ELSEVIER

Theoretical Computer Science 230 (2000) 247–256

Theoretical Computer Science

www.elsevier.com/locate/tcs

Note

The inclusion problem for some subclasses of context-free languages

Peter R.J. Asveld, Anton Nijholt *

Department of Computer Science, Twente University of Technology, P.O. Box 217, 7500 AE Enschede, The Netherlands

Communicated by G. Rozenberg

Abstract

By a reduction to Post's Correspondence Problem we provide a direct proof of the known fact that the inclusion problem for unambiguous context-free grammars is undecidable. The argument or some straightforward modification also applies to some other subclasses of context-free languages such as linear languages, sequential languages, and DSC-languages (i.e., languages generated by context-free grammars with disjunct syntactic categories). We also consider instances of the problem "Is $L(D_1) \subseteq L(D_2)$?" where D_1 and D_2 are taken from possibly different descriptor families of subclasses of context-free languages.

Keywords: Inclusion problem; Unambiguous context-free language; (restricted) Context-free grammar; (restricted) Deterministic push-down automaton

When we change a context-free language by e.g. modifying its grammar G_1 into a new context-free grammar G_2, the obvious questions are: What is the relationship between $L(G_1)$ and $L(G_2)$? Are they equal? Is one language a (proper) subset of the other one? Are these two languages incomparable? In answering questions of this type the (decidability of the) inclusion problem plays a principal part.

Consider two descriptors $\mathbf{D}_1$ and $\mathbf{D}_2$ of subclasses of context-free languages; e.g., $\mathbf{D}_1$ and $\mathbf{D}_2$ are two particular kinds of context-free grammars or push-down automata. Then the *inclusion* or *containment problem* for $(\mathbf{D}_1, \mathbf{D}_2)$ is the question whether for arbitrary $D_1 \in \mathbf{D}_1$ and $D_2 \in \mathbf{D}_2$ the inclusion $L(D_1) \subseteq L(D_2)$ holds. In case $\mathbf{D}_1 = \mathbf{D}_2$ we refer to this problem as the *inclusion problem* for $\mathbf{D}_1$.

* Corresponding author.
E-mail address: anijholt@cs.utwente.nl (A. Nijholt)

0304-3975/00/$ - see front matter
PII: S0304-3975(99)00113-9

It is well known that the inclusion problem for regular languages is decidable, whereas it is undecidable for context-free languages. Originally, this latter fact has been proved in [1]; [10] contains an alternative proof, and many text books, like [11, 14, 12, 18], also provide a proof of this undecidability result.

A similar conclusion has been obtained for deterministic context-free languages [5], and it has been established in [3] that the inclusion problem for simple deterministic languages is undecidable too; see also [8]. Clearly, these facts imply the undecidability of the inclusion problem for unambiguous context-free languages.

In this note we provide an alternative, direct proof of this latter fact (Theorem 1) which consists of a reduction to Post's Correspondence Problem over two-letter alphabets. As a consequence of this proof we obtain the undecidability of the inclusion problem for linear and sequential languages (Corollary 2). A slight modification of the argument yields the undecidability of the inclusion problem for context-free grammars with disjunct syntactic categories (Theorem 3). This result also follows from the undecidability of the inclusion problem for NTS (or nonterminal separating) languages established in [16]. Finally, we consider some consequences for inclusion problems of the form $(\mathbf{D}_1, \mathbf{D}_2)$ with $\mathbf{D}_1 \neq \mathbf{D}_2$ (Theorems 5–7; Table 1), and we survey the open problems in the area (Table 1).

The emphasis in this note is on the application of the proof technique used in establishing Theorem 1 and on surveying results with respect to the inclusion problem rather than deriving new results. Actually, only Corollary 2 and its consequences (see Table 1), Theorems 5–7, and the proofs of Theorems 1 and 3 seem to be new.

