Ramsey-based Inclusion for Visibly Pushdown Languages

Friedmann, Klaedtke, Lange, ICALP'14.

Goal: · Check inclusion among languages accepted by visibly pushdowns.

· Do not use determinization nor complementation.

· Instead, generalize Ramsey-based approaches
from Büchi automata to visibly pushdowns.

1. Visibly Pushdown Automata

Idea: · Restrict non-deterministic pushdowns:
input symbol determines when the automaton pushes or pops.

· Consequence: for a given input word,
stack height is identical for the same position in all runs.

· Hence, given several visibly pushdowns over the same alphabet, when reading a word their different stacks agree on the height, and we can form a product:

$$\begin{array}{|c|}\hline a \\ \hline b \\ \hline c \\ \hline\end{array} \times \begin{array}{|c|}\hline x \\ \hline y \\ \hline z \\ \hline\end{array} = \begin{array}{|c|}\hline (a,x) \\ \hline (b,y) \\ \hline (c,z) \\ \hline\end{array}$$

More formally,
visibly pushdown languages are closed
under intersection and complement.

Motivation: · Verification of recursive programs
↳ Visibly push/pop = call/return.
↳ $L(VPL_1) \subseteq L(VPL_2)$ EXPTIME-complete.
↳ Can express properties like
"an acquired lock must be released in the same procedure".

Note:

- Boxes are not easy to generalize:
  - ↳ Need finitely many boxes but
  - ↳ have to take care of an unbounded stack.
- Does not work for general pushdowns, universality undecidable.

Definition:

- A *partitioned alphabet* is a finite set Σ of the form
$$\Sigma = \Sigma_{int} \uplus \Sigma_{call} \uplus \Sigma_{ret}.$$
We attach an opening bracket $\langle$ to call symbols and a closing bracket $\rangle$ to return symbols.
- A *nested word* is a word $w \in \Sigma^* \cup \Sigma^\omega$.
Call or return letters without a matching bracket are called *pending*.
- We use $NW^{*/\omega}(\Sigma)$ for the set of all *well-matched* nested words that do not have pending calls/returns.

Example:

Let $\Sigma_{int} = \{a\}$, $\Sigma_{call} = \{b, c\}$, $\Sigma_{ret} = \{d\}$.

Consider

$$w = a \ \ d\rangle \ \ \langle b \ \ a \ \ \langle c \ \ d\rangle \ \ d\rangle \ \ \langle b \ \ \langle c$$

Definition:

- A *visibly-pushdown automaton (VPA)* is a tuple

$\mathcal{A} = (Q, \Gamma, \Sigma, \delta, q_I, \Omega)$

with

- Q = finite set of states, $q_I \in Q$ initial state,
- Γ = finite stack alphabet, $\bot \notin \Gamma$,
- $\Sigma = \Sigma_{int} \uplus \Sigma_{call} \uplus \Sigma_{ret}$ partitioned alphabet
- $\delta = \delta_{int} \uplus \delta_{call} \uplus \delta_{ret}$ with

$$\delta_{int} : Q \times \Sigma_{int} \longrightarrow \mathcal{P}(Q)$$
$$\delta_{call} : Q \times \Sigma_{call} \longrightarrow \mathcal{P}(Q \times \Gamma)$$
$$\delta_{ret} : Q \times (\Gamma \uplus \{\bot\}) \times \Sigma_{ret} \longrightarrow \mathcal{P}(Q)$$

transition relation

- $\Omega : Q \longrightarrow \mathbb{N}$ priority function.

- We write $\Gamma_\bot$ for $\Gamma \uplus \{\bot\}$.

The <u>size</u> of $\mathcal{A}$ is $|Q|$.

The <u>index</u> of $\mathcal{A}$ is $|\Omega(Q)|$.

- A <u>run</u> of $\mathcal{A}$ on $w = w_0 w_1 \ldots \in \Sigma^\omega$ is a word

$$(q_0, \gamma_0)(q_1, \gamma_1) \ldots \in (Q \times \Gamma_\bot^{\,+})^\omega$$

with $(q_0, \gamma_0) = (q_I, \bot)$

and so that for each $i \in \mathbb{N}$ the following holds:

(1) If $w_i \in \Sigma_{int}$ then $q_{i+1} \in \delta(q_i, w_i)$ and $\gamma_{i+1} = \gamma_i$.

(2) If $w_i \in \Sigma_{call}$ then $(q_{i+1}, B) \in \delta(q_i, w_i)$ and $\gamma_{i+1} = B.\gamma_i$, for some $B \in \Gamma$.

