

Deciding DPDA Equivalence is Primitive Recursive

Colin Stirling

Division of Informatics
University of Edinburgh
email: cps@dcs.ed.ac.uk

Abstract. Recently Sénizergues showed decidability of the equivalence problem for deterministic pushdown automata. The proof of decidability is two semi-decision procedures that do not give a complexity upper bound for the problem. Here we show that there is a simpler deterministic decision procedure that has a primitive recursive upper bound.

1 Introduction

Recently Sénizergues showed decidability of the equivalence problem for deterministic pushdown automata, that language equivalence is decidable for deterministic context-free languages, see [6–9]. These proofs of decidability involve two semi-decision procedures that do not give a complexity upper bound for the problem. Here we show that there is a simpler deterministic decision procedure (first described in [10]) that has a primitive recursive upper bound.

2 Preliminaries

A deterministic pushdown automaton, a DPDA, consists of finite sets of states P , stack symbols S , terminals A and basic transitions T . A basic transition is $pS \xrightarrow{a} q\alpha$ where p, q are states in P , $a \in A \cup \{\varepsilon\}$, S is a stack symbol in S and α is a sequence of stack symbols in S^* . Basic transitions are restricted.

if $pS \xrightarrow{a} q\alpha \in T$ and $pS \xrightarrow{a} r\beta \in T$ and $a \in A \cup \{\varepsilon\}$, then $q = r$ and $\alpha = \beta$
if $pS \xrightarrow{\varepsilon} q\alpha \in T$ and $pS \xrightarrow{a} r\beta \in T$ then $a = \varepsilon$

A configuration of a DPDA has the form $p\delta$ where $p \in P$ and $\delta \in S^*$. The transitions of a configuration are determined by the following prefix rule: if $pS \xrightarrow{a} q\alpha \in T$ then $pS\beta \xrightarrow{a} q\alpha\beta$. The transition relation \xrightarrow{a} , $a \in A \cup \{\varepsilon\}$, is extended to words \xrightarrow{w} , $w \in A^*$. First, $p\alpha \xrightarrow{\varepsilon} p_n\alpha_n$, if $p_n = p$ and $\alpha_n = \alpha$ or there is a sequence of transitions $p\alpha \xrightarrow{\varepsilon} p_1\alpha_1 \xrightarrow{\varepsilon} \dots \xrightarrow{\varepsilon} p_n\alpha_n$. If $w = av \in A^+$, then $p\alpha \xrightarrow{w} q\beta$ if $p\alpha \xrightarrow{\varepsilon} p'\alpha' \xrightarrow{a} q'\beta' \xrightarrow{v} q\beta$. The language accepted by a configuration $p\delta$, $L(p\delta)$, is the set of words $\{w \in A^* : \exists q \in P. p\delta \xrightarrow{w} q\epsilon\}$. Acceptance is by empty stack. The DPDA problem is whether $L(p\alpha) = L(q\beta)$.

Moreover, one can assume that the DPDA is in normal form: if $pS \xrightarrow{a} q\alpha \in \mathsf{T}$, then $|\alpha| \leq 2$ and if $pS \xrightarrow{\varepsilon} q\alpha \in \mathsf{T}$ then $\alpha = \varepsilon$.

An important step in the decidability proof is a syntactic representation of DPDA configurations that dispenses with ε -transitions. The key is *non-deterministic* pushdown automata with a single state and without ε -transitions, introduced by Harrison and Havel [3] as grammars, and further studied in [2, 4]. Because the state is redundant, a configuration of a pushdown automaton with a single state is a sequence of stack symbols. Ingredients of such an automaton without ε -transitions, an SDA, are finite sets of stack symbols S , terminals A and basic transitions T of the form $S \xrightarrow{a} \alpha$ where $a \in \mathsf{A}$, $S \in \mathsf{S}$ and $\alpha \in \mathsf{S}^*$. A configuration of an SDA is a sequence of stack symbols whose transitions are determined by the prefix rule: if $S \xrightarrow{a} \alpha \in \mathsf{T}$, then $S\beta \xrightarrow{a} \alpha\beta$. The language $\mathsf{L}(\alpha)$ accepted, or generated, by a configuration α is the set $\{w \in \mathsf{A}^* : \alpha \xrightarrow{w} \varepsilon\}$, so acceptance is again by empty stack. We assume the SDA is in normal form: if $S \xrightarrow{a} \alpha \in \mathsf{T}$, then $|\alpha| \leq 2$ and no element of S is redundant ($S \in \mathsf{S}$ is redundant, if $\mathsf{L}(S) = \emptyset$).

If the SDA is deterministic, then the decision problem is decidable, as proved by Korenjak and Hopcroft [5]. However, the languages generable by deterministic SDA are strictly contained in those generable by DPDA. Instead of assuming determinism, Harrison and Havel include an extra component, \equiv , in the definition of an SDA that is an equivalence relation on stack symbols that partitions S . The relation, \equiv , on S is extended to a relation on sequences of stack symbols, and the same relation, \equiv , is used for the extension: $\alpha \equiv \beta$ if, either $\alpha = \beta$, or $\alpha = \delta X \alpha'$ and $\beta = \delta Y \beta'$ and $X \equiv Y$ and $X \neq Y$. Some simple properties of \equiv are: $\alpha\beta \equiv \alpha$ if, and only if, $\beta = \varepsilon$; $\alpha \equiv \beta$ if, and only if, $\delta\alpha \equiv \delta\beta$; if $\alpha \equiv \beta$ and $\gamma \equiv \delta$, then $\alpha\gamma \equiv \beta\delta$; if $\alpha \equiv \beta$ and $\alpha \neq \beta$, then $\alpha\gamma \equiv \beta\delta$; if $\alpha\gamma \equiv \beta\delta$ and $|\alpha| = |\beta|$, then $\alpha \equiv \beta$.

Definition 1 The relation \equiv on S is *strict* when the following two conditions hold. (1) If $X \equiv Y$ and $X \xrightarrow{a} \alpha$ and $Y \xrightarrow{a} \beta$, then $\alpha \equiv \beta$. (2) If $X \equiv Y$ and $X \xrightarrow{a} \alpha$ and $Y \xrightarrow{a} \alpha$, then $X = Y$. An SDA is *strict* (deterministic) if its partition is strict.