Theorem 1. *Let G_1 and G_2 be unambiguous context-free grammars. Then the problem "Is $L(G_1) \subseteq L(G_2)$?" is undecidable.*

Proof. Let I be an instance of Post's Correspondence Problem (PCP) over a two-letter alphabet, i.e., $I = (\alpha_1, \ldots, \alpha_n; \beta_1, \ldots, \beta_n)$ with $\alpha_i, \beta_i \in \Delta^+$ $(1 \leqslant i \leqslant n)$ and $\Delta = \{a, b\}$. Let Θ be an alphabet of n new symbols, say $\Theta = \{a_1, \ldots, a_n\}$, and define the homomorphism $h : (\Theta \cup \Delta)^* \to \Delta^*$ by

$$h(a_i) = \lambda \quad \text{for all } i \ (1 \leqslant i \leqslant n),$$

$$h(a) = a,$$

$$h(b) = b,$$

(λ denotes the empty word). Consider the context-free grammar $G_I = (V, \Sigma, P_I, S)$ with $\Sigma = \Theta \cup \Delta \cup \{c\}$, $V = \Sigma \cup \{S\}$ and P_I consists of the productions

$$S \to a_i \alpha_i S \beta_i^R \quad \text{for all } i \ (1 \leqslant i \leqslant n),$$

$$S \to a_i \alpha_i c \beta_i^R \quad \text{for all } i \ (1 \leqslant i \leqslant n),$$

where R is the reversal or mirror operation. Then we have

$$L(G_I) = \{a_{i_1} \alpha_{i_1} \cdots a_{i_k} \alpha_{i_k} c (\beta_{i_1} \cdots \beta_{i_k})^R \mid k \geqslant 1, 1 \leqslant i_j \leqslant n, \text{ for all } j \text{ with } 1 \leqslant j \leqslant k\}$$

Table 1

$\subseteq$ \ $\cup$	Regular	Linear	Unambiguous	Sequential	Deterministic	Simple deterministic	Real-time strict deterministic	Super-deterministic	LL(k)	NTS	DSC	Context-free
Regular	TD	?	D [0]	?	D [11]	D [11*]	D [11*]	D [11*]	D [11*]	D [11*]	?	U [11]
Linear	TD	U [0]	U [0]	U [0]	U [0]	?	?	U [4*]	?	?	U [0]	U [11*]
Unambiguous	TD	U [0]	U [5*]	U [0]	U [5*]	U [3*]	U [3*]	U [4*]	U [3*]	U [16*]	U [16*]	U [5*]
Sequential	TD	U [0]	U [0]	U [0]	U [0]	?	?	U [4*]	?	?	U [0]	U [11*]
Deterministic	TD	U [0]	U [5*]	U [0]	U [5]	U [3*]	U [3*]	U [4*]	U [3*]	U [16*]	U [16*]	U [11*]
Simple deterministic	TD	U [0]	U [3*]	U [0]	U [3*]	U [3]	U [3*]	U [0]	U [3*]	?	U [0]	U [11*]
Real-time strict deterministic	TD	U [0]	U [3*]	U [0]	U [3*]	U [3*]	U [3*]	U [0]	U [3*]	?	U [0]	U [11*]
Super-deterministic	TD	U [0]	U [4*]	U [0]	U [4*]	?	?	U [4]	?	?	U [0]	U [11*]
LL(k)	TD	U [0]	U [3*]	U [0]	U [3*]	U [3*]	U [3*]	U [4*]	U [3*]	?	U [0]	U [11*]
NTS	TD	U [0]	U [16*]	U [0]	U [16*]	?	?	U [4*]	?	U [16*]	U [16*]	U [11*]
DSC	TD	U [0]	U [16*]	U [0]	U [16*]	?	?	U [4*]	?	U [16*]	U [16*]	U [16*]
Context-free	TD	U [0]	U [5*]	U [0]	U [5*]	U [3*]	U [3*]	U [4*]	U [3*]	U [16*]	U [16*]	U [1]

TD = trivially decidable, D = decidable, U = undecidable, ? = open problem.

and G_I is unambiguous. Next we define the context-free grammar $G=(V_0,\Sigma,P,S)$ with alphabet $V_0=\Sigma\cup\{S,A,B,C,D\}$ and P consisting of the productions

