

More generally: a^n leads from A to A^{2^n}. Therefore, it is not straightforward to construct a linear bounded machine recognizing the language of the transducer.

The last remarks leads to encode the vertices of the graph in order to keep their length linear. In this case it becomes difficult to compute the "next vertex function". Especially if some branches of the transducer produce a sub-graph with a linear growth, and some other with an exponential growth.

The next section exposes a different approach which avoids those difficulties.

3.2 Construction of the LBM

Let $G \in \Sigma^* \times \mathcal{A} \times \Sigma^*$ be a rational graph recognized by a transducer T. For each a in $\mathcal{A}$, we denote by T_a the transducer recognizing $\xrightarrow[G]{a}$. We construct a LBM recognizing the trace $L(G, I, F)$, where I and F are rational sets.

Roughly speaking, our solution is to simulate the transitions of G in parallel: for example, let us consider a path $U \xrightarrow{a} V \xrightarrow{b} W$ in G and suppose that the first transition of T_a is of the form $q \xrightarrow{\epsilon/X} q'$. Then X is the first element of V, thus we can activate T_b knowing that X is the first non-empty left-hand-side label.

Using this observation we only need to keep on the tape of the machine one state for each transducer T_a, plus some bounded information corresponding to what it might consume and what it has produced (and that has not been consumed yet).

By Lemma 2.9 we suppose that I and F are singletons containing respectively \$ and $\#$.

A transducer is *normalized* if all its transitions are of the form, $p \xrightarrow{u/v} q$, where $|u| + |v| = 1$. It is straightforward to see that any rational graph can be generated by a normalized transducer.

In order to present the LBM that we construct, we simply give *moves* corresponding to obvious sequences of ordinary LBM transitions. Let $M = (Q, W, \triangleright, F, \mathcal{R})$ be a LBM, where W contains the elements of Σ, the states of T, ε and left and right end-markers respectively denoted by $\vartriangle$ and $\triangledown$. The elements of Q are not described in details, we only need to specify two macro states (allowing to initiate a *move*): $\triangleright$ and $\blacktriangleright$, $\triangleright$ being the initial state of the machine. The set F contains a single state ($\Diamond$).

The initial configuration is the following:

$$\vartriangle \triangleright\ w\, \square^{|w|+1}\, \triangledown.$$

There are $|w|+1$ blank symbols after w because the first *transitions* produce this configuration:

$$\vartriangle \blacktriangleright\ \$\, i_{w(0)}\, \varepsilon\, i_{w(1)} \cdots\, i_{w(|w|)}\, \varepsilon\, \triangledown.$$

For this configuration, each symbol $i_{w(k)}$ is the initial state of the transducer corresponding to the letter $w(k)$ (denoted by $T_{w(k)}$). In each configuration of this machine, the even positions correspond to states of the transducers (the machine has to simulate $|w|$ transducers). For example, let us suppose that $A\ q_a\ B\ q_b\ C$ is a factor of some configuration of the machine, the letter A corresponds to the left hand side of a transition in transducer T_a *starting* from q_a, B corresponds to the right hand-side of some transition in T_a *ending* in state q_a (it can be interpreted as: *transducer T_a has produced B and has to consume A*). It is the same for state q_b. There are three different moves of the machine:

The machine checks for success each time state $\blacktriangleright$ reaches $\#$ followed by $\triangledown$ (if $\triangledown$ follows any other non-ε letter there is no transition, the run fails). It also checks for success if $\blacktriangleright$ reaches ε followed by a f_{w_i} (a final state of T_{w_i}). In those cases, the machine checks whether for all i, q_{a_i} equals f_{a_i} and A_i equals ε; if it is the case, it is a success. Indeed, it means that everything

Label	Transducer	Initial position	Final position	Comment
Move (a)	$q_a \xrightarrow[T_a]{A/\varepsilon} q'_a$	$\cdots \blacktriangleright A\, q_a\, C\, q_b \cdots$	$\cdots \varepsilon\, q'_a \blacktriangleright C\, q_b \cdots$	$A, C \in \Sigma \cup \{\$, \#, \varepsilon\}$ q_a state of T_a q_b state of T_b or ∇
Move (b)	$q_a \xrightarrow[T_a]{\varepsilon/B} q'_a$	$\cdots \blacktriangleright A\, q_a\, \varepsilon\, q_b \cdots$	$\cdots A\, q'_a \blacktriangleright B\, q_b \cdots$	$A \in \Sigma \cup \{\$, \#, \varepsilon\}$ q_a state of T_a q_b state of T_b or ∇
Move (c)	—	$\cdots B\, q_b \blacktriangleright C\, q_a \cdots$	$\cdots \blacktriangleright B\, q_b\, C\, q_a \cdots$	$C = \varepsilon$ **or** $q_a = \nabla$ and $C = \#$

that has been produced has been consumed, and that each transducer is in an acceptable state (a final state). If there is no success, the machine proceeds to move (a), (b) or (c).

Lemma 3.2. *The languages recognized by the machine M is the trace, $L(G, \$, \#)$, of G.*

Proof. First, we prove that $L(M) \subseteq L_G$. Consider a word $w \in L(M)$. From a successful run in M, we can deduce paths in the transducers corresponding to the letters of w: all moves done by the machine can be done by a transducer, except those to the left which correspond to a "change of transducer".

