Decidability and Complexity of DPDA Language Equivalence via 1st Order Grammars

Petr Jančar

Dept of Computer Science
Technical University Ostrava (FEI VŠB-TUO), Czech Republic
www.cs.vsb.cz/jancar

Talk at the event (workshop)
Pushdown Automata and Vector Addition Systems:
A New Look At Two Classical Problems
LSV, ENS Cachan, France
http://www.lsv.ens-cachan.fr/Events/Pavas/
20 January 2011

(This is a modified version of the slides used at the talk, with an added summary and further remarks at the end.)

Language equivalence of deterministic pushdown automata

Example of a (formal) language L over a finite alphabet Σ, so $L \subseteq \Sigma^*$:

$\Sigma = \{0,1,2,3,4,5,6,7,8,9,+,-,*,(,)\} \cup \{\dashv\}$

L ... language of arithmetic expressions

e.g., the word (sequence)

$u = \boxed{5+28*(318-5*24)+562\dashv}$ is in L,

$v = \boxed{5+28*(318-(5*24)+562\dashv}$ is not in L

We can view a deterministic pushdown automaton M as a program

- with fixed finite memory; program+memory...finite control unit,
- with a potentially unbounded stack (LIFO, access to the top),
- reading the input word from left-to-right,
- accepting when reading the endmarker $\dashv$ and having the stack empty.

Decidability of $L(M_1) \stackrel{?}{=} L(M_2)$ was open since 1960s (stated in a paper by Ginsburg, Greibach). Another formulation: $L(p\alpha) \stackrel{?}{=} L(q\beta)$ for configurations of the same M (p ... control state, α ... stack content).

Solution (for DPDA language equivalence)

- Sénizergues G.:
  L(A)=L(B)? Decidability results from complete formal systems.
  Theoretical Computer Science 251(1-2): 1-166 (2001)
  (a preliminary version appeared at ICALP'97; Gödel prize 2002)
- Stirling C.: Decidability of DPDA equivalence.
  Theoretical Computer Science 255, 1-31, 2001
- Sénizergues G.: L(A)=L(B)? A simplified decidability proof.
  Theoretical Computer Science 281(1-2): 555-608 (2002)
- Stirling C.: Deciding DPDA equivalence is primitive recursive.
  ICALP 2002, Lecture Notes in Computer Science 2380, 821-832,
  Springer 2002 (longer draft paper on the author's web page, 38 pages)

- Sénizergues G.: The Bisimulation Problem for Equational Graphs of Finite Out-Degree. SIAM J.Comput., 34(5), 1025–1106 (2005)
  (a preliminary version appeared at FOCS'98)

Outline (of a novel presentation of the decidability)

- Reduction to trace equivalence $p\alpha \stackrel{?}{\sim} q\beta$ for a given (ε-popping) dpda.
- An approach for deciding trace equiv. (or bisimilarity) on deterministic labelled transition systems where states are structured objects.
- Crucial notions and ideas:

  - offending words, i.e. the shortest traces w witnessing $\mathcal{O}_0 \not\sim \mathcal{O}'_0$ for the given initial pair $(\mathcal{O}_0, \mathcal{O}'_0)$ (if nonequivalent);

  - tools for recognizing "non-offending (prefixes of) traces":

    - finite basis $\mathcal{B}$ containing "schemas" (templates, shapes, ...) $(E(x_1, \dots, x_n), F(x_1, \dots, x_n))$ of pairs of (equivalent) objects;

    - (sub)object replacement $E(\mathcal{O}_1) \rightarrow E(\mathcal{O}_2)$, for easier recognizing non-offending prefixes (by creating basis-instances).
- Soundness: Success for $(\mathcal{O}_0, \mathcal{O}'_0)$ and all (worst) instances of schemas in $\mathcal{B}$ implies $\mathcal{O}_0 \sim \mathcal{O}'_0$. (Stratification $\sim_0 \supseteq \sim_1 \supseteq \cdots \supseteq \sim$; congruence).
- Completeness: determ-1st-order grammars have sufficient bases $\mathcal{B}$; objects = terms = ordered-trees; we also allow regular infinite trees.
- Dpda configurations can be easily transformed to terms=trees.

Possible additional remarks

- An upper complexity bound for det-1st-order grammar trace equivalence (and dpda language equivalence):

  $\boxed{2 \uparrow\uparrow g(n)}$

  where g is an elementary function
  and $\uparrow\uparrow$ denotes tetration (iterated exponentiation).
  (The bound applies to the length of offending words.)
- Decidability of bisimulation equivalence for the general nondeterministic 1st order grammars
  (or nondeterministic pda with restricted use of ε-steps).

Note. This presentation is based on the paper made public at http://arxiv.org/abs/1010.4760 (version 3 from December 2010) but with some new modifications.

Note. In the 1st version of the slides put here (LSV-page), Slide 49 contained some remarks to the above case for nondeterministic grammars, which I also tried to handle in my arxiv-paper. Géraud Sénizergues found a serious bug in this part. More information is on the new Slide 49.

Trace equivalence on LTSs ; deterministic LTSs

A labelled transition system (LTS) is a tuple
$(\mathcal{S}, \mathcal{A}, \{\xrightarrow{a}\}_{a\in\mathcal{A}})$ where $\xrightarrow{a} \subseteq \mathcal{S} \times \mathcal{S}$.
We assume the action set $\mathcal{A}$ finite, while the state set $\mathcal{S}$ can be infinite.
$\mathcal{O} \in \mathcal{S}$ enables (trace) $w = a_1 a_2 \ldots a_m \in \mathcal{A}^*$, denoted $\boxed{\mathcal{O} \xrightarrow{w}}$,
if $\mathcal{O} \xrightarrow{a_1} \mathcal{O}_1 \xrightarrow{a_2} \mathcal{O}_2 \cdots \xrightarrow{a_m} \mathcal{O}_m$ for some $\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_m$.
$\mathrm{TRAC}(\mathcal{O}) = \{w \in \mathcal{A}^* \mid \mathcal{O} \xrightarrow{w}\}$ (the action sequences w enabled by $\mathcal{O}$).
Trace equivalence: $\boxed{\mathcal{O}_1 \sim \mathcal{O}_2} \Leftrightarrow_{df} \boxed{\mathrm{TRAC}(\mathcal{O}_1) = \mathrm{TRAC}(\mathcal{O}_2)}$.

LTS $(\mathcal{S}, \mathcal{A}, \{\xrightarrow{a}\}_{a\in\mathcal{A}})$ is deterministic (a det-LTS)
if each $\xrightarrow{a}$ is a partial function
(if $\mathcal{O} \xrightarrow{a} \mathcal{O}_1$ and $\mathcal{O} \xrightarrow{a} \mathcal{O}_2$ then $\mathcal{O}_1 = \mathcal{O}_2$).

Example of a (finite) det-LTS, with $\mathcal{S} = \{p, q, r, s, t, D\}$, $\mathcal{A} = \{a, b, c\}$:

$$\boxed{\begin{array}{l} p \xrightarrow{a} p,\; p \xrightarrow{b} r,\; q \xrightarrow{a} q,\; q \xrightarrow{b} r \\ r \xrightarrow{c} r \\ s \xrightarrow{a} t,\; s \xrightarrow{b} r,\; t \xrightarrow{a} D,\; t \xrightarrow{b} t \end{array}}$$

$p \overset{?}{\sim} s$ (NO: $p \xrightarrow{aaa}$, $\neg(s \xrightarrow{aaa})$)

$p \overset{?}{\sim} q$ (YES, why ?)

Basis. Schemas (for objects, object-pairs, transition rules).

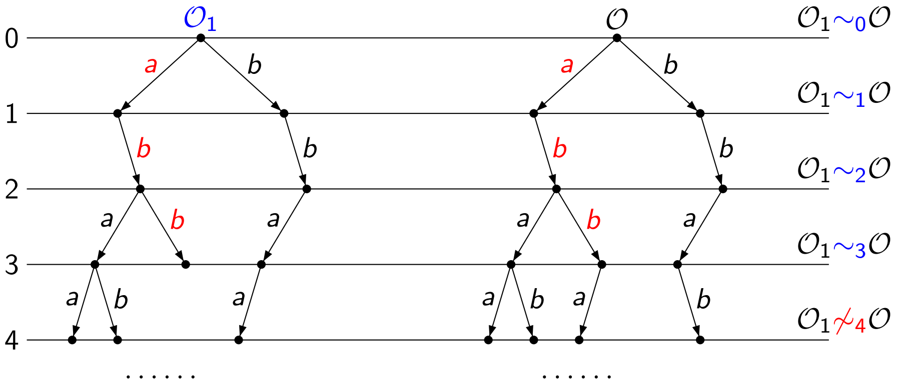
For (a generator $\mathcal{G}$ of) an LTS $(\mathcal{S}_{\mathcal{G}}, \mathcal{A}, \{\xrightarrow{a}\}_{a\in\mathcal{A}})$, we aim to suggest
a finite basis $\mathcal{B} = \{\, (x_1, x_1), \dots \,\}$ containing
pair-schemas (templates, shapes, ...) $(E(x_1, \dots, x_n), F(x_1, \dots, x_n))$.

