IA159 Formal Verification Methods

LTL Model Checking of Pushdown Systems

Jan Strejček

Faculty of Informatics
Masaryk University

Focus and sources

Focus

- pushdown systems
- representation of sets of configurations
- computing all predecessors: checking safety properties
- state-based LTL model checking

Sources

- J. Esparza, D. Hansel, P. Rossmanith, and S. Schwoon: *Efficient algorithms for model checking pushdown systems*, CAV 2000, LNCS 1855, Springer, 2000.
- S. Schwoon: *Model-Checking Pushdown Systems*, PhD thesis, TUM, 2002.

Motivation

Pushdown systems can be used to precisely model sequential programs with procedure calls, recursion, and both local and global variables.

Pushdown systems

A pushdown system is a triple $\mathcal{P} = (P, \Gamma, \Delta)$, where

- P is a finite set of control locations,
- Γ is a finite stack alphabet,
- $\Delta \subseteq (P \times \Gamma) \times (P \times \Gamma^*)$ is a finite set of transition rules.

We write $\langle q, \gamma \rangle \hookrightarrow \langle q', w \rangle$ instead of $((q, \gamma), (q', w)) \in \Delta$.

We do not consider any input alphabet as we do not use pushdown systems as language acceptors.

Definitions

- a configuration of $\mathcal{P}$ is a pair $\langle p, w \rangle \in P \times \Gamma^*$, where w is a stack content (the topmost symbol is on the left)
- the set of all configurations is denoted by $\mathcal{C}$
- an immediate successor relation on configurations is defined in standard way
- reachability relation $\Rightarrow \subseteq \mathcal{C} \times \mathcal{C}$ is the reflexive and transitive closure of the immediate successor relation
- $\stackrel{+}{\Rightarrow} \subseteq \mathcal{C} \times \mathcal{C}$ is the transitive closure of the immediate successor relation
- given a set $C \subseteq \mathcal{C}$ of configurations, we define the set of their predecessors as

$$pre^*(C) = \{c \in \mathcal{C} \mid \exists c' \in C \,.\, c \Rightarrow c'\}$$

$\mathcal{P}$-automata

$\mathcal{P}$-automata

- are finite automata used to represent sets of configurations
- use Γ as an alphabet
- have one initial state for every control location of the pushdown (we use P as the set of initial states)

Given a pushdown system $\mathcal{P} = (P, \Gamma, \Delta)$, a $\mathcal{P}$-automaton (or simply automaton) is a tuple $\mathcal{A} = (Q, \Gamma, \delta, P, F)$ where

- Q is a finite set of states such that $P \subseteq Q$,
- $\delta \subseteq Q \times \Gamma \times Q$ is a set of transitions,
- $F \subseteq Q$ is a set of final states.

More definitions

- a (reflexive and transitive) transition relation $\rightarrow \subseteq Q \times \Gamma^* \times Q$ is defined in a standard way
- $\mathcal{P}$-automaton $\mathcal{A}$ represents the set of configurations

$$Conf(\mathcal{A}) = \{\langle p, w \rangle \mid \exists q \in F \,.\, p \xrightarrow{w} q\}$$

- a set of configurations of $\mathcal{P}$ is called regular if it is recognized by some $\mathcal{P}$-automaton

Notation convention

In the rest of this section, we use

- $p, p', p'', \ldots$ to denote initial states of an automaton (i.e. elements of P)
- $s, s', s'', \ldots$ to denote non-initial states, and
- $q, q', q'', \ldots$ to denote arbitrary states (initial or not).

Verification of pushdown systems: the first step

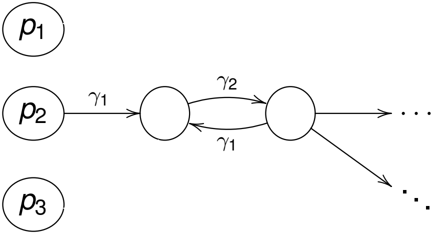
Computing $pre^*(C)$ for a regular set C

Statements

1. Given a pushdown system $\mathcal{P}$ and a regular set of configurations C, the set $pre^*(C)$ is again regular.
2. If C is defined by a $\mathcal{P}$-automaton $\mathcal{A}$, then the automaton $\mathcal{A}_{pre^*}$ representing $pre^*(C)$ is effectively constructible.