Second, we prove that $L_G \subseteq L(M)$. Let w be a word in L_G, and suppose that $n = |w|$. There is a path in G between $\$$ and $\#$ labelled w: $\$ \xrightarrow{w_1} u_1 \xrightarrow{w_2} u_2 \cdots u_{n-1} \xrightarrow{w_n} \#$. Therefore we have: $\$\, T_{w_1} u_1$, $u_1 T_{w_2} u_2$, $\cdots$ $u_{n-1} T_{w_n} \#$.

To construct a successful run of M, we use, for all i, a path in T_{w_i} labelled u_{i-1}/u_i. We define a new transducer T'_{w_i} as a copy of the transducer T_{w_i}, where each ε is replaced by a letter E (not in Σ). Thus $u_{i-1} T_{w_i} u_i$ implies $u'_{i-1} T'_{w_i} u''_i$ with:

$$u'_{i-1} = E^{k_1} u_{i-1}(1) E^{k_2} \cdots E^{k_{|u_{i-1}|}} u_{i-1}(|u_{i-1}|) E^{k_{|u_{i-1}|+1}}.$$

Each E means that a transition labelled ε on the left, in T_{w_i} has been followed. The word u''_i is similar. Each letter in u'_{i-1} witnesses for a transition in T_{w_i}, and therefore corresponds to a letter in u''_i (thus, for all i, $|u'_{i-1}| = |u''_i|$).

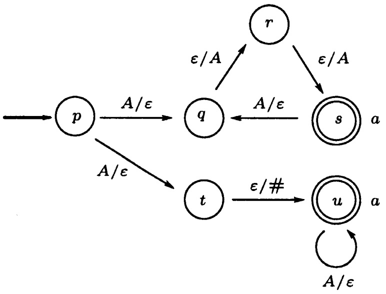
Now we use these words u'_i to construct a successful run in M. The function: `first`, over words, returns the first letter of a word (nothing if it is the empty word), and the function `tail` erases the first letter of a word. This process constructs a successful run of M:

```
Set $i := 1$ /* index of the transducer */
Set $A_1 :=$ \$ and, for all $i \geqslant 2$, $A_i := \varepsilon$
Repeat
    Case:
      $u''_i = \varepsilon$              :Follow corresponding move (c)
                                $i := i - 1$
      first $(u'_{i-1}) = E$:$A_{i+1}$ :=first($u''_i$) (first($u''_i$) $\neq E$)
                                tail($u'_{i-1}$), tail($u''_i$), $i := i + 1$
                                Follow corresponding move (b)
      first $(u'_{i-1}) = A_i$:$A_i := \varepsilon$
                                tail($u'_{i-1}$), tail($u''_i$), $i := i + 1$
                                Follow corresponding move (a)
      Else                      :$i := i - 1$, follow corresponding move (c)
Until (For all $i$, $u'_i = u''_i = \varepsilon$)
```

From the construction of the u'_i and u''_i, this process will always be able to follow a transition. Since all transitions to the right remove letters, the process eventually meets the "out" condition and therefore succeeds. This process yields a path in M, recognizing w which concludes the proof. □

This construction is illustrated for the graph of Example 3.1.

Example 3.3. The first step consist of normalizing the transducer, and to separate initial and final states.



Once the transducer is transformed, we consider the trace of the graph from A to # (the vertex A correspond to the vertex \$ in the construction of the machine). Now, let us consider the word a^3 which labels following path:

$$A \xrightarrow{a} A^2 \xrightarrow{a} A^4 \xrightarrow{a} \#$$

The configurations of the machine are presented on the left. Internal states corresponding to the process are presented on the right (we omit the initial configuration $\triangle\triangleright$ $aaa\square\square\square\square\triangledown$):

$$\triangle\blacktriangleright Ap\varepsilon p\varepsilon p\varepsilon\triangledown \qquad \begin{array}{lll} u'_0 = AEE & u'_1 = AEEAEE & u'_2 = AEAAA \\ u''_1 = EAA & u''_2 = EAAEAA & u''_3 = E\#EEE \\ i := 1 & & \end{array}$$

Apply move (a)

$$\triangle\ \varepsilon q\blacktriangleright \varepsilon p\varepsilon p\varepsilon\triangledown \qquad \begin{array}{lll} u'_0 = EE & u'_1 = AEEAEE & u'_2 = AEAAA \\ u''_1 = AA & u''_2 = EAAEAA & u''_3 = E\#EEE \\ & i := 2 & \end{array}$$

Apply move (c)

$$\triangle\blacktriangleright \varepsilon q\varepsilon p\varepsilon p\varepsilon\triangledown \qquad \begin{array}{lll} u'_0 = EE & u'_1 = AEEAEE & u'_2 = AEAAA \\ u''_1 = AA & u''_2 = EAAEAA & u''_3 = E\#EEE \\ i := 1 & & \end{array}$$

Apply move (b)

$$\triangle\ \varepsilon r\blacktriangleright Ap\varepsilon p\varepsilon\triangledown \qquad \begin{array}{lll} u'_0 = E & u'_1 = AEEAEE & u'_2 = AEAAA \\ u''_1 = A & u''_2 = EAAEAA & u''_3 = E\#EEE \\ & i := 2 & \end{array}$$

Apply moves (a),(c),(b)

$$\triangle\ \varepsilon r\varepsilon r\blacktriangleright Ap\varepsilon\triangledown \qquad \begin{array}{lll} u'_0 = E & u'_1 = EAEE & u'_2 = AEAAA \\ u''_1 = A & u''_2 = AEAA & u''_3 = E\#EEE \\ & & i := 3 \end{array}$$