(1) $S\to Aa\,|\,Ab$
(2) $A\to Aa\,|\,Ab\,|\,D$
(3) $S\to \xi aB\,|\,\xi bB$
(4) $B\to \xi aB\,|\,\xi bB\,|\,D$
(5) $D\to \xi aDa\,|\,\xi bDb\,|\,\xi aDb\,|\,\xi bDa\,|\,c$
(6) $S\to C$
(7) $C\to \xi aCa\,|\,\xi bCb\,|\,\xi aDb\,|\,\xi bDa$

with $\xi\in\{\lambda\}\cup\Theta$. It is easy to see that

$$L(G)=\{wcv\,|\,w\in((\Theta\cup\{\lambda\})\Delta)^+,\ v\in\Delta^+,\ h(w)\neq v^R\},$$

as well as the following facts: for each $wcv\in L(G)$ we have

(a) $|h(w)|<|v|$ if and only if wcv has been derived using the rules (1), (2) and (5) only,
(b) $|h(w)|>|v|$ if and only if wcv has been derived using the rules (3)–(5) only,
(c) $|h(w)|=|v|$ if and only if wcv has been derived using the rules (5)–(7) only,

where as usual $|x|$ denotes the length of the string x. Using this observation it is straightforward to show that G is unambiguous.

Now suppose the instance I has a solution. Thus there exists a sequence $a_{i_1}a_{i_2}\cdots a_{i_k}$ such that

$$h(a_{i_1}\alpha_{i_1}\cdots a_{i_k}\alpha_{i_k})=\beta_{i_1}\cdots\beta_{i_k}.$$

This means that $L(G_I)-L(G)\neq\emptyset$, and consequently $L(G_I)$ is not included in the language $L(G)$.

Conversely, suppose $L(G_I)$ is not included in $L(G)$. Then there exists a string wcv in $L(G_I)$ with $h(w)=v^R$. But then the sequence of symbols from Θ that occur from left to right in w determines a solution for I.

Summarizing, we have that the inclusion problem for unambiguous context-free grammars is reducible to PCP. Hence it is undecidable. □

Notice that both grammars constructed in the proof are linear and sequential. Remember that a context-free grammar $G=(V,\Sigma,P,S)$ is called *sequential* [6] if $V-\Sigma$ can be provided with a linear order $\leqslant$ such that for each rule $A\to w$, $A\leqslant B$ holds for all nonterminal symbols B that occur in w. (The linear order for the grammar G in our proof is: $S\leqslant A\leqslant B\leqslant C\leqslant D$). Therefore we have

Corollary 2. *Let G_1 and G_2 be unambiguous sequential linear context-free grammars. Then the problem "Is $L(G_1)\subseteq L(G_2)$?" is undecidable.*

Next we turn to context-free grammars that possess disjunct syntactic categories or that satisfy the NTS (nonterminal separating) property. A *DSC-grammar* or a *context-free grammar with disjunct syntactic categories* is a context-free grammar $G=(V,\Sigma,P,X)$ with $X\subseteq V-\Sigma$, such that for all $A,B\in V-\Sigma$, $A\neq B$ implies $L(G,A)\cap L(G,B)=\emptyset$, where for each A, $L(G,A)=\{w\in V^*\mid A\Rightarrow^* w\}$. The language generated by a DSC-grammar $G=(V,\Sigma,P,X)$ is defined by $L(G)=\{w\in\Sigma^*\mid A\Rightarrow^* w \text{ for some } A\in X\}$.

A context-free grammar $G=(V,\Sigma,P,X)$ with $X\subseteq V-\Sigma$ is an *NTS-grammar* (or satisfies the nonterminal separating property [2, 16]) if for all $A\in V-\Sigma$, and for all $w\in V^*$, $A\Rightarrow^* w$ holds if, and only, if $A\Leftrightarrow^* w$, where $\Leftrightarrow^*$ is the reflexive and transitive closure of the union of $\Rightarrow$ and its converse relation $\Leftarrow$. So, roughly spoken, $A\Leftrightarrow^* w$ means that w may be obtained from A by using the productions of P in both directions. The language generated by an NTS-grammar $G=(V,\Sigma,P,X)$ is defined by $L(G)=\{w\in\Sigma^*\mid A\Rightarrow^* w \text{ for some } A\in X\}$.

Each NTS-grammar has disjunct syntactic categories [2]; but the converse does not hold.[1] For instance, the language $\{a^n b^n \mid n \geqslant 1\} \cup \{a^n b^{2n} \mid n \geqslant 1\}$ is not an NTS-language [2], but it is easy to show that this language can be generated by a DSC-grammar. The inclusion problem for NTS-grammars is undecidable [16], which also implies the undecidability of the inclusion problem for DSC-grammars. Here we provide a direct proof of this latter statement.