Given a set $\mathcal{S}$ of objects:

- An (object-)schema $\boxed{E(x_1, \dots, x_n)}$ is a function $E : \mathcal{S}^n \to \mathcal{S}$.
  We sometimes use bracket-free notation $Yx_1 \dots x_n$ for $Y(x_1, \dots, x_n)$.
- E represents the subset $\textsc{Inst}(E) \subseteq \mathcal{S}$ of instances $\boxed{E(\mathcal{O}_1, \dots, \mathcal{O}_n)}$.
- The instances of a pair-schema $(E(x_1, \dots, x_n), F(x_1, \dots, x_n))$ are the pairs $(E(\mathcal{O}_1, \dots, \mathcal{O}_n), F(\mathcal{O}_1, \dots, \mathcal{O}_n))$.
- $\textsc{Inst}(\mathcal{B})$ is the set of instances of pair-schemas in $\mathcal{B}$.
- The instances of a transition-schema $Yx_1 \dots x_n \xrightarrow{a} E(x_1, \dots, x_n)$
  are transitions $Y\mathcal{O}_1 \dots \mathcal{O}_n \xrightarrow{a} E(\mathcal{O}_1, \dots, \mathcal{O}_n)$.

A pair-schema is sound (equivalent) if each of its instances is an equivalent pair (a pair of trace-equivalent objects=states).
A basis $\mathcal{B}$ is sound if each pair-schema in $\mathcal{B}$ is sound.

Stratification of trace equivalence, equivalence-level



$\mathcal{O}_1 \not\sim_5 \mathcal{O}$

The equivalence level: $\textsc{EqLv}(\mathcal{O}_1, \mathcal{O}) = 3$

Offending words: $\mathrm{OW}(\mathcal{O}_1, \mathcal{O}) = \{abb(a), bba(a, b)\}$.

Observe: ($\boxed{\mathcal{O}_1 \sim_2 \mathcal{O}}$ and) $\boxed{\mathcal{O}_1 \sim_2 \mathcal{O}_2}$ $\Rightarrow$ $\boxed{\mathcal{O}_2 \sim_2 \mathcal{O}}$

($\boxed{\textsc{EqLv}(\mathcal{O}_1, \mathcal{O}) = 3}$ and) $\boxed{\mathcal{O}_1 \sim_4 \mathcal{O}_2}$ $\Rightarrow$

$\boxed{\textsc{EqLv}(\mathcal{O}_2, \mathcal{O}) = 3}$ and $\boxed{\mathrm{OW}(\mathcal{O}_2, \mathcal{O}) = \mathrm{OW}(\mathcal{O}_1, \mathcal{O})}$

How $\textsc{EqLv}$ changes by performing action a ?

Stratification, equivalence level (formally)

For $k \in \mathbb{N}$, $\boxed{\mathcal{O} \sim_k \mathcal{O}'}$ $\Leftrightarrow_{df}$ $\forall w \in \mathcal{A}^*, |w| \leq k : \mathcal{O} \xrightarrow{w} \Leftrightarrow \mathcal{O}' \xrightarrow{w}$.

Observation. $\boxed{\sim_0 \supseteq \sim_1 \supseteq \sim_2 \supseteq \cdots \supseteq \sim}$. $\boxed{\mathcal{O} \sim \mathcal{O}' \Leftrightarrow \forall k : \mathcal{O} \sim_k \mathcal{O}'}$.

Equivalence level: $\boxed{\textsc{EqLv}(\mathcal{O}, \mathcal{O}') = \omega}$ $\Leftrightarrow_{df}$ $\mathcal{O} \sim \mathcal{O}'$.

$\boxed{\textsc{EqLv}(\mathcal{O}, \mathcal{O}') = k}$ $\Leftrightarrow_{df}$ $\mathcal{O} \sim_k \mathcal{O}' \wedge \mathcal{O} \not\sim_{k+1} \mathcal{O}'$.

$\boxed{\mathrm{OW}(\mathcal{O}, \mathcal{O}')}$ is the set of offending words, i.e., of the shortest words belonging to precisely one of the sets $\textsc{Trac}(\mathcal{O})$, $\textsc{Trac}(\mathcal{O}')$.

Observation. If $\mathcal{O} \not\sim \mathcal{O}'$ then $\textsc{EqLv}(\mathcal{O}, \mathcal{O}') = |w| - 1$ for any offending w.

Observation. In det-LTS,

by (performing) $u \in \mathcal{A}^*$, the eq-level can drop by at most $|u|$;
it drops precisely by $|u|$ for offending prefixes:

if $(\mathcal{O}, \mathcal{O}') \xrightarrow{u} (\mathcal{O}_1, \mathcal{O}'_1)$ then

$\textsc{EqLv}(\mathcal{O}, \mathcal{O}') - |u| \leq \textsc{EqLv}(\mathcal{O}_1, \mathcal{O}'_1)$;

$=$ iff $u \in \textsc{Pref}(\mathrm{OW}(\mathcal{O}, \mathcal{O}'))$ (for $u \neq \varepsilon$).

Observations for (sub)object replacement; congruence

Object replacement:

$$\boxed{\begin{array}{l}\mathrm{EqLv}(\mathcal{O}_1,\mathcal{O}_2)\geq k \text{ (i.e. } \mathcal{O}_1 \sim_k \mathcal{O}_2) \\ \mathrm{EqLv}(\mathcal{O}_1,\mathcal{O}) \geq k \text{ (i.e. } \mathcal{O}_1 \sim_k \mathcal{O})\end{array}} \quad \Downarrow \quad \boxed{\mathrm{EqLv}(\mathcal{O}_2,\mathcal{O})\geq k}$$

$$\boxed{\begin{array}{l}\mathrm{EqLv}(\mathcal{O}_1,\mathcal{O}_2)\geq k{+}1 \\ \mathrm{EqLv}(\mathcal{O}_1,\mathcal{O}) = k\end{array}} \quad \Downarrow \quad \boxed{\begin{array}{l}\mathrm{EqLv}(\mathcal{O}_2,\mathcal{O}) = k \text{ , and} \\ \mathrm{OW}(\mathcal{O}_2,\mathcal{O}) = \mathrm{OW}(\mathcal{O}_1,\mathcal{O})\end{array}}$$

Subobject replacement in the case of

$\boxed{\text{congruence}}$, i.e. $\mathcal{O}_1 \sim_k \mathcal{O}_2 \Rightarrow E(\mathcal{O}_1) \sim_k E(\mathcal{O}_2)$ (for all $k \in \mathbb{N}$):

$$\boxed{\begin{array}{l}\mathrm{EqLv}(\mathcal{O}_1,\mathcal{O}_2)\geq k \\ \mathrm{EqLv}(\boxed{E(\mathcal{O}_1)},\mathcal{O}) \geq k\end{array}} \quad \Downarrow \quad \boxed{\mathrm{EqLv}(\boxed{E(\mathcal{O}_2)},\mathcal{O})\geq k}$$

$$\boxed{\begin{array}{l}\mathrm{EqLv}(\mathcal{O}_1,\mathcal{O}_2)\geq k{+}1 \\ \mathrm{EqLv}(\boxed{E(\mathcal{O}_1)},\mathcal{O}) = k\end{array}} \quad \Downarrow \quad \boxed{\begin{array}{l}\mathrm{EqLv}(\boxed{E(\mathcal{O}_2)},\mathcal{O}) = k \text{ and} \\ \mathrm{OW}(\boxed{E(\mathcal{O}_2)},\mathcal{O}) = \mathrm{OW}(\boxed{E(\mathcal{O}_1)},\mathcal{O})\end{array}}$$

An idea for an "abstract algorithm"; an invariant

Given an LTS-generator $\mathcal{G}$ and a basis $\mathcal{B}$, for each (initial) pair $(\mathcal{O}_0, \mathcal{O}'_0)$ (of elements of $\mathcal{S}_\mathcal{G}$), we will define inductively a predicate

$\models_{(\mathcal{O}_0,\mathcal{O}'_0)} \subseteq \mathcal{A}^* \times ((\mathcal{S}_\mathcal{G} \times \mathcal{S}_\mathcal{G}) \cup \{\mathrm{NOP}, \mathrm{FAIL}\})$ ($\models$ if $(\mathcal{O}_0, \mathcal{O}'_0)$ clear).

- $\boxed{u \models (\mathcal{O}, \mathcal{O}')}$ ... (e.g., $\varepsilon \models (\mathcal{O}_0, \mathcal{O}'_0)$ is the axiom); read as "$u \in \mathcal{A}^*$ can be labelled (by the "algorithm") with the pair $(\mathcal{O}, \mathcal{O}')$".
- $\boxed{u \models \mathrm{NOP}}$ .... "u can be labelled with NOP"; it is intended to imply that u is Not an Offending Prefix if $\mathcal{B}$ is sound. (thus $\varepsilon \models \mathrm{NOP}$ is intended to imply $\mathcal{O}_0 \sim \mathcal{O}'_0$ if $\mathcal{B}$ is sound).
- $\boxed{\varepsilon \models \mathrm{FAIL}}$ ...... this is intended to imply $\mathcal{O}_0 \not\sim \mathcal{O}'_0$.

