TESTING FOR GRAMMATICAL COVERINGS

D.J. ROSENKRANTZ*† and H.B. HUNT, III**

Computer Science Department, State University of New York at Albany, Albany, NY 12222, U.S.A.

Communicated by M.A. Harrison
Received May 1983
Revised September 1984

Abstract. For any given binary relation ρ defined on the context-free grammars, there is the associated computational problem of determining, for a pair of grammars (G, H), if $G \rho H$. We study the complexity of this problem for a number of grammatical similarity relations whose definitions involve mappings between the symbols of related grammars. The relations considered include Reynolds covering, weak Reynolds covering, onto grammar homomorphism, grammar isomorphism, interpretation of grammar forms, and weak interpretation of grammar forms.

A single general theorem is used to show that the computational problem associated with each of these grammatical relations, except grammar isomorphism, is NP-complete. In contrast, deterministic polynomial time algorithms are presented for testing if $G \rho H$, when H is structurally unambiguous and the relation ρ is Reynolds covering, onto grammar homomorphism, or grammar isomorphism. These results provide a rare example of a nontrivial natural algebraic and/or combinatorial structure, namely the unambiguous context-free grammars, with polynomial time algorithms for homomorphism, onto homomorphism, and isomorphism.

We also show that the grammar isomorphism problem is polynomially equivalent to the graph isomorphism problem.

1. Introduction

A variety of binary relations on context-free grammars have been defined in the literature. These relations model some concept of similarity between two grammars. Besides their intrinsic mathematical interest, concepts of grammatical similarity are relevant to compiler construction. Often one grammar is used as a reference grammar to which semantic specifications are added. But a second grammar is used for parsing. These two grammars are called the *semantic grammar* and the *parsing grammar* respectively. The reason for using two grammars is as follows. The semantic grammar is a more natural basis to which semantic specifications can be added. But it may not belong to a given class of grammars having an efficient parsing method.

* The first author's research was supported in part by the National Science Foundation under Grants MCS 78-03157 and MCS 82-03237.

† Present affiliation: Phoenix Data Systems, Albany, NY, U.S.A.

** The second author's research was supported in part by the National Science Foundation under Grant MCS 80-03353.

A survey of grammatical similarity relations appears in [22]. The most extensively studied ways of precisely defining a similarity relationship between the semantic and parsing grammars involve mappings between productions or between nonterminals. Relationships defined in terms of a map between productions are called *coverings* [11, 17]. In this paper we study covering-type relationships that are based on maps between grammatical symbols. Such relationships include Reynolds covering, weak Reynolds covering, onto grammar homomorphism, grammar isomorphism, interpretation of grammar forms, and strict interpretation of grammar forms.

For any grammatical similarity relation ρ, there is the associated computational problem of determining, for a given pair of grammars (G, H), if $G \rho H$. In applications to compilers, this is the problem of determining if a semantic grammar and a possible parsing grammar are appropriately related. In this paper we study the complexity of such problems.

In Section 2 we give definitions and notation.

In Section 3 we show that, for a number of grammar relations, the set of pairs of related grammars is NP-complete. These problems are NP-complete even when the grammars are regular grammars. The relations considered include Reynolds covering, weak Reynolds covering, onto grammar homomorphism, skeletal grammars, interpretation, and strict interpretation. A single theorem and proof apply to many different relations. Consequently, our results are insensitive to many possible variations in the definitions of particular similarity relations. This is desirable since it is often not obvious just what the formal definition corresponding to an intuitive concept of 'grammars with closely related structure' should be. We also present results relevant to grammar forms and their associated grammatical families [5]. In contrast to the NP-completeness results in Section 3, testing a pair of grammars for left covering or right covering [11] is undecidable for context-free grammars and is PSPACE-complete for regular grammars [14].

In Section 4 we present deterministic polynomial time algorithms for testing for Reynolds covering, onto grammar homomorphism, and grammar isomorphism when the parsing grammar, i.e., H, is structurally unambiguous. Thus, since the class of structurally unambiguous context-free grammars contains the class of unambiguous context-free grammars, there is a deterministic polynomial time algorithm for testing for Reynolds covering, onto grammar homomorphism, and grammar isomorphism when the proposed parsing grammar is a member of most of the grammar classes corresponding to parsing methods used in compilers [1, 16]. Thus, although testing for Reynolds covering and for onto grammar homomorphism are likely to be hard in general, there are efficient algorithms for the cases of practical interest.

In Section 5 we show that testing for grammar isomorphism and for several related problems are polynomially equivalent to the graph isomorphism problem.

A preliminary version of some of the results in Sections 3 and 5 has appeared in [12], and a preliminary version of some of the results in Section 4 has appeared in [13].

2. Definitions and notation

In this section, a number of definitions needed to read this paper are presented. We begin with definitions concerning binary relations.

Definition 2.1. Let A be a nonempty set. Let σ and ρ be binary relations on A. We say that relation σ *includes* relation ρ if and only if, for all x, y in A, $x \rho y$ implies $x \sigma y$.

Definition 2.2. Let A be a nonempty set. Let ρ, σ, and τ be binary relations on A. We say that relation σ is *between* relations ρ and τ if and only if σ includes ρ and τ includes σ.

We assume that the reader is familiar with the basic definitions and results concerning context-free grammars and parsing; otherwise, see [1].

We denote the empty word by λ.

Definition 2.3. A *context-free* grammar $G=(N, \Sigma, P, S)$ is a four-tuple, where N and Σ are disjoint finite sets of *nonterminals* and *terminals* respectively, the *start symbol* S is an element of N, and P, the set of *productions*, is a finite subset of $N \times (N \cup \Sigma)^*$.

Productions are written in the form $A \to \alpha$ rather than (A, α). In the remainder of this paper, we will use 'grammar' as an abbreviation for 'context-free grammar'.

Definition 2.4. A *regular grammar* $G=(N, \Sigma, P, S)$ is a grammar such that if $A \to \alpha$ is a production in P, then α is in $\Sigma \cup \Sigma \bullet N$.

Henceforth, we assume that all grammars are *reduced*, i.e., that each nonterminal occurs in some derivation of a terminal string, since this simplifies some of the algorithms. The reader should note that there are efficient polynomial time algorithms for reducing grammar [1, 16].

Definition 2.5. A *structure* is a derivation tree with all nonterminal labels deleted. Two derivation trees are *structurally equivalent* if they have the same structure. A grammar is *structurally unambiguous* if no two derivation trees generated by the grammar are structurally equivalent.

Definition 2.6. Two productions of a grammar

$$A \to \alpha_1 \alpha_2 \dots \alpha_k \quad \text{and} \quad B \to \beta_1 \beta_2 \dots \beta_m,$$

where each α_i and β_j is a single grammatical symbol, are *compatible* if $k=m$ and, for all i, $1 \leq i \leq k$, α_i and β_i are identical terminal symbol or α_i and β_i are both nonterminal symbols. Two productions are *incompatible* if they are not compatible.