(3) If $w_i \in \Sigma_{ret}$ and $B.u$ with $B \in \Gamma_\bot$ and $u \in \Gamma_\bot^{\,*}$, then $q_{i+1} \in \delta_{ret}(q_i, B, w_i)$ and $\gamma_{i+1} = u$ if $u \neq \varepsilon$, $\gamma_{i+1} = \bot$ otherwise.

The run is *accepting*, if

$$\max\{\Omega(q) \mid q \in \inf(q_0 q_1 \dots)\} \text{ is } \underline{\text{even}}.$$

- Runs on finite words are defined as expected. Acceptance is by reaching a state with even priority.

- The language is

$$\mathcal{L}^{*/\omega}(\mathcal{A}) := \{w \in \mathbb{N}W^{*/\omega}(\Sigma) \mid \mathcal{A} \text{ has an accepting run on } w\}.$$

Definition (Priority and Reward Ordering):

- For a set S, let $+$ always denote an element not contained in S. Write

$$S_+ := S \uplus \{+\}.$$

Why? Use $+$ for partial functions into S.

- Define two orderings on $\mathbb{N}_+$.

The *priority ordering* is

$$0 \sqsubset 1 \sqsubset 2 \sqsubset \dots \sqsubset +.$$

The *reward ordering* is

$$+ < \dots < 5 < 3 < 1 < 0 < 2 < 4 < \dots$$

Note: $+$ is maximal for $\sqsubset$ and minimal for $<$.

Write $\bigsqcup S$ resp. $\bigvee S$ for the maximum in a finite non-empty set S wrt. $\sqsubseteq$ resp. $\leq$.

Idea: The reward ordering expresses how valuable a priority is for acceptance.

Small odd priorities are easier to subsume and hence better for acceptance than large odd priorities.
$\dagger$ stands for non-existence of a run.

2. Universality Checking

Goal: Give an algorithm to check whether $L^\omega(A) = NW^\omega(\Sigma)$.

Throughout the section, fix
$$A = (Q, \Gamma, \Sigma, \delta, q_I, \Omega).$$

Approach: Use a suitable notion of boxes.

Definition:
We define three kinds of transition profiles (TPs).

- An int-TP is a function
$$Q \times Q \to \Omega(Q)_\dagger.$$
We associate with $a \in \Sigma_{int}$ the int-TP f_a defined by
$$f_a(q, q') := \begin{cases} \Omega(q'), & \text{if } q' \in \delta_{int}(q, a) \\ \dagger, & \text{otherwise.} \end{cases}$$

- A call-TP is a function
$$Q \times \Gamma \times Q \to \Omega(Q)_\dagger.$$
We associate with $a \in \Sigma_{call}$ the call-TP f_a defined by
$$f_a(q, B, q') := \begin{cases} \Omega(q'), & \text{if } (q' ; B) \in \delta_{call}(q, a) \\ \dagger, & \text{otherwise.} \end{cases}$$

- A ret-TP is a function
  $$Q \times T_{\perp}^{!} \times Q \longrightarrow \mathcal{R}(Q)_{\top}.$$
  We associate with $a \in \Sigma_{ret}$
  the ret-TP f_a defined by
  $$f_a(q, B, q') := \begin{cases} \mathcal{R}(q'), & \text{if } q' \in \delta_{ret}(q, B, a) \\ \top, & \text{otherwise.} \end{cases}$$
- A TP of the form f_a with $a \in \Sigma$ is called atomic.
  For $\tau \in \{int, ret, call\}$, define the set of atomic TPs
  $$T_\tau := \{ f_a \mid a \in \Sigma_\tau \}.$$

Idea:

- TPs describe the behavior of A on single letters.
- To describe the behavior of A on words, compose TPs.
- Composition $f ; g$ can only be applied
  to TPs of certain kind:
  They describe the behavior of A on words such that
  after reading the word the stack height has changed by ≤ 1.

Definition (TP Composition):

Let f, g be TPs.
We define six compositions, depending on the types of f and g.

- If f and g are int-TPs, then
  $$(f ; g)(q, q') := \bigvee \{ f(q, q'') \sqcup g(q'', q') \mid q'' \in Q \}$$
- If f is an int-TP and g is a call-TP or a ret-TP, then
  $$(f ; g)(q, B, q') := \bigvee \{ f(q, q'') \sqcup g(q'', B, q') \mid q'' \in Q \}$$

$(g;f)(q,B,q') := \bigvee \{ g(q,B,q'') \sqcup f(q'',q') \mid q'' \in Q \}$.

- If f is a <u>call-TP</u> and g is a <u>ret-TP</u>, then

$(f;g)(q,q') := \bigvee \{ f(q,B,q'') \sqcup g(q'',B,q') \mid q'' \in Q, B \in T \}$.