Proposition 1 Assume a strict SDA. (1) If $\alpha \xrightarrow{w} \alpha'$ and $\beta \xrightarrow{w} \beta'$ and $\alpha \equiv \beta$ then $\alpha' \equiv \beta'$. (2) If $\alpha \xrightarrow{w} \alpha'$ and $\beta \xrightarrow{w} \alpha'$ and $\alpha \equiv \beta$ then $\alpha = \beta$. (3) If $\alpha \equiv \beta$ and $w \in \mathsf{L}(\alpha)$, then for all words v , and $a \in \mathsf{A}$, $wav \notin \mathsf{L}(\beta)$. (4) If $\alpha \equiv \beta$ and $\alpha \neq \beta$, then $\mathsf{L}(\alpha) \cap \mathsf{L}(\beta) = \emptyset$.

The definition of a configuration of a strict SDA is extended to sets of sequences of stack symbols, $\{\alpha_1, \dots, \alpha_n\}$, written in sum form $\alpha_1 + \dots + \alpha_n$. Two sum configurations are equal, written using $=$, if they are the same set. A degenerate case is the empty sum, written \emptyset . The language of a sum configuration is defined using union: $\mathsf{L}(\alpha_1 + \dots + \alpha_n) = \bigcup \{\mathsf{L}(\alpha_i) : 1 \leq i \leq n\}$.

Definition 2 A sum configuration $\beta_1 + \dots + \beta_n$ is *admissible*, if $\beta_i \equiv \beta_j$ for each pair of components, and $\beta_i \neq \beta_j$ when $i \neq j$.

In [4] admissible configurations are called “associates”. A simple corollary of Proposition 1 is that admissibility is preserved by word transitions: if $\{\beta_1, \dots, \beta_n\}$ is admissible, then for any $w \in \mathbf{A}^*$, $\{\beta'_i : \beta_i \xrightarrow{w} \beta'_i, 1 \leq i \leq n\}$ is admissible.

A strict SDA can be determined, by determining the basic transitions \mathbf{T} to \mathbf{T}^d : for each stack symbol X and $a \in \mathbf{A}$, the transitions $X \xrightarrow{a} \alpha_1, \dots, X \xrightarrow{a} \alpha_n$ in \mathbf{T} are replaced by the single transition $X \xrightarrow{a} \alpha_1 + \dots + \alpha_n$ in \mathbf{T}^d . The resulting sum configuration is admissible. For each stack symbol X and $a \in \mathbf{A}$ there is a unique transition $X \xrightarrow{a} \sum \alpha_i \in \mathbf{T}^d$ (where the empty sum is \emptyset). The prefix rule for generating transitions is extended to admissible configurations: if $X_1\beta_1 + \dots + X_m\beta_m$ is admissible and $X_i \xrightarrow{a} \sum \alpha_{ij} \in \mathbf{T}^d$ for each i , then $X_1\beta_1 + \dots + X_m\beta_m \xrightarrow{a} \sum \alpha_{1j}\beta_1 + \dots + \sum \alpha_{mj}\beta_m$. The resulting configuration is admissible.

Admissible configurations of strict SDAs generate the same languages as configurations of DPDA with empty stack acceptance [3]. The DPDA problem is the same problem as deciding language (or, bisimulation) equivalence between admissible configurations of a determined strict SDA in normal form.

3 Heads, tails and extensions

Assume a fixed determined strict SDA in normal form with ingredients \mathbf{S} , \mathbf{A} , \mathbf{T}_d and \equiv . We assume a total ordering on \mathbf{A} , and we say that u is shorter than v if $|u| < |v|$ or $|u| = |v|$ and u is lexicographically smaller than v . Let E, F, G, \dots range over admissible configurations. The configuration E after the word u , $E \cdot u$, is the unique admissible configuration F such that $E \xrightarrow{u} F$. The language accepted by E , $\mathbf{L}(E)$, is $\{u : (E \cdot u) = \epsilon\}$. E and F are language equivalent, $E \sim F$, if they accept the same language, $\mathbf{L}(E) = \mathbf{L}(F)$. Language equivalence can also be approximated. If $n \geq 0$, then E and F are n -equivalent, $E \sim_n F$, provided that they reject the same words whose length is at most n : for all w such that $|w| \leq n$, $(E \cdot w) = \emptyset$ if, and only if, $(F \cdot w) = \emptyset$.

Proposition 1 (1) $E \sim F$ if, and only if, for all $n \geq 0$, $E \sim_n F$. (2) If $E \not\sim F$ and $E, F \neq \emptyset$, then there is an $n \geq 0$ such that $E \sim_n F$ and $E \not\sim_{n+1} F$. (3) $E \sim F$ if, and only if, for all $u \in \mathbf{A}^*$, $(E \cdot u) \sim (F \cdot u)$. (4) $E \sim_n F$ if, and only if, for all $u \in \mathbf{A}^*$ where $|u| \leq n$, $(E \cdot u) \sim_{n-|u|} (F \cdot u)$. (5) If $E \sim_n F$ and $0 \leq m < n$, then $E \sim_m F$. (6) If $E \sim_n F$ and $F \not\sim_n G$, then $E \not\sim_n G$.

Definition 1 (1) For each stack symbol X , the word $w(X)$ is the shortest word in the set $\{u : (X \cdot u) = \epsilon\}$. (2) The *norm* M of an SDA in normal form is $\max\{|w(X)| : X \in \mathbf{S}\}$.