Next, we give the definitions of several grammatical similarity relations involving mappings from nonterminals to nonterminals, but for which terminals are unchanged. These relations include onto grammar homomorphism [11], grammar isomorphism [11], Reynolds cover [18, 11], weak Reynolds cover [18, 11], and skeletal grammars [7]. Note that Reynolds cover corresponds to a grammar homomorphism that need not be onto. This concept was called a homomorphism in [9]. A skeletal grammar string homomorphism [7] is similar to an onto homomorphism, except that if there is a production of the form $A \rightarrow B$ in the semantic grammar, where A and B both map into the same nonterminal of the parsing grammar, then the parsing grammar does not contain the corresponding production (whose left and right side would be identical). We note that Schnorr [20] defined a concept called grammar homomorphism that is more general than the concept of grammar homomorphism considered here.

Definition 2.7. Let $G=(M, \Sigma, P, S)$ and $H=(N, \Sigma, Q, T)$ be grammars. Let f be a homomorphism from $(M \cup \Sigma)^*$ into $(N \cup \Sigma)^*$ such that $f(M) \subset N$ and f is the identity on Σ. Let $f(P)=\{f(A) \rightarrow f(\gamma) \mid A \rightarrow \gamma \text{ is in } P\}$.

(1) We say that f is a *(grammar) homomorphism* from G onto H if $f(S)=T$ and $f(P)=Q$. If in addition f is one-to-one, we say that f is a *(grammar) isomorphism* from G onto H. We say that G and H are *(grammar) isomorphic* if there is a (grammar) isomorphism from G onto H.

(2) We say that G is *Reynolds covered* by H (or H *Reynolds covers* G) if there exists a homomorphism from $(M \cup \Sigma)^*$ into $(N \cup \Sigma)^*$ as above such that $f(S)=T$ and $f(P) \subset Q$.

(3) We say that G is *weak Reynolds covered* by H (or H *weak Reynolds covers* G) if there exists a homomorphism f from $(M \cup \Sigma)^*$ into $(N \cup \Sigma)^*$ as above, such that $f(S)=T$ and, for all productions $A \rightarrow \gamma$ in P,

$$f(A) \overset{*}{\underset{H}{\Rightarrow}} f(\gamma).$$

(4) We say that H is a *skeletal grammar* for G if there exists a homomorphism f from $(M \cup \Sigma)^*$ into $(N \cup \Sigma)^*$ as above, such that $f(S)=f(T)$, and letting $f'(P)$ be $f(P)$ with all productions of the form $C \rightarrow C$ deleted, $f'(P)=Q$.

Next, we give the definitions of several grammatical similarity relations involving more general maps which can involve both nonterminals and terminals. These relations include strict interpretations [6, 10, 15], some special cases of strict interpretations [5], and some generalizations that are useful in expressing the general results in Section 3. Note that a variant definition of strict interpretation appears in [2]. The concept of strict interpretation is applicable to compilers where the semantic grammar is a strict interpretation of the parsing grammar, and several terminal symbols of the semantic grammar are represented by the same terminal symbol of the parsing grammar. These terminal symbols can then be represented as a lexical token with the same class part, but different value parts [16].

Definition 2.8. Let $G=(M, \Sigma, P, S)$ and $H=(N, \Delta, Q, T)$ be grammars. Let μ be a substitution on $(N \cup \Delta)^*$, such that $\mu(a)$ is a finite subset of Σ^* for each a in Δ, and $\mu(A)$ is a subset of M for each A in N. In addition, let μ have the properties that $\mu(A) \cap \mu(B)=\emptyset$ for each A and B in N with $A \neq B$, and that S is in $\mu(T)$.

(1) G is an *interpretation* of H if for each production $A \rightarrow \gamma$ of P there is a production $B \rightarrow \xi$ in Q such that A is in $\mu(B)$ and γ is in $\mu(\xi)$.

(2) G is a *strict interpretation* of H if G is an interpretation of H, $\mu(a)$ is a subset of Σ for each a in Δ, and $\mu(a) \cap \mu(b)=\emptyset$ for each a and b in Δ with $a \neq b$.

(3) G is an *onto strict interpretation* of H if G is a strict interpretation of H and, for each production $B \rightarrow \xi$ in Q, there is a production $A \rightarrow \gamma$ in P such that A is in $\mu(B)$ and γ is in $\mu(\xi)$.

(4) G is an *isomorphic strict interpretation* of H if G is an onto strict interpretation of H and μ is one-to-one.

(5) G is a *generalized strict interpretation* of H if $\mu(a)$ is a subset of Σ for each a in Δ; $\mu(a) \cap \mu(b)=\emptyset$ for each a and b in Δ with $a \neq b$; and for each production $A \rightarrow \gamma$ of P there is a B in N and ξ in $(N \cup \Delta)^*$ such that A is in $\mu(B)$, γ is in $\mu(\xi)$, and $B \underset{H}{\overset{*}{\Rightarrow}} \xi$.

(6) G is a *weak interpretation* of H if either G is an interpretation of H or G is a generalized strict interpretation of H.

An equivalent definition of strict interpretation is that there is a homomorphism f from $(M \cup \Sigma)^*$ into $(N \cup \Delta)^*$ such that $f(M) \subset N$, $f(\Sigma) \subset \Delta$, $f(S)=T$, and $f(P) \subset Q$. Thus, Reynolds covered by, onto homomorphism from, and isomorphism from are special cases of strict interpretation of, onto strict interpretation of, and isomorphic strict interpretation of, respectively, where each terminal maps into itself.

3. Hard covering relations

In this section we study the complexity of determining if a pair of grammars are related by an onto homomorphism, a Reynolds cover, a weak Reynolds cover, an interpretation of grammar forms, a strict interpretation of grammar forms, an onto strict interpretation of grammar forms, or the skeletal grammar relation. A single theorem and proof are presented that imply, for each of these relations, that the problem of determining if a pair of grammars are related by the relation is NP-hard. We also generalize the concept of 'a family of grammars represented by a grammar form' [5, 6] and give simple sufficient conditions for the decidability of the problem of determining if two 'generalized' grammar forms represent the same family of grammars.

Theorem 3.1. *Let* Hom-Onto *and* Weak-Interp *be the binary relations on the set of grammars defined by—for all grammars G and H,*

(1) *G* HOM-ONTO *H if and only if there is a grammar homomorphism from G onto H, and*

(2) *G* WEAK-INTERP *H if and only if G is a weak interpretation of H.*

Let σ be any binary relation on the set of grammars between the relations HOM-ONTO *and* WEAK-INTERP. *Then each of the sets*

$$R_\sigma = \{(G, H) \mid G \text{ and } H \text{ are regular grammars, and } G\,\sigma\,H\}$$

and

$$C_\sigma = \{(G, H) \mid G \text{ and } H \text{ are grammars, and } G\,\sigma\,H\}$$

is NP-*hard.*

Proof. The proof consists in showing that there is a deterministic polynomially time-bounded reduction of the language

CLIQUE $= \{(J, \hat{k}) \mid J$ is an undirected graph with $n \geq 1$ nodes, $\hat{k}$ is the unary numeral for a nonnegative integer $k \leq n$, and J has a clique of size $k\}$

to the sets R_σ and C_σ, for each such binary relation σ on the set of grammars. The set CLIQUE is known to be NP-hard [8].

