Properties of NLC Graph Languages squeezed with Bipartite Graphs

Dong Hoon Lee

Department of Computer Sciences, Korea University
Seochang 208, Chochiwon, Yongi-gun, Chung-Nam, 339-700, KOREA
e-mail: donghlee@kusccgx.korea.ac.kr

Abstract: For a graph grammar G and a graph-theoretical property ψ, let $L_\psi(G)$ be the *language* $L(G)$ *squeezed with* ψ, i.e., $L_\psi(G) = L(G) \cap L(\psi)$. This paper presents results on language-theoretic properties (such as closure, membership, and other decision properties) of NLC, B-NLC, and NU-NLC languages squeezed with bipartite graphs. It turns out that the family of NLC languages is not closed under squeezing with bipartite graphs. We also show that for each NLC (B-NLC) grammar G, the language squeezed with bipartite graphs, $L_{bip}(G)$, is in PSPACE (NP) and there is an NLC (B-NLC) grammar G' such that $L_{bip}(G')$ is PSAPCE-complete (NP-complete). We further show that for each NU-NLC grammar G, the set of all complete bipartite graphs in $L(G)$ is in NSPACE($\log n$).

Properties of NLC Graph Languages squeezed with Bipartite Graphs

Dong Hoon Lee

Department of Computer Sciences, Korea University
Seochang 208, Chochiwon, Yongi-gun, Chung-Nam, 339-700, KOREA
e-mail: donghlee@kusccgx.korea.ac.kr

グラフ文法 G とグラフ理論的性質ψについて，$L_\psi(G)$ をψで圧縮された言語 $L(G)$，すなわち $L_\psi(G) = L(G) \cap L(\psi)$ であるとする．本文では 2 部グラフで圧縮された NLC, B-NLC，NU-NLC 言語の閉包性，所属やその他の決定的性質などの言語理論的性質を示した．それによって NLC 言語の族は 2 部グラフによる圧縮については閉じてないことがわかる．また，各 NLC(B-NLC) 文法 G について，2 部グラフで圧縮された言語 $L_{bip}(G)$ は PSPACE(NP) であり，$L_{bip}(G')$ が PSPACE-完全 (NP-完全) である NLC(B-NLC) 文法 G'が存在することを示し，さらに各 NU-NLC 文法 G について $L(G)$ の中の全ての完全 2 部グラフの集合は NSPACE($\log n$) であることを示す．

1 Introduction

Graph grammars describe graphs by replacing a graph by a graph in a derivation step. Within the theory of graph grammars, we are usually interested in describing a set of graphs with a certain graph-theoretical property ψ (such as planarity, connectedness, bounded degreeness, and so on) by a grammar G, so that we can decide whether or not an arbitrary graph possesses the property ψ by solving the membership problem for G. From a language-theoretical point of view, we are also interested in whether or not a grammar G generates a graph with the property ψ, finitely many such graphs, only such graphs. We refer to [4, 8-10, 21] for various applications and approaches of the theory of graph grammars.

One particular graph-grammar model studied intensively during the past decade is the node-label-controlled (NLC) grammars [16-18], where in one step a single node is replaced by a graph and the embedding of the introduced graph into the existing graph is based on node labels only. Many variations of NLC grammars have been studied in the literature. Examples are boundary NLC (B-NLC) grammars [23-25] in which no two nonterminal nodes are allowed to be adjacent in any sentential form and neighborhood-uniform NLC (NU-NLC) grammars [19] in which each node of the right-hand side of a production is connected either to all neighbors of the replaced node or to none.

For a graph-theoretical property ψ, let $L(\psi)$ be the set of all graphs satisfying ψ and, for a graph grammar G, let $L_\psi(G)$ be the *language $L(G)$ squeezed with ψ*, i.e., $L_\psi(G) = L(G) \cap L(\psi)$ [17]. Properties of squeezed graph languages have been studied much in the literature. Janssens and Rozenberg [18] studied decision problems of various squeezed NLC languages. Rozenberg and Welzl [23] showed that, for a B-NLC grammar G and a positive integer k, the membership in $L(G)$ of connected graphs of maximal degree at most k can be decided in polynomial time. Rozenberg and Welzl [24] further studied the closure properties of B-NLC languages squeezed with various classes of graphs. Courcelle [5] extended many of the results in [24] by showing that, for a *confluent* NLC grammar G (in which derived graphs are invariant under the order of applications of production rules) and a graph-theoretical property ψ expressible in monadic second-order logic, a grammar G' of the same type as G such that $L(G') = L_\psi(G)$ can be effectively constructed and the decision problems "$L_\psi(G) = \emptyset$?" and "$L(G) = L_\psi(G)$?" are decidable. (Confluent NLC grammars include, e.g., B-NLC and NU-NLC grammars. Graphs expressible in monadic second-order logic include, e.g., trees, connected graphs, planar graphs, bipartite graphs, and cyclic graphs.) Engelfriet and Leih [12, 13] showed that the membership problem for Lin-eNCE (B-eNCE) grammars generating connected graphs of bounded degree is in NLOG (LOGCFL). Lautemann [22] showed that the membership problem for hyperedge replacement systems generating connected graphs of bounded degree is in LOGCFL. (A hyperedge replacement system generates hypergraphs by replacing a hyperedge by a hypergraph.) Aalbersberg et al. [1] considered various restrictions on regular DNLC grammars and on graph languages generated by them with the goal of classifying the membership complexity.