Apply moves (a),(c),(b)

$$\triangle\ \varepsilon r\varepsilon r\varepsilon u\blacktriangleright \#\triangledown \qquad \begin{array}{lll} u'_0 = E & u'_1 = EAEE & u'_2 = AAA \\ u''_1 = A & u''_2 = AEAA & u''_3 = EEE \\ & & i := 4 \end{array}$$

Apply moves (c),(c),(b),(a), then (c),(c),(c)

$$\triangle\blacktriangleright \varepsilon r\varepsilon s\varepsilon u\#\triangledown \qquad \begin{array}{lll} u'_0 = E & u'_1 = AEE & u'_2 = AA \\ u''_1 = A & u''_2 = EAA & u''_3 = EE \\ i := 1 & & \end{array}$$

Finally, following these moves: (a,b,c,a,b,c,c,a,b), the process finishes. We have computed a successful path from the transducer.

From Lemma 3.2 it is easy to prove the desired result.

Proposition 3.4. *Traces of rational graphs are context-sensitive languages.*

Proof. First, we transform the graph in order to consider the trace between two vertices. Then we construct the corresponding machine M which recognizes the same language,by Lemma 3.2. Thus the traces of rational graphs are Csl. □

4 From context-sensitive languages to synchronized graphs

In the previous section, we proved that the traces of rational graphs are the context-sensitive languages. Thus any trace of a synchronized graph is a

context-sensitive language. Conversely, we show that any context-sensitive language is the trace of a synchronized graph. The proof uses Penttonen's characterization of Csl. It is a detailed construction of [19].
Let L be a context-sensitive language. We construct a synchronized graph whose traces between two finite sets is L.
By Theorem 2.12, there exists a 2-system Γ such that L is obtained by derivation of the linear language L_{lin}. Recall that the derivation rules of non-terminal words are of the form $AB \to AC$. Consider a transducer having transitions $(A,B)\xrightarrow{A/C}(A,C)$ for each (AB,AC) of Γ_2. Any derivation $AB_1\xrightarrow{\Gamma_2}AB_2\ldots\xrightarrow{\Gamma_2}AB_m$ of a word of length 2 corresponds to an arc $A^m \to B_1B_2\ldots B_m$ on the graph. Following this idea, we first get a rational synchronized graph G_{Lin} such that $L = L(G_{Lin}, L_{Lin}, \{\epsilon\})$. Then, we transform G_{Lin} into a graph G having a rational set of vertices L_{Rat} such that $L = L(G, L_{Rat}, \{\epsilon\})$. Finally, using Lemma 2.9, we obtain finite initial and final sets of states.

4.1 Traces from the linear language L_{Lin}

Let T_2 be the transducer defined from Γ_2 by:

$$\begin{array}{llll}
I \xrightarrow{[A/[B} (B,A,B) & \text{for all } A,B\in\Sigma & & \text{(type 1)}\\
(A,B,C) \xrightarrow{B/D} (A,B,D) & \text{for all } A,B,C,D\in\Sigma & & \\
 & \text{such that } BC \underset{\Gamma_2}{\longrightarrow} BD & & \text{(type 2)}\\
(A,B,C) \xrightarrow{D/C} (A,D,C) & \text{for all } A,B,C,D\in\Sigma & & \text{(type 3)}\\
(A,B,C) \xrightarrow{]A/]} F & \text{for all } A,B,C\in\Sigma & & \text{(type 4)}
\end{array}$$

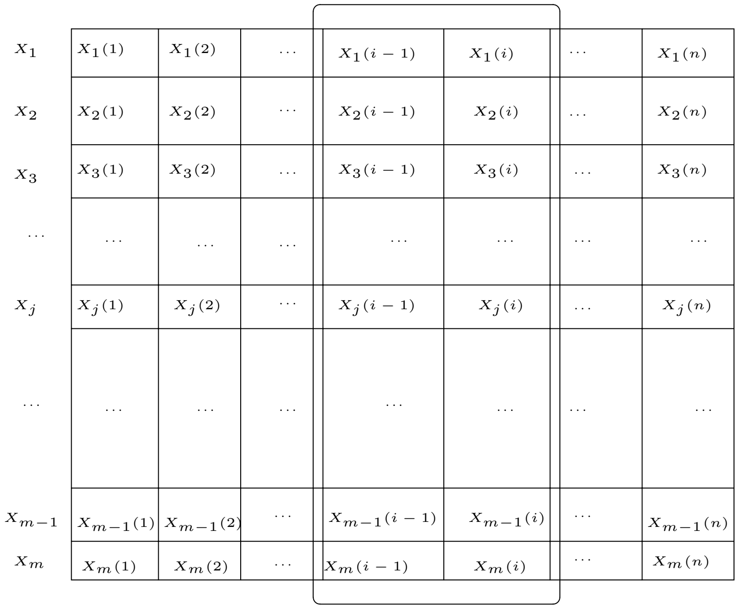
This transducer starting at I and ending at F recognizes pairs of the form
$$([AA_1\ldots A_m]B, [BB_1\ldots B_m])$$
meaning that under the successive contexts $A, A_1, \ldots, A_m$ the letter B can be rewritten successively $B, B_1, \ldots, B_m$. If the context does not change: $A_i = A_{i+1}$, one can apply a rule $A_iB_i \underset{\Gamma_2}{\longrightarrow} A_{i+1}B_{i+1}$. If the context changes: $A_i \neq A_{i+1}$, we copy the letter $B_i = B_{i+1}$. The first component of states of T_2 stores the first word of the derivation.
Note that the relation R_2 recognized by T_2 is of bounded length difference.