The operation $+$, as used in sum form, can be extended: if E and F are admissible and $E \cup F$ is admissible and E, F are disjoint, $E \cap F = \emptyset$, then $E + F$ is the admissible configuration $E \cup F$. Sequential composition, written as juxtaposition, is also used: if E and F are admissible, then EF is the configuration $\{\beta\gamma : \beta \in E \text{ and } \gamma \in F\}$, that is admissible. Some properties are: if $E + F$ is admissible and $u \in \mathbf{L}(E)$, then $uv \notin \mathbf{L}(F)$; if $E + F$ is admissible,

then $\mathsf{L}(E) \cap \mathsf{L}(F) = \emptyset$; $\mathsf{L}(EF) = \{uv : u \in \mathsf{L}(E) \text{ and } v \in \mathsf{L}(F)\}$. Also, the following identities hold: $E + \emptyset = E = \emptyset + E$, $E\emptyset = \emptyset = \emptyset E$, $E\varepsilon = E = \varepsilon E$, $(E + F)G = EG + FG$ and $G(E + F) = GE + GF$. Admissible configurations can have different “shapes”, using $+$ and sequential composition.

Definition 2 Assume $k \geq 1$ and $E = \{\beta'_1\delta_1, \dots, \beta'_m\delta_m\}$ and $|\beta'_i| = k$, or $|\beta'_i| < k$ and $\delta_i = \varepsilon$, for each $i : 1 \leq i \leq m$. If β_1, \dots, β_n are the distinct elements in $\{\beta'_1, \dots, \beta'_m\}$ and $H_l = \{\delta_j : \beta'_j = \beta_l\}$ for each $l : 1 \leq l \leq n$, then $E = \beta_1 H_1 + \dots + \beta_n H_n$ is in k -head form.

Fact 1 If $E = \beta_1 G_1 + \dots + \beta_n G_n$ is in k -head form, then $\beta_1 + \dots + \beta_n$ is admissible and each G_i is admissible and different from \emptyset .

Proposition 2 Assume $E = \beta_1 G_1 + \dots + \beta_n G_n$ is in k -head form, and each $\beta_j = X_1^j \dots X_{k_j}^j$. If $(\beta_i \cdot u) = X_m^i \dots X_{k_i}^i$, then $(E \cdot u) = E_1 G_1 + \dots + E_n G_n$ where $E_i = (\beta_i \cdot u)$ and for $j \neq i$, either $E_j = \emptyset$ or $E_j = X_m^j \dots X_{k_j}^j$ and $X_1^j \dots X_{m-1}^j$ is the same sequence as $X_1^i \dots X_{m-1}^i$.

Definition 3 $E = E_1 G_1 + \dots + E_n G_n$ is in head/tail form, if the head $E_1 + \dots + E_n$ is admissible and at least one $E_i \neq \emptyset$, and each tail $G_i \neq \emptyset$.

If a configuration E is presented as $E_1 G_1 + \dots + E_n G_n$, then we assume that it fulfills the conditions of Definition 3 of a head/tail form. If E is $E_1 G_1 + \dots + E_n G_n$ and F is $F_1 G_1 + \dots + F_n G_n$, then the imbalance between E and F , relative to this presentation, is $\max\{|E_i|, |F_i| : 1 \leq i \leq n\}$.

Proposition 3 Assume $E = E_1 G_1 + \dots + E_n G_n$. (1) If $(E_i \cdot u) = \varepsilon$, then for all $j \neq i$, $(E_j \cdot u) = \emptyset$ and $(E \cdot u) = G_i$. (2) If $(E_i \cdot u) \neq \emptyset$, then $(E \cdot u) = (E_1 \cdot u)G_1 + \dots + (E_n \cdot u)G_n$. (3) If $H_i \neq \emptyset$ for all $i : 1 \leq i \leq n$, then $E_1 H_1 + \dots + E_n H_n$ is a head/tail form. (4) If each $H_i \neq \emptyset$ and each $E_i \neq \varepsilon$ and for each j such that $E_j \neq \emptyset$, $H_j \sim_m G_j$, then $E \sim_{m+1} E_1 H_1 + \dots + E_n H_n$. (5) If each $H_i \neq \emptyset$ and for each j such that $E_j \neq \emptyset$, $H_j \sim G_j$, then $E \sim E_1 H_1 + \dots + E_n H_n$.

Proposition 4 Assume $E = E_1 G_1 + \dots + E_n G_n$, $F = F_1 G_1 + \dots + F_n G_n$, $E' = E_1 H_1 + \dots + E_n H_n$ and $F' = F_1 H_1 + \dots + F_n H_n$. If $E \sim_m F$ and $E' \not\sim_m F'$, then there is a word u , $|u| \leq m$, and an i such that either 1 or 2. (1) $(E' \cdot u) = H_i$ and $(F' \cdot u) = (F_1 \cdot u)H_1 + \dots + (F_n \cdot u)H_n$ and $(F' \cdot u) \neq \emptyset$ and $(E' \cdot u) \not\sim_{m-|u|} (F' \cdot u)$. (2) $(F' \cdot u) = H_i$ and $(E' \cdot u) = (E_1 \cdot u)H_1 + \dots + (E_n \cdot u)H_n$ and $(E' \cdot u) \neq \emptyset$ and $(E' \cdot u) \not\sim_{m-|u|} (F' \cdot u)$.

Definition 4 If $E = E_1 G_1 + \dots + E_n G_n$ and $F = F_1 H_1 + \dots + F_m H_m$, then F in its head/tail form is a tail extension of E in its head/tail form provided that each $H_i = K_1^i G_1 + \dots + K_n^i G_n$, $1 \leq i \leq m$. When F is a tail extension of E , the associated extension e is the m -tuple $(K_1^1 + \dots + K_n^1, \dots, K_1^m + \dots + K_n^m)$ without the G_i s, and F is said to extend E by e .

Proposition 5 If $E = E_1 G_1 + \dots + E_l G_l$ and $E' = E'_1 G'_1 + \dots + E'_m G'_m$ and $E'' = E''_1 G''_1 + \dots + E''_n G''_n$ and E' extends E by $e = (J_1^1 + \dots + J_l^1, \dots, J_1^m + \dots + J_l^m)$ and E'' extends E' by $f = (K_1^1 + \dots + K_m^1, \dots, K_1^n + \dots + K_m^n)$, then E'' extends

E by $ef = (H_1^1 + \dots + H_l^1, \dots, H_1^n + \dots + H_l^n)$ where $H_j^i = K_1^i J_j^1 + \dots + K_m^i J_j^m$ and $|ef| \leq |e| + |f|$.