We present results on the properties of node replacement graph languages (in particular, NLC, B-NLC, and Lin-NLC languages) squeezed with bipartite graphs. The paper is organized as follows. Section 2 contains basic definitions on graphs and NLC graph grammars. Section 3 discusses closure properties. It is known that the family of B-NLC languages is closed under squeezing with (the properties of being) bipartite graphs [5]. We show that the family of NLC languages is not closed under squeezing with bipartite graphs. Section 4 discusses membership complexity and undecidability results for other decision problems. We show that, for each NLC grammar G, $L_{bip}(G)$ is in PSPACE (NP) and there is an NLC (B-NLC) grammar G' such that $L_{bip}(G')$ is PSAPCE-complete (NP-complete). We further show that for each NU-NLC grammar G, the set of all complete bipartite graphs in $L(G)$ is in NSPACE(log$n$). Our NLOG result is not covered by other efficient recognition algorithms shown in the literature since NU-NLC grammars can generate degree-unbounded bipartite graphs.

2 Preliminaries

The definitions relating to graph-grammar theory are mostly from [16]. In the sequel, the empty set is denoted by $\emptyset$ and, for a finite set A, the cardinality of A is denoted by $\#A$. For a string x, the length of x is denoted by $|x|$.

A (node-labeled undirected) *graph* is a system $H = (V, E, \Sigma, \phi)$, where V is a finite set of *nodes*, E is a set of two-element subsets of V (the set of *edges*), Σ is a finite set of *node labels*, and ϕ: $V \rightarrow \Sigma$ is a *node-labeling* *function*. For convenience, the different components of H are denoted by V_H, E_H, Σ_H, and ϕ_H. H is called a *graph over* Σ; the set of all graphs over Σ is denoted by GR_Σ.

Let H be a graph. Two nodes v, w in H are *neighbors* (or *adjacent*) if $v, w \in E_H$. The *degree* of a node v in H is the number of nodes adjacent to v. H is *degree-bounded* if each node of H is of degree at most k, for some fixed $k \geq 0$. A node in H is a *leaf* if it is adjacent to exactly one node. A sequence $p = (v_1, v_2, \ldots, v_r), r \geq 2$, of distinct nodes in V_H is a *path* (of length $r-1$) between v_1 and v_r if v_i and v_{i+1} are neighbors for each $i = 1, 2, \ldots, r-1$. If p is a path of length at least two, then a sequence $(v_1, \ldots, v_r, v_{r+1})$ is a *cycle* if v_r and v_{r+1} are neighbors and $v_1 = v_{r+1}$. H is *connected* if there is a path between each pair of nodes in H; otherwise, H is *disconnected*. H is a *tree* if it is a connected and does not contain a cycle. H is a *chain* if it is a tree with two leaves. H is a bipartite graph if its node set can be partitioned into V_1 and V_2 so that $E_H \subseteq V_1 \times V_2$; H is a *complete bipartite graph* if for each $v_1 \in V_1$ and for each $v_2 \in V_2$, $(v_1, v_2) \in E_H$. H is a *discrete graph* if $V_H \neq \emptyset$ and $E_H = \emptyset$. H is the *empty graph*, denoted by Λ, if $V_H = \emptyset$.

A *graph language* is any subset of GR_Σ, where Σ is an alphabet of node labels. For a graph-theoretical property ψ, the set of all graphs in GR_Σ satisfying ψ is denoted by $L_\Sigma(\psi)$, or simply by $L(\psi)$ if the underlying alphabet Σ is understood.

A *node-label-controlled (NLC) grammar* [16] is a system $G = (\Sigma, \Delta, P, C, S)$, where Σ is an alphabet of *node labels*, $\Delta \subseteq \Sigma$ is an alphabet of *terminal node labels* (the labels in $\Sigma - \Delta$ are *nonterminal node labels*), P is a finite set of *productions* of the form (X, Y) where $X \in \Sigma - \Delta$ and $Y \in G_\Sigma$, C is a subset of $\Sigma \times \Sigma$ (the *embedding relation*), and $S \in \Sigma - \Delta$ is the *initial nonterminal*.