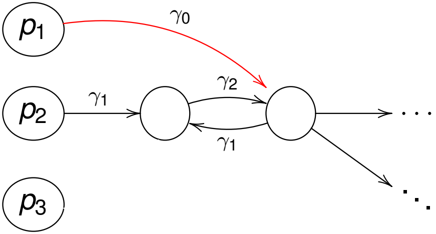
Intuition

$$\langle p_1, \gamma_0 \rangle \hookrightarrow \langle p_2, \gamma_1\gamma_2 \rangle$$
$$\langle p_3, \gamma_3 \rangle \hookrightarrow \langle p_1, \gamma_0\gamma_1 \rangle$$



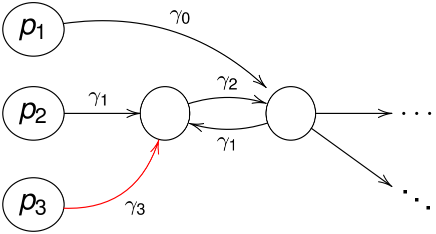
Intuition

$$\langle p_1, \gamma_0 \rangle \hookrightarrow \langle p_2, \gamma_1\gamma_2 \rangle$$
$$\langle p_3, \gamma_3 \rangle \hookrightarrow \langle p_1, \gamma_0\gamma_1 \rangle$$



Intuition

$$\langle p_1, \gamma_0 \rangle \hookrightarrow \langle p_2, \gamma_1 \gamma_2 \rangle$$
$$\langle p_3, \gamma_3 \rangle \hookrightarrow \langle p_1, \gamma_0 \gamma_1 \rangle$$



Idea

Let $\mathcal{P}$ be a pushdown system and $\mathcal{A}$ be a $\mathcal{P}$-automaton. We assume (w.l.o.g.) that $\mathcal{A}$ has no transition leading to an initial state. The automaton $\mathcal{A}_{pre^*}$ is obtained from $\mathcal{A}$ by addition of new transitions according to the following rule:

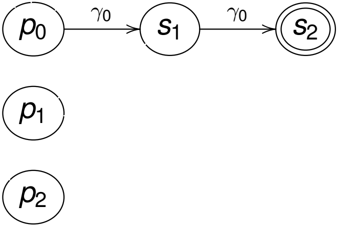
Saturation rule

If $\langle p, \gamma \rangle \hookrightarrow \langle p', w \rangle$ and $p' \xrightarrow{w} q$ in the current automaton, add a transition (p, γ, q).

- we apply this rule repeatedly until we reach a fixpoint
- a fixpoint exists as the number of possible new transitions is finite
- the resulting $\mathcal{P}$-automaton is $\mathcal{A}_{pre^*}$

Example

$\mathcal{A}$

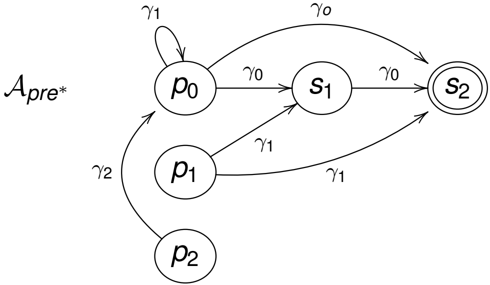


transition rules of $\mathcal{P}$:

$\langle p_0, \gamma_0 \rangle \hookrightarrow \langle p_1, \gamma_1 \gamma_0 \rangle \qquad \langle p_2, \gamma_2 \rangle \hookrightarrow \langle p_0, \gamma_1 \rangle$

$\langle p_1, \gamma_1 \rangle \hookrightarrow \langle p_2, \gamma_2 \gamma_0 \rangle \qquad \langle p_0, \gamma_1 \rangle \hookrightarrow \langle p_0, \varepsilon \rangle$

Example



transition rules of $\mathcal{P}$:

$$\langle p_0, \gamma_0 \rangle \hookrightarrow \langle p_1, \gamma_1\gamma_0 \rangle \qquad \langle p_2, \gamma_2 \rangle \hookrightarrow \langle p_0, \gamma_1 \rangle$$

$$\langle p_1, \gamma_1 \rangle \hookrightarrow \langle p_2, \gamma_2\gamma_0 \rangle \qquad \langle p_0, \gamma_1 \rangle \hookrightarrow \langle p_0, \varepsilon \rangle$$

Normal form

A pushdown system is in normal form if every rule $\langle p, \gamma \rangle \hookrightarrow \langle p', w \rangle$ satisfies $|w| \leq 2$.

Any pushdown system can be transformed into normal form with only linear size increase.

Algorithm: notes

We give an algorithm that, for a given $\mathcal{A}$, computes transitions of $\mathcal{A}_{pre^*}$. The rest of the automaton $\mathcal{A}_{pre^*}$ is identical to $\mathcal{A}$.