Extensions are matrices, written in a linear notation. A special case is when the tails are the same: if $E = E_1 G_1 + \dots + E_n G_n$ and $F = F_1 G_1 + \dots + F_n G_n$, then F extends E by $e = (\varepsilon + \emptyset + \dots + \emptyset, \dots, \emptyset + \emptyset + \dots + \varepsilon)$, and e is then abbreviated to the identity (ε).

4 The decision procedure

The procedure for deciding $E \sim F$ is to build a goal directed proof tree, a tableau, with initial goal $E \doteq F$, “is $E \sim F$?”, using proof rules that reduce goals to subgoals. There are just three rules. The first is UNF, for “unfold”.

$$\frac{E \doteq F}{(E \cdot a_1) \doteq (F \cdot a_1) \quad \dots \quad (E \cdot a_k) \doteq (F \cdot a_k)} \mathbf{A} = \{a_1, \dots, a_k\}$$

The goal is true, if, and only if, all the subgoals are true. A finer version of soundness uses approximants: if $E \not\sim_{m+1} F$, then at least one subgoal fails at level m , $(E \cdot a) \not\sim_m (F \cdot a)$. If $E' \doteq F'$ is a subgoal that is a result of m consecutive applications of UNF (and no other rule) to $E \doteq F$, then there is a word u such that $|u| = m$ and $E' = (E \cdot u)$ and $F' = (F \cdot u)$.

$$\begin{array}{l} \text{BAL(R)} \quad F \doteq X_1 H_1 + \dots + X_k H_k \\ \quad \quad \quad \vdots \\ \quad \quad \quad F' \doteq E_1 H_1 + \dots + E_k H_k \\ \text{BAL(L)} \quad \frac{F' \doteq E_1 (F \cdot w(X_1)) + \dots + E_k (F \cdot w(X_k))}{X_1 H_1 + \dots + X_k H_k} \doteq F \\ \quad \quad \quad \vdots \\ \quad \quad \quad \frac{E_1 H_1 + \dots + E_k H_k}{E_1 (F \cdot w(X_1)) + \dots + E_k (F \cdot w(X_k))} \doteq F' \end{array} \quad \text{C}$$

where C is the condition

1. Each $E_i \neq \varepsilon$ and at least one $H_i \neq \varepsilon$.
2. There are precisely $\max\{|w(X_i)| : E_i \neq \emptyset \text{ for } 1 \leq i \leq k\}$ applications of UNF between the top goal and the bottom goal, and no application of any other rule.
3. If u is the word associated with the sequence of UNFs, then $E_i = (X_i \cdot u)$ for each $i : 1 \leq i \leq k$.

Fig. 1. Balance rules

The balance rules in Figure 1 involve two premises: the second premise goal reduces to the (balanced) subgoal beneath it provided that the first premise is

above it (on the path back to the root goal). An application of BAL *uses* F , if F is the configuration in the initial goal of the rule, see Figure 1. The BAL rules are sound and complete.

Proposition 1 (1) *If $X_1H_1 + \dots + X_kH_k \sim F$ and $E_1H_1 + \dots + E_kH_k \sim F'$, then $E_1(F \cdot w(X_1)) + \dots + E_k(F \cdot w(X_k)) \sim F'$. (2) If $X_1H_1 + \dots + X_kH_k \sim_{n+m} F$ and $E_1H_1 + \dots + E_kH_k \not\sim_{n+1} F'$ and each $E_i \neq \varepsilon$ and $m \geq \max\{|w(X_i)| : E_i \neq \emptyset\}$, then $E_1(F \cdot w(X_1)) + \dots + E_k(F \cdot w(X_k)) \not\sim_{n+1} F'$.*

It is intended that there be a unique tableau associated with any initial goal. So there are restrictions on which rule is to be applied when. First, the initial premise of a BAL is the one that is “closest” to the goal and, therefore, the one that involves the least number of applications of UNF. To resolve which rule should be applied, the following priority order is assumed: (1) if BAL(L) is permitted, then apply BAL(L), (2) if BAL(R) is permitted, then apply BAL(R), (3) otherwise, apply UNF. However, whether an application of BAL is permitted involves more than fulfillment of the side condition. It also depends on the previous application of a BAL. The motivation for this restriction is Proposition 2, below. Initially, both BALs are permitted provided the side conditions hold. If an application of BAL uses F , then the resulting goal contains the configuration $E_1(F \cdot w(X_1)) + \dots + E_k(F \cdot w(X_k))$. E_i is a “top” of the application of BAL and $(F \cdot w(X_i))$ is a “bottom”. Assume an application of BAL(L). A subsequent application of BAL(L) is permitted provided the side condition of the rule is fulfilled. However, BAL(R) is not permitted until a bottom of the previous application of BAL(L) is exposed and the side condition of the rule is true. Between the application of BAL(L) and the goal $G_1 \doteq H_1$, below,

$$\begin{array}{r}
F \\
\vdots \\
E_1(F \cdot w(X_1)) + \dots + E_k(F \cdot w(X_k)) \doteq H \\
\vdots \\
(F \cdot w(X_i)) = G_1 \doteq H_1
\end{array}
\begin{array}{l}
\\
\text{BAL(L)} \\
\\
\text{UNFs} \\
\\
\end{array}$$

there are no other applications of BAL(L), and G_1 is a bottom, $(F \cdot w(X_i))$, of the previous application of BAL(L). BAL(R) is now permitted provided it uses configuration G_1 , or a configuration beneath it, when the side condition holds. BAL(R) is not permitted using a configuration from a goal above $G_1 \doteq H_1$, even when the side condition is true. The strategy is to apply a BAL rule whenever it is permitted, and if both BAL rules are permitted, then priority lies with BAL(L). If BAL(R) is applied, then the strategy is to repeatedly apply BAL(R), and to use UNF otherwise. BAL(L) is only permitted once a bottom of the previous application of BAL(R) becomes the right hand configuration of a goal and the side condition holds. One consequence is that when building a tableau proof tree, there is just one choice of which rule to apply next to any subgoal. A branch of a tableau from a subgoal $g(0)$ is a sequence of goals that start from $g(0)$.