<u>Note</u>:

- We take the maximum value on paths according to the priority ordering.
- Then we take the maximum over these values according to the reward ordering.

We generalize ; to sets:

$F;G := \{ f;g \mid f \in F, g \in G, f;g \text{ defined} \}$.

We map words to TPs.

↳ We map $a \in \Sigma$ to f_a.

↳ If $u \in \Sigma^+$ is mapped to f
and $v \in \Sigma^+$ is mapped to g,
then $u.v$ is mapped to $f;g$, provided this is defined.

The relation is indeed a function.

<u>Lemma</u>:

If $(h;f);(g;h)$ and $h;((f;g);h)$ are both defined, then they are equal.

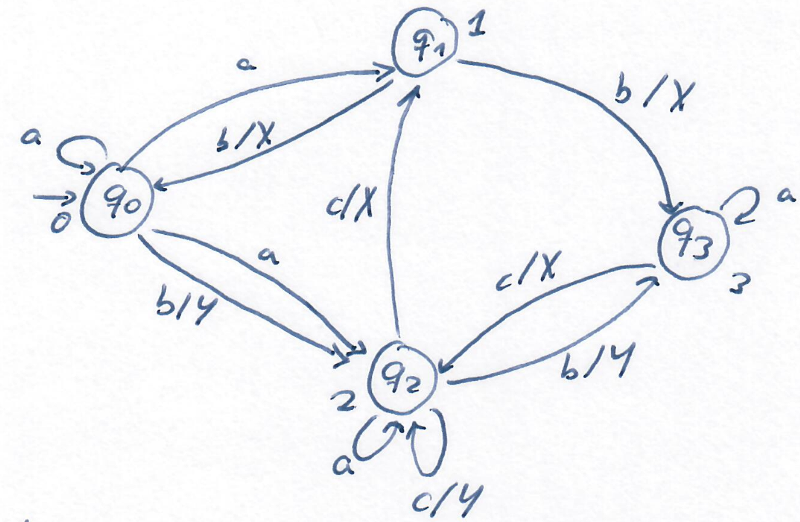
We write f_u for the TP of u.
Note that $f_u = f_v$ may hold for $u \neq v$.
In this case, the behavior on u is the same as on v.

Example:

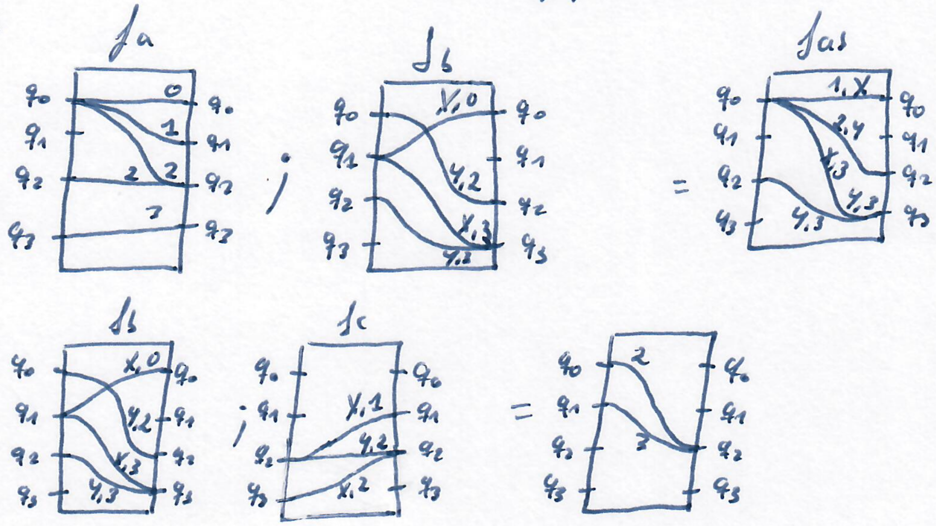
Let $\Sigma_{int} = \{a\}$, $\Sigma_{call} = \{b\}$, $\Sigma_{ret} = \{c\}$.

Let $\Gamma = \{X, Y\}$.

Let A:



Then



Definition:

Let I be the least solution to the equation

$$X = T_{int} \cup T_{call} ; T_{ret} \cup T_{call} ; X ; T_{ret} \cup X ; X,$$

when interpreting the equation over the powerset lattice of int-TPs.

Lemma:

- $\cup$ and $;$ are monotonic as operations over sets of int-TPs.
- Hence, the least solution can be found using Kleene iteration in a finite number of steps.