Example 4.1. Let $\Gamma_2 = \{(AB,AC),(AC,AD),(DA,DE),(EA,EE)\}$. We have $[AAA]B\ R_2\ [BCD]$ because under the context A, letter B can be rewritten to C and then to D. The following derivation:
$$ABAA \underset{\Gamma_2}{\longrightarrow} ACAA \underset{\Gamma_2}{\longrightarrow} ADAA \underset{\Gamma_2}{\longrightarrow} ADEA \underset{\Gamma_2}{\longrightarrow} ADEE$$
is represented as follows:

A	B	A	A
A	C	A	A
A	D	E	A
A	D	E	E

We have $[AAAAA]B\ R_2\ [BCDDD]$ and $[BCDDD]A\ R_2\ [AAAEE]$ and $[AAAEE]A\ \ R_2\ \ [AAAAE]$.

Consider a word $X_1 \in L_{Lin}$ of length n and a derivation $X_1 \underset{\Gamma_2}{\longrightarrow} X_2 \underset{\Gamma_2}{\longrightarrow} \ldots \underset{\Gamma_2}{\longrightarrow} X_m$ represented by the following figure.



x_1	$x_1(1)$	$x_1(2)$	$\cdots$	$x_1(i-1)$	$x_1(i)$	$\cdots$	$x_1(n)$
x_2	$x_2(1)$	$x_2(2)$	$\cdots$	$x_2(i-1)$	$x_2(i)$	$\cdots$	$x_2(n)$
x_3	$x_3(1)$	$x_3(2)$	$\cdots$	$x_3(i-1)$	$x_3(i)$	$\cdots$	$x_3(n)$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
x_j	$x_j(1)$	$x_j(2)$	$\cdots$	$x_j(i-1)$	$x_j(i)$	$\cdots$	$x_j(n)$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
x_{m-1}	$x_{m-1}(1)$	$x_{m-1}(2)$	$\cdots$	$x_{m-1}(i-1)$	$x_{m-1}(i)$	$\cdots$	$x_{m-1}(n)$
x_m	$x_m(1)$	$x_m(2)$	$\cdots$	$x_m(i-1)$	$x_m(i)$	$\cdots$	$x_m(n)$

The transducer T_2 produces pairs corresponding to m successive letters of adjacent positions: Given the m successive letters at a position i, it yields the m successive letters at position $i+1$.
For any words $X, Y \in \Sigma^*$ of length n, we denote by $X \bigtriangleup Y$ the cardinal of $\{\ 1 \leq i \leq n \mid X(i) \neq Y(i)\ \}$. The following technical lemma states that any two consecutive columns are recognized by T_2.

Lemma 4.2. *The two following properties are equivalent:*
a) $X_1 \underset{\Gamma_2}{\longrightarrow} X_2 \underset{\Gamma_2}{\longrightarrow} \ldots \underset{\Gamma_2}{\longrightarrow} X_m$
b) $[X_1(i-1)X_2(i-1)\ldots X_m(i-1)]X_1(i)\ R_2\ [X_1(i)\ldots X_m(i)]$ *for all* $2 \leq i \leq |X_1|$
and $|X_{j-1}| = |X_j|$ *and* $X_{j-1} \bigtriangleup X_j = 1$ *and* $X_{j-1}(1) = X_j(1)$ *for all* $2 \leq j \leq m$.

Proof. **i)** By definition of Γ_2, we have, for all $2 \leq j \leq m$,

$$|X_{j-1}| = |X_j| \text{ and } X_{j-1} \triangle X_j = 1 \text{ and } X_{j-1}(1) = X_j(1)\,.$$

We show that

$$[X_1(i-1)X_2(i-1)\ldots X_m(i-1)]X_1(i) \;\; R_2 \;\; [X_1(i)\ldots X_m(i)]$$

by induction on $m \geq 1$.
Basis case : $m = 1$. For all $2 \leq i \leq |X_1|$, we have

$$[X_1(i-1)]X_1(i) \; R_2 \; [X_1(i)]$$

considering the path

$$I \xrightarrow[T_{\Gamma_2}]{[X_1(i-1)/[X_1(i)} (X_1(i), X_1(i-1), X_1(i)) \xrightarrow[T_2]{]X_1(i)/]} F$$

Inductive case : $m \Longrightarrow m+1$.
Suppose the implication for a derivation of length m and let $X_1 \underset{\Gamma_2}{\longrightarrow} \ldots \underset{\Gamma_2}{\longrightarrow} X_m \underset{\Gamma_2}{\longrightarrow} X_{m+1}$.
There exists $2 \leq k \leq |X_1|$ such that $X_m(k) \neq X_{m+1}(k)$ and for all $i \neq k,\; X_m(i) = X_{m+1}(i)$.
Let $2 \leq i \leq |X_1|$. We want to show that

$$[X_1(i-1)\ldots X_m(i-1)X_{m+1}(i-1)]X_1(i) \;\; R_2 \;\; [X_1(i)\ldots X_{m+1}(i)]$$