Theorem 1 [The extension theorem] *Assume there are two families of goals $g(i), h(i), 1 \leq i \leq 2^{n-k}$, and each goal $g(i)$ has the form $E_1G_1^i + \dots + E_nG_n^i \doteq F_1G_1^i + \dots + F_nG_n^i$ and each goal $h(i)$ has the form $E_1H_1^i + \dots + E_nH_n^i \doteq F_1H_1^i + \dots + F_nH_n^i$, and k is the number of distinct knowns for the family $g(i) \cup h(i)$ at level m . Assume extensions e_1, \dots, e_{n-k} such that for each e_j and $i \geq 0$ $g(2^j i + 2^{j-1} + 1)$ extends $g(2^j i + 2^{j-1})$ by e_j and $h(2^j i + 2^{j-1} + 1)$ extends $h(2^j i + 2^{j-1})$ by e_j . If each goal $g(i)$ is true at level $m, i : 1 \leq i \leq 2^{n-k}$, and each goal $h(j), j : 1 \leq j < 2^{n-k}$, is true at level m , then $h(2^{n-k})$ is true at level m .*

Proof: The proof is by induction on $n - k$. For the base case assume $n - k = 0$. Assume that there are two goals $g(1), E_1G_1 + \dots + E_nG_n \doteq F_1G_1 + \dots + F_nG_n$, abbreviated to $E \doteq F$, and $h(1), E_1H_1 + \dots + E_nH_n \doteq F_1H_1 + \dots + F_nH_n$, abbreviated to $E' \doteq F'$, and n distinct knowns for $\{g(1), h(1)\}$ at level m . That is, for each $i : 1 \leq i \leq n$ either $G_i \sim_m G_j$ and $H_i \sim_m H_j$ and $j < i$, or $G_i \sim_m J_1G_1 + \dots + J_nG_n$ and $H_i \sim_m J_1H_1 + \dots + J_nH_n$ where $J_i \neq \varepsilon$. We prove that if $g(1)$ is true at level $m, E \sim_m F$, then $h(1)$ is also true at level $m, E' \sim_m F'$. Suppose not. Then, without loss of generality, by Proposition 4 of Section 3 there is a word $u, |u| \leq m$, and $(E' \cdot u) = H_i$ and $(F' \cdot u) = (F_1 \cdot u)H_1 + \dots + (F_n \cdot u)H_n$ and $(F' \cdot u) \neq \emptyset$ and $(E' \cdot u) \not\sim_{m-|u|} (F' \cdot u)$. However, because $E \sim_m F$ it follows that $G_i \sim_{m-|u|} (F_1 \cdot u)G_1 + \dots + (F_n \cdot u)G_n$. The first case is that $G_i \sim_m G_j$ and $H_i \sim_m H_j$ where $j < i$. Therefore, using Proposition 1 of Section 3, $G_j \sim_{m-|u|} (F_1 \cdot u)G_1 + \dots + (F_n \cdot u)G_n$ and $H_j \not\sim_{m-|u|} (F_1 \cdot u)H_1 + \dots + (F_n \cdot u)H_n$. This can only be repeated finitely many times, $G_j \sim_m G_{j'}$ and $H_j \sim_m H_{j'}$, and $j' < j$. Hence, the second case must then occur, $G_i \sim_m J_1G_1 + \dots + J_nG_n$ and $H_i \sim_m J_1H_1 + \dots + J_nH_n$ and no $J_i = \varepsilon$. Using Proposition 1 of Section 3, $J_1G_1 + \dots + J_nG_n \sim_{m-|u|} (F_1 \cdot u)G_1 + \dots + (F_n \cdot u)G_n$ and $J_1H_1 + \dots + J_nH_n \not\sim_{m-|u|} (F_1 \cdot u)H_1 + \dots + (F_n \cdot u)H_n$. Now Proposition 4 of Section 3, as above, is applied again. Therefore, by repeating this argument a contradiction will be obtained.

For the general step, assume that it holds for $n - k < t$ and consider $n - k = t$. So, there are two families of goals $g(i), h(i), 1 \leq i \leq 2^{n-k}$, and each goal $g(i)$ has the form $E_1G_1^i + \dots + E_nG_n^i \doteq F_1G_1^i + \dots + F_nG_n^i$ and each goal $h(i)$ has the form $E_1H_1^i + \dots + E_nH_n^i \doteq F_1H_1^i + \dots + F_nH_n^i$, and k is the number of distinct knowns for the family $g(i) \cup h(i)$ at level m . Assume extensions e_1, \dots, e_{n-k} such that for each e_j and $i \geq 0, g(2^j i + 2^{j-1} + 1)$ extends $g(2^j i + 2^{j-1})$ by e_j and $h(2^j i + 2^{j-1} + 1)$ extends $h(2^j i + 2^{j-1})$ by e_j . Assume that each goal $g(i)$ is true at level $m, i : 1 \leq i \leq 2^{n-k}$, and each goal $h(j), j : 1 \leq j < 2^{n-k}$, is true at level m . The aim is to show that $h(2^{n-k})$ is also true at level m . Suppose not. Let $q = 2^{n-k}$. Therefore, $E_1H_1^q + \dots + E_nH_n^q \not\sim_m F_1H_1^q + \dots + F_nH_n^q$ and $E_1G_1^q + \dots + E_nG_n^q \sim_m F_1G_1^q + \dots + F_nG_n^q$. Without loss of generality, by Proposition 4 of Section 3 there is a word $u, |u| \leq m$, and $(E_1H_1^q + \dots + E_nH_n^q \cdot u) = H_i^q$ and $(F_1H_1^q + \dots + F_nH_n^q \cdot u) = (F_1 \cdot u)H_1^q + \dots + (F_n \cdot u)H_n^q$ and $H_i^q \not\sim_{m-|u|} (F_1 \cdot u)H_1^q + \dots + (F_n \cdot u)H_n^q$ and for all $j : 1 \leq j \leq q, G_i^j \sim_{m-|u|} (F_1 \cdot u)G_1^j + \dots + (F_n \cdot u)G_n^j$ and for all $j : 1 \leq j < q, H_i^j \sim_{m-|u|} (F_1 \cdot u)H_1^j + \dots + (F_n \cdot u)H_n^j$. There are two cases. First