Theorem 3. *Let G_1 and G_2 be context-free grammars with disjunct syntactic categories. Then the problem "Is $L(G_1) \subseteq L(G_2)$?" is undecidable.*

Proof. We slightly change the proof of Theorem 1. First, we observe that G_I is trivially a DSC-grammar. Secondly, we replace the grammar G in that proof by $G_0 = (V_0, \Sigma, P_0, X_0)$ with $\Sigma = \Delta \cup \Theta \cup \{c\}, \Delta = \{a, b\}, V_0 = \Sigma \cup \{S, T, C, D, E\}, X_0 = \{S, T, C, D\}$ and P_0 consists of the productions

$$S \to \xi aS \mid \xi bS \mid \xi aC \mid \xi bC \mid \xi aD \mid \xi bD \mid \xi aE \mid \xi bE$$

$$T \to Ta \mid Tb \mid Ca \mid Cb \mid Da \mid Db \mid Ea \mid Eb$$

$$C \to \xi aCa \mid \xi bCb \mid \xi aDa \mid \xi bDb$$

$$D \to \xi aDb \mid \xi bDa \mid \xi aCb \mid \xi bCa \mid \xi aEb \mid \xi bEa$$

$$E \to \xi aEa \mid \xi bEb \mid c$$

with $\xi \in \{\lambda\} \cup \Theta$. Then it is easy to see that

$$L(G_0, S) = \{wcv \mid w \in ((\Theta \cup \{\lambda\})\Delta)^*,\ v \in \Delta^*,\ |h(w)| > |v|\},$$

$$L(G_0, T) = \{wcv \mid w \in ((\Theta \cup \{\lambda\})\Delta)^*,\ v \in \Delta^*,\ |h(w)| < |v|\},$$

$$L(G_0, C) = \{wcv \mid w \in ((\Theta \cup \{\lambda\})\Delta)^*,\ v \in \Delta^*,\ |h(w)| = |v| \geqslant 1,\ h(w) \neq v^R,\ 1 : h(w) = 1 : v^R\},$$

$$L(G_0, D) = \{wcv \mid w \in ((\Theta \cup \{\lambda\})\Delta)^*,\ v \in \Delta^*,\ |h(w)| = |v| \geqslant 1,\ 1 : h(w) \neq 1 : v^R\},$$

$$L(G_0, E) = \{wcv \mid w \in ((\Theta \cup \{\lambda\})\Delta)^*,\ v \in \Delta^*,\ h(w) = v^R\},$$

where $1 : x$ denotes the first symbol of the string x. Hence G_0 is a DSC-grammar. Moreover, it is straightforward to prove that $L(G_0) = L(G)$. □

[1] This latter observation and the following example are due to Jan Anne Hogendorp.

Although G_I is an NTS-grammar, it is unlikely that this proof can be modified in order to provide an alternative way of establishing the undecidability of the inclusion problem for NTS-grammars [16]. More concretely, $L(G_0)$ is probably not an NTS-language.

We assume the reader to be familiar with the notion of deterministic PDA (push-down automaton) and restricted variants such as simple deterministic PDA and real-time strict deterministic PDA; cf. [8] for an excellent survey. However, we will recall the definition of the somewhat less known concept of super-deterministic PDA [4, 7].

Definition. Let $M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ be a deterministic push-down automaton with $F\subseteq Q\times(\Gamma\cup\{\lambda\})$ rather than $F\subseteq Q$. For each rule (q,a,A,p,y) in δ, the pair (q,A) is called the *mode* of the rule with input a; if $a=\lambda$, this is a λ-rule. If no rule is defined for (q,A) in $Q\times\Gamma$, it is a *blocking mode*; (q,λ) is also called a blocking mode. The elements of F are called *accepting modes*. A pair (q,yA) with $q\in Q$, $y\in\Gamma^*$, and $A\in\Gamma$ is a *configuration of M with mode* (q,A), for which we use the notation $m(q,yA)=(q,A)$. A configuration (q,yA) is in *reading mode*, if no λ-rule is defined for mode (q,A), and (q,A) is not a blocking mode.

M is *super-deterministic* if it is finite delay and for all accessible configurations in reading mode (q,s_1), (q,s_2), (q_1,t_1) and (q_2,t_2) in $Q\times\Gamma^*$ and $a\in\Sigma$, if $(q,s_1)\vdash^a(q_1,t_1)$ and $(q,s_2)\vdash^a(q_2,t_2)$, then $q_1=q_2$ and $|s_1|-|t_1|=|s_2|-|t_2|$.