Goal: Use the elements of $\mathcal{I}$
to characterize non-universality of $\mathcal{A}$.

The following will be helpful.

Lemma:

If $f \in \mathcal{I}$, then there is $w \in Nw^+(\Sigma)$ with $f = f_w$.

Definition:

Let f be an int-TP.

(i) f is idempotent, if $f;f = f$.
Note that only int-TPs can be idempotent.

(ii) For $q \in Q$, let

$$f(q) := \{ q' \in Q \mid f(q, q') \neq \top \}.$$

Also use

$$f(Q) := \bigcup_{q \in Q} f(q).$$

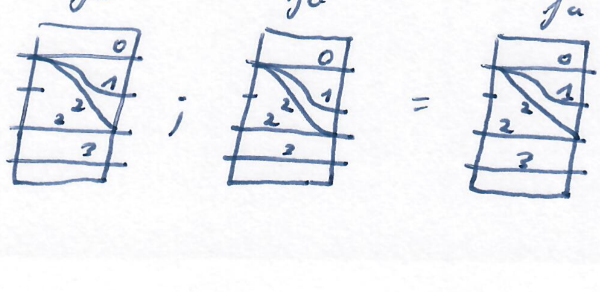
(iii) f is bad for the set $Q' \subseteq Q$, if

$$\forall q \in f(Q') : \ f(q,q) \text{ is } \top \text{ or odd.}$$

By definition, any TP is bad for $\emptyset$.
A TP is good if it is not bad.
We only consider good/bad for idempotent TPs.

Example:

Consider $f_a ; f_a = f_a$ idempotent.



- Since $f_a(q_2, q_2) = 2$,

  f_a is <u>good for</u> any $Q' \subseteq \{q_0, \ldots, q_3\}$ with $q_2 \in Q'$.

  Intuition: There is at least one run on a^ω
  that starts in q_2 and loops through q_2 infinitely often.
  On this run, 2 is the highest priority that occurs infinitely often.
  So if v is a prefix with a run leading to q_2,
  $v.a^\omega$ is <u>accepted</u> by the VPA.

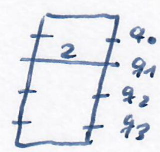
- We have that f_a is bad for $\{q_1, q_3\}$.

  Why? $f(q_1, q_1) = 1$ and $f(q_3, q_3) = 3$.

  So if there is a prefix v for which
  all runs starting in the initial state end in q_1 or q_3,
  then $v.a^\omega$ is <u>not accepted</u> by the VPA.

- Another idempotent int-TP is

$$g := f_b ; ((f_b ; f_c) ; f_c) =$$



Then g is bad for every $Q' \subseteq Q$ with $q_1 \notin Q'$.

The following theorem characterizes universality.

<u>Theorem:</u>

$\mathcal{L}^\omega(A) \neq NW^\omega(\Sigma)$ iff

there are $f, g \in \mathcal{I}$ with g idempotent
and bad for $f(q_I)$.

Algorithm:

$N \leftarrow T_{int} \cup T_{call}; T_{ret}$ // N = Newly generated TPs

$T \leftarrow N$

while $N \neq \emptyset$ **do**

  for all $(f_u, f_r) \in N \times T \cup T \times N$ **do**

    if f_r idempotent and bad for $f_u(q_I)$ **then**

      return universality does not hold, witness $u.v^\omega$ // early termination.

    fi

  od

  $N \leftarrow (N; T \cup T; N \cup T_{call}; N; T_{ret}) \setminus T$

  $T \leftarrow T \cup N$

od

return universality holds.

Note:

To return a witness, we have to maintain representatives.

Theorem:

Let $size(A) = n \geq 1$ and $index(A) = k \geq 2$.

Let $m = \max\{|\Sigma|, 1, |P|\}$.

The above algorithm runs in time $m^3 2^{O(n^2 \log k)}$.

Tuning:

- Store TPs in a hash table.
- Maintain pointers to newly generated TPs.
- Maintain pointers to idempotent TPs.
- Implement the following antichain idea that does not improve the worst-case complexity but is valuable in practice.

Antichain Idea (De Wulf, Doyen, Henzinger, Raskin. CAV '06)

For the badness check,

it is sufficient to know $\underbrace{f_u(q_I)}$ with $f_u \in T$:

sets $Q' \subseteq Q$ for which
all runs on some well-matched word
end in some state Q'.

We can maintain a set R
storing this information:

Init: $R := \{(\varepsilon, \{q_I\})\}$

Update: $R := R \cup \{(u.v, f_v(Q')) \mid (u, Q') \in R, f_v \in T\}$
after reassigning T/N.