is that each G_i^j and H_i^j , $1 \leq j \leq q$, is a known at level m . The proof proceeds as in the base case, but here with respect to the whole family of goals. Therefore, at some point the second case happens, that G_i^j and H_i^j are not knowns. However, each even goal $g(2j)$ and $h(2j)$ extends the previous goal $g(2j-1)$ and $h(2j-1)$ by e_1 . Therefore, each $G_i^{2j} \sim_{m-|u|} (F_1 \cdot u)G_1^{2j} + \dots + (F_n \cdot u)G_n^{2j}$ becomes $J_1 G_1^{2j-1} + \dots + J_n G_n^{2j-1} \sim_{m-|u|} F'_1 G_1^{2j-1} + \dots + F'_n G_n^{2j-1}$ by substituting in the entries of e_1 , and $J_1 H_1^{2j-1} + \dots + J_n H_n^{2j-1} \sim_{m-|u|} F'_1 H_1^{2j-1} + \dots + F'_n H_n^{2j-1}$ when $j < (q-1)$ and $J_1 H_1^{q-1} + \dots + J_n H_n^{q-1} \not\sim_{m-|u|} F'_1 H_1^{q-1} + \dots + F'_n H_n^{q-1}$. Let these goals be $g'(i)$, $h'(i)$, $1 \leq i \leq 2^{n-(k+1)}$ and consider the extensions $e'_1, \dots, e'_{n-(k+1)}$ where $e'_i = e_1 e_{i+1}$. Using Proposition 5 of Section 3, for each e'_j and $i \geq 0$, $g'(2^j i + 2^{j-1} + 1)$ extends $g'(2^j i + 2^{j-1})$ by e'_j and the same for the $h'(i)$ s. Moreover, the number of knowns is now $(k+1)$ because the previous k knowns at level m remain knowns at level $m-|u|$ and there is the new known at level $m-|u|$ because for each j , $G_i^j \sim_{m-|u|} (F_1 \cdot u)G_1^j + \dots + (F_n \cdot u)G_n^j$ and $H_i^j \sim_{m-|u|} (F_1 \cdot u)H_1^j + \dots + (F_n \cdot u)H_n^j$. By the induction hypothesis if every $g'(i)$, $1 \leq i \leq 2^{n-(k+1)}$ is true at level $m-|u|$ and every $h'(i)$, $1 \leq i < 2^{n-(k+1)}$, is true at level $m-|u|$, then $h'(2^{n-(k+1)})$ is also true at level $m-|u|$, which contradicts that $J_1 H_1^{q-1} + \dots + J_n H_n^{q-1} \not\sim_{m-|u|} F'_1 H_1^{q-1} + \dots + F'_n H_n^{q-1}$. \square

Definition 2 Assume $d(0), \dots, d(l)$ is a branch of goals and $d(l)$ is $E_1 H_1 + \dots + E_n H_n \doteq F_1 H_1 + \dots + F_n H_n$. Goal $d(l)$ obeys the extension theorem if the following hold: (1) There are families of goals $g(i)$, $h(i)$, $1 \leq i \leq 2^n$ belonging to $\{d(0), \dots, d(l)\}$, and each goal $g(i)$ has the form $E_1 G_1^i + \dots + E_n G_n^i \doteq F_1 G_1^i + \dots + F_n G_n^i$ and each goal $h(i)$ has the form $E_1 H_1^i + \dots + E_n H_n^i \doteq F_1 H_1^i + \dots + F_n H_n^i$. (2) The goal $h(2^n)$ is $d(l)$ and there is at least one application of UNF between goal $h(2^n - 1)$ and $d(l)$. (3) There are extensions e_1, \dots, e_n such that for each e_j and $i \geq 0$, $g(2^j i + 2^{j-1} + 1)$ extends $g(2^j i + 2^{j-1})$ by e_j and $h(2^j i + 2^{j-1} + 1)$ extends $h(2^j i + 2^{j-1})$ by e_j .

Definition 3 If $g(0), \dots, g(n)$ where $g(0)$ is the root goal is a branch of goals, then $g(n)$ is a *final goal* in the following circumstances. (1) If $g(n)$ is an identity $E \doteq E$, then $g(n)$ is a successful final goal. (2) If $g(n)$ obeys the extension theorem, then $g(n)$ is a successful final goal. (3) If $g(n)$ has the form $E \doteq \emptyset$ or $\emptyset \doteq E$ and $E \neq \emptyset$, then $g(n)$ is an unsuccessful final goal.

The deterministic procedure that decides whether $E \sim F$ is defined iteratively.

1. Stage 0: start with the root goal $g(0)$, $E \doteq F$, that becomes a frontier node of the branch $g(0)$.
2. Stage $n+1$: if a current frontier node $g(n)$ of branch $g(0), \dots, g(n)$ is an unsuccessful final goal, then halt and return “unsuccessful tableau”; if each frontier node $g(n)$ of branch $g(0), \dots, g(n)$ is a successful final goal, then return “successful tableau”; otherwise, for each frontier node $g(n)$ of branch $g(0), \dots, g(n)$ that is not a final goal, apply the next rule to it, and the subgoals that result are the new frontier nodes of the extended branches.

5 Correctness and complexity of the decision procedure

For the complexity upper bound, some basis functions are introduced assuming a fixed SDA with s stack symbols and largest partition p : a DPDA in normal form with s_1 stack symbols and p_1 states is transformed into an SDA with at most $s_1 \times p_1^2$ stack symbols and whose largest partition is no more than p_1 . The size of an admissible configuration $E = \beta_1 + \dots + \beta_n$, written $|E|$, is $\max\{|\beta_i| : 1 \leq i \leq n\}$. Let $\text{goal}(h)$ be the number of different goals $E \doteq F$ such that $|E|, |F| \leq h$. Let $\text{width}(h)$ be the maximum n of an admissible configuration $\beta_1 + \dots + \beta_n$ such that $|\beta_i| \leq h$ (so, $\text{width}(h) \leq p^h$). The function $\text{ext}(d, w)$ is the number of different extensions e such that $|e| \leq d$ and e has width at most w (so, $\text{ext}(d, w) \leq 2^{s^{dw}}$). The other measure used is the norm M that is bounded by 2^s . Some auxiliary functions are now defined from the basis functions.