A production $\pi = (X, Y)$ of and NLC grammar $G = (\Sigma, \Delta, P, C, S)$ is applied to a node v of a graph $H \in GR_\Sigma$, with $\phi_H(v) = X$, as follows. We replace the node v and all of its incident edges in H by (an isomorphic copy of) the graph Y and add an edge between a node x in Y and a (former) neighbor y of v if and only if $(\phi_Y(x), \phi_H(y)) \in C$. This results in a graph $K \in GR_\Sigma$ and we write $H \Rightarrow_{(v,\pi)} K$ (or simply $H \Rightarrow K$). A graph $H \in GR_\Sigma$ such that $S \Rightarrow^* H$ is called a *sentential form* of G. The set of all sentential forms of G is denoted by $I(G)$. The *language generated by* G is $L(G) = I(G) \cap GR_\Delta$.

An NLC grammar G is a *boundary NLC (B-NLC) grammar* [23] if no two nonterminal nodes are adjacent in the right-hand side of any of its productions. G is a *linear NLC (Lin-NLC) grammar* [11] if the right-hand side of each production contains at most one nonterminal node. A graph language L is an *NLC (B-NLC, or Lin-NLC) language* if there is an NLC (B-NLC, or Lin-NLC) grammar G such that $L = L(G)$. The Lin-NLC class is a proper subclass of the B-NLC class [12], which is in turn a proper subclass of the NLC class [23]. Each NLC (B-NLC, or Lin-NLC) language not containing Λ can be generated by an NLC (B-NLC, or Lin-NLC) grammar with no Λ – *production* (i.e., a production whose right-hand side is Λ) [7, 14]. We shall assume throughout this paper that each grammar is in such a normal form unless stated otherwise.

Let G be a graph grammar of any type and let ψ be a graph-theoretical property. The *language* $L(G)$ *squeezed with* ψ, denoted by $L_\psi(G)$, is the set $L(G) \cap L(\psi)$. In particular, the set of all bipartite graphs in $L(G)$ is denoted by $L_{bip}(G)$.

3 Closure Properties

Let ψ be a graph-theoretical property. A family of graph languages is *closed under squeezing with* ψ if each of its languages, when squeezed with ψ, is again in the family.

Closure under squeezing has been considered often in the literature. Rozenberg and Welzl [24] showed that B-NLC languages are effectively closed under squeezing with several graph-theoretical properties, such as connectedness, k-colorableness, and nonplanarity. Courcelle [5] presented several metatheorems on this property. In particular, B-NLC languages are effectively closed under squeezing with graph-theoretical properties that can be expressed in monadic second-order logic. As the property of a graph being a bipartite graph can be expressed in monadic second-order logic [6], the closure result in [5] yields the following theorem:

Theorem 3.1. *The family of B-NLC languages is effectively closed under squeezing with bipartite graphs.*

Theorem 3.2. *The family of NLC languages is not* *closed under squeezing with bipartite graphs.*

Proof. We shall define an NLC grammar G_0 such that $L_{bip}(G_0)$ is not an NLC language. In fact, each bipartite graph in $L(G_0)$ is a complete bipartite graph, and so, the proof works also for complete bipartite graphs.

Let $\pi = \{a_0, a_1, a_2\}$ and $\pi' = \{b_0, b_1, b_2\}$ be alphabets of node labels. Let f be a mapping on nonnegative integers, defined by $f(k) = k \bmod 3$, $k \geq 0$. For each $i \geq 0$, let T_i be the complete binary tree of depth i such that

1. for each j, $0 \leq j < i$, all nodes at the level j (the root is at the level 0) are labeled by $a_{f(j)}$ and
2. all nodes at the bottom level (the level i) are labeled by $b_{f(i)}$.

For each $i \geq 0$, let B_i be a bipartite graph obtained from T_i by deleting all edges, adding a node labeled by $\#$, and finally establishing an edge between the new node and every other node. So B_i has 2^{i+1} nodes. The tree T_2 and the bipartite graph B_2 are shown in Fig. 1 and Fig. 2, respectively.

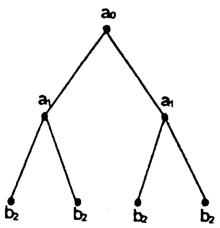

Fig. 1. The graph T_2.

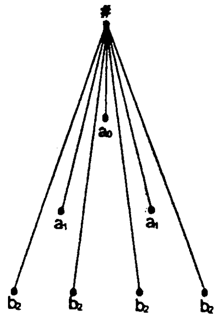

Fig. 2. The graph B_2

Let $L_0 = \{B_i | i \geq 0\}$. For each NLC grammar G, there is a context-free string grammar G' such that $L(G') = \{a_n | n = \#V_H$ for some $H \in L(G)\}$ [14, Lemma16]. Note that $\#V_{B_i} = 2^{i+1}$ for each $i \geq 0$. Clearly, the set $\{a^{2^{i+1}} | i \geq 0\}$ is not a context-free string language. Therefore, L_0 is not an NLC language. We shall show, however, that there is an NLC grammar G_0 such that $L_{bip}(G_0)$ is exactly L_0.