The algorithm uses sets *rel* and *trans* containing the transitions that are known to belong to $\mathcal{A}_{pre^*}$:

- *rel* contains transitions that have already been examined
- no transition is examined more than once
- when we have a rule $\langle p, \gamma \rangle \hookrightarrow \langle p', \gamma' \gamma'' \rangle$ and transitions $t_1 = (p', \gamma', q')$ and $t_2 = (q', \gamma'', q'')$ (where q, q' are arbitrary states), we have to add transition (p, γ, q'')
- we do it in such a way that whenever we examine t_1, we check if there is a corresponding $t_2 \in rel$ and we add an extra rule $\langle p, \gamma \rangle \hookrightarrow \langle q', \gamma'' \rangle$ to a set of such extra rules Δ'
- the extra rule guarantees that if a suitable t_2 will be examined in the future, (p, γ, q'') will be added.

Algorithm

Input: a pushdown system $\mathcal{P} = (P, \Gamma, \Delta)$ in normal form
a $\mathcal{P}$-automaton $\mathcal{A}=(Q, \Gamma, \delta, P, F)$ without transitions into P
Output: the set of transitions of $\mathcal{A}_{pre^*}$

$rel := \emptyset;\ trans := \delta;\ \Delta' := \emptyset;$
forall $\langle p, \gamma \rangle \hookrightarrow \langle p', \varepsilon \rangle \in \Delta$ **do** $trans := trans \cup \{(p, \gamma, p')\};$
while $trans \neq \emptyset$ **do**
    pop $t = (q, \gamma, q')$ from *trans*;
    if $t \notin rel$ **then**
        $rel := rel \cup \{t\};$
        forall $\langle p_1, \gamma_1 \rangle \hookrightarrow \langle q, \gamma \rangle \in (\Delta \cup \Delta')$ **do**
            $trans := trans \cup \{(p_1, \gamma_1, q')\};$
        forall $\langle p_1, \gamma_1 \rangle \hookrightarrow \langle q, \gamma\gamma_2 \rangle \in \Delta$ **do**
            $\Delta' := \Delta' \cup \{\langle p_1, \gamma_1 \rangle \hookrightarrow \langle q', \gamma_2 \rangle\};$
            forall $(q', \gamma_2, q'') \in rel$ **do**
                $trans := trans \cup \{(p_1, \gamma_1, q'')\};$
return *rel*

Theorem

Theorem

Let $\mathcal{P} = (P, \Gamma, \Delta)$ be a pushdown system and $\mathcal{A} = (Q, \Gamma, \delta, P, F)$ be a $\mathcal{P}$-automaton. There exists an automaton $\mathcal{A}_{pre^*}$ recognizing $pre^*(Conf(\mathcal{A}))$. Moreover, $\mathcal{A}_{pre^*}$ can be constructed in $\mathcal{O}(|Q|^2 \cdot |\Delta|)$ time and $\mathcal{O}(|Q| \cdot |\Delta| + |\delta|)$ space.

Proof

- We can assume that every transition is added to *trans* at most once. This can be done (without asymptotic loss of time) by storing all transitions which are ever added to *trans* in an additional hash table.
- Further, we assume that there is at least one rule in Δ for every $\gamma \in \Gamma$ (transitions of $\mathcal{A}$ under some γ not satisfying this assumption can be moved directly to *rel*).
- The number of transitions in δ as well as the number of iterations of the while-loop is bounded by $|Q|^2 \cdot |\Delta|$.

Proof: time complexity

- Line 10 is executed for each combination of a rule $\langle p_1, \gamma_1 \rangle \hookrightarrow \langle q, \gamma\gamma_2 \rangle$ and a transition $(q, \gamma, q') \in \mathit{trans}$, i.e. at most $|Q| \cdot |\Delta|$ times.
- Hence, $|\Delta'| \leq |Q| \cdot |\Delta|$.
- For the loop starting at line 11, q' and γ_2 are fixed. Thus, line 12 is executed at most $|Q|^2 \cdot |\Delta|$ times.
- Line 8 is executed for each combination of a rule $\langle p_1, \gamma_1 \rangle \hookrightarrow \langle q, \gamma \rangle \in (\Delta \cup \Delta')$ and a transition $(q, \gamma, q') \in \mathit{trans}$. As $|\Delta'| \leq |Q| \cdot |\Delta|$, line 8 is executed at most $\mathcal{O}(|Q|^2 \cdot |\Delta|)$ times.

As a conclusion, the algorithm takes $\mathcal{O}(|Q|^2 \cdot |\Delta|)$ time.

Proof: space complexity

Memory is needed for storing *rel*, *trans*, and Δ'.

- The size of Δ' is in $\mathcal{O}(|Q| \cdot |\Delta|)$.
- Line 1 adds $|\delta|$ transitions to *trans*.
- Line 2 adds at most $|\Delta|$ transitions to *trans*.
- In lines 8 and 12, p_1 and γ_1 are given by the head of a rule in Δ (note that every rule in Δ' have the same head as some rule in Δ). Hence, lines 8 and 12 add at most $|Q| \cdot |\Delta|$ different transitions.