By inductive hypothesis, we have

$$[X_1(i-1)\ldots X_m(i-1)]X_1(i) \;\; R_2 \;\; [X_1(i)\ldots X_m(i)]$$

By definition of the transducer T_2, we have

$$I \overset{[X_1(i-1)\ldots X_m(i-1)/[X_1(i)\ldots X_m(i)}{\underset{T_2}{\Longrightarrow}} (X_1(i), X_m(i-1), X_m(i))$$

We distinguish the two complementary cases below.
Case 1 : $i \neq k$. Then $X_m(i) = X_{m+1}(i)$ and we add an arc of type 3.

$$(X_1(i), X_m(i-1), X_m(i)) \xrightarrow[T_2]{X_{m+1}(i-1)/X_{m+1}(i)}$$
$$(X_1(i), X_{m+1}(i-1), X_{m+1}(i))$$

Case 2 : $i = k$. We have the rule $X_m(i-1)X_m(i) \; \Gamma_2 \; X_{m+1}(i-1)X_{m+1}(i)$.
The following arc of type 2 is associated to previous rule:

$$(X_1(i), X_m(i-1), X_m(i)) \xrightarrow[T_2]{X_{m+1}(i-1)/X_{m+1}(i)}$$
$$(X_1(i), X_{m+1}(i-1), X_{m+1}(i))$$

Finally, we add the arc leading to the final state:

$$(X_1(i), X_{m+1}(i-1), X_{m+1}(i)) \xrightarrow[T_2]{]X_1(i)/]} F$$

We get the result for $m+1$ and the direct implication.

ii) Conversely, we prove that (b) $\Longrightarrow$ (a).
Suppose that $[X_1(i-1)\ldots X_m(i-1)]X_1(i)\ R_2\ [X_1(i)\ldots X_m(i)]$ for all $2 \le i \le |X_1|$
and $|X_{j-1}| = |X_j|$ and $X_{j-1} \triangle X_j = 1$ and $X_1(j-1) = X_1(j)$ for all $2 \le j \le m$.
Let $2 \le j \le m$. Let us show that $X_{j-1} \xrightarrow[\Gamma_2]{} X_j$.
As $X_{j-1} \triangle X_j = 1$, there exists a unique $2 \le k \le |X_1|$ such that $X_{j-1}(k) \neq X_j(k)$.
Moreover $X_j(1) = X_j(1)$ thus $k \neq 1$ and $X_{j-1}(k-1) = X_j(k-1)$.
We have $[X_1(k-1)\ldots X_m(k-1)]X_1(k)\ R_2\ [X_1(k)\ldots X_m(k)]$.
By definition of the transducer T_2, the following arc exists

$$(X_1(k), X_{j-1}(k-1), X_{j-1}(k)) \xrightarrow[T_2]{X_j(k-1)/X_j(k)} (X_1(k), X_j(k-1), X_j(k))$$

This arc is of type 2 and gives the existence of the following rule of Γ_2

$$X_{j-1}(k-1)X_{j-1}(k) \longrightarrow X_j(k-1)X_j(k)$$

Thus for any $2 \le j \le m$, $X_{j-1} \xrightarrow[\Gamma_2]{} X_j$ i.e. a) holds. $\square$

The transducer T_2 recognizes arcs of the form $[U]A \to [AV]$. In order to create paths on the graph, we add to T_2 the set of transitions $\{F \xrightarrow{A/A} F \mid A \in \Sigma\}$. New arcs are of the form $[U]AW \to [AV]W$ where W is a suffix of the initial word of the derivation. If $X_1 \in L_{Lin}$ with $|X_1| = n$ and $X_1 \xrightarrow[\Gamma_2]{m-1} X_m$, the graph G_{Lin} contains the following path:

$$[X_1(1)^m]X_1(2)\ldots X_1(n) \to [X_1(2)\ldots X_m(2)]X_1(3)\ldots X_1(n)\ \ldots \\ \to [X_m(1)\ldots X_m(n)].$$

It remains to add arcs of the form $[U] \to \varepsilon$ for any word U and to label arcs of G. Since X_m is derived by Γ_1 in a word of L, the last letter of each column gives labels of arcs. Thus, we set $[UA]BW \xrightarrow[G_{Lin}]{a} [BV]W$ for each $a \in \mathcal{A}$ such that $A \xrightarrow[\gamma_1]{} a$. The graph G_{Lin} obtained is left-synchronized graph, and verifies that $L = L(G_{Lin}, L_{Lin}, \{\varepsilon\})$.

4.2 Traces from a rational set

The problem is that L_{Lin} is not rational. In order to reduce L_{Lin} to a rational set, we complete T_2 into a transducer generating words of L_{Lin} successively from left to right.
Let Gr be a grammar in Greibach normal form generating L_{Lin} from a non-terminal S. Each rule of Gr is of the form $Z \rightarrow AW$ where $Z \in \Sigma_r$ is a non-terminal of Gr, $A \in \Sigma$ is a terminal (which is also a non-terminal of Γ) and $W \in \Sigma_r^*$ is a non-terminal word of Gr. Let the transducer

$$T_2' := T_2 \cup \{F \xrightarrow{Z/U} F' \mid (Z,U) \in Gr\} \cup \{F' \xrightarrow{Z/Z} F' \mid Z \in \Sigma_r\},$$

where F' is a new state of the transducer. We denote by R_2' the relation recognized by T_2' from I to F'. This relation is still of bounded length difference. Let

$$L_{Rat} := \{\, [A^m]BW \mid S \xrightarrow[Gr]{2} ABW \wedge A, B \in \Sigma \wedge W \in \Sigma_r^* \wedge m \geq 1 \,\}.$$

Let us reformulate Lemma 4.2 for derivations starting from L_{Lin}.