For each $k \in \{0, 1, 2\}$, let $\sigma(k) = (k+1) \bmod 3$ and $\rho(k) = (k+2) \bmod 3$. Let $\gamma_0, \gamma_1, \gamma_2$ be the graphs shown in Fig. 3. Let $G_0 = (\Sigma, \Delta, P, C, S)$, where

$$
\begin{aligned}
\Sigma &= \{S, \#\} \cup \pi \cup \pi' \cup \{A_i \mid i = 0, 1, 2\},\\
\Delta &= \pi \cup \pi' \cup \{\#\},\\
P &= \{(S, {}^{\#}\bullet\!\!-\!\!\bullet^{A_0})\} \cup (A_i, \gamma_i), (A_i, \bullet^{b_i}) | i = 0, 1, 2\},\\
C &= \Sigma \times \Sigma - \{(a_i, a_i), (a_i, A_{\sigma(i)}), (b_i, b_i), (A_i, a_{\rho(i)})\\
&\quad | i = 0, 1, 2\}.\\
S &= \bullet^{S}.
\end{aligned}
$$

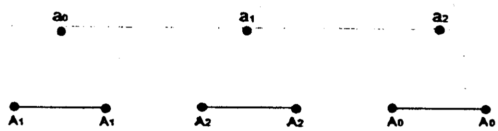

Fig. 3. The graph γ_0, γ_1, and γ_2.

For each $i \geq 0$, let B_i' be the graph obtained from B_i by replacing the label $b_{f(i)}$ of each leaf by the nonterminal $A_{f(i)}$ and by adding edges between leaves to make the subgraph of B_i' induced by the leaves of B_i a complete graph. For example, B_2' is shown in Fig. 4.

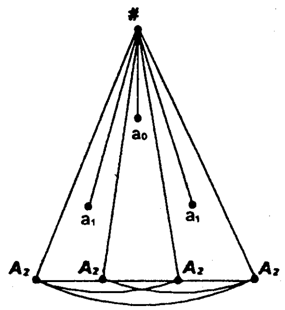

Fig. 4. The graph B_2'.

Claim1. $L_0 \subseteq L_{bip}(G_0)$.

Proof of Claim 1. We first show, by an induction on $i \geq 0$, that $B_i' \in I(G_0)$. The induction basis is obviously true. Assume that $B_i' \in I(G_0)$ for all $i \leq n$. Then, B_{n+1}' can be obtained by applying the production $(A_{f(n)}, \gamma_{f(n)})$ to each node in B_n' labeled by $A_{f(n)}$. Appropriate connections between nodes in the new graph are guaranteed due to the embedding relation C. Note that, in any intermediate graph between

B_n' and B_{n+1}', each nonterminal node is adjacent to all other nonterminal nodes and to all terminal nodes at the same level. Now, each B_i, $i \geq 0$, can be derived from B_i' by applying the production $(A_{f(i)}, \bullet^{b_{f(i)}})$ to each node in B_i' labeled by $A_{f(i)}$. Again, appropriate connections between nodes are guaranteed by the embedding relation C. So, Claim 1 holds.

Claim2. $L_{bip}(G_0) \subseteq L_0$.

Proof of Claim 2. Let $X \in L_{bip}(G_0)$. Let $D = (X_0, X_1, \ldots, X_t)$, where $X_0 = \bullet^S$ and $X_t = X$, be a derivation for X in G_0. We first show that $X_{2^i} = B_i'$ for each $i \geq 0$ such that $2^i < t$. This will be done by an induction on i. The induction basis is clearly true. Assume that $X_{2^i} = B_i'$ for all $i \leq n$. We need to prove that, if $2^{n+1} < t$, then $X_{2^{n+1}} = B_{n+1}'$. Let $2^{n+1} < t$. By the induction hypothesis, $X_{2^n} = B_n'$. There are 2^n nonterminal nodes in B_n'. We claim that, in the subderivation $X_{2^n} \Rightarrow^{2^n} X_{2^{n+1}}$ of D, the 2^n productions used are exactly the production $(A_{f(n)}, \gamma_{f(n)})$ applied to each nonterminal node of B_n'. (This claim yields $X_{2^{n+1}} = B_{n+1}'$, as shown in the proof of Claim 1.) Suppose to the contrary that the claim were not true. Then there are two cases:

1. the production $(A_{f(n)}, \bullet^{b_{f(n)}})$ is applied to some nonterminal node of B_n', or

2. the production $(A_{f(n+1)}, \gamma_{f(n+1)})$ or $(A_{f(n+1)}, \bullet^{b_{f(n+1)}})$ is applied to a node created by the application of $(A_{f(n)}, \gamma_{f(n)})$ to a node of B_n'.