The antichain idea is to

optimize R by removing (u, Q')
if there is (u', Q'') with $Q'' \subseteq Q'$. // Badness is more likely
to hold for Q''.

3. Inclusion Checking

Setting:
- For simplicity, we consider a single VPA
and check inclusion between languages of states q_I^1 and q_I^2.
- The case of two VPAs can be reduced to this one
by taking their disjoint union.
- For the remainder of the section, let
$A_i = (Q, P, \Sigma, \delta, q_I^i, \Omega)$ with $i = 1, 2$.

Definition:
- A tagged transition profile of type int, an int-TTP for short,
is an element
$(Q \times \Omega(Q) \times Q) \times (Q \times Q \to \Omega(Q)_\bot)$.

We write (p, c, p', f) as $f^{(p,c,p')}$ to indicate

we have the int-TP f extended with a tuple of states and priority.

- A <u>call-TTP</u> is from

$$(Q \times T' \times \Omega(Q) \times Q) \times (Q \times T' \times Q \to \Omega(Q)_+),$$

written as $f^{(p,B,c,p')}$.

- A <u>ret-TTP</u> is from

$$(Q \times \Omega(Q) \times T_\perp \times Q) \times (Q \times T'_\perp \times Q \to \Omega(Q)_+),$$

written as $f^{(p,c,B,p')}$.

<u>Intuition:</u>

Consider the int-TTP $f^{(p,c,p')}$.

- Then f captures essential information about <u>all</u> runs of A_2 on a well-matched word $u \in \Sigma^+$.
- The attached information (p, c, p') describes the existence of <u>some</u> run of A_1 on u. The run starts in p, ends in p', and has c as the maximal priority.

<u>Definition:</u>

With each letter $a \in \Sigma$, we associate a set F_a of TTPs:

(1) If $a \in \Sigma_{int}$, then $F_a := \{ \underbrace{f_a}_{\text{as before}}^{(p,\Omega(p'),p')} \mid p' \in \delta_{int}(p,a)\}$.

(2) If $a \in \Sigma_{call}$, then $F_a := \{ f_a^{(p,B,\Omega(p'),p')} \mid (p', B) \in \delta_{call}(p,a)\}$.

(3) If $a \in \Sigma_{ret}$, then $F_a := \{ f_a^{(p,\Omega(p'),B,p')} \mid p' \in \delta_{ret}(p,B,a)\}$.

As before, composition is limited to certain cases:

$$\text{int-TTP} \,;\, \text{int-TTP} = \text{int-TTP}$$

$$\text{int-TTP} \,;\, \begin{matrix}\text{call-TTP}\\ \text{ret}\end{matrix} = \begin{matrix}\text{call-TTP}\\ \text{ret}\end{matrix}$$

$$\text{call-TTP} \,;\, \text{ret-TTP} = \text{int-TTP}.$$

Definition:

- Let $f^{(p,c,p')}$ and $g^{(p',c',p'')}$ be int-TTPs:

$$f^{(p,c,p')} \,;\, g^{(p',c',p'')} := (f;g)^{(p,\, c \cup c',\, p'')}.$$

- Let $f^{(p,c,p')}$ be an int-TTP and $g^{(q,B,c',q')}$ be a call-TTP:

$$f^{(p,c,p')} \,;\, g^{(q,B,c',q')} := (f;g)^{(p,B,\, c \cup c',\, q')}, \quad \text{if } q = p'.$$

$$g^{(q,B,c',q')} \,;\, f^{(p,c,p')} := (g;f)^{(q,B,\, c \cup c',\, p')}, \quad \text{if } p = q'.$$

For ret-TTPs, the definition is similar.

- Let $f^{(p,B,c,p')}$ be a call-TTP and $g^{(p',c',B,p'')}$ a ret-TTP:

$$f^{(p,B,c,p')} \,;\, g^{(p',c',B,p'')} := (f;g)^{(p,\, c \cup c',\, p'')}.$$

Note that B is the same symbol in both annotations.

- As before, we generalize the composition to sets of TTPs.

Definition:

Let I be the least solution to the equation

$$X = T_{int} \cup T_{call} ; T_{ret} \cup T_{call} ; X ; T_{ret} \cup X ; X,$$

where $T_\tau := \bigcup_{a \in \Sigma_\tau} F_a$ with $\tau \in \{int, call, ret\}$.

Theorem:

$L^\omega(A_1) \not\subseteq L^\omega(A_2)$ iff there are $f^{(q_I,c,p)}, g^{(p,d,p)} \in I$

with (i) d even and (ii) g idempotent and bad for $f(q_I^2)$.