Lemma 4.3. *Let* $X_1, \ldots, X_m \in \Sigma^*$ *and* $n = |X_1|$.
The two following properties are equivalent:
a) $X_1 \xrightarrow[\Gamma_2]{} X_2 \xrightarrow[\Gamma_2]{} \ldots \xrightarrow[\Gamma_2]{} X_m$ *and* $X_1 \in L_{Lin}$
b) *There exists* $W_1, \ldots, W_{n-1} \in \Sigma_r^*$ *such that*

$$\begin{aligned}
&[X_1(1)\ldots X_m(1)]X_1(2)W_1 \in L_{Rat} \textit{ and } W_{n-1} = \varepsilon \\
&\textit{and } [X_1(n{-}1)\ldots X_m(n{-}1)]X_1(n)R_2[X_1(n)\ldots X_m(n)] \\
&\textit{and } [X_1(i{-}1)\ldots X_m(i{-}1)]X_1(i)W_{i-1}R_2'[X_1(i)\ldots X_m(i)]X_1(i{+}1)W_i \\
&\qquad\qquad\qquad \textit{for all } 2{\leq}i{<}n \\
&\textit{and } |X_{j-1}| = |X_j| \textit{ and } X_{j-1} \triangle X_j{=}1 \textit{ and } X_{j-1}(1){=}X_j(1) \\
&\qquad\qquad\qquad \textit{for all } 2{\leq}j{\leq}m.
\end{aligned}$$

Proof. **i)** We suppose (a) and show (b).
Since $X_1 \in L_{Lin}$, we consider the derivation from S to X_1 according to Gr: there exists non-terminal words $W_1, \ldots, W_{n-2}$ of Gr such that

$$\begin{aligned}
S &\xrightarrow[Gr]{2} X_1(1)X_1(2)W_1 \xrightarrow[Gr]{} \ldots \xrightarrow[Gr]{} X_1(1)\ldots X_1(n-1)W_{n-2} \\
&\xrightarrow[Gr]{} X_1(1)\ldots X_1(n)
\end{aligned}$$

By Lemma 4.2, we have for all $2 \leq i \leq |X_1|$,

$$[X_1(i-1)\ldots X_m(i-1)]X_1(i) \ R_2 \ [X_1(i)\ldots X_m(i)],$$

$|X_{j-1}| = |X_j|$, $X_{j-1} \triangle X_j = 1$ and $X_{j-1}(1) = X_j(1)$ for all $2 \leq j \leq m$.
Let $2 \leq i \leq n-1$. We know that W_i is obtained from W_{i-1} by the rewriting of the non-terminal $W_{i-1}(1)$:

$$W_{i-1} \;=\; ZV \underset{Gr}{\longrightarrow} UV \;=\; X_2(i+1)W_i.$$

We complete the preceding path leading to F with the arc $F \overset{Z/U}{\longrightarrow} F'$ and then with arcs $F' \underset{T_2'}{\overset{Z/Z}{\longrightarrow}} F'$ for V. Thus, we have

$$[X_1(i-1)...X_m(i-1)]X_1(i)W_{i-1} \;\; R_2' \;\; [X_1(i)...X_m(i)]X_1(i+1)W_i.$$

ii) We suppose (b) and show (a).
We cut the paths

$$[X_1(i-1)...X_m(i-1)]X_1(i)W_{i-1} \;\; R_2' \;\; [X_1(i)...X_m(i)]X_1(i+1)W_i$$

which become

$$[X_1(i-1)X_2(i-1)...X_m(i-1)]X_1(i) \;\; R_2 \;\; [X_1(i)...X_m(i)]$$

By Lemma 4.2, we have $X_1 \underset{\Gamma_2}{\longrightarrow} X_2 \underset{\Gamma_2}{\longrightarrow} \ldots \underset{\Gamma_2}{\longrightarrow} X_m$.
By hypothesis $[X_1(1)...X_m(1)]X_1(2)W_1 \in L_{Rat}$ and $X_1(1) = \ldots = X_m(1)$. So $S \underset{Gr}{\overset{2}{\longrightarrow}} X_1(1)X_1(2)W_1$. Thus $S \underset{Gr}{\overset{*}{\longrightarrow}} X_1(1)...X_1(n) \;=\; X_1$ hence $X_1 \in L_{Lin}$. □

The transducer T_2' successively generates letters of X_1. It remains to label arcs of the recognized graph to get a left-synchronized graph such that the language of path labels leading from the rational vertex set L_{Rat} to the final vertex set $\{\varepsilon\}$ is the context-sensitive language defined by Γ. As in Section 4.1, any arc of the form $[UA]BW \to [BV]W$ is labelled $a \in \mathcal{A}$ if $A \underset{\Gamma_1}{\longrightarrow} a$.