Consider the first case. Let the production $(A_{f(n)}, \bullet^{b_{f(n)}})$ be applied to a nonterminal node of B_n'; call this node v after being (rewritten and) relabeled by $b_{f(n)}$. As $2^{n+1} < t$, there is at least one nonterminal node of B_n' to which the production $(A_{f(n)}, \gamma_{f(n)})$ is applied. Let w be any node labeled by $a_{f(n)}$ introduced by such a rewriting. As $(b_{f(n)}, A_{f(n)})$, $(a_{f(n)}, A_{f(n)})$, $(b_{f(n)}, a_{f(n)})$, and $(a_{f(n)}, b_{f(n)})$ are all in C, the nodes v and w are adjacent in $X_{2^{n+1}}$. As $(b_{f(n)}, \#)$ and $(a_{f(n)}, \#)$ are in C, v, w and $\bullet^{\#}$ make a cycle of length three. As the nodes in this cycle are labeled by terminals, no edge in the cycle can be removed in the subsequent derivation steps. So, X is not a bipartite graph, a contradiction.

Now, consider the second case. Namely, a node (say v) at the level $n+1$, labeled by $A_{f(n+1)}$, is rewritten before a node (say w) at the level n, labeled by $A_{f(n)}$, is rewritten. Clearly, v and w are neighbors. The node v will be relabeled by $a_{f(n+1)}$ or $b_{f(n+1)}$ by the rewriting; call it $\overline{v}$ after being relabeled. The node $\overline{v}$ is adjacent to w before w is rewritten, since $(a_{f(n+1)}, A_{f(n)})$ and $(b_{f(n+1)}, A_{f(n)})$ are in C. Later, when w is rewritten, it is relabeled by $a_{f(n)}$ or $b_{f(n)}$; call this new terminal node $\overline{w}$. Note that $(a_{f(n)}, a_{f(n+1)})$, $(a_{f(n)}, b_{f(n+1)})$, $(b_{f(n)}, a_{f(n+1)})$, and $(b_{f(n)}, b_{f(n+1)})$ are all in C. That is, the nodes $\overline{v}$ and $\overline{w}$ are adjacent in $X_{2^{n+1}}$. For the same reason as in the first case, $\overline{v}$, $\overline{w}$, and $\bullet^{\#}$ are in a cycle of length three that cannot be removed in the subsequent derivation steps. So, X is not a bipartite graph, a contradiction.

Let m be the maximum integer such that $2^m < t$. Then, $X_{2^m} = B_m'$. The graph B_m' contains 2^m nonterminal nodes. In going from X_{2^m} to X_t, we must apply the production $(A_{f(m)}, \bullet^{b_{f(m)}})$ exactly 2_m times, once for each nonterminal node of B_m' since, otherwise, m would not be the maximum integer such that $2^m < t$. (Note that X_t does not contain nonterminal nodes.) The reader can easily check that the graph $X_t (= X)$ is B_m, and so, $X \in L_0$. This completes the proof of Claim 2. ♣

4 Decision Problems

4.1 Membership problems

We consider the membership problem for graph languages squeezed with bipartite graphs, i.e., the question "$X \in L_{bip}(G)$?", where X is a graph and G is a (fixed) graph grammar. We shall assume that the input graph X is given by its adjacency list, encoded into a string. Our goal in this section is to determine the upper and lower bounds of this problem for NLC grammars and their restrictions and develop efficient recognition algorithms when possible.

Lemma 4.1. *Let Σ be an alphabet. The set of all bipartite graphs in GR_Σ is in NLOG.*

Proof. Let X be an arbitrary graph in GR_Σ. A cycle of odd length in X, if any, can be guessed on log space. As NLOG is closed under complementation [15], nonexistence of cycle of odd length in X can also be tested on log space. ♣

Theorem 4.2. *For every NLC grammar G, $L_{bip}(G)$ is in PSPACE. There is an NLC grammar G' such that $L_{bip}(G')$ is PSPACE-complete.*

Proof. For each NLC grammar G, $L(G)$ is in PSPACE[3]. $L_{bip}(G) \in$ PSPACE from this and Lemma 4.1. It is known in [2] that there is a context-sensitive string grammar for which the membership problem is PSPACE-complete. Let G_0' be such a grammar. There exists an algorithm which constructs an NLC grammar G_0 such that every connected graph X derived by G_0 is a "chain graph" of the form

where $a_i \neq \#$ for all i, $1 \leq i \leq n$, such that $\{W(X) \mid X \text{ is a connected graph in } L(G_0)\} = L(G_0')$, where $W(X) = a_1 a_2 \cdots a_n$ [17]. We can easily transform an input instance w of $L(G_0')$ into a chain graph X, an instance of $L_{bip}(G_0)$. Clearly, $w \in L(G_0')$ if and only if $X \in L_{bip}(G_0)$. ♣