We directly get that the algorithm needs $\mathcal{O}(|Q| \cdot |\Delta| + |\delta|)$ space. As this is also the size of the result *rel*, the algorithm is optimal with respect to the memory usage.

Notes

- the algorithm can be used to verify safety property: given an automaton $\mathcal{A}$ representing error configurations, we can compute $\mathcal{A}_{pre^*}$, i.e. the set of all configurations from which an error configuration is reachable
- there is a similar algorithm computing, for a given regular set of configurations C, the set of all successors

$$post^*(C) = \{c' \in \mathcal{C} \mid \exists c \in C \,.\, c \Rightarrow c'\}$$

Theorem

Let $\mathcal{P} = (P, \Gamma, \Delta)$ be a pushdown system and $\mathcal{A} = (Q, \Gamma, \delta, P, F)$ be a $\mathcal{P}$-automaton. There exists an automaton $\mathcal{A}_{post^*}$ recognizing $post^*(Conf(\mathcal{A}))$. Moreover, $\mathcal{A}_{post^*}$ can be constructed in $\mathcal{O}(|P| \cdot |\Delta| \cdot (|Q| + |\Delta|) + |P| \cdot |\delta|)$ time and space.

Verification of pushdown systems: the second step

LTL model checking

The problem

The global state-based LTL model checking problem for pushdown systems

Compute the set of all configurations of a given pushdown system $\mathcal{P}$ that violate a given LTL formula φ (where a configuration c violates φ if there is a path starting from c and not satisfying φ).

Extending pushdown systems

- state-based $\implies$ validity of atomic propositions
- labelling function $L : (P \times \Gamma) \to 2^{AP}$ assigns valid atomic propositions to every pair (p, γ) of a control location p and a topmost stack symbol γ
- pushdown system $\mathcal{P}$ and L define Kripke structure
  - states = configurations of $\mathcal{P}$
  - transition relation = immediate successor relation
  - no initial states (global model checking)
  - labelling function is an extension of L: $L(\langle p, \gamma w \rangle) = L(p, \gamma)$

The schema

pushdown system $\mathcal{P}$
with labelling function L

Kripke structure
with infinitely many states

LTL formula φ

Büchi automaton $\mathcal{A}_{\neg\varphi}$
words over $2^{AP(\varphi)}$ violating φ

product: Büchi pushdown system $\mathcal{BP}$

Accepting run problem
to compute the set of all configurations
of $\mathcal{BP}$ where accepting runs start

Büchi pushdown system

Büchi pushdown system = pushdown system with a set of accepting control locations.

An accepting run of a Büchi pushdown system is a path passing through some accepting control location infinitely often.

Product

Product of

- a pushdown system $\mathcal{P} = (P, \Gamma, \Delta)$ with a labelling L and
- a Büchi automaton $\mathcal{A}_{\neg\varphi} = (2^{AP(\varphi)}, Q, \delta, q_0, F)$

is a Büchi pushdown system $\mathcal{BP} = ((P \times Q), \Gamma, \Delta', G)$, where

$$\langle (p,q), \gamma \rangle \hookrightarrow \langle (p',q'), w \rangle \in \Delta' \quad \text{if} \quad \begin{array}{l} \langle p, \gamma \rangle \hookrightarrow \langle p', w \rangle \in \Delta \text{ and} \\ q' \in \delta(q, L(p,\gamma) \cap AP(\varphi)) \end{array}$$

and $G = P \times F$.

Clearly, a configuration $\langle p, w \rangle$ of $\mathcal{P}$ violates φ if $\mathcal{BP}$ has an accepting run starting from $\langle (p, q_0), w \rangle$.

Accepting run problem

The original model checking problem reduces to the following:

The accepting run problem

Compute the set $\mathcal{C}_a$ of configurations c of $\mathcal{BP}$ such that $\mathcal{BP}$ has an accepting run starting from c.

Repeating heads

$\Rightarrow$ denotes the (reflexive and transitive) reachability relation.
$\stackrel{+}{\Rightarrow}$ denotes the (transitive) reachability relation.

We define the relation $\stackrel{r}{\Rightarrow}$ on configurations of $\mathcal{BP}$ as

$$c \stackrel{r}{\Rightarrow} c' \quad \text{if} \quad c \Rightarrow \langle g, u \rangle \stackrel{+}{\Rightarrow} c'$$
for some configuration $\langle g, u \rangle$ with $g \in G$.

The head of a rule $\langle p, \gamma \rangle \hookrightarrow \langle p', w \rangle$ is the configuration $\langle p, \gamma \rangle$.
A head $\langle p, \gamma \rangle$ is repeating if $\langle p, \gamma \rangle \stackrel{r}{\