Proposition 4.4. *Any context-sensitive language is trace of a synchronized graph.*

Proof. Let L be a context-sensitive language. There exists a 2-system Γ such that

$$L \;=\; \{\, v \in \mathcal{A}^* \mid \exists\, u \in L_{Lin} \,,\; u \underset{\Gamma}{\overset{*}{\longrightarrow}} v \,\}.$$

For all letter $a \in \mathcal{A}$, we denote by

$$\Sigma_a \;:=\; \{\, A \in \Sigma \mid A \underset{\Gamma_1}{\longrightarrow} a \,\}$$

the set of non-terminals generating the terminal a in Γ.
We define the graph G_0 such that for any $a \in \mathcal{A}$,

$$\underset{G_0}{\overset{a}{\longrightarrow}} \;:=\; R_2' \,\cap\, [\Sigma^*\Sigma_a]\Sigma\Sigma_r^* {\times} ([\Sigma^+]\Sigma\Sigma_r^* \cup [\Sigma^+]\Sigma_r^*).$$

Since R'_2 is a bounded length difference relation, so G_0 is by Proposition 2.8 and Theorem 2.7.
In particular, G_0 is left-synchronized and the following graph:

$$G \ := \ G_0 \ \cup \ \bigcup_{a \in \mathcal{A}} [\Sigma^* \Sigma_a] \times \{a\} \times \varepsilon$$

is also left-synchronized since $[\Sigma^* \Sigma_a] \times \{a\} \times \varepsilon$ is recognizable ([10]) for all $a \in \mathcal{A}$.
We recall that

$$L_{Rat} \ := \ \{ \ [A^m] BW \mid S \xrightarrow[Gr]{*} ABW \ \wedge \ m \geq 1 \ \}$$

where S is the axiom of Gr. We have

$$\begin{array}{l}
u \in L \text{ with } |u| = n > 1 \\
\quad \Longleftrightarrow \ \text{(By definition)} \\
\text{there exists} \quad X_1, \ldots, X_m \in \Sigma^* \text{ of length } n \text{ such that} \\
X_1 \in L_{Lin} \text{ and } X_1 \xrightarrow[\Gamma_2]{} X_2 \xrightarrow[\Gamma_2]{} \ldots \xrightarrow[\Gamma_2]{} X_m \\
\text{and } X_m(i) \xrightarrow[\Gamma_1]{} u(i) \text{ for all } 1 \leq i \leq n \\
\quad \Longleftrightarrow \ \text{(by Lemma 4.3)} \\
\text{there exists non-terminal words } W_1, \ldots, W_{n-1} \text{ of } Gr \text{ such that} \\
[X_1(1) \ldots X_m(1)] X_1(2) W_1 \in L_{Rat} \\
\text{and } W_{n-1} = \varepsilon \text{ and such that} \\
|X_1|, [X_1(i) \ldots X_m(i)] X_1(i+1) W_i \\
\xrightarrow[G_0]{u(i)} [X_1(1) \ldots X_m(1)] X_1(2) W_1 \\
\xrightarrow[G_0]{u(1)} [X_1(2) \ldots X_m(2)] X_1(3) W_2 \\
\xrightarrow[G_0]{u(2)} \ldots \xrightarrow[G_0]{u(n-1)} [X_1(n) \ldots X_m(n)] \\
\text{and } X_m(n) \in \Sigma_{u(n)} \\
\quad \Longleftrightarrow \ \text{(By definition)} \\
u \in L(G, L_{Rat}, \{\varepsilon\})
\end{array}$$

Thus

$$L \ = \ L(G, L_{Rat}, \{\varepsilon\}) \ \cup \ \{ \ u \in L \mid |u| \leq 1 \ \}$$

and it remains to apply Lemma 2.9. □

This leads to the main result of this paper:

Theorem 4.5. *Context-sensitive languages are exactly the traces of synchronized graphs between finite sets.*

Proof. Any synchronized graph is a rational graph, hence any trace of a synchronized graph is a context-sensitive language by Proposition 3.4. Proposition 4.4 ensures the converse. □

4.3 Letter to letter graphs

Using Lemma 2.9, the previous section defined Csl as traces of synchronized graphs from and to finite sets of vertices. In this section, we study traces with initial and final rational sets. Provided this extension, the traces of letter to letter graphs are Csl.
Indeed, the synchronized relation of bounded length difference R'_2, we have used in the proof of Proposition 4.4, can be completed into a letter-to-letter relation.

Lemma 4.6. *Let $R \subseteq \Sigma^* \times \Sigma^*$ be a left-synchronized relation and let $\diamond$ be a symbol such that $\diamond \notin \Sigma$. We can transform R into a letter-to-letter relation R_l such that*
$\forall (U, V) \in \Sigma^* \times \Sigma^*, \forall n \geq 0,$

$$(U \underset{R}{\xrightarrow{n}} V) \iff (\exists k \geq 0, \exists k' \geq 0 \text{ such that } U\diamond^k \underset{R_l}{\xrightarrow{n}} V\diamond^{k'})$$

Proof. Let T be a left-synchronized transducer recognizing R. The transducer T_l is built from T by replacing each arc of the form $p \xrightarrow{\epsilon/A} q$ (respectively $p \xrightarrow{A/\epsilon} q$) with $A \in \Sigma$ by the arc $p \xrightarrow{\diamond/A} q$ (respectively $p \xrightarrow{A/\diamond} q$). Then for each final vertex f of T, we create a new final state f' of T_l and add the arcs $f \xrightarrow{\diamond/\diamond} f'$ and $f' \xrightarrow{\diamond/\diamond} f'$. □

Let us reformulate Proposition 4.4.