Definition 1 (1) The function $f_1(k) = M(k + 1 + M^2 + 2M) + \text{goal}(M)$. (2) The function $f_2[d, h, w](n)$ is defined recursively: $f_2[d, h, w](0) = \text{goal}(h)$, $f_2[d, h, w](j+1) = \text{ext}(f_2[d, h, w](j) \times d, w) \times f_2[d, h, w](j)$. (3) Let $d = M^2 + 2M$, $h = M^2 + 5M + 2$, $w = \text{width}(h)$ and $b = f_2[d, h, w](w)$. Then, $f(n) = n \times d^b \times f_1(n + bd)$.

Theorem 1 *If $E \not\sim F$, then the decision procedure terminates with “unsuccessful tableau”.*

In the following result, symmetric properties hold if BAL(R) replaces BAL(L) in parts 2 and 4.

Proposition 1 *Assume that $g(0), \dots, g(n)$ is a sequence of goals in a branch of a tableau and $g(0)$ is $E \doteq F$. (1) If $g(0)$ is the root goal and $E = \beta_1 G_1 + \dots + \beta_m G_m$ and $F = \delta_1 H_1 + \dots + \delta_l H_l$ are in k -head form and $|E|, |F| > k$, and there is no application of BAL between $g(0)$ and $g(kM)$, then there are goals $g(i)$ and $g(j)$, $1 \leq i, j \leq kM$, such that $g(i)$ is $G_{i'} \doteq F'$ and $g(j)$ is $E' \doteq H_{j'}$ for some i' and j' . (2) If $g(0)$ is a result of BAL(L) using F' and there is no application of a BAL between $g(0)$ and $g(k)$ where $k = M^2 + M$ and $E = E_1(F' \cdot w(X_1)) + \dots + E_k(F' \cdot w(X_k))$, then there is a goal $g(i)$, $i : 1 \leq i \leq k$, that has the form $(F' \cdot w(X_j)) \doteq F''$. (3) If $g(0)$ is a result of BAL using F' and the next application of BAL is $g(k)$ using F'' , then $|F''| \leq |F'| + M^2 + 2M$. (4) If $g(0)$ is a result of BAL(L) using F' and the next application of BAL is $g(k)$ using F'' and $k \geq M^3 + 3M^2 + 3M$, then $|F''| < |F'|$.*

Lemma 1 *Assume that $g(0), \dots, g(n)$ is a sequence of goals in a branch and $g(0)$ is $E \doteq F$ and there are no applications of BAL in this branch. If $|E|, |F| \leq k$, and $n \geq f_1(k)$, then the branch contains a successful final goal.*

Lemma 1 establishes termination of the decision procedure for a sufficiently long branch of goals that does not involve applications of BAL. An analysis is now developed for branches that involve repeated applications of BAL.

Definition 2 Assume F_0, F_1, \dots, F_t are successive configurations used in applications of BAL in a branch and assume words u_1, \dots, u_t such that $F_i =$

$(F_0 \cdot u_1 \dots u_i)$. Let $F_i = \beta_1 G_1 + \dots + \beta_n G_n$ be in m -head form, $m \geq 1$. F_i is m -low in the interval $F_j F_{j+1} \dots F_{j+k}$, $j \geq i$ and $j+k \leq t$, if there is a prefix v of $u_{j+1} \dots u_{j+k}$ such that $(F_i \cdot u_{i+1} \dots u_j v) = G_l$ for some l , and F_i is then said to be m -low with G_l using v in this interval. The definition is extended to 0-low: F_i is 0-low with F_i using ε in the interval $F_j F_{j+1} \dots F_{j+k}$ provided that $j = i$ and F_i is not m -low for any $m > 0$ in this interval.

Proposition 2 Assume F_0, F_1, \dots, F_t are successive configurations used in applications of BAL in a branch and assume words u_1, \dots, u_t such that $F_i = (F_0 \cdot u_1 \dots u_i)$. Assume F_i is m_1 -low in $F_j F_{j+1}$, $m_1 \geq 0$ and $j \geq i$ with G using v , and F_i is not m'_1 -low for any $m'_1 > m_1$ in $F_j F_{j+1} \dots F_t$. Assume F_{j+1} is m_2 -low in $F_l F_{l+1}$, $j+1 \leq l < t$, and $m_2 \geq 0$, with G' and F_{j+1} is not m'_2 -low in $F_l F_{l+1}$ for any $m'_2 > m_2$. (1) If $G = X_1 H_1 + \dots + X_k H_k$ is in 1-head form and $u_{j+1} = vv'$, then for all prefixes w of $v' u_{j+1} \dots u_t$, $(G \cdot w) = (X_1 \cdot w) H_1 + \dots + (X_k \cdot w) H_k$. (2) $|F_{j+1}| \leq |G| + (M^2 + 2M)$ and $m_2 \leq (M^2 + 2M)$. (3) If $G = \beta_1 G_1 + \dots + \beta_n G_n$ is in $M+1$ -head form, then the goal that is the result of BAL using F_{j+1} has the head/tail form $E_1 G_1 + \dots + E_n G_n \doteq F_1 G_1 + \dots + F_n G_n$, where $|E_i|, |F_i| \leq M^2 + 5M + 2$. (4) If $G = \beta_1 G_1 + \dots + \beta_n G_n$ and $G' = \delta_1 G'_1 + \dots + \delta_{n'} G'_{n'}$ are in $M+1$ -head form, then in these forms G' is an extension of G with extension e and $|e| \leq M^2 + 2M$.