The language $T(M)$ accepted by M by final state (accepting mode) is

$$T(M)=\{w\in\Sigma^*\mid(q_0,Z_0)\vdash^w(q,s)\text{ and }m(q,s)\in F\},$$

and the language $L(M)$ accepted by M by final state and empty store is

$$L(M)=\{w\in\Sigma^*\mid(q_0,Z_0)\vdash^w(q,\lambda)\text{ and }(q,\lambda)\in F\}.$$

A language L_0 over Σ_0 is *super-deterministic* if there is a super-deterministic PDA M such that either $L_0=T(M)$ or $L_0\$=T(M)$ for some symbol \$ not in Σ_0.

The inclusion problem for super-deterministic PDAs highly depends on the way in which a language is accepted; viz.

Theorem 4 (Greibach and Friedman [7] and Friedman and Greibach [4], respectively). *The inclusion problem is decidable for languages accepted by super-deterministic PDAs by final state and empty store. In case of acceptance by final state only, the inclusion problem is undecidable.*

Next we consider a few consequences for inclusion problems of the form $(\mathbf{D}_1,\mathbf{D}_2)$ in which $\mathbf{D}_1$ may differ from $\mathbf{D}_2$. In the sequel we restrict our attention to super-deterministic PDAs that accept by final state.

Theorem 5. *Let* $\mathbf{D}_1$ *and* $\mathbf{D}_2$ *be equal to one of the following descriptors*:

- *linear context-free grammar*,

- *sequential context-free grammar*,
- *unambiguous context-free grammar*,
- *deterministic push-down automaton*,
- *context-free grammar with disjunct syntactic categories*,
- *context-free grammar.*

Then the inclusion problem for $(\mathbf{D}_1, \mathbf{D}_2)$ *is undecidable. The same conclusion holds if* $\mathbf{D}_1$ *is taken equal to* "*NTS-grammar*".

Proof. These statements directly follow from the proofs of the previous results and the fact that $L(G_1)$ is an NTS-language. □

It remains an open problem whether "super-deterministic push-down automaton (acceptance by final state)" can be added to the list in Theorem 5; cf. Theorem 6.

Note that in [4] a variant of PCP – viz. the so-called *variant correspondence problem* or VCP – is used to establish the undecidability of the inclusion problem for super-deterministic PDAs. An instance I of such a VCP consists of two lists of n $(n \geqslant 1)$ nonempty words over Δ: $I = (\alpha_1, \ldots, \alpha_n; \beta_1, \ldots, \beta_n)$, where $|\alpha_1| < |\beta_1|$, and $|\alpha_i| \leqslant |\beta_i|$ for each i $(2 \leqslant i \leqslant n)$. For a symbol a in Δ, I has an a-marked solution $(i_1, \ldots, i_t)$, if $x_1 x_{i_1} \cdots x_{i_t} a$ is a prefix of $y_1 y_{i_1} \cdots y_{i_t}$ and $2 \leqslant i_1, \ldots, i_t \leqslant n$. And the question whether such an a-marked solution exists is undecidable [4]. From the argument in [4], it follows that the symbol a does not occur in the string $x_1 x_{i_1} \cdots x_{i_t}$. We will use this observation in the proof of Theorem 6(b).

Theorem 6. (a) *Let* $\mathbf{D}_1$ *be equal to one of the following descriptors*:

- *simple deterministic push-down automaton*,
- *real-time strict deterministic push-down automaton*,
- *LL(*k*)-grammar*,
- *super-deterministic push-down automaton*,

and let $\mathbf{D}_2$ *be equal to either* "*linear context-free grammar*", "*sequential grammar*", *or* "*context-free grammar with disjunct syntactic categories*". *Then the inclusion problem for* $(\mathbf{D}_1, \mathbf{D}_2)$ *is undecidable.*

(b) *The inclusion problem for* $(\mathbf{D}_1, \mathbf{D}_2)$ *is undecidable in case* $\mathbf{D}_1$ *equals* "*simple deterministic push-down automaton*" *or* "*real-time strict deterministic push-down automaton*", *and* $\mathbf{D}_2$ *is* "*super-deterministic push-down automaton*".