Proposition 4.7. *Any context-sensitive language is the language $L(G, L_{Rat}, F_{Rat})$ of path labels leading from a rational set of vertices L_{Rat} to another F_{Rat} and where G is a letter-to-letter rational graph.*

Proof. Using Proposition 2.8 we get that R'_2 is a left-synchronized relation. Let $\diamond$ be a symbol such that $\diamond \notin \Sigma \cup \Sigma_r$. Using Lemma 4.6, R'_2 is completed into a letter-to-letter relation R_l. The result is obtained by adapting the proof of Proposition 4.4 with

$$\underset{G_0}{\xrightarrow{a}} := R_l \cap \; [\Sigma^* \Sigma_a] \Sigma \Sigma_r^* \diamond^* \times ([\Sigma^+] \Sigma \Sigma_r^* \diamond^* \cup [\Sigma^+] \diamond^*)$$

$$G := G_0 \;\cup\; \bigcup_{a \in \mathcal{A}} \{\, [UA]\diamond^k \xrightarrow{a} \$^{|[UA]|+k} \mid U \in \Sigma^* \;\wedge\; A \in \Sigma_a \,\}$$

$$L_{Rat} := \{\, [A^m]BW\diamond^k \mid S \underset{Gr}{\xrightarrow{2}} ABW \;\wedge\; m \geq 1 \;\wedge\; k \geq 0\}$$

and

$$F_{Rat} := \$^+ \qquad \square$$

5 Conclusion

In this paper, we have established a connection between context-sensitive languages and rational graphs. We have been able to prove that the traces of these graphs are context-sensitive languages, and that the context-sensitive languages are traces of letter-to-letter rational graphs with initial and final rational sets. The proof of the latter result relies on the Penttonen normal form for context-sensitive languages, it is indeed possible to avoid the use of this form: this has been done by Carayol [2] and Meyer [13], those proofs adapt our construction to produce a rational graph from a linearly bounded Turing machine.

Our result might give an interesting approach to Kuroda's conjecture [12]: do the deterministic context-sensitive languages (i.e., generated using a deterministic LBM) coincide with context-sensitive languages? An easier question would be to characterize the traces of deterministic rational graphs. This question is still unsolved.

References

1. Blumensath, A., Grädel, E. (2000) Automatic structures. Proceedings of 15th IEEE Symposium on Logic in Computer Science LICS 2000, pp. 51–62
2. Carayol, A. (2001) Notions of determinism for rational graphs. Private communication
3. Caucal, D. (2003) On the transition graphs of Turing machines. Theoretical Computer Science 296(2): 195–223
4. Caucal, D. (2003) On transition graphs having a decidable monadic theory. Theoretical Computer Science 290(1): 79–115
5. Chomsky, N. (1956) Three models for the description of language. IRE Transactions on Information Theory 2(3): 113–124
6. Colcombet, T. (2002) On families of graphs having a decidable first order theory with reachability. Proceedings of the 29th International Conference on Automata, Languages, and Programming, vol. 2380 (Lecture Notes in Computer Science). Springer, Berlin Heidelberg New York
7. Courcelle, B. (1990) Handbook of Theoretical Computer Science, chap. Graph rewriting: an algebraic and logic approach. Elsevier
8. Elgot, C., Mezei, J. (1965) On relations defined by generalized finite automata. IBM 9: 47–68

9. Epstein, D., Cannon, J.W., Holt, D.F., Levy, S.V.F., Paterson, M.S., Thurston (1992) Word processing in groups. Jones and Barlett publishers
10. Frougny, C., Sakarovitch, J. (1993) Synchronized rational relations of finite and infinite words. Theoretical Computer Science 108: 45–82
11. Kleene, S.C. (1956) Representation of events in nerve nets and finite automata. In: Shannon, C.E., McCarthy, J. (eds.) Automata studies. Princeton, pp. 3–40
12. Kuroda, S.-Y. (1964) Classes of languages and linear-bounded automata. Information and Control 7(2): 207–223
13. Meyer, A. (2002) Sur des sous-familles de graphes rationnels. Rapport de DEA, Université de Rennes 1
14. Morvan, C. (2000) On rational graphs. In: Tiuryn, J. (ed.) Fossacs 00, vol. 1784 (LNCS), pp. 252–266. ETAPS 2000 best theoretical paper Award
15. Morvan, C., Stirling, C. (2001) Rational graphs trace context-sensitive languages. In: Pultr, A., Sgall, J. (eds.) MFCS 01, vol. 2136 (LNCS), pp. 548–559
16. Muller, D., Schupp, P. (1985) The theory of ends, pushdown automata, and second-order logic. Theoretical Computer Science 37: 51–75
17. Penttonen, M. (1974) One-sided and two-sided context in formal grammars. Information and Control 25(4): 371–392
18. Rispal, C. (2001) Graphes rationnels synchronisés. Rapport de DEA, Université de Rennes 1
19. Rispal, C. (2002) Synchronized graphs trace the context-sensitive languages. In: Mayr, R., Kucera, A. (ed.) Infinity 02, vol. 68. (ENTCS)
20. Greibach, S., Ginsburg, S. (1969) Abstract families of languages. Mem. Am. Math. Soc. 87
21. Sénizergues, G. (1992) Definability in weak monadic second-order logic of some infinite graphs. In: Dagstuhl seminar on Automata theory: Infinite computations, Warden, Germany, vol. 28, p. 16
22. Urvoy, T. (2002) Abstract families of graphs. In: Toyama, M., Ito, M. (ed.) DLT 02, vol. 2450. (LNCS), pp. 381–392