Theorem 4.3. *For every B-NLC grammar G, $L_{bip}(G)$ is in NP. There is a Lin-NLC grammar G' such that $L_{bip}(G')$ is NP-complete.*

Proof. For each B-NLC grammar G, $L(G) \in$ NP [23]. $L_{bip}(G) \in$ NP from this and Lemma 4.1. In [1, Theorem 3.5(2)], a so-called regular directed NLC grammar $\overline{G'}$ was constructed, where $L(\overline{G'})$ contains bipartite graphs only and $L(\overline{G'})$ is NP-complete. By removing edge directions from the right-hand sides of the productions of $\overline{G'}$, we can obtain a Lin-NLC grammar G' satisfying the second statement of the theorem. ♣

An NLC grammar $G = (\Sigma, \Delta, P, C, S)$ is a *neighborhood-uniform NLC (NU-NLC) grammar* [19] if, for each $d \in \Sigma$, either $d \times \Sigma \subseteq C$ (d is a *connecting label*) or $(d \times \Sigma) \cap C = \emptyset$ (d is a *disconnecting label*).

The set of all complete bipartite graphs over an alphbet Δ can be generated by an NU-NLC grammar. Consider the NU-NLC grammar $G = (\Sigma, \Delta, P, C, S)$ such that

$$\begin{aligned} \Sigma &= \{S,\ A,\ B\} \cup \Delta, \\ P &= \{(S,\ {}^{A}\bullet\!\!-\!\!\bullet^{A}),\ (A,{}^{A}\bullet\ \bullet^{A}),\ (B,{}^{B}\bullet\ \bullet^{B}), \\ &\quad (A,\ \bullet^{\sigma})\ (B,\ \bullet^{\sigma})\ \mid\ \sigma \in \Delta\}, \\ C &= \Sigma \times \Sigma. \end{aligned}$$

A typical derivation in G is shown in Fig. 5, where $\sigma_i \in \Delta$ for each $i \in \{1,\ 2, \ldots, k\}$. Starting with the last sentential form in this derivation, each A-labeled node can just disappear or can be rewritten by using a process similar to the one as shown in Fig. 5. It should not be difficult to see that $L(G)$ is the set of all complete bipartite graphs in GR_{Δ}.

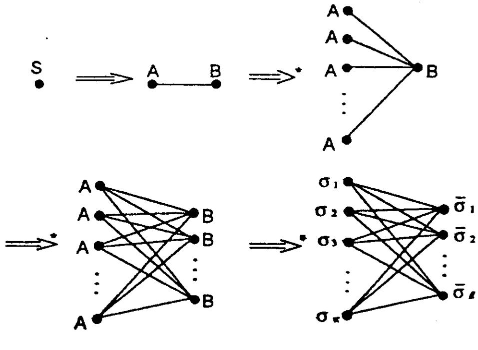

FIG. 5. A typical derivation in G.

NU-NLC grammars are confluent and, as a result, NU-NLC grammars have many nice combinatorial and language-theoretical properties [5, 17, 19]. (NU-NLC grammars are called context-free NLC grammars in [17].) In particular, many decision problems, which are undecidable for NLC grammars, are decidable for NU-NLC grammars.

We consider NU-NLC languages squeezed with complete bipartite graphs, and present their efficient recognition algorithms. (Our consideration of this class has been motivated by the NP-completeness result stated in Theorem 4.3.) In general, complete bipartite graphs is degree-unbounded. Thus, our results have merits compared with other efficient recognition algorithms for squeezed graph languages, that mostly contain graphs of bounded degree only [1, 12, 13, 22, 23]. Our results are in the style of the results in [1], i.e., efficient recognition has been obtained by imposing structural restrictions (NU) on NLC grammars.