(Intended) Invariant

eq-level drops by at most $\|u\|$; it drops by $\|u\|$ for offending prefixes, i.e.:

if $\boxed{u \models (\mathcal{O}, \mathcal{O}')}$ then $\boxed{\mathrm{EqLv}(\mathcal{O}_0, \mathcal{O}'_0) - |u| \leq \mathrm{EqLv}(\mathcal{O}, \mathcal{O}')}$;
moreover, if $u \in \mathrm{Pref}(\mathrm{OW}(\mathcal{O}_0, \mathcal{O}'_0))$ then ... $=$ ...

and $\boxed{\mathrm{OW}(\mathcal{O}, \mathcal{O}') = u \backslash \mathrm{OW}(\mathcal{O}_0, \mathcal{O}'_0)}$ (the left quotient operation $u \backslash L$)

Derivation system defining $\models_{(\mathcal{O}_0,\mathcal{O}'_0)}$, given an LTS and $\mathcal{B}$

Axiom $\boxed{\varepsilon \models (\mathcal{O}_0, \mathcal{O}'_0)}$ (the system is parametrized by an initial pair)

1. (Basic transition) (this is the first derivation (or deduction) rule)

   $\boxed{u \models (\mathcal{O}, \mathcal{O}')}$ $\boxed{\mathcal{O} \sim_1 \mathcal{O}'}$ $\boxed{(\mathcal{O}, \mathcal{O}') \xrightarrow{a} (\mathcal{O}_1, \mathcal{O}'_1)}$ $\Rightarrow$ $\boxed{ua \models (\mathcal{O}_1, \mathcal{O}'_1)}$

2. ((Sub)object replacement)

   $\boxed{u \models (E(\mathcal{O}_1), \mathcal{O})}$ $\boxed{|v| < |u|}$ $\boxed{v \models (\mathcal{O}_1, \mathcal{O}_2)}$ $\Rightarrow$ $\boxed{u \models (E(\mathcal{O}_2), \mathcal{O})}$

3. (Symmetry) $\boxed{u \models (\mathcal{O}, \mathcal{O}')}$ $\Rightarrow$ $\boxed{u \models (\mathcal{O}', \mathcal{O})}$

4. (Basis) $\boxed{u \neq \varepsilon}$ $\boxed{u \models (\mathcal{O}, \mathcal{O}')}$ $\boxed{(\mathcal{O}, \mathcal{O}') \in \textsc{Inst}(\mathcal{B})}$ $\Rightarrow$ $\boxed{u \models \mathrm{NOP}}$

   ("u is not an offending prefix if $\mathcal{B}$ is sound")

5. (Bottom-up progression)

   $\boxed{u \models (\mathcal{O}, \mathcal{O}')}$ $\boxed{\mathcal{O} \sim_1 \mathcal{O}'}$ $\boxed{\forall a, \mathcal{O} \xrightarrow{a} : ua \models \mathrm{NOP}}$ $\Rightarrow$ $\boxed{u \models \mathrm{NOP}}$

6. (Rejection) $\boxed{u \models (\mathcal{O}, \mathcal{O}')}$, $\boxed{\mathcal{O} \not\sim_1 \mathcal{O}'}$ $\Rightarrow$ $\boxed{\varepsilon \models \mathrm{FAIL}}$ ("$\mathcal{O}_0 \not\sim \mathcal{O}'_0$")

An infinite state LTS with structured objects as states

Example. We have a finite generator $\mathcal{G}$ of an LTS $(\mathcal{S}_{\mathcal{G}}, \mathcal{A}, \{\xrightarrow{a}\}_{a\in\mathcal{A}})$, here a set of (root) rewrite rules (transition schemas):

$$\boxed{\begin{array}{l} Ax_1 \xrightarrow{a} AAx_1 \\ Ax_1 \xrightarrow{b} x_1 \\ Bx_1 \xrightarrow{a} BBx_1 \\ Bx_1 \xrightarrow{b} x_1 \end{array}}$$

$\mathcal{A} = \{a, b\}$

$\mathcal{S}_{\mathcal{G}} = \{A, B\}^* \cdot \{\perp\}$

Transitions $\alpha \xrightarrow{x} \beta$ are instances of transition schemas.

E.g., by substitution $(Ax_1 \xrightarrow{a} AAx_1)[\alpha/x_1]$ we get

$A\alpha \xrightarrow{a} AA\alpha$ (for every $\alpha \in \{A, B\}^* \cdot \{\perp\}$).

We can ask, e.g., $A\perp \overset{?}{\sim} B\perp$.
In the vertical notation,
we view strings as (thin) trees.
(Hint. "Guess" the basis
$\mathcal{B} = \{(x_1, x_1), (Ax_1, Bx_1)\}$.)

$$\begin{tikzcd} \boxed{A} \arrow[d] & \overset{?}{\sim} & \boxed{B} \arrow[d] \\ \boxed{\perp} & & \boxed{\perp} \end{tikzcd}$$

Note. We could extend $\mathcal{S}_{\mathcal{G}}$ with infinite strings, say from $\{A, B\}^\omega$;
regular infinite strings can be finitely presented ... $\alpha\,\beta^\omega$.

Soundness of $\models$ in the case of det-LTS and congruence

We now assume det-LTSs, and congruence in Subobject replacement.

Claim. $\boxed{\mathcal{O}_0 \sim \mathcal{O}'_0,\ u \models (\mathcal{O}, \mathcal{O}')} \Rightarrow \boxed{\mathcal{O} \sim \mathcal{O}',\ \mathrm{T{\small RAC}}(\mathcal{O}) = u \backslash \mathrm{T{\small RAC}}(\mathcal{O}_0)}$.

$\mathcal{O}_0 \not\sim \mathcal{O}'_0$ iff $\varepsilon \models \mathrm{FAIL}$. (By induction: axiom and rules 1.,2.,3. (and 6.).)

Soundness Lemma.

> If $\varepsilon \models_{(\mathcal{O},\mathcal{O}')} \mathrm{NOP}$ for all $(\mathcal{O}, \mathcal{O}') \in \{(\mathcal{O}_0, \mathcal{O}'_0)\} \cup \mathrm{I{\small NST}}(\mathcal{B})$
> then $\mathcal{B}$ is sound and $\mathcal{O}_0 \sim \mathcal{O}'_0$.

Proof. First show (inductively) the following (invariant):

if $\quad u \models (\mathcal{O}, \mathcal{O}') \quad$ then $\qquad \mathrm{E{\small Q}L{\small V}}(\mathcal{O}_0, \mathcal{O}'_0) - |u| \leq \mathrm{E{\small Q}L{\small V}}(\mathcal{O}, \mathcal{O}')$;

morever, if $u \in \mathrm{P{\small REF}}(\mathrm{OW}(\mathcal{O}_0, \mathcal{O}'_0))$ then $\quad \ldots = \ldots$

and $\mathrm{OW}(\mathcal{O}, \mathcal{O}') = u \backslash \mathrm{OW}(\mathcal{O}_0, \mathcal{O}'_0)$.

Then proceed by contradiction:

Suppose the assumption holds but there is some

$(\mathcal{O}, \mathcal{O}') \in \{(\mathcal{O}_0, \mathcal{O}'_0)\} \cup \mathrm{I{\small NST}}(\mathcal{B})$ with the $\boxed{\text{least finite eq-level}}$.

Take the $\boxed{\text{longest (offending) prefix}}$ u of some $w \in \mathrm{OW}(\mathcal{O}, \mathcal{O}')$ such that $u \models_{(\mathcal{O},\mathcal{O}')} \mathrm{NOP}$. What derivation rule has derived this?? None could!

Extensions (of subobject replacement), keeping soundness

- ((Sub)object replacement)
  $u \models (E(\mathcal{O}_1), \mathcal{O})$, $|v| < |u|$, $v \models (\mathcal{O}_1, \mathcal{O}_2) \implies u \models (E(\mathcal{O}_2), \mathcal{O})$

- ("Dummy" subobject replacement) (later ... unreachable subtrees)
  If $u \models (E(\mathcal{O}_1), \mathcal{O})$ and E has a specified property implying $E(\mathcal{O}_1) \sim E(\mathcal{O}_2)$ for any O_2 then $u \models (E(\mathcal{O}_2), \mathcal{O})$.
- (Limit subobject replacement) (crucial for us)
  If $u \models (E(\mathcal{O}_1), \mathcal{O})$, $|v| < |u|$, $v \models (\mathcal{O}_1, F(\mathcal{O}_1))$
  (hence $u \models (E(F(\mathcal{O}_1)), \mathcal{O})$,
  $u \models (E(F(F(\mathcal{O}_1))), \mathcal{O})$, $\ldots$)
  then $\boxed{u \models (E(F^\omega(\mathcal{O}_1)), \mathcal{O})}$ ... if well-defined and "congruent".

- (Basis application) (we do not use; just shows there are other options)
  If $u \neq \varepsilon$, $u \models (E(\mathcal{O}_1), \mathcal{O})$, and $(\mathcal{O}_1, \mathcal{O}_2)$ is a basis-instance, then $u \models (E(\mathcal{O}_2), \mathcal{O})$.
- Other (general) options keeping soundness ... ??

1st order grammars (a generalization of Greibach-NF CFG)

A 1st order grammar is a tuple $\mathcal{G} = (\mathcal{N}, \mathcal{A}, \mathcal{R})$, where
$\mathcal{N}$ is a finite set of ranked nonterminals (arity : $\mathcal{N} \to \mathbb{N}$)
(view nonterminals as "functions composing objects from subobjects"),
$\mathcal{A}$ is a finite set of actions (terminals), and
$\mathcal{R}$ is a finite set of (root) rewriting rules of the type

$$\boxed{Xx_1x_2\ldots x_m \xrightarrow{a} E(x_1, x_2, \ldots, x_m)}$$

where $X \in \mathcal{N}$, $m = \text{arity}(X)$, $a \in \mathcal{A}$, E a finite term over $\mathcal{N}$.