Let J be an undirected graph with $n \geq 1$ nodes. Let the set of nodes of J be $\{N_i \mid 1 \leq i \leq n\}$. Let k be a nonnegative integer such that $1 \leq k \leq n$. Consider the mapping from $(J, \hat{k})$ to the pair of regular grammars G and H given as follows:

$G = (\{S\} \cup \{A_i \mid 1 \leq i \leq k\} \cup \{N_i \mid 1 \leq i \leq n\}, \{a, b, c\}, P, S)$, where

$P = P_1 \cup P_2 \cup P_3$,

$P_1 = \{S \to a\, A_i \mid 1 \leq i \leq k\} \cup \{S \to a\, N_i \mid 1 \leq i \leq n\}$,

$P_2 = \{A_i \to b\, A_j \mid i \neq j, 1 \leq i, j \leq k\} \cup \{N_i \to b\, N_j \mid$ the graph J contains an edge connecting nodes N_i and $N_j\}$, and

$P_3 = \{A_i \to c \mid 1 \leq i \leq k\} \cup \{N_i \to c \mid 1 \leq i \leq n\}$;

$H = (\{T\} \cup \{N_i \mid 1 \leq i \leq n\}, \{a, b, c\}, Q, T)$, where

$Q = Q_1 \cup Q_2 \cup Q_3$,

$Q_1 = \{T \to a\, N_i \mid 1 \leq i \leq n\}$,

$Q_2 = \{N_i \to b\, N_j \mid$ the graph J contains an edge connecting nodes N_i and $N_j\}$, and

$Q_3 = \{N_i \to c \mid 1 \leq i \leq n\}$.

Clearly, the regular grammars G and H can be constructed from $(J, \hat{k})$ in deterministic polynomial time. We claim further that

(1) if J has a clique of size k, then G HOM-ONTO H, and

(2) if J does not have a clique of size k, then $\sim(G$ WEAK-INTERP $H)$.

Claims (1) and (2) imply that J has a clique of size k if and only if (G, H) is in R_σ, and thus, imply that the language CLIQUE is deterministic polynomially time-bounded reducible to R_σ.

We now prove claims (1) and (2).

Proof of claim (1). Suppose J has a clique of size k. Let one such clique be $\{N_{\nu_1}, N_{\nu_2}, \ldots, N_{\nu_k}\}$. Let f be the function from the alphabet of G to the alphabet of H defined by

(i) $f(S) = T$,

(ii) $f(A_i) = N_{\nu_i}$ for $1 \leq i \leq k$,

(iii) $f(N_i) = N_i$ for $1 \leq i \leq n$, and

(iv) $f(a) = a, f(b) = b, f(c) = c$.

Then f is an onto homomorphism from G to H. Thus, G HOM-ONTO H, and, thus, $G \sigma H$.

Proof of claim (2). Suppose J does not have a clique of size k, but G is a weak interpretation of H under substitution μ. For each i such that $1 \leq i \leq k$, A_i is the image of only one nonterminal of H under μ. If A_i were in $\mu(T)$, then corresponding to the production $S \to a A_i$ in P_1, Q would have the production $T \to aT$. Thus, A_i is in $\mu(N_{\nu_i})$ for some ν_i. Corresponding to the production $A_i \to b A_j$ in P_2, Q_2 must have the production $N_{\nu_i} \to b N_{\nu_j}$. If $\nu_i = \nu_j$, then Q_2 would have the production $N_{\nu_i} \to b N_{\nu_i}$. Since Q_2 does not contain such a production, $i \neq j$, and thus, $\nu_i \neq \nu_j$. Thus $N_{\nu_1}, N_{\nu_2}, \ldots, N_{\nu_k}$ are k distinct nonterminals corresponding to a clique of size k in the graph J, a contradiction.

Finally, we note that the NP-hardness of the language R_σ directly implies the NP-hardness of the language C_σ. □

Theorem 3.2. *The following sets are* NP-*complete*:

(1) $\{(G, H) \mid$ *G and H are grammars* [*or G and H are regular grammars*] *and there is a homomorphism from G onto H*$\}$;

(2) $\{(G, H) \mid$ *G and H are grammars* [*or G and H are regular grammars*] *and G is Reynolds covered by H*$\}$;

(3) $\{(G, H) \mid$ *G and H are grammars* [*or G and H are regular grammars*] *and G is weak Reynolds covered by H*$\}$;

(4) $\{(G, H) \mid$ *G and H are grammars* [*or G and H are regular grammars*] *and G is an interpretation of H*$\}$;

(5) $\{(G, H) \mid$ *G and H are grammars* [*or G and H are regular grammars*] *and G is a strict interpretation of H*$\}$;

(6) $\{(G, H) \mid$ *G and H are grammars* [*or G and H are regular grammars*] *and G is an onto strict interpretation of H*$\}$;

(7) $\{(G, H) \mid$ *G and H are grammars* [*or G and H are regular grammars*] *and G is a generalized strict interpretation of H*$\}$; *and*

(8) $\{(G, H) \mid$ *G and H are grammars* [*or G and H are regular grammars*] *and H is a skeletal grammar for G*$\}$.

Proof. From Theorem 3.1, each of sets (1)–(7) is NP-hard. Since a skeletal grammar for a regular grammar is the image of an onto homomorphism, the sets of (8) are also NP-hard. Finally, it can easily be seen that each of the sets (1)–(8) is in NP. □

Next, we generalize the concept of 'a grammar form representing a family of grammars' [5, 6, 10, 15] by using an arbitrary grammatical similarity relation to define the set of interpretations of a given grammar.

Definition 3.3. Let ρ be a binary relation on the set of grammars. Let G be a grammar. The *family of grammars of G induced by the relation ρ*, denoted by $\Gamma_\rho(G)$, is the set

$$\{K \mid K \text{ is a grammar and } G \rho K\}.$$

Two grammars G and H are *strongly equivalent grammar forms under ρ* if $\Gamma_\rho(G) = \Gamma_\rho(H)$.

Theorem 3.4. *Let ρ be any reflexive and transitive binary relation on the set of grammars. Then, for all grammars G and H, $G \rho H$ if and only if $\Gamma_\rho(G) \subset \Gamma_\rho(H)$.*

Proof. Suppose $\Gamma_\rho(G) \supset \Gamma_\rho(H)$. Since ρ is reflexive, $H \rho H$, and so H is in $\Gamma_\rho(H)$. But then H is in $\Gamma_\rho(G)$, so $G \rho H$.

Now suppose $G \rho H$. Suppose K is in $\Gamma_\rho(H)$. Then $H \rho K$. Since ρ is transitive, $G \rho K$, so K is in $\Gamma_\rho(G)$. Thus, $\Gamma_\rho(G) \supset \Gamma_\rho(K)$. □

Each of the grammatical relations of Theorem 3.2 is both reflexive and transitive (provided the skeletal grammar relationship is restricted to grammars that do not have productions of the form $A \to A$). Thus the result in [5] that it is decidable if the families of grammars associated with two grammar forms are equal can be generalized as follows.