Proof. (a) Once again we adapt the proof of Theorem 1; viz. we construct a deterministic push-down automaton M_1 that accepts the language $L(G_I)$. The set of rules δ of $M_I = (\{q\}, \Sigma, \Sigma \cup \{Z_0\}, \delta, q, Z_0, F)$ is defined by

$$
\begin{array}{ll}
(q, a_i, Z_0, q, \beta_i a_i \alpha_i^R) & \text{for each } i \ (1 \leqslant i \leqslant n), \\
(q, x, x, q, \lambda) & \text{for each } x \text{ in } \Delta, \\
(q, a_j, a_i, q, \beta_j a_j \alpha_j^R) & \text{for each } i \text{ and } j \ (1 \leqslant i, j \leqslant n), \\
(q, c, a_i, q, \lambda) & \text{for each } i \ (1 \leqslant i \leqslant n).
\end{array}
$$

The push-down automaton M_I accepts the language $L(G_I)$ by final state and empty stack. It is straightforward to show that M_I is simple deterministic ($F=\{q\}$) as well as super-deterministic ($F=\{(q,\lambda)\}$).

(b) It suffices to show that the language

$$L_1=\{a_{i_t}\cdots a_{i_1}a_1\alpha_1\alpha_{i_1}\cdots\alpha_{i_t}a \mid t\geqslant 1;\ 2\leqslant i_1,\ldots,i_t\leqslant n\}$$

is simple deterministic; cf. [4]. Note that $\alpha_{i_j}\in\Sigma_0^*$ with $\Sigma_0=\Delta-\{a\}$ $(1\leqslant j\leqslant t)$, $\Sigma_0\cap\Sigma_1=\emptyset$, and $a\notin\Sigma_0\cup\Sigma_1$ with $\Sigma_1=\{a_1,\ldots,a_n\}$. The deterministic push-down automaton $M_1=(\{q\},\Delta\cup\Sigma_1,\Delta\cup\Sigma_1\cup\{Z_0\},\delta_1,q,Z_0,\{q\})$, where δ_1 is defined by

$$\begin{array}{ll}(q,a_i,Z_0,q,a\alpha_i^Ra_i) & \text{for each } i\ (2\leqslant i\leqslant n),\\ (q,x,x,q,\lambda) & \text{for each } x \text{ in } \Delta-\{\alpha\},\\ (q,a_j,a_i,q,\alpha_j^Ra_j) & \text{for each } i \text{ and } j\ (2\leqslant i,j\leqslant n),\\ (q,a_1,a_i,q,\alpha_1^R) & \text{for each } i\ (2\leqslant i\leqslant n),\end{array}$$

accepts L_1 by final state and empty stack. Clearly, M_1 is simple deterministic. □

In Table 1 we summarize known results with respect to the inclusion problem; it also includes the cases considered in the present paper to which we refer by [0]. A reference in Table 1 provided with an asterisk, e.g. [n^*], means that the result is not mentioned in [n] explicitly, but it follows from [n]: either trivially, or it can be inferred from the argument in [n] by observing that for the languages L_i in [n], it is obvious to construct descriptors D_i ($D_i\in\mathbf{D}_i$, $i=1,2$) such that $L(D_i)=L_i$. An example of a slightly less obvious construction is the proof of Theorem 6(b): rather than proving that the language L_1 is super-deterministic as in [4], we now show that L_1 is simple deterministic and, consequently, real-time strict deterministic.

Of course, Table 1 may be viewed as an extension of the appropriate row from Fig. 14.2 on p. 230 in [11]. A table similar to Table 1 surveying the equivalence problem for some subclasses of context-free languages can be found in [9].

Finally, we will discuss some decidable cases from Table 1. The inclusion problem for $(\mathbf{D}_1,\mathbf{D}_2)$, where $\mathbf{D}_2$ is any descriptor for the regular languages, is trivially decidable in the following sense; see also p. 204 in [12]. Let $D_i\in\mathbf{D}_i$ $(i=1,2)$, and $R=L(D_2)$ be regular. Because for each $\mathbf{D}_1$ in Table 1, we can effectively construct a context-free grammar G_1 such that $L(G_1)=L(D_1)$, we have

$$\text{“}L(D_1)\subseteq R?\text{”}\ \Leftrightarrow\ \text{“}L(G_1)\subseteq R?\text{”}\ \Leftrightarrow\ \text{“}L(G_1)\cap\bar{R}=\emptyset?\text{”}.$$

The latter question is decidable, since (i) the complement $\bar{R}$ of R is regular, (ii) the family of context-free languages is effectively closed under intersection with regular sets, and (iii) the emptiness problem for context-free languages is decidable.