E.g., $Xx_1x_2 \xrightarrow{b} x_2$, $Yx_1x_2x_3 \xrightarrow{c} Xx_2x_2$, $Z \xrightarrow{c} Y\bot\bot\bot$, ...
$Yx_1x_2x_3 \xrightarrow{a} Y_1 Y_2 x_1 x_2 x_3 x_3 Y_3 x_1 x_2 x_3$ $[Y_1(Y_2(x_1, x_2, x_3), x_3, Y_3(x_1, x_2, x_3))]$.

A finite term over $\mathcal{N}$ is either

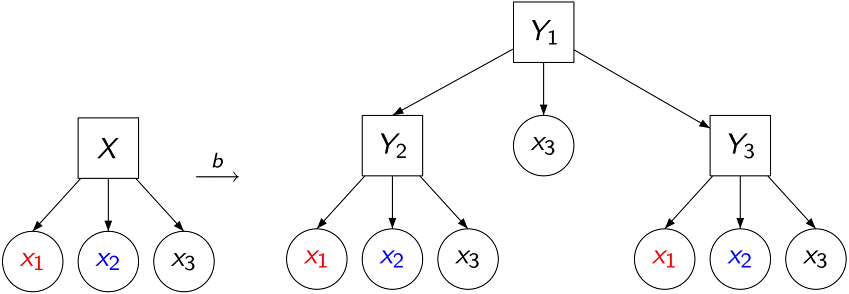
- $\bot$ ... the empty term, or
- x_i ... a variable from an assumed set $\mathcal{V} = \{x_1, x_2, \ldots\}$, or
- $YG_1G_2\ldots G_m$ where $Y \in \mathcal{N}$, $m = \text{arity}(Y)$, and G_i are finite terms.

An infinite term over $\mathcal{N}$ is of the form $YG_1G_2\ldots G_m$ where $Y \in \mathcal{N}$, $m = \text{arity}(Y)$, and G_i are finite or infinite terms, at least one being infinite.

1st-order grammar $\mathcal{G} = (\mathcal{N}, \mathcal{A}, \mathcal{R})$ as a generator of LTS$_\mathcal{G}$

A (root) rewriting rule $Xx_1x_2\ldots x_m \xrightarrow{a} E(x_1, x_2, \ldots, x_m)$ is a transition schema; its instances are transitions $XG_1G_2\ldots G_m \xrightarrow{a} E(G_1, G_2, \ldots, G_m)$.

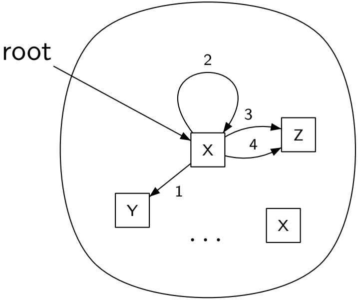
The terms can be naturally represented by (ordered) trees. E.g., the rule $Xx_1x_2x_3 \xrightarrow{b} Y_1(Y_2(x_1, x_2, x_3), x_3, Y_3(x_1, x_2, x_3))$ can be presented as

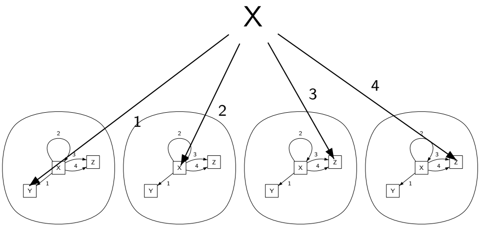


So the terms=trees are the states of the LTS$_\mathcal{G}$. We are mainly interested in comparing ground terms (denoted T, U, V, W), with no occurences of variables $x_i \in \mathcal{V}$, but we include all finite terms and regular infinite terms.

Finite presentations of regular infinite trees (our objects)

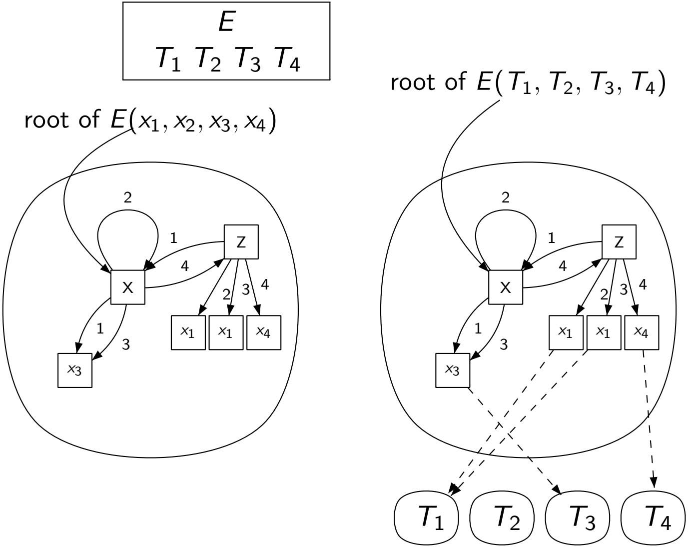
A regular tree has only finitely many pairwise nonisomorphic subtrees.
A finite directed (multi)graph naturally represents an (infinite) tree by its unfolding. Each node is labelled with $X \in \mathcal{N}$, or $\perp$, or $x_i \in \mathcal{V}$; its outgoing edges $e_1, e_2, \ldots, e_{\text{arity}(X)}$ are ordered. One node is specified as the root. (Nullary nonterminals, $\perp$, and variables x_i are leaves, with no outgoing edges.)





Note. Each regular tree can be finitely presented. The number of regular trees presented within a bounded presentation size is bounded.

A head-tails presentation $E(T_1, T_2, \dots, T_n)$ of a tree



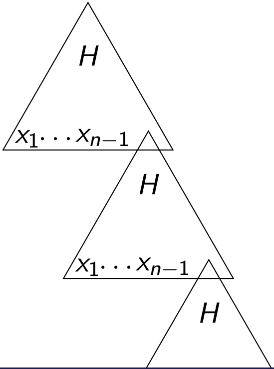
Here $E(x_1, x_2, x_3, x_4) = X(x_3, X(\dots), x_3, Z(X(\dots), x_1, x_1, x_4))$

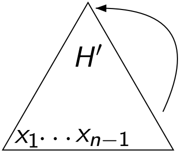
Congruence property ; limit subtree replacement

We observe congruence property: $T_1 \sim_k T_2 \Rightarrow E(T_1) \sim_k E(T_2)$.

Also for limit subtree replacement: $T_1 \sim_k F(T_1) \Rightarrow E(T_1) \sim_k E(F(T_1))$
$\Rightarrow E(T_1) \sim_k E(F(F(T_1))) \Rightarrow \cdots \Rightarrow E(T_1) \sim_k E(\text{LIM-REP}(T_1, F(T_1)))$
where $\text{LIM-REP}(T_1, F(T_1)) = F(F(F(\ldots)))$.

Example: $\boxed{\text{LIM-REP}(T_n, H(T_1, \ldots, T_{n-1}, T_n)) = H'(T_1, \ldots, T_{n-1})}$
(formally $\text{LIM-REP}(T_n, F(T_n))$ where $F(x_1) = H(T_1, \ldots, T_{n-1}, x_1))$





Starting with $H(x_1, \ldots, x_n)$, repeatedly replace x_n with $H(x_1, \ldots, x_n)$ ...
$H'(x_1, \ldots, x_{n-1})$ is the limit;
it has a simple finite presentation.

Deterministic 1st order grammars

A 1st-order grammar $\mathcal{G} = (\mathcal{N}, \mathcal{A}, \mathcal{R})$ is deterministic if
for each pair $X \in \mathcal{N}$, $a \in \mathcal{A}$ there is
at most one rule of the type $X x_1 x_2 \ldots x_m \xrightarrow{a} E(x_1, x_2, \ldots, x_m)$.

Then $\mathrm{LTS}_{\mathcal{G}}$ is deterministic.

The next slide shows a (variant of our general)
derivation system for deterministic 1st-order grammars
(i.e., an inductive definition of the predicates $\models_{(T_0, T_0')}$).
We thus assume a given det-1st-order grammar $\mathcal{G}$ and a finite basis $\mathcal{B}$.

A basis $\mathcal{B}$ is now a finite set of pairs

$$(E(x_1, \ldots, x_n), F(x_1, \ldots, x_n))$$

where E, F are regular terms=trees
(in which variables from $\{x_1, \ldots, x_n\}$ can occur).

Derivation system (given $\mathcal{G},\mathcal{B}$) with axiom $\varepsilon \models (T_0, T_0')$

- (Basic transition) (the first derivation (or deduction) rule)
  $u \models (T, T'),\ T \sim_1 T',\ (T, T') \xrightarrow{a} (T_1, T_1') \Rightarrow ua \models (T_1, T_1')$
- (Subtree replacement)
  $u \models (E(T_1), T),\ |v| < |u|,\ v \models (T_1, T_2) \Rightarrow u \models (E(T_2), T)$
- (Limit subtree replacement) $\boxed{u \models (E(T_1), T)}$ $\boxed{|v| < |u|}$
  $\boxed{v \models (T_1, F(T_1))} \Rightarrow \boxed{u \models (E(\text{LIM-REP}(T_1, F(T_1))), T)}$
- (Unreachable subtree replacement) If $\boxed{u \models (E(T_1), T)}$ and there is no w such that $E(x_1) \xrightarrow{w} x_1$ then $\boxed{u \models (E(T_2), T)}$ for any T_2.
- (Symmetry) If $u \models (T, T')$ then $u \models (T', T)$.
- (Basis) If $\boxed{u \neq \varepsilon}$, $\boxed{u \models (E(T_1, \dots, T_n), F(T_1, \dots, T_n))}$, and $\boxed{(E(x_1, \dots, x_n), F(x_1, \dots, x_n))}$ is in $\mathcal{B}$ then $\boxed{u \models \mathrm{NOP}}$.
- (Bottom-up progression) If $u \models (T, T'),\ T \sim_1 T'$, and $ua \models \mathrm{NOP}$ for all a, $T \xrightarrow{a}$, then $u \models \mathrm{NOP}$.
- (Rejection) If $u \models (T, T')$ and $T \not\sim_1 T'$ then $\varepsilon \models \mathrm{FAIL}$.