Theorem 3.5. *The set*

$$\{(G, H) \mid G \text{ and } H \text{ are strongly equivalent grammar forms under } \rho\}$$

is in NP *when ρ is any of the relations: onto homomorphism, Reynolds covers, weak Reynolds cover, interpretation, strict interpretation, onto strict interpretation, generalized strict interpretation, and skeletal grammar (for grammars without productions of the form $A \to A$).*

Proof. The proof immediately follows from Theorems 3.2 and 3.4. □

Theorem 3.6. *The set*

$$\{(G, H) \mid G \text{ and } H \text{ are strongly equivalent grammar forms under } \rho\}$$

is NP-*complete when ρ is any of the relations*: ***Reynolds covers, weak Reynolds covers, interpretation, strict interpretation, and generalized strict interpretation.***

Proof. Membership in NP follows from Theorem 3.5. NP-hardness follows from noting that, in the proof of Theorem 3.1, the constructed grammars G and H are always related in one direction by each of the specified relations. □

The three relations included in Theorem 3.5, but not in Theorem 3.6, are onto homomorphism, onto strict interpretation, and skeletal grammar (for grammars without productions of the form $A \rightarrow A$). It follows directly from the definitions of these relations that two grammars are strongly equivalent grammar forms under the first or the third of these relations if and only if they are isomorphic grammars; and two grammars are strongly equivalent grammar forms under the second of these relations if and only if the grammars are related by isomorphic strict interpretation. In Section 5 we show that testing for grammar isomorphism or for isomorphic strict interpretation is polynomially equivalent to testing for graph isomorphism. Thus, these three grammar form equivalence problems are polynomially equivalent to graph isomorphism.

4. Covering by structurally unambiguous grammars

In this section we present deterministic polynomial time algorithms for testing, for a grammar G and a structurally unambiguous grammar H, if G is Reynolds covered by H, if there is an onto homomorphism from G to H, or if there is an isomorphism from G to H. Moreover, if such a Reynolds cover, onto homomorphism, or isomorphism exists, the algorithms output appropriate functions from the nonterminals of G to the nonterminals of H. Since the class of structurally unambiguous grammars properly contains the class of unambiguous grammars, deterministic polynomial time algorithms exist when H is in Γ, for any class Γ, of unambiguous grammars. Thus, deterministic polynomial time algorithms exist when H is a member of most of the grammar classes corresponding to parsing methods used in compilers.

Theorem 4.1. *There is a polynomial time algorithm for testing, for grammar G and structurally unambiguous grammar H, if H Reynolds covers G. Moreover, if H Reynolds covers G, then this algorithm outputs an appropriate function from the nonterminals of G to the nonterminals of H.*

Proof. Let $G = (M, \Sigma, P, S)$, and $H = (N, \Sigma, Q, T)$. The basis of the algorithm is finding, for each A in M, a derivation tree generated by G containing A and a structurally equivalent derivation tree τ generated by H. Since H is structurally unambiguous, if there is a Reynolds cover, then A must map into the label of the corresponding node of τ. However, these derivation trees are *not* explicitly constructed since they may be exponential in the sizes of G and H. Rather, each tree is compactly represented without producing it in entirety.

The algorithm consists of four steps.

[1] *Step* 1 of the algorithm consists of computing, for each A in M, a set $\textsc{Match}(A)$ defined in terms of a specific tree, called $\textsc{Tree}(A)$, that A generates.

$$\textsc{Match}(A) = \{B \mid B \text{ is in } N, \text{ and } B \text{ generates a tree that is structurally equivalent to } \textsc{Tree}(A)\}.$$

The sets $\textsc{Match}(A)$, for A in M, are computed in a manner related to the test for aliveness in [16]. The trees $\textsc{Tree}(A)$, for A in M, are not explicitly computed. The computation of the $\textsc{Match}$ sets is done as follows:

If $\textsc{Match}(A)$ has not already been computed and there is a production

$$A \to \alpha_1 \alpha_2 \ldots \alpha_k$$

in P such that each α_i is either a terminal or a nonterminal for which $\textsc{Match}(\alpha_i)$ has already been computed, then choose one such production and let $\textsc{Match}(A) = \{B \mid B$ is in N; and there is a compatible production

$$B \to \beta_1 \beta_2 \cdots \beta_k$$

in Q for which if β_i is a nonterminal, it is in $\textsc{Match}(\alpha_i)\}$.

As an example consider the grammars G and H of Fig. 1(a). From productions 2, 7, and 10, $\textsc{Match}(A) = \{D, E\}$. From productions 4, 8, 11, and 15, $\textsc{Match}(B) = \{D, E, G\}$. From productions 1, 6, and 16, $\textsc{Match}(S) = \{T, G\}$. From productions 5, 14, and 17, $\textsc{Match}(C) = \{F, H\}$. Although they are not explicitly constructed, the $\textsc{Tree}$ of each nonterminal is shown in Fig. 1(b).

[2] *Step* 2 consists of computing, for each A in M, an *incomplete reachability tree* for A, called $\textsc{Reach}(A)$. An incomplete reachability tree is a derivation tree produced by G having S as its root and frontier nodes that are elements of $\Sigma \cup M$. $\textsc{Reach}(A)$ is an incomplete reachability tree in which A is one of the frontier nodes. A reachability tree for each nonterminal of G can be found in polynomial time in a manner analogous to the reachability test in [16]. The number of productions used in each reachability tree is bounded by $|M|$. The computation of these trees is done as follows:

> $\textsc{Reach}(S)$ consists of a single node labeled S. If $\textsc{Reach}(B)$ has not yet been computed, and there is a production $A \to \gamma$ where B is a symbol in γ, and $\textsc{Reach}(A)$ has been computed, then choose one such production and let $\textsc{Reach}(B)$ be $\textsc{Reach}(A)$ with the symbols of γ appended as descendants of some occurrence of A on the frontier of $\textsc{Reach}(A)$. As an example, $\textsc{Reach}$ trees for grammar G of Fig. 1(a) are shown in Fig. 1(c).

[3] *Step* 3 consists of computing, for each A in M, a set $\textsc{Candidate}(A)$ of members of N that A can map into under a Reynolds cover.

For each A in M, define DERIV(A) to be the complete derivation tree obtained by appending to each nonterminal B on the frontier of REACH(A), the subtree TREE(B). (More accurately, the root of TREE(B) is merged with the occurrence of B on the frontier of REACH(A).) Suppose that we distinguish some particular node that is labelled with A and occurs on the frontier of REACH(A). This node then becomes a distinguished node labelled A in the tree DERIV(A). Any derivation tree H that is structurally equivalent to DERIV(A) contains a node corresponding to this distinguished node. This corresponding node becomes a distinguished node of H. Define

CANDIDATE(A) = $\{B \mid B$ is in N; and B labels the distinguished node of a derivation tree of H that is structurally equivalent to DERIV(A)$\}$.

Since, in DERIV(A), the subtree headed by the distinguished node is identical to TREE(A), CANDIDATE(A) is a subset of MATCH(A). The set CANDIDATE(A) can be computed without constructing DERIV(A). The computation uses REACH(A) and the MATCH sets, as follows:

For each B in MATCH(A), associate a subset of N with each nonterminal node of REACH(A) as follows. Associate $\{B\}$ with one occurrence of A on the frontier of REACH(A). With every other nonterminal α on the frontier, associate MATCH(α). The sets associated with the interior nodes of REACH(A) are computed in the following bottom-up manner. With each interior node associate the set of nonterminals C in N for which

(i) there is a production $C \to \beta_1\beta_2 \ldots \beta_k$ in Q, and

(ii) the interior node has k immediate descendants such that, for $1 \leq i \leq k$, if the ith immediate descendant is a terminal, then β_i is the same terminal and if the ith immediate descendant is a nonterminal, then β_i is one of the nonterminals associated with the descendant.