Lemma 2 Assume that there is a subsequence of goals in a branch $g(i)$, $E_1^i G_1^i + \dots + E_{n_i}^i G_{n_i}^i \doteq F_1^i G_1^i + \dots + F_{n_i}^i G_{n_i}^i$, $i : 0 \leq i \leq t$ and each $g(j+1)$ extends $g(j)$ by e_j where $|e_j| \leq d$ and each head $|E_j^i|, |F_j^i| \leq h$. If $w = \text{width}(h)$ and $t \geq f_2[d, h, w](w)$, then there is a successful final goal in the branch.

Theorem 2 (1) If $E \sim F$, then the decision procedure terminates with “successful” tableau. (2) If $|E|, |F| < n$, then the decision procedure with root $E \doteq F$ terminates within $f(n)$ steps.

Proof: Assume that $E \sim F$. The tableau for $E \doteq F$ is built using the rules of the proof system. By completeness of rule application at each stage each frontier goal is true. Therefore, it is not possible to reach an unsuccessful final goal. It is also not possible to become stuck because UNF can always apply. Hence the only issue is that the decision procedure doesn't terminate. Assume that there is an infinite branch of goals $g(0), \dots, g(n), \dots$ where $g(0)$ is the root goal $E \doteq F$ that does not contain a successful final goal. If there are only finitely many applications of BAL, then there is an infinite suffix of goals $g(i), \dots, g(n), \dots$ and there are no applications of BAL. However, by Lemma 1 there must be a successful final goal. Otherwise there are infinitely many applications of BAL. Let $F_0, F_1 \dots$ be the successive configurations used in applications of BAL. Start with F_0 and find the largest m_0 and the first interval $F_{i_1-1} F_{i_1}$ such that F_0 is m_0 -low in this interval. By assumption, F_0 is not m' -low, for any $m' > m_0$ in any later interval. Next, consider F_{i_1} . Find the largest m_1 and the first interval $F_{i_2-1} F_{i_2}$, $i_2 > i_1$, such that F_1 is m_1 -low in this interval. Using Proposition 2, the application of BAL using F_{i_1} , the goal $g(k_1)$, has a head/tail form $E_1^1 G_1^1 + \dots + E_{n_1}^1 G_{n_1}^1 \doteq F_1^1 G_1^1 + \dots + F_{n_1}^1 G_{n_1}^1$ and the application of BAL using F_{i_2} , the

goal $g(k_2)$, has head/tail form $E_1^2 G_1^2 + \dots + E_{n_2}^2 G_{n_2}^2 \doteq F_1^2 G_1^2 + \dots + F_{n_2}^2 G_{n_2}^2$ where $|E_j^i|, |F_j^i| \leq h = M^2 + 5M + 2$ and $g(k_2)$ extends $g(k_1)$ by e_1 where $|e_1| \leq d = M^2 + 2M$. The argument is now repeated giving an infinite subsequence of goals $g(k_1), g(k_2), \dots, g(k_n), \dots$ such that each $g(k_{i+1})$ is an extension of $g(k_i)$ by e_i with $|e_i| \leq d$, and the heads have bounded size no more than h . Therefore, by Lemma 2 there is a successful final goal within this subsequence. For part 2, the argument above is refined. Assume a branch $g(0), \dots, g(t)$ where $g(0)$ is the root $E \doteq F$, and $|E|, |F| \leq n$ and $t \geq f(n)$. If there are no applications of BAL, then by Lemma 1 there is a successful final goal within $f_1(n)$ steps and $f_1(n) \leq f(n)$. Let F_0, F_1, \dots, F_l be successive configurations used in applications of BAL. Let d and h be as defined above and let $w = \text{width}(h)$ and let $b = f_2[d, h, w](w)$. Via Proposition 2, if no F_i has a low point that is greater than 0, then for $l \leq b$ the goals that are the result of these applications of BAL obey Lemma 2. By Proposition 1, $|F_{i+1}| \leq |F_i| + d$. Hence, it follows that the largest F_i is bounded by $u = n + bd$. Moreover, it follows that the maximum number of steps between F_i and F_{i+1} is at most $f_1(u)$ (because, otherwise there must be a successful final goal using Lemma 1). For the general case let $f'(b)$ be the number of successive configurations used in applications of BAL such that there is a subsequence of length b such that the goals that are the result of these applications of BAL obey Lemma 2. F_0 may have n low points. Therefore, $f'(b) = n \times f'(b-1)$. Subsequent F_i have only d low points, so $f'(b-1) = d \times f'(b-2)$. The overall bound is therefore $n \times d^b$. The maximum number of steps between F_i and F_{i+1} is again at most $f_1(u)$. Therefore, within $f(n)$ steps it follows that the decision procedure must terminate. \square

References

1. Ginsberg, S., and Greibach, S. (1966). Deterministic context-free languages. *Information and Control*, 620-648.
2. Harrison, M. (1978). *Introduction to Formal Language Theory*, Addison-Wesley.
3. Harrison, M., and Havel, I. (1973). Strict deterministic grammars. *Journal of Computer and System Sciences*, **7**, 237-277.
4. Harrison, M., Havel, I., and Yehudai, A. (1979). On equivalence of grammars through transformation trees. *Theoretical Computer Science*, **9**, 173-205.
5. Korenjak, A and Hopcroft, J. (1966). Simple deterministic languages. *Procs. 7th Annual IEEE Symposium on Switching and Automata Theory*, 36-46.
6. Sénizergues, G. (1997). The equivalence problem for deterministic pushdown automata is decidable. *Lecture Notes in Computer Science*, **1256**, 671-681.
7. Sénizergues, G. (2001). $L(A) = L(B)$? decidability results from complete formal systems. *Theoretical Computer Science*, **251**, 1-166.
8. Sénizergues, G. (2001). $L(A) = L(B)$? a simplified decidability proof. pp.1-58. To appear in *Theoretical Computer Science*.
9. Stirling, C. (2001). Decidability of DPDA equivalence. *Theoretical Computer Science*, **255**, 1-31.
10. Stirling, C. (2001). An introduction to decidability of DPDA equivalence. *Lecture Notes in Computer Science*, **2245**, 42-56.