Finite proofs of $T_0 \sim T'_0$ (for det-1st-order grammars)

Soundness has been already proven:

> If $\varepsilon \models_{(T,T')} \mathrm{NOP}$ for all $(T, T') \in \{(T_0, T'_0)\} \cup \textsc{Inst}(\mathcal{B})$
> then $\mathcal{B}$ is sound and $T_0 \sim T'_0$.

But note: Though $\mathcal{B}$ is finite, $\textsc{Inst}(\mathcal{B})$ is (generally) infinite!

For each $\boxed{(E(x_1, \dots, x_n), F(x_1, \dots, x_n))}$ in $\mathcal{B}$ we consider the worst instance: each x_i is treated as an object for which $x_i \sim x_i$ but $x_i \not\sim_1 H$ if $H \neq x_i$. (In particular, $x_1 \not\sim_1 x_2$.)

If we insist on ground terms in the basis, we can assume a fixed countable set of special leaves $\mathcal{L} = \{L_1, L_2, L_3, \dots\}$ $(\mathcal{L} \cap \mathcal{N} = \emptyset)$ and the rules $L_1 \xrightarrow{\ell_1} \perp$, $L_2 \xrightarrow{\ell_2} \perp$, $L_3 \xrightarrow{\ell_3} \perp$, $\dots$ where $\ell_i \neq \ell_j$ for $i \neq j$, and $\ell_i \notin \mathcal{A}$.

$\boxed{(E(L_1, \dots, L_n), F(L_1, \dots, L_n))}$ is defined as the worst instance of (E, F).

Soundness for det-1st-order grammars:

> If $\varepsilon \models_{(T,T')} \mathrm{NOP}$ for all $(T, T') \in \{(T_0, T'_0)\} \cup \textsc{WorstInst}(\mathcal{B})$
> then $\mathcal{B}$ is sound and $T_0 \sim T'_0$.

Soundness of the worst instances; exposing equations

We say that $u \in \mathcal{A}^*$ exposes an equation for the pair (of heads)

$\boxed{E(x_1,\dots,x_n) \mid F(x_1,\dots,x_n)}$

if $E(x_1,\dots,x_n) \xrightarrow{u} x_i$, $F(x_1,\dots,x_n) \xrightarrow{u} H(x_1,\dots,x_n)$, or vice versa,
where $H(x_1,\dots,x_n) \neq x_i$ (but might be $H = x_j$ for $i \neq j$).

Formally we write $\boxed{x_i \doteq H(x_1,\dots,x_n)}$.

Observation. For such u,
$\mathrm{EQLV}(E(T_1,\dots,T_n), F(T_1,\dots,T_n)) - |u| \leq \mathrm{EQLV}(T_i, H(T_1,\dots,T_n))$.
Hence $E(T_1,\dots,T_n) \sim F(T_1,\dots,T_n)$ implies $T_i \sim H(T_1,\dots,T_n)$.

Claim. (Soundness of verifying just the "special-leaves basis instances".)

$\boxed{\mathrm{EQLV}(E(L_1,..,L_n), F(L_1,..,L_n))} \leq \boxed{\mathrm{EQLV}(E(T_1,..,T_n), F(T_1,..,T_n))}$

(if $L_1,\dots,L_n$ do not occur in E, F).

Proof easily follows from the following:

Note. If $\boxed{\begin{array}{c} E \\ T'_1 \dots T'_n \end{array}} \sim_k \boxed{\begin{array}{c} F \\ T'_1 \dots T'_n \end{array}}$ and $\boxed{\begin{array}{c} E \\ T_1 \dots T_n \end{array}} \not\sim_k \boxed{\begin{array}{c} F \\ T_1 \dots T_n \end{array}}$

then an offending prefix (on the rhs) exposes an equation for E, F.

Dpda problem reduces to det-1st-order grammar problem

Proposition.

> Dpda language equivalence can be effectively reduced to det-1st-order grammar trace equivalence.

$(M, p\alpha, q\beta)$ transformed to $(\mathcal{G}, T_0, T_0')$ where $L(p\alpha) = L(q\beta)$ iff $T_0 \sim T_0'$.

This is accomplished by

- reducing dpda language equivalence to dpda trace equivalence,
- transforming dpda to ε-popping form,
- representing dpda configurations as trees
  (naturally, adding a control state to each stack symbol),
- "precomputing the ε-moves" (trimming the configuration-tree).

The next slides sketch the ideas in more detail.
(*Note.* This part can be skipped, when one prefers to look at the completeness of $\models$ for 1st-order grammars.)

ε-popping dpda and their *trace equivalence* problem

stable	unstable
$q_1 A$	
	$q_2 A$
$q_3 A$	
$q_1 B$	
	$q_2 B$
$q_3 B$	

$q_1 A \xrightarrow{a} q_1 BB$

$q_1 A \xrightarrow{b} ..$

$q_2 A \xrightarrow{\varepsilon} q_3$

$q_3 A \xrightarrow{a} ..$

$q_3 A \xrightarrow{b} q_1$

$q_1 B \xrightarrow{a} q_2 AA$

$q_1 B \xrightarrow{b} q_3 BA$

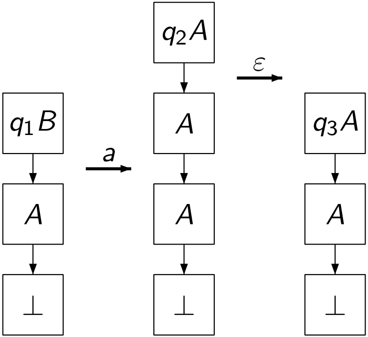
$q_2 B \xrightarrow{\varepsilon} q_1$

$q_3 B \ldots$

$M = (Q, \mathcal{A}, \Gamma, \mathcal{R})$ .....

$Q = \{q_1, q_2, q_3\}$, $\mathcal{A} = \{a, b\}$, $\Gamma = \{A, B\}$

(Root rewrite) rules $\mathcal{R}$ ...



Trace equivalence problem (for ε-pop-dpda):

$\boxed{\text{Given } M,\ p\alpha,\ q\beta}$, is $\boxed{p\alpha \sim q\beta}$?

$(\forall w \in \mathcal{A}^*\colon\ p\alpha \xrightarrow{w} \text{ iff } q\beta \xrightarrow{w})$

A (routine) reduction from dpda-language to dpda-trace

Instance	Question	Remark
2 dpda M_1, M_2	$L_{FS}(M_1) \stackrel{?}{=} L_{FS}(M_2)$	classical setting
2 dpda M_1, M_2	$L_{\perp}(M_1) \stackrel{?}{=} L_{\perp}(M_2)$	(1)
dpda M, conf. $p\alpha, q\beta$	$L(p\alpha) \stackrel{?}{=} L(q\beta)$	(2)
ε-popping-dpda M, $p\alpha, q\beta$	$L(p\alpha) \stackrel{?}{=} L(q\beta)$	(3)
ε-popping-dpda M, $p\alpha, q\beta$	$p\alpha \stackrel{?}{\sim} q\beta$	(4)

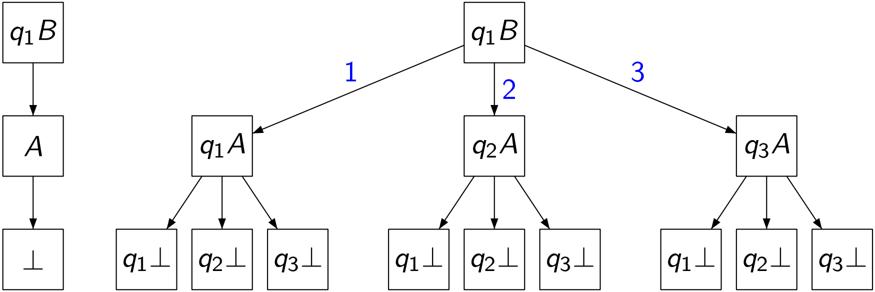
Remarks:

1. If $L_1, L_2 \subseteq \Sigma^*$ and $\$ \not\in \Sigma$ then $L_1 = L_2 \Leftrightarrow L_1 \cdot \{\$\} = L_2 \cdot \{\$\}$.
   So further $L(p\alpha) = \{ w \in \mathcal{A}^* \mid p\alpha\perp \xrightarrow{w} q\perp \text{ for some } q \}$.
2. The disjoint union of two dpda's is a dpda.
3. ... $pA \xrightarrow{\varepsilon} qB\alpha$ where $qB \xrightarrow{a} q'\beta$ replace with $pA \xrightarrow{a} q'\beta\alpha$ ...
   ... if pA enables infinite ε-sequence, remove it ...
4. Add an 'error' state q_{err}, rules $q_{err}A \xrightarrow{a} q_{err}A$ ($\forall A \in \Gamma, a \in \mathcal{A}$),
   complete every 'missing' rule $pA \xrightarrow{a}$ .. by $pA \xrightarrow{a} q_{err}A$.
   (We get: $L(p\alpha) = L(q\beta)$ iff $\forall w \in \mathcal{A}^* : p\alpha \xrightarrow{w} \Leftrightarrow q\beta \xrightarrow{w}$.)