If T is one of the nonterminals associated with the root of REACH(A), then B is included in CANDIDATE(A).

As an example, Fig. 1(d) shows the computation of the CANDIDATE sets for the grammars of Fig. 1(a). The result of the computation is that CANDIDATE(S) = $\{T\}$, CANDIDATE(A) = $\{D\}$, CANDIDATE(B) = $\{D\}$, and CANDIDATE(C) = $\{F\}$.

To see that the computation correctly determines if B is in CANDIDATE(A), note that each node of REACH(A) corresponds to a node of the complete tree DERIV(A), and also corresponds to the subtree of DERIV(A) headed by that node. From the definition of MATCH, each member of N that the computation associates with a frontier node of REACH(A) generates the image of the corresponding subtree of DERIV(A). Each member of N that the computation associates with an interior node of REACH(A) generates the image of the corresponding subtree of DERIV(A) because the member of N has a context-free production whose right-hand side symbols are associated with the immediate descendants of the interior node and generate the corresponding subtrees. Thus, CANDIDATE(A) is correctly computed.

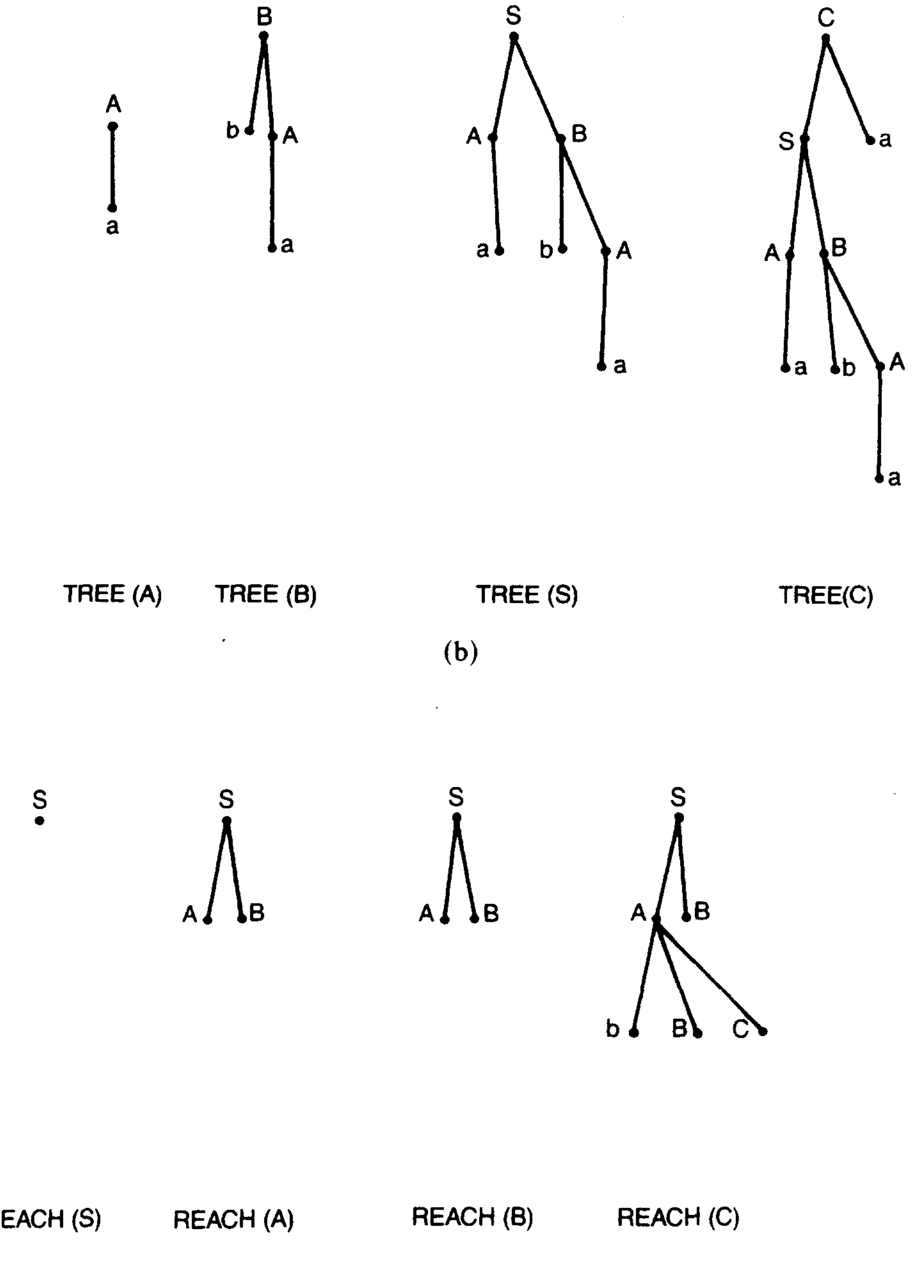
1. S→A B
2. A→a
3. A→b B C
4. B→b A
5. C→S a

Grammar G

6. T→D D
7. D→a
8. D→b D
9. D→b D f
10. E→a
11. E→b D
12. E→b D H
13. F→c
14. F→T a
15. G→b E
16. G→E E
17. H→Ta

Grammar H

(a)

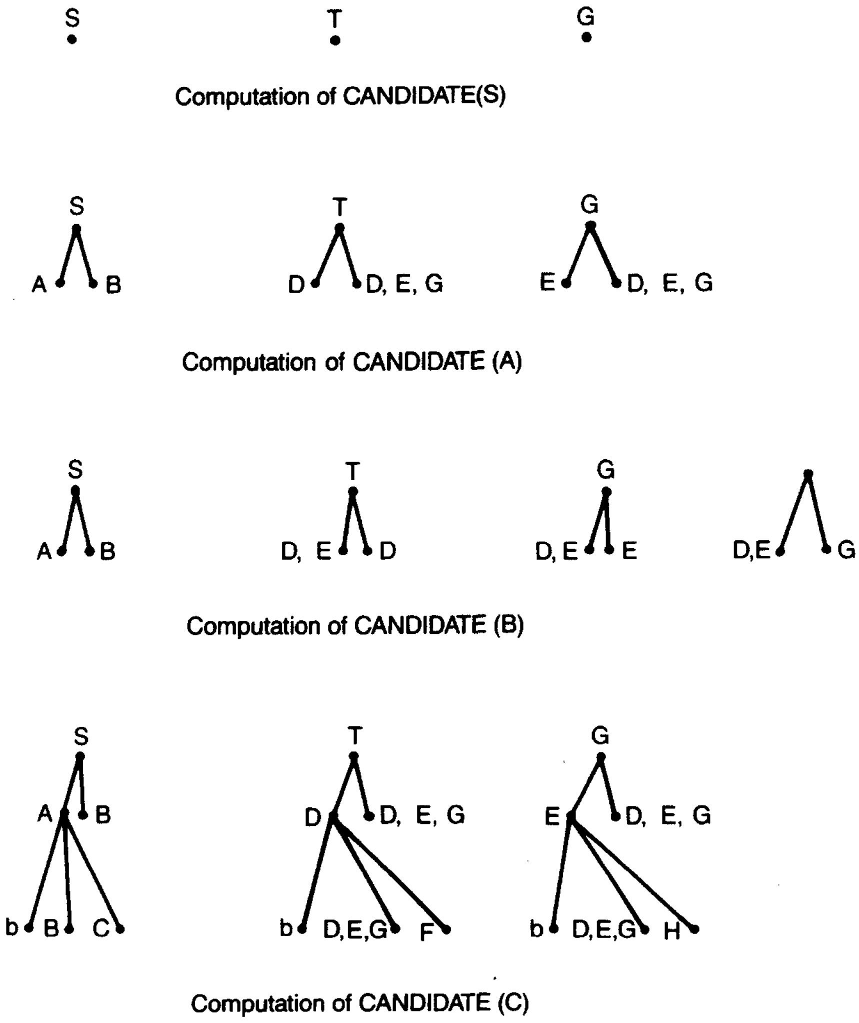


(c)