Theorem 4.4. *For each NU-NLC grammar G, the set of all complete bipartite graphs in $L(G)$, denoted by $L_{cbip}(G)$, is in NSPACE($\log n$).*

Proof. Let $G = (\Sigma,\ \Delta,\ P,\ C,\ S)$ be an NU-NLC grammar. Suppose that $X \in L_{cbip}(G)$ and let $D = (X_0,\ X_1, \ldots, X_n), X_0 = \bullet^S$, be a derivation for X in G. It is obvious that an odd cycle in any sentential form cannot disappear in further derivation. So, each X_i is bipartite. We show that each X_i is a complete bipartite graph. For each X_i, let V_i^1 and V_i^2 be the partition of V_{X_i}. Let $v_1 \in V_i^1$ and $v_2 \in V_i^2$ be nodes such that (v_1, v_2) is an edge of X_i (note that each X_i is connected because X_n is connected). Suppose to the contrary that there is a node v_3 such that $(v_1, v_3), (v_2, v_3) \notin E_{X_i}$. For each $j \in \{1, 2, 3\}$, let u_j be a node in X_n such that $u_j = v_j$ if v_j is a terminal node or u_j is a node generated from v_j if v_j is a nonterminal node. Then, u_3 is adjacent to none of u_1 and u_2 in X_n. This is a contradiction to the fact that X_n is a complete bipartite graph and hence u_3 is adjacent to u_1 or u_2. This proves the claim.

Let A be the set of all complete bipartite graphs in GR_Σ and let H be a graph in A. Let V_H^1, V_H^2 be a partition of nodes of H and let $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$, for $k = \#\Sigma$. Then, H can be fully encoded up to isomorphism by a string $\mu(H) \in \Sigma^*\$\Sigma^*$ defined by

$$\mu(H) = \sigma_1^{l_1}\sigma_2^{l_2}\cdots\sigma_k^{l_k}\$\sigma_1^{r_1}\sigma_2^{r_2}\cdots\sigma_k^{r_k},$$

where for each $1 \leq i \leq k$, $l_i(r_i)$ is the number of nodes labeled σ_i in $V_H^1(V_H^2$, respectively).

Let ρ be any efficient encoding function that transforms a string $\mu(H) \in \Sigma^*\$\Sigma^*$ into another string of nonnegative integers, defined by

$$\rho(\mu(H)) = l_1 l_2 \cdots l_k \$ r_1 r_2 \cdots r_k.$$

Let L_G be the set of all strings of the form $\rho(\mu(H))$, where there exist graphs $H_i \in A, 1 \leq i \leq n$, such that (1) H_1 is a single node labeled by S, (2) $H_i \Rightarrow H_{i+1}$ for $1 \leq i \leq n-1$, and (3) $H_n = H$. It is not difficult to see that L_G can be accepted by a nondeterministic linear-bounded automaton, say M_G.

Now, let X be an input graph. The membership of X in $L_{cbip}(G)$ can be tested by checking if there is a partition (V_X^1, V_X^2) of the node set of X such that the string $\rho(\mu(X))$ corresponding to it is in L_G. Such a partition can be guessed on log space and the transformation from X into $\rho(\mu(X))$ can be done on log space. Run M_G on $\rho(\mu(X))$. This needs $|\rho(\mu(X))|$ space, which is log space with respect to the size of $|X|$. ♣

4.2 Other decision problmes

We consider language-theoretic decision problems other than the membership problem, such as emptiness, finiteness, equivalence of graph languages squeezed with bipartite graphs.

Theorem 4.5(**Courcelle** [5]). *For a B-NLC grammar G and a graph-theoretical property ψ expressible in monadic second-order logic, the problem of whether or not (1) $L_\psi(G) = \emptyset$, (2) $L_\psi(G)$ is finite, and (3) $L_\psi(G)$ is of bounded degree.*

The problems of whether or not $L(G)$ is empty and whether or not $L(G)$ is finite can be easily seen to be decidable for an NLC grammar by the known methods for context-free string grammars. The problem of whether or not $L(G)$ is of bounded degree is decidable for an NLC grammar G [20]. These problems are undecidable when squeezed with bipartite graphs.

Theorem 4.6. *It is undecidable for an NLC grammar G whether or not (1) $L_{bip}(G) = \emptyset$, (2) $L_{bip}(G)$ is finite, and (3) $L_{bip}(G)$ is of bounded degree.*

Proof. We reduce the problem of whether or not an NLC language contains a discrete graph, which is undecidable [18], to the problems stated in the theorem. (The proof for the result in [18] is quite complicated.) Let $G = (\Sigma, \Delta, P, C, S)$ be an NLC grammar. Let $\overline{\#}$, $\#$ be symbols not in Σ. Let G' be an NLC grammar defined by

$$\begin{aligned} G' \;=\; & (\Sigma \cup \{\overline{\#}, \#\}, \Delta \cup \{\#\}, P', \\ & C \cup ((\Sigma \cup \{\overline{\#}, \#\}) \times \{\#\}), S'), \end{aligned}$$

where $P' = P \cup \{(S', \bullet\!\overset{S}{-\!-}\!\bullet\!\overset{\#}{-\!-}\!\bullet^{\overline{\#}}), (\overline{\#}, {}^{\#}\bullet \;\; \bullet^{\overline{\#}}), (\overline{\#}, \bullet^{\#})\}$. It is easy to see that $L(G)$ contains a discrete graph if and only if $L_{bip}(G') \neq \emptyset$ if and only if $L_{bip}(G')$ is infinite if and only if $L_{bip}(G')$ is of unbounded degree. ♣

References

[1] IJ. J. Aalbersberg, J. Engelfriet and G. Rozenberg, "The complexity of regular DNLC graph languages," *J. Comput. System Sci.* Vol. 40, pp. 376-404, 1990.