(D)pda configuration as trees

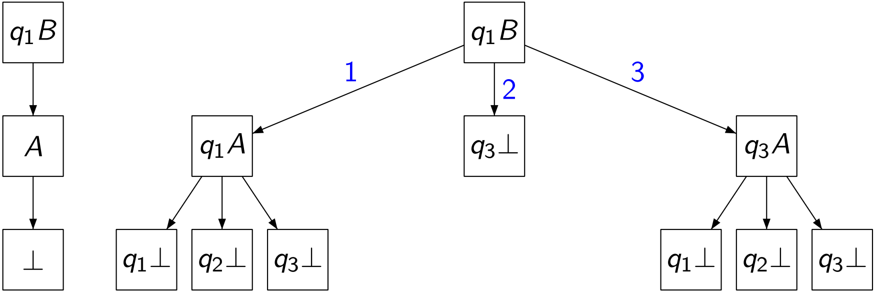
Assume

$Q = \{q_1, q_2, q_3\}$, a rule $\boxed{q_2 A \xrightarrow{\varepsilon} q_3}$, and $q_1 B$, $q_1 A$, $q_3 A$ are stable.



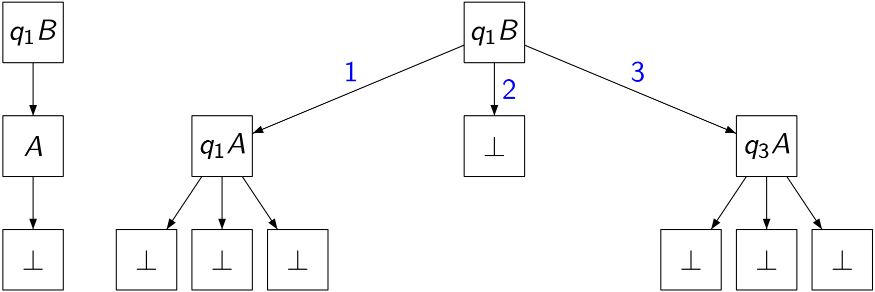
"Precomputing ε-(popping) moves" – trimming the tree

$Q = \{q_1, q_2, q_3\}$, a rule $\boxed{q_2 A \xrightarrow{\varepsilon} q_3}$, and $q_1 B$, $q_1 A$, $q_3 A$ are stable.



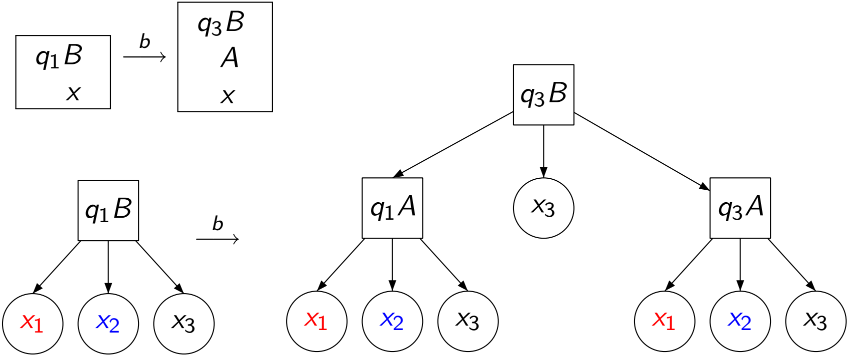
Unifying the leaves

$Q = \{q_1, q_2, q_3\}$, a rule $\boxed{q_2 A \xrightarrow{\varepsilon} q_3}$, and $q_1 B$, $q_1 A$, $q_3 A$ are stable.



Adjusting (root rewriting) rules

A rule $q_1 B \xrightarrow{b} q_3 BA$, i.e., (recall the rule $\boxed{q_2 A \xrightarrow{\varepsilon} q_3}$)



$\boxed{q_1 B} x_1 x_2 x_3 \xrightarrow{b} \boxed{q_3 B} \left(\boxed{q_1 A} x_1 x_2 x_3 \,,\, x_3 \,,\, \boxed{q_3 A} x_1 x_2 x_3 \right) = E(x_1, x_2, x_3)$

Another example: $q_3 A \xrightarrow{b} q_1$ ..... $\boxed{q_3 A} x_1 x_2 x_3 \xrightarrow{b} x_1 = F(x_1, x_2, x_3)$

Completeness of predicates $\models_{(T_0,T'_0)}$, and decidability

Soundness Lemma. (det-LTSs, congruence)

If $\varepsilon \models_{(\mathcal{O},\mathcal{O}')} \mathrm{NOP}$ for all $(\mathcal{O},\mathcal{O}') \in \{(\mathcal{O}_0,\mathcal{O}'_0)\} \cup \textsc{Inst}(\mathcal{B})$ then $\mathcal{B}$ is sound and $\mathcal{O}_0 \sim \mathcal{O}'_0$.

Completeness Lemma.

For each det-1st-order grammar $\mathcal{G}$ there is a finite sound basis $\mathcal{B}$ of pairs of regular trees such that for each pair $T_0 \sim T'_0$ of (equivalent) regular trees we have $\varepsilon \models_{(T_0,T'_0)} \mathrm{NOP}$.

Using Soundness Lemma (with the worst basis-instances), Completeness Lemma, and simple observations on (algorithmic) manipulations with finite presentations of regular trees, it is straightforward to derive:

Theorem. Trace equivalence for deterministic 1st order grammars (and thus also language equivalence for dpda) is decidable.
Moreover, for each grammar $\mathcal{G}$ there is a (computable) finite basis $\mathcal{B}$ for which "the predicate $\models$ is sound and complete".

So now we are going to prove Completeness Lemma ...

Refresh: Derivation system $(\mathcal{G}, \mathcal{B})$ with axiom $\varepsilon \models (T_0, T_0')$

- (Basic transition)
  $u \models (T, T'),\ T \sim_1 T',\ (T, T') \xrightarrow{a} (T_1, T_1') \Rightarrow ua \models (T_1, T_1')$
- (Subtree replacement)
  $u \models (E(T_1), T),\ |v| < |u|,\ v \models (T_1, T_2) \Rightarrow u \models (E(T_2), T)$
- (Limit subtree replacement) $u \models (E(T_1), T),\ |v| < |u|,$
  $v \models (T_1, F(T_1)) \Rightarrow u \models (E(\text{LIM-REP}(T_1, F(T_1))), T)$
- (Unreachable subtree replacement) If $u \models (E(T_1), T)$ and there is no w such that $E(x_1) \xrightarrow{w} x_1$ then $u \models (E(T_2), T)$ for any T_2.
- (Symmetry) If $u \models (T, T')$ then $u \models (T', T)$.

- (Basis) If $u \neq \varepsilon$, $u \models (E(T_1, \dots, T_n), F(T_1, \dots, T_n))$, and $(E(x_1, \dots, x_n), F(x_1, \dots, x_n))$ is in $\mathcal{B}$ then $u \models \text{NOP}$.
- (Bottom-up progression) If $u \models (T, T')$, $T \sim_1 T'$, and $ua \models \text{NOP}$ for all a, $T \xrightarrow{a}$, then $u \models \text{NOP}$.
- (Rejection) If $u \models (T, T')$ and $T \not\sim_1 T'$ then $\varepsilon \models \text{FAIL}$.

Completeness Lemma shown by contradiction

Given a det-1st-order grammar $\mathcal{G} = (\mathcal{N}, \mathcal{A}, \mathcal{R})$, (imagine we have) put in basis $\mathcal{B}$ all sound pair-schemas $(E(x_1, \ldots, x_n), F(x_1, \ldots, x_n))$ where the presentation size of (regular trees) E, F is bounded by a large constant SIZE (somehow determined by $\mathcal{G}$).

Now assume a pair W_0, W_0' of (possibly very large) regular trees such that

$$\boxed{W_0 \sim W_0' \;\text{ and } \varepsilon \not\models_{(W_0, W_0')} \mathrm{NOP}}$$

We write $\models$ instead of $\models_{(W_0,W_0')}$; recall that $u \models (T, T')$ implies $T \sim T'$.

There is necessarily some $a_1 \in \mathcal{A}$ such that $(W_0, W_0') \xrightarrow{a_1} (W_1, W_1')$ and $a_1 \not\models \mathrm{NOP}$, some $a_2 \in \mathcal{A}$ such that $(W_1, W_1') \xrightarrow{a_2} (W_2, W_2')$ and $a_1 a_2 \not\models \mathrm{NOP}$; etc. ... This gives an infinite word $\mu = a_1 a_2 a_3 \ldots$.

$$(W_0, W_0') \xrightarrow{a_1} (W_1, W_1') \xrightarrow{a_2} (W_2, W_2') \xrightarrow{a_3} \cdots$$

Since we cannot derive $u \models \mathrm{NOP}$ for any prefix u of μ, there is no repeat, i.e. no two prefixes $u_1 \neq u_2$ with $u_1 \models (T, T')$, $u_2 \models (T, T')$ (otherwise we would get $u_2 \models \mathrm{NOP}$).