Theorem 7. *The inclusion problem for* $(\mathbf{D}_1,\mathbf{D}_2)$ *is decidable in case* $\mathbf{D}_2$ *equals* “*unambiguous context-free grammar*” *and* $\mathbf{D}_1$ *is any descriptor of the regular languages.*

Proof. Let R be a regular language and let L_0 be an unambiguous context-free language. Clearly, $R \subseteq L_0$ holds, if and only if $R \cap L_0 = R$. Now the language $R \cap L_0$ is unambiguous by Theorem 6.4.1 from [8]. Then the result follows from the fact that the question "Is $L = R$?" is decidable for regular R and unambiguous L [15]; see also [9]. □

For the complexity of some (trivially) decidable entries mentioned in Table 1, we refer to [13, 17]. Even for the simplest case of Table 1 – viz. the inclusion problem for **D**, where **D** is any descriptor of the regular languages, i.e., the case corresponding to the left-upper corner of Table 1 – the inclusion problem is PSPACE-complete. Deterministic polynomial time-bounded algorithms have only been obtained for restricted cases of this entry, viz. for unambiguous descriptors, and for descriptors with bounded ambiguity; see [17] for details.

Acknowledgements

We are indebted to Rieks op den Akker and Jan Anne Hogendorp for some remarks on an earlier version of this note.

References

[1] Y. Bar-Hillel, M. Perles, E. Shamir, On formal properties of simple phrase-structure grammars, Z. Phonetik, Sprachwiss. Kommunikationsforsch. 14 (1961) 143–172. Reprinted as Chapter 9 in Y. Bar-Hillel: Language and Information, Addison-Wesley, Reading, MA, 1964, pp. 116–150.
[2] L. Boasson, G. Sénizergues, NTS languages are deterministic and congruential, J. Comput. System Sci. 31 (1985) 332–342.
[3] E.P. Friedman, The inclusion problem for simple languages, Theoret. Comput. Sci. 1 (1976) 297–316.
[4] E.P. Friedman, S.A. Greibach, Superdeterministic DPDAs: the method of accepting does affect decision problems, J. Comput. System Sci. 19 (1979) 79–117.
[5] S. Ginsburg, S.A. Greibach, Deterministic context-free languages, Inform. and Control 9 (1966) 620–648.
[6] S. Ginsburg, H.G. Rice, Two families of languages related to ALGOL, J. Assoc. Comput. Mach. 9 (1962) 350–371.
[7] S.A. Greibach, E.P. Friedman, Superdeterministic PDAs: a subcase with a decidable inclusion problem, J. Assoc. Comput. Mach. 27 (1980) 675–700.
[8] M.A. Harrison, Introduction to Formal Language Theory, Addison-Wesley, Reading, MA, 1978.
[9] M.A. Harrison, I.M. Havel, A. Yehudai, On equivalence of grammars through transformation trees, Theoret. Comput. Sci. 9 (1979) 173–205.
[10] J. Hartmanis, Context-free languages and Turing machine computations, in: J.T. Schwartz (Ed.), Mathematical Aspects of Computer Science, Proc. Symp. in Appl. Math., vol. XIX, American Mathematical Society, Providence, RI, 1967, pp. 42–51.
[11] J.E. Hopcroft, J.D. Ullman, Formal Languages and their Relation to Automata, Ch. 14, Addison-Wesley, Reading, MA, 1969.
[12] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, Reading, MA, 1979.
[13] H.B. Hunt III, D.J. Rosenkrantz, T.G. Szymanski, On the equivalence, containment, and covering problems for the regular and context-free languages, J. Comput. System Sci. 12 (1976) 222–268.
[14] A. Salomaa, Formal Languages, Academic Press, New York, 1973.

[15] A.L. Semenov, Algorithmic problems for power series and context-free grammars, Soviet Math. Dokl. 14 (1973) 1319–1322.
[16] G. Sénizergues, The equivalence and inclusion problems for NTS languages, J. Comput. System Sci. 31 (1985) 303–331.
[17] R.E. Stearns, H.B. Hunt III, On the equivalence and containment problems for unambiguous regular expressions, regular grammars and finite automata, SIAM J. Comput. 14 (1985) 598–611.
[18] D. Wood, Theory of Computation, Harper & Row, New York/Wiley, New York, 1987.