[2] R. V. Book, "Comparing complexity classes," *J. Comput. System Sci.* Vol. 9, pp. 213-229, 1974.

[3] F. J. Brandenberg, "On polynomial time graph grammars," *STACS* 88 (R. Cori and M. Wirsing, Eds.), Lecture Notes in Computer Science 294, 227-236, 1988.

[4] V. Claus, H. Ehrig, and G. Rozenberg (Eds.), *Graph Grammars and Their Application to Computer Science and Biology*, Lecture Notes in Computer Science 73 (Springer-Verlag, Berlin, 1979).

[5] B. Courcelle, "An axiomatic definition of context-free rewriting and its application to NLC graph grammars," *Theoret. Comput. Sci.* Vol. 55, pp. 141-181, 1987.

[6] B. Courcelle, "Graph rewriting: an algebraic and logic approach," in *Handbook of Theoretical Computer Science* (J. V. Leeuwen, Ed.), Vol. B, Chap. 5, Elsevier, Amsterdam, 1990.

[7] A. Ehrenfeucht, M. Main, and G. Rozenberg, "Restrictions on NLC graph grammars," *Theoret. Comput. Sci.* Vol. 31, pp. 211-223, 1984.

[8] H. Ehrig, H.-J. Kreowski and G. Rozenberg (Eds.), *Graph Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science 532 (Springer-Verlag, Berlin, 1991).

[9] H. Ehrig, M. Nagl, and G. Rozenberg (Eds.), *Graph Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science 153 (Springer-Verlag, Berlin, 1983).

[10] H. Ehrig, M. Nagl, G. Rozenberg, and A. Rosenfeld (Eds.), *Graph Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science 291 (Springer-Verlag, Berlin, 1987).

[11] J. Engelfriet and G. Leih, "Nonterminal Bounded NLC graph grammars," *Theoret. Comput. Sci.* Vol. 59, pp. 309-315, 1988.

[12] J. Engelfriet and G. Leih, "Linear graph grammars: power and complexity," *Inform. Comput.* Vol. 81, pp. 88-121, 1989.

[13] J. Engelfriet and G. Leih, "Complexity of boundary graph languages," *Theoretical Informatics and Applications* Vol. 24, pp. 267-274, 1990.

[14] J. Engelfriet, G. Leih, and E. Welzl, "Boundary graph grammars with dynamic edge relabeling," *J. Comput. System Sci.* Vol. 40, pp. 307-345, 1990.

[15] N. Immerman, "Nondeterministic space is closed under complementation," *SIAM J. Comput.* Vol. 17, pp. 935-938, 1988.

[16] D. Janssens and G. Rozenberg, "On the structure of node-label-controlled graph languages," *Inform. Sci.* Vol. 20, pp. 191-216, 1980.

[17] D. Janssens and G. Rozenberg, "Restrictions, extensions, and variations of NLC grammars," *Inform. Sci.* Vol. 20, pp. 217-244, 1980.

[18] D. Janssens and G. Rozenberg, "Decision problems for node label controlled graph grammars," *J. Comput. System Sci.* Vol. 22, pp. 144-177, 1981.

[19] D. Janssens and G. Rozenberg, "Neighborhood-uniform NLC grammars," *Computer Vision, Graphics, and Image Processing* Vol. 35, pp. 131-151, 1986.

[20] D. Janssens, G. Rozenberg, and E. Welzl, "The bounded degree problem for NLC grammars is decidable," *J. Comput. System Sci.* Vol. 33, pp. 415-422, 1986.

[21] H.-J. Kreowski and G. Rozenberg, "On structured graph grammars. I and II," *Inform. Sci.* Vol. 52, pp. 185-246, 1990.

[22] C. Lautemann, "The complexity of graph languages generated by hyperedge replacement," *Acta Inform.* Vol. 27, pp. 399-421, 1990.

[23] G. Rozenberg and E. Welzl, "Boundary NLC graph grammars - Basic definitions, normal forms, and complexity," *Inform. Control* Vol. 69, pp. 136-167, 1986.

[24] G. Rozenberg and E. Welzl, "Graph theoretic closure properties of the family of boundary NLC graph languages," *Acta Inform.* Vol. 23, pp. 289-309, 1986.

[25] G. Rozenberg and E. Welzl, "Combinatorial properties of boundary NLC graph languages," *Discrete Applied Math.* Vol. 16, pp. 59-73, 1987.