("Hint.") Equations for possible limit-subtree-replacement

If u is a prefix of μ, $u \neq \varepsilon$, and

$u \models \boxed{\begin{array}{c} E \\ T_1 \dots T_n \end{array}} \boxed{\begin{array}{c} F \\ T_1 \dots T_n \end{array}}$ (implying $E(T_1, \dots, T_n) \sim F(T_1, \dots, T_n)$)

where E, F (i.e., $E(x_1, \dots, x_n), F(x_1, \dots, x_n)$) have presentation size bounded by SIZE

then (E, F) is not in $\mathcal{B}$ (otherwise $u \models \mathrm{NOP}$) and thus $E(L_1, \dots, L_n) \not\sim F(L_1, \dots, L_n)$. Hence there is a shortest $v \in \mathcal{A}^*$ exposing an equation for E, F (recall Note on slide 24);

w.l.o.g. assume $uv \models (T_n, \boxed{\begin{array}{c} H \\ T_1 \dots T_n \end{array}})$.

Thus for every prefix uw of μ where $|w| > |v|$:

if $\quad uw \models (E'(T_n), F')$

then this T_n can be replaced with $H(T_1, \dots, T_n)$, and thus with the appropriate $H'(T_1, \dots, T_{n-1})$ by limit-subtree-replacement;

hence $\quad uw \models (E'(H'(T_1, \dots, T_{n-1})), F')$.

Words exposing subtrees ("going down")

For $T \xrightarrow{v}$, we say that
T goes down by (in) v, in other words v exposes a root-successor of T,
if $T = XU_1 \dots U_m$ and $Xx_1 \dots x_m \xrightarrow{v'} x_i$
for some prefix v' of v and some $i \in \{1, 2, \dots, m\}$.

(Given grammar $\mathcal{G} = (\mathcal{N}, \mathcal{A}, \mathcal{R})$,) we can attach
to each $X \in \mathcal{N}$ and each $j, 1 \leq j \leq m = \text{arity}(X)$

---

a shortest word $w(X, j)$ such that

$$\boxed{\begin{matrix} X \\ x_1 \dots x_m \end{matrix}} \xrightarrow{w(X,j)} x_j$$

if there is one. Otherwise we let $w(X, j) = \varepsilon$.

Let $M \in \mathbb{N}$ be (the least) such that

$$M > \max\{\, \mathit{length}(w(X, j)) \mid X \in \mathcal{N},\ 1 \leq j \leq \text{arity}(X)\,\}$$

Grammar-determined (number or function) bounds

The (least) number M obviously exists and is determined by $\mathcal{G}$, independently of the fixed (W_0, W_0') and μ.

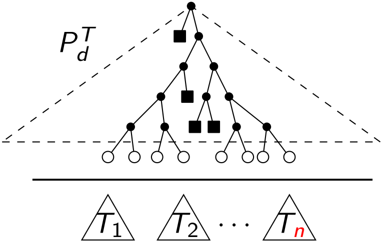
Grammar $\mathcal{G}$ obviously also determines the (component-wise least) function $\textsc{B-Inc} : \mathbb{N} \to \mathbb{N}$ ("Bounding Increase") satisfying the following:
if $Xx_1 \ldots x_m \xrightarrow{v} F(x_1, \ldots, x_m)$ (where $X \in \mathcal{N}$, $v \in \mathcal{A}^*$)
then the depth of the finite tree F (the maximal length of branches) is bounded by $\textsc{B-Inc}(|v|)$.

Note. It is now irrelevant that we can compute (bounds on) M and $\textsc{B-Inc}$. When we say a $\mathcal{G}$-determined number (function), we mean that such a number (function) exists and is only dependent on $\mathcal{G}$ (not on W_0, W_0', μ).

We later use some other numbers M_1, M_2, M_3, M_4 which are obviously $\mathcal{G}$-determined.
When we say that a number (discussed in some context) is $\mathcal{G}$-bounded, we mean that there is a $\mathcal{G}$-determined upper bound for the number.

(For $d \in \mathbb{N}$), each T has *d-prefix form* with *d-prefix* P_d^T

The $(d+1)$-th node of each branch (if the branch is not shorter) is replaced by a (fresh) variable-leaf ... the head P_d^T is a d-prefix.



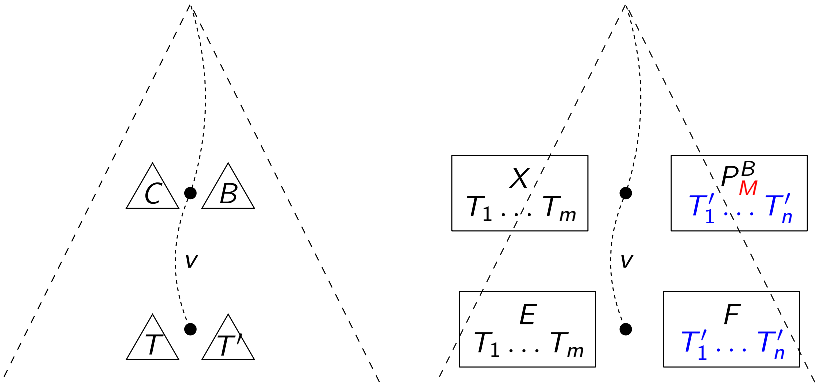
d levels

$n \leq b^d$ variable-leaves
(b ... branching degree)

Observation. If $\boxed{\begin{array}{c} P_d^T \\ T_1 \cdots T_n \end{array}} \xrightarrow{u} T'$ where $|u| \leq d$

then $P_d^T(x_1, \ldots, x_n) \xrightarrow{u} E(x_1, \ldots, x_n)$ and $T' = \boxed{\begin{array}{c} E \\ T_1 \cdots T_n \end{array}}$.

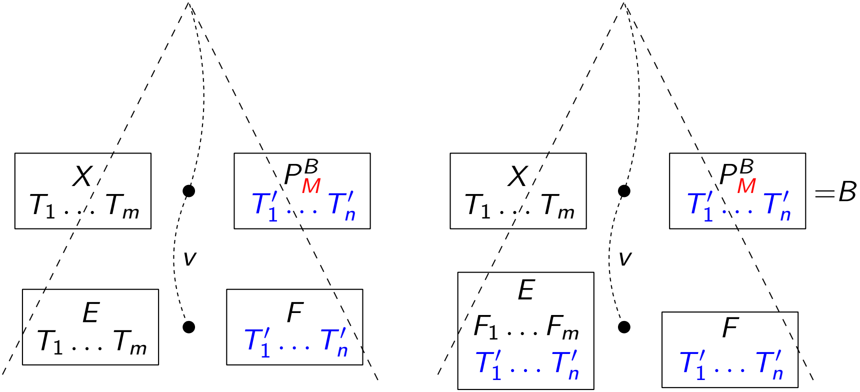
Balancing (possibility) with the (rhs) balancing pivot B



Suppose the trees on the left are in the form on the right, for a prefix uv of μ we have $u \models (C, B)$, and $(C, B) \xrightarrow{v} (T, T')$, so $uv \models (T, T')$, $\boxed{C \text{ does not go down by } v}$, $|v| = M$.

Replace each $\boxed{\boxed{T_j}}$ by $\left(B = \boxed{\begin{matrix} P^B_M \\ T'_1 \dots T'_n \end{matrix}} \xrightarrow{w(X,j)} \right) \boxed{\boxed{\begin{matrix} F_j \\ T'_1 \dots T'_n \end{matrix}}}$

The result of balancing (determined by B, X, v)



The depth of finite trees $E, F_1, \dots, F_m, F$ is bounded by some M_1.

Note that the balancing result $uv \models (E'(T'_1, \dots, T'_n), F(T'_1, \dots, T'_n))$, (where $E' = E(F_1, \dots, F_m)$) is determined by B, X, v.

Corollary. We cannot get the same pivot for infinitely many prefixes of μ (for no more than some M_2 prefixes); otherwise we would get a repeat.

Balancing strategy along our fixed infinite μ

Starting from $\varepsilon \models (W_0, W_0')$, we must have the first possibility to balance, with some pivot B_1, otherwise W_i, W_i' would go down in each segment of μ of length M, and they thus would range over finitely many trees, due to regularity; then we would get a repeat.

Having pivot B_i (rhs, w.l.o.g.), in $u \models (XT_1 \ldots T_m, B_i)$, we (re)start from the balancing result $uv \models (E'(T_1', \ldots, T_n'), F(T_1', \ldots, T_n'))$, where $E' = E(F_1, \ldots, F_m)$, (thus redefining W_j for $j \geq |uv|$).

In the next segment of μ of length M either the lhs-tree does not go down, we then do balancing with a further rhs-pivot B_{i+1}, or the lhs-tree does go down, then we "look" a bit further ...

There is obviously a $\mathcal{G}$-determined M_3 such that in the segment of length M_3 of μ, which starts from the balancing result with B_i, we either get B_{i+1} on the same side as B_i, or (some $F_j(T_1' \ldots T_n')$ on the lhs is exposed, and thus) the trees on both sides are reachable from B_i, by a word of length $\leq M_3$ – after this moment we take B_{i+1} as the first possible pivot on either side (there must be one by regularity).

Infinite pivot path; a "stair-base" subtree V_1 of pivot B_1

We thus get an infinite "pivot-path" $B_1 \xrightarrow{u_1} B_2 \xrightarrow{u_2} B_3 \xrightarrow{u_3} \cdots$ where $u_1 u_2 u_3 \ldots$ arises from a suffix of μ by replacing segments of length $\leq M_3$.