Fig. 1. (a) Grammars G and H. (b) The trees TREE(X) for X a nonterminal of G. (c) The REACH trees.

Because H is structurally unambiguous, H generates at most one derivation tree that is structurally equivalent to DERIV(A). Thus, for each A in M, the set CANDIDATE(A) has at most one member.

[4] *Step* 4 consists of testing whether the map corresponding to the CANDIDATE sets represents a Reynolds cover.



Computation of CANDIDATE(S)

Computation of CANDIDATE (A)

Computation of CANDIDATE (B)

Computation of CANDIDATE (C)

(d)

Fig. 1. (d) Computation of CANDIDATE sets.

If the set CANDIDATE(A) is empty for some A in M, then the algorithm halts with output "NO". Otherwise, CANDIDATE represents a map from M to N. Let f be the homomorphism from $(M \cup \Sigma)^*$ to $(N \cup \Sigma)^*$ that corresponds to CANDIDATE on M and is the identity on Σ. Then test whether f satisfies the definition of a Reynolds cover (Definition 2.7). If f is a Reynolds cover, the algorithm halts with output "YES" and the map from M to N. If not, the algorithm halts with output "NO".

In the example of Fig. 1, the computed CANDIDATE map does indeed represent a Reynolds cover.

Corollary 4.2. (a) *There is a polynomial time algorithm that determines if there exists an onto homomorphism from a given grammar G to a given structurally unambiguous grammar H, and outputs an appropriate nonterminal map if it exists.*

(b) *There is a polynomial time algorithm to determine if there exists an isomorphism from a given grammar G to a given structurally unambiguous grammar H, and outputs an appropriate nonterminal map if it exists.*

Proof. Onto homomorphism and isomorphisms are special cases of Reynolds covers. As shown in the proof of Theorem 4.1, since H is structurally unambiguous, there is at most one Reynolds cover of G by H. □

Next, we note that in using the algorithms of Theorem 4.1 and Corollary 4.2, there is no problem in checking that grammar H is structurally unambiguous. A structural ambiguity test is implicit in [21]. A deterministic polynomial time structural ambiguity test for regular grammars is given in [3]. A deterministic polynomial time structural ambiguity test for arbitrary context-free grammars is given in [19]. Hence, Theorem 4.1 and Corollary 4.2 give the following.

Corollary 4.3. *The sets*

(1) $\{(G, H) \mid G$ *is a grammar, H is a structurally unambiguous grammar, and G is Reynolds covered by H*$\}$,

(2) $\{(G, H) \mid G$ *is a grammar, H is a structurally unambiguous grammar, and there is an onto homomorphism from G to H*$\}$, *and*

(3) $\{(G, H) \mid G$ *is a grammar, H is a structurally unambiguous grammar, and there is an isomorphism from G to H*$\}$

are each recognizable in deterministic polynomial time.

5. Isomorphism problems

In this section we show that the problems of determining if a pair of grammars are related by isomorphism, or by isomorphic strict interpretation, are polynomially equivalent to the graph isomorphism problem. The main point of this result is that the grammar problems are no harder than the graph problem, even though the right-hand side of a grammatical production can contain several nonterminal symbols.

Booth [4] showed that the graph isomorphism problem and the problem of testing pairs of deterministic finite automata for isomorphic strict interpretation are polynomially equivalent. (The relation between finite automata was called 'isomorphism' in [4], but corresponds to isomorphic strict interpretation because a terminal need not be mapped into itself.) Note that the well-known polynomial time decidability of the state equivalence problem for deterministic finite automata [1, 16] implies that the isomorphism problem (in the sense used in this paper) is decidable deterministically in polynomial time for deterministic finite automata.

Theorem 5.1. *The following sets are polynomially equivalent*:

(1) $\{(G, H) \mid G$ *and H are isomorphic grammars*$\}$;

(2) $\{(G, H) \mid G$ *and H are isomorphic regular grammars*$\}$;

(3) $\{(J, K) \mid J$ *and K are isomorphic graphs*$\}$;

(4) $\{(G, H) \mid$ *G and H are regular grammars, and G is an isomorphic strict interpretation of* $H\}$; *and*

(5) $\{(G, H) \mid$ *G and H are grammars and G is an isomorphic strict interpretation of* $H\}$.

Proof. (a) A polynomial time reduction of problem (1) to problem (2).

Let $G=(M, \Sigma, P, S)$ be a grammar. Let $A \to \gamma$ be a member of P. We define the *template* of $A \to \gamma$ to be the string obtained from $A\gamma$ by replacing each occurrence of a nonterminal with an N_j, where if nonterminal B is the ith distinct nonterminal occurring in $A\gamma$, each occurrence of B is replaced by N_i. For example, the templates of $S \to aBcSSdCB$ and $A \to aDcAAdBD$ are identical and equal to $N_1aN_2cN_1N_1dN_3N_2$. Suppose G has t distinct templates. Let the templates of G be numbered in lexicographic order so that $i<j$ implies that template i lexicographically precedes template j. For each template i, let there be p_i productions of G with template i and let there be n_i distinct N_j's appearing in template i. For instance, $n_i=3$ for the template given above.

Let Z_{ijk} be the kth distinct nonterminal appearing in the jth production with template i. For instance, if the template given above is template 5, and its second production is $A \to aDcAAdBD$, then Z_{521} is A, Z_{522} is D, and Z_{523} is B. (Note that each Z_{ijk} is in M.)

Let $R(G)=(M', \Sigma', P', S)$ be the regular grammar defined by

(1) $M'=M \cup \{A_{ij} \mid 1 \leq i \leq t, 1 \leq j \leq p_i\}$,

(2) $\Sigma'=\{a\} \cup \{b_{ik} \mid 1 \leq i \leq t, 1 \leq k \leq n_i\}$, and

(3) $P'=P_1 \cup P_2 \cup P_3$, where

$P_1=\{S \to aA_{ij} \mid 1 \leq i \leq t, 1 \leq j \leq p_i\}$,

$P_2=\{A_{ij} \to b_{ik}Z_{ijk} \mid 1 \leq i \leq t, 1 \leq j \leq p_i, 1 \leq k \leq n_i\}$, and

$P_3=\{X \to a \mid X \text{ is in } M\}$.

We claim that two grammars $G=(M, \Sigma, P, S)$ and $H=(N, \Delta, Q, T)$ are isomorphic if and only if they have the same sets of templates and the regular grammars $R(G)=(M', \Sigma', P', S)$ and $R(H)=(N', \Delta', Q', T)$ above are isomorphic. Here we sketch the proof of the 'if' part of the claim. Thus, assume that G and H have the same sets of templates and that $R(G)$ and $R(H)$ are isomorphic.

Since we are assuming that all grammars are reduced, the assumption that G and H have the same set of templates implies that Σ equals Δ.

Let t and t' be the numbers of templates of G and of H respectively. By assumption, G and H have the same set of templates. Moreover, this set was lexicographically ordered during the constructions of $R(G)$ and of $R(H)$. Thus,

(1) $t=t'$ and, for $1 \leq i \leq t$, template i of G equals template i of H.

Thus, for $1 \leq i \leq t$, letting n_i and n'_i equal the numbers of distinct nonterminals appearing in template i of G and in template i of H,

(2) $n_i=n'_i$.

Thus $\Sigma'=\Delta'$.

Let ϕ be an isomorphism from $R(G)$ to $R(H)$. By assumption, such an isomorphism exists. We show that ϕ induces an isomorphism $\hat{\phi}$ from G to H. Since ϕ is a Reynolds cover, inspection of the productions of $R(G)$ and $R(H)$ shows that $\phi(S)=T$, $\phi(X)$ is in N for X in M, and $\phi(A_{ij})=A_{im}$ for some m. For $1\leq i\leq t$, let p_i and p'_i be the number of productions of G and H with template i. Since ϕ is an isomorphism, $p_i=p'_i$.

Let $\hat{\phi}$ be the mapping from $M\cup\Sigma$ defined as follows:

$$\hat{\phi}(\alpha)=\begin{cases}\alpha & \text{for all } \alpha \text{ in } \Sigma,\\ \phi(\alpha) & \text{for all } \alpha \text{ in } N.\end{cases}$$

Then, $\hat{\phi}$ is a one-to-one map from M to N.

For $1\leq i\leq t$ and $1\leq j\leq p_i$, corresponding to production j for template i in P, grammar $R(G)$ has productions $A_{ij}\rightarrow b_{i1}Z_{ij1}, A_{ij}\rightarrow b_{i2}Z_{ij2},\ldots, A_{ij}\rightarrow b_{in_i}A_{ijn_i}$. Suppose $\phi(A_{ij})=A_{im}$. Then $R(H)$ has productions $A_{im}\rightarrow b_{i1}\phi(Z_{ij1})$, $A_{im}\rightarrow b_{i2}\phi(Z_{ij2})$, $\ldots, A_{im}\rightarrow b_{in_i}\phi(Z_{ijn_i})$. Thus, production j for template i in P maps into production m for template i in Q. Thus, $\hat{\phi}$ is an isomorphism from G to H.

(b) A polynomial time reduction of problem (2) to problem (3).

Let $G=(N,\Sigma,P,S)$ be a regular grammar. A graph GRAPH(G), can be constructed from G as described below. An example of the construction is shown in Fig. 2,

Let the members of Σ be $\{a_1,a_2,\ldots,a_m\}$. Then GRAPH$(G)=(N',E)$ where the set of nodes $N'=N_1\cup N_2\cup N_3\cup N_4$, and the set of edges $E=E_1\cup E_2\cup E_3\cup E_4\cup E_5\cup E_6\cup E_7$, as described below:

$$N_1=\{[A,i]\mid A \text{ is in } N \text{ and } 1\leq i\leq 4\},$$

$$N_2=\{[A\rightarrow a_kB,i]\mid A\rightarrow a_kB \text{ is in } P \text{ and } 1\leq i\leq k+1\},$$

$$N_3=\{[A\rightarrow a_k,i]\mid A\rightarrow a_k \text{ is in } P \text{ and } 1\leq i\leq k\},$$

$$N_4=\{\text{START}\},$$

$$E_1=\{([A,i],[A,j])\mid A \text{ is in } N,\ i\neq j,\ 1\leq i\leq 4,\ 1\leq j\leq 4\},$$

$$E_2=\{([A\rightarrow a_kB,i],[A\rightarrow a_kB,i+1])\mid A\rightarrow a_kB \text{ is in } P,\ 1\leq i\leq k\},$$

$$E_3=\{([A\rightarrow a_k,i],[A\rightarrow a_k,i+1])\mid A\rightarrow a_k \text{ is in } P,\ 1\leq i\leq k\},$$

$$E_4=\{([A,i],[A\rightarrow a_kB,1])\mid A\rightarrow a_kB \text{ is in } P,\ 1\leq i\leq 2\},$$

$$E_5=\{([A\rightarrow a_kB,k+1],[B,4])\mid A\rightarrow a_kB \text{ is in } P\},$$

$$E_6=\{([A,i],[A\rightarrow a_k,1])\mid A\rightarrow a_kB \text{ is in } P,\ 1\leq i\leq 2\}, \text{ and}$$

$$E_7=\{(\text{START},[S,4])\}.$$

GRAPH(G) encodes grammar G as follows. The nodes of N_1 form a disjoint set of 4-cliques, where each 4-clique corresponds to a nonterminal of G. The other

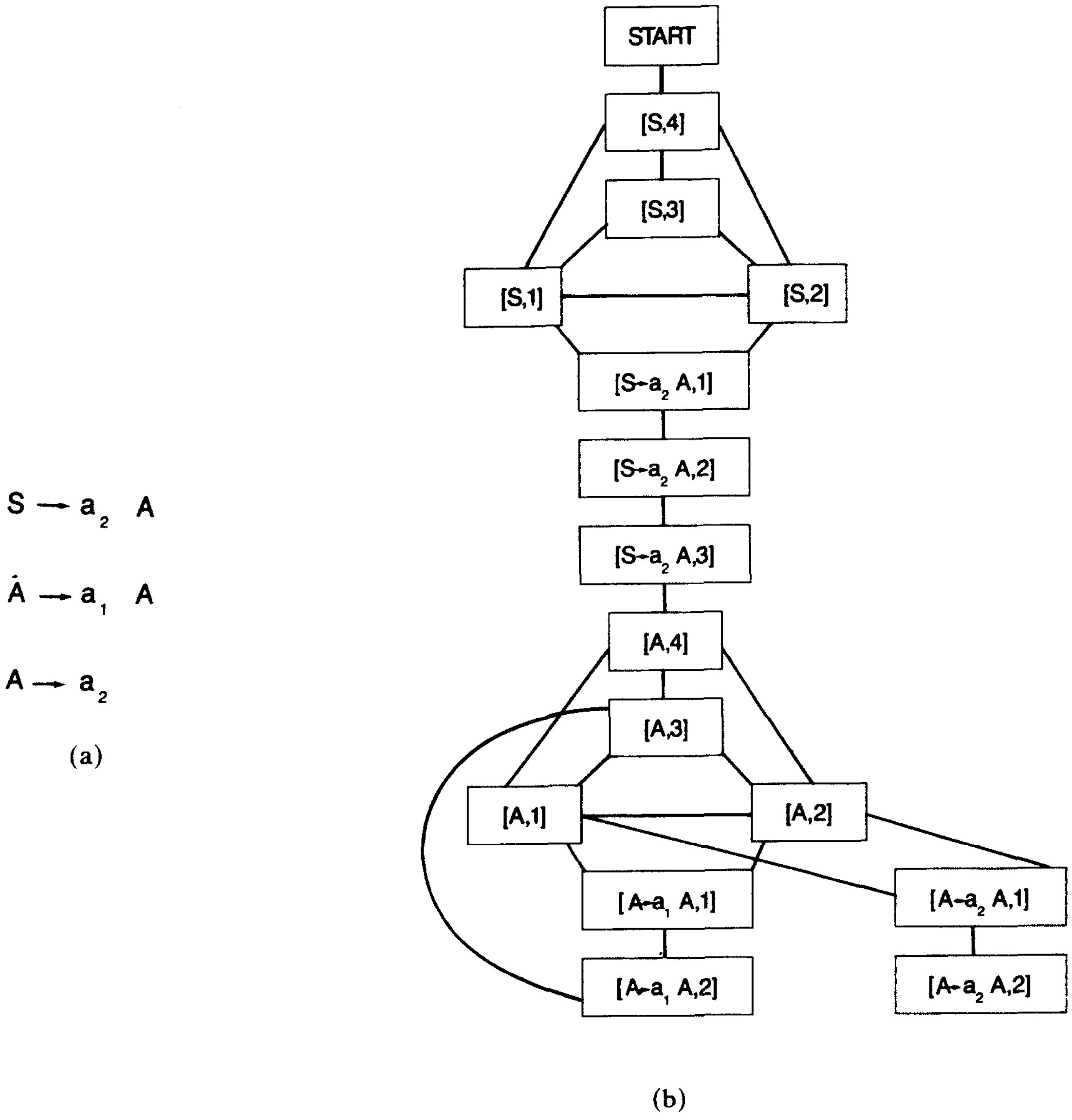


Fig. 2. (a) Grammar *G*. (b) The corresponding GRAPH(*G*).

nodes of the graph do not participate in 4-cliques. The 4-clique for the starting nonterminal is distinguished by E_7. Each production is encoded by a chain in the graph, where the length of the chain encodes the terminal occurring in the production. A production $A \rightarrow a_k B$ is encoded as a chain with $k+1$ nodes, where the chain connects two nodes of the clique for A with one node of the clique for B. Having at least two nodes in each such chain ensures that these nodes are uninvolved in a 4-clique. A production $A \rightarrow a_k$ is encoded as a chain of k nodes, with the chain connected to two nodes of the clique for A.

Assume that two regular grammars G and H have the same terminal alphabet, and the symbols in this alphabet are enumerated in the same order. We claim that, under this assumption, the two grammars are isomorphic if and only if the two graphs GRAPH(G) and GRAPH(H) are isomorphic. The details of the proof are left to the reader.

(c) The polynomial time reducibility of problem (3) to problem (4) follows from the reduction of graph isomorphism to deterministic finite state automata isomorphism in [4].

(d) The polynomial time reducibility of problem (4) to problem (5) is trivial, since problem (4) is a special case of problem (5).

(e) Polynomial time reduction of problem (5) to problem (1).

Let $G=(N, \Sigma, P, S)$ be a grammar. Let $\#$ be a special symbol. Let $Z(G)$ be the grammar $(N \cup \Sigma, \#, P \cup \{a \rightarrow \# \mid a \text{ is in } \Sigma\}, S)$. Then grammar G is an isomorphic strict interpretation of grammar H if and only if grammars $Z(G)$ and $Z(H)$ are isomorphic. □

References

[1] A.V. Aho and J.D. Ullman, *The Theory of Parsing, Translation, and Compiling, Vols. 1 and 2* (Prentice-Hall, Englewood Cliffs, NJ, 1972 and 1973).

[2] E. Bertsch, An observation on relative parsing time, *J. Assoc. Computing Mach.* **22** (1975) 493-498.

[3] R. Book, S. Even, S. Greibach and G. Ott, Ambiguity in graphs and expressions, *Proc. 3rd Ann. Princeton Conf. on Information Science and Systems* (1969) 345-349.

[4] K.S. Booth, Isomorphism testing for graphs, semigroups, and finite automata are polynomially equivalent problems, *SIAM J. Computing* **7** (1978) 273-279.

[5] A.B. Cremers and S. Ginsburg, Context-free grammar forms, *J. Computer and System Sciences* **11** (1975) 86-117.

[6] A.B. Cremers, S. Ginsburg and E.H. Spanier, The structure of context-free grammatical families, *J. Computer and System Sciences* **15** (1977) 262-279.

[7] A.J. Demers, Skeletal LR parsing, *Proc. IEEE 15th Annual Symp. on Switching and Automata Theory*, New Orleans, LA (1974) 185-198.

[8] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, San Francisco, CA, 1979).

[9] S. Ginsburg and M.A. Harrison, Bracketed context-free languages, *J. Computer and System Sciences* **1** (1967) 1-23.

[10] S. Ginsburg, B. Leong, O. Mayer and D. Wotschke, On strict interpretations of grammar forms, *Math. Systems Theory* **12** (1979) 233-252.

[11] J.N. Gray and M.A. Harrison, On the covering and reduction problems for context-free grammars, *J. Assoc. Computing Mach.* **19** (1972) 675-698.

[12] H.B. Hunt, III and D.J. Rosenkrantz, Complexity of grammatical similarity relationships, Preliminary report, *Proc. Conf. on Theoretical Computer Science*, Waterloo, Canada (1977) 139-145.

[13] H.B. Hunt, III and D.J. Rosenkrantz, Efficient algorithms for structural similarity of grammars, *Proc. 7th Ann. ACM Symp. on Principles of Programming Languages*, Las Vegas, NV (1980) 213-219.

[14] H.B. Hunt, III, D.J. Rosenkrantz and T.G. Szymanski, On the equivalence, containment and covering problems for the regular and context-free languages, *J. Computer and Systems Sciences* **12** (1976) 222-268.

[15] B. Leong and D. Wotschke, The influence of productions on derivations and parsing, *Conf. Rec. 3rd ACM Symp. On Principles of Programming Languages*, Atlanta, GA (1976) 1-11.

[16] P.M. Lewis, II, D.J. Rosenkrantz and R.E. Stearns, *Compiler Design Theory* (Addison-Wesley, Reading, MA, 1976).

[17] A. Nijholt, Context-free grammars: Covers, normal forms, and parsing, *Lecture Notes in Computer Science* **93** (Springer, Berlin, 1980).

[18] J.C. Reynolds and R. Haskell, Grammatical coverings, Unpublished manuscript, 1970.

[19] D.J. Rosenkrantz and H.B. Hunt, III, Efficient algorithms for automatic construction and compactification of parsing grammars, Submitted for publication.

[20] C.P. Schnorr, Transformational classes of grammars, *Information and Control* **14** (1969) 252-277.

[21] J.W. Thatcher, Tree automata: An informal survey, in: A.V. Aho, ed., *Currents in the Theory of Computing* (Prentice-Hall, Englewood Cliffs, NJ, 1973) 143-172.

[22] D. Wood, Grammar and L forms: An introduction, *Lecture Notes in Computer Science* **91** (Springer, Berlin, 1980).