Observation. $\boxed{\text{In each "short" segment: either a pivot or going down}}$:

There is $\mathcal{G}$-determined M_4 such that in any pivot-path segment $U_0 \xrightarrow{b_1} U_1 \xrightarrow{b_2} \cdots \xrightarrow{b_{M_4}} U_{M_4}$ there is a pivot ($U_i = B_j$ for some i, j) or U_0 goes down in this segment.

Define V_1 as the subtree of B_1, so $B_1 = F(V_1)$, which is exposed by $u_1 u_2 u_3 \ldots$ ($F(x_1) \xrightarrow{w} x_1$ for a prefix w of $u_1 u_2 u_3 \ldots$) but none of the proper subtrees of V_1 is exposed by $u_1 u_2 u_3 \ldots$.
If there was no such V_1, the pivot path would be exposing infinitely often isomorphic subtrees of B_1 and by the above Observation we would necessarily get infinitely often the same pivot (a contradiction).

So we have $B_1 \longrightarrow \cdots\cdots\cdots\cdots\cdots\cdots \longrightarrow V_1 \rightarrow B_k \longrightarrow \cdots$ where the root of $V_1 = Y(\cdots)$ (i.e., the tree $Yx_1 \ldots x_m$) enables the whole infinite suffix of the pivot-path after exposing V_1.

$\mathcal{G}$-determined least n with a $\mathcal{G}$-determined function g

Recall $B_1 \longrightarrow \cdots\cdots\cdots \longrightarrow V_1 \rightarrow B_k \longrightarrow B_{k+1} \longrightarrow B_{k+2} \longrightarrow \cdots$
where $V_1 = Y(\cdots)$ does not go down (in the whole infinite suffix).

Let $\boxed{\begin{array}{c} P_M^{V_1} \\ T_1 \ldots T_n \end{array}}$ be the M-prefix form of V_1; note: n is $\mathcal{G}$-bounded.

The balancing results with B_{k+j} are $\boxed{\begin{array}{c} E_{k+j} \\ T_1 \ldots T_n \end{array}}$ $\boxed{\begin{array}{c} F_{k+j} \\ T_1 \ldots T_n \end{array}}$ where
E_{k+j}, F_{k+j} are finite heads; (the length of their branches, and thus also) their presentation size is bounded by a grammar-determined function $f(j)$ (recall Observation and function $\textrm{B-I}\textsc{nc}$).
So without assuming anything specific on W_0, W'_0, μ, we know that there exists (some, and thus also) the least (grammar-determined) n
for which there is a grammar-determined function $g : \mathbb{N} \rightarrow \mathbb{N}$ guaranteeing:
there are some (unspecified) trees $T_1, \ldots, T_n$ and infinitely many (nonempty, increasing) prefixes $w_0, w_1, w_2, \ldots$ of μ such that
$w_j \models (E_j(T_1, \ldots, T_n), F_j(T_1, \ldots, T_n))$ where E_j, F_j are (generally) regular trees with the presentation size bounded by $g(j)$.

Using the "Hint" (an equation) to get a contradiction

Let us have the least n with a function g as discussed. We can assume that (we have chosen) $\mathrm{SIZE} > g(0)$ and thus, necessarily, $n > 0$.
$(E_0(x_1, \ldots, x_n), F_0(x_1, \ldots, x_n))$ cannot be a sound pair-schema, and there is thus a shortest word v exposing an equation for (E_0, F_0), say $x_n \doteq H(x_1, \ldots, x_n)$. As previously, we define
$H'(x_1, \ldots, x_{n-1}) = H(x_1, \ldots, x_{n-1}, H(x_1, \ldots, x_{n-1}, H(\ldots)))$.
Since the number of possible E_0, F_0 (whose size is bounded by $g(0)$) is $\mathcal{G}$-bounded, the length of v and thus also the presentation size of (H and) H' is $\mathcal{G}$-bounded. By noting

$w_0 \models (E_0(T_1, \ldots, T_n), F_0(T_1, \ldots, T_n))$

$w_0 v \models (T_n, H(T_1, \ldots, T_n))$

$w_k \models (E'_k(T_1, \ldots, T_{n-1}), F'_k(T_1, \ldots, T_{n-1}))$
$w_{k+1} \models (E'_{k+1}(T_1, \ldots, T_{n-1}), F'_{k+1}(T_1, \ldots, T_{n-1}))$
$\ldots$ (for some $\mathcal{G}$-bounded $k \in \mathbb{N}$)

where $E'_{k+j}(x_1, \ldots, x_{n-1}) = E_{k+j}(x_1, \ldots, x_{n-1}, H'(x_1, \ldots, x_{n-1}))$,
and $F'_{k+j}(x_1, \ldots, x_{n-1}) = F_{k+j}(x_1, \ldots, x_{n-1}, H'(x_1, \ldots, x_{n-1}))$
we derive a contradiction with the minimality of n.

Informal summary and some further remarks (1)

Outline on slide 4 has been realized, using several notions, ideas and claims which can be easily explained and observed.

We have used the brute-force breadth-first search for a shortest trace witnessing nonequivalence of the given pair of objects, i.e. an offending word, the notion of a finite basis of schemas (templates, shapes) of equivalent pairs, and the subobject replacement option helping to recognize non-offending (prefixes of) traces by creating basis instances.

Soundness of the process is guaranteed if we have the congruence property of the stratified trace equivalence and deterministic LTSs, guaranteeing that the eq-level of two objects can drop at most by 1 by performing one (common) action.

Completeness for det-1st-order grammars is based on a few ideas. Firstly we note that a nonequivalent schema with at least one equivalent instance must yield an "equation" (a "linear dependency" among component-objects); such equations can be used for subtree (limit) replacement for sufficiently long prefixes (traces). Secondly, an idea reminding the "stair-sequences" of pda computations, is used:

Informal summary and some further remarks (2)

A situation when the tree on some side does not go down in a bounded segment of a trace allows a balancing step, arriving at a pair with bounded (differing) "heads" and the same "tails". Using this balancing with a slight care allows us to extract a "stair-base" along a potentially infinite trace. Using the mentioned ideas, the existence of a sufficient basis for each det-1st-order grammar is derived in a straightforward manner.

Remark (PJ): Géraud Sénizergues remarked that all the ideas are present in his original paper, and my presentation is isomorphic with his. This is well possible; I do not claim having come with any new fundamental idea. I can just say that it would be very difficult for me to present and prove such an isomorphism.

For bounding the computational complexity, in fact the length of offending words (by $2 \uparrow\uparrow g(n)$) for non-equivalent trees in 1st-order grammars, we would need to look more closely at the "stair sequences" of balancing results along a potential offending word.
A rough idea can be got from the following observation.

Informal summary and some further remarks (3)

Claim. If $u_1 \models (E(T), F(T))$, $w_1 \models (E(G(T)), F(G(T)))$,
$u_2 \models (E(V), F(V))$, $w_2 \models (E(G(V)), F(G(V)))$,
where $|u_1| < |w_1|$, $|u_2| < |w_2|$, and w_1 is a proper prefix of w_2
then w_2 is not a prefix of an offending word.

Proof. Suppose w_2 is an offending prefix. Then $(E(x_1), F(x_1))$ is not an equivalent schema and there is a shortest v exposing an equation $x_1 \doteq H(x_1)$.

Exercise. Show a contradiction by noting that the eq-levels of the pairs $(E(G(T)), F(G(T)))$, $(E(G(V)), F(G(V)))$ must be both equal to $(E(G(H')), F(G(H'))$, where H' arises from $H(x_1)$ by limit-replacement of x_1 with $H(x_1)$.

Important is to note that we deduced that w_2 is not an offending prefix without knowing the length of v exposing the equation. The idea can be generalized (we had 1 tail and needed 4 pairs, for 2 tails we need 8 pairs, for 3 tails 16 pairs, ...), and "stair-sequences" of balancing results, with bounded stairs, yield the overall bound ...

Informal summary and some further remarks (4)

Remark (PJ): Colin Stirling was the first to use the above idea to derive a primitive recursive complexity bound. I have had a closer look at the method, deriving the upper bound $2 \uparrow\uparrow g(n)$ on the length of offending words (in the framework of det-1st-order grammars). Doing this, I derived an auxiliary proposition (Proposition 26 in my arxiv-version-3 paper); this seems to be a variant (or an instance) of the Subwords Lemma in Sénizergues' ICALP'03 paper. (I will refer to this in later version(s).)

Informal summary and some further remarks (5)

I claimed in the arxiv-paper (and at the end of the talk) that the decidability of bisimulation equivalence for (nondeterministic) 1st-order grammars can be shown in principle in the same way as for the det-1st-order grammars, after solving some technical problems.
In the discussion Géraud Sénizergues doubted that my approach works for bisimilarity, so I was writing here previously: ... either I have a serious problem in my proof (which should be demonstrated), or our approaches are not very closely isomorphic ...
Géraud later really demonstrated a serious problem, see http://arxiv.org/abs/1101.5046. The crucial point is something which was trivial in the soundness proof of the determ-case: any offending word decreased the real eq-level of respective pairs of objects, and this property was not influenced by the subtree replacement (from larger eq-levels).
This is true in the nondeterministic case for relative eq-levels wrt a fixed Defender's strategy, but one must be careful when mixing them with the real (absolute) eq-levels. I was not enough careful, which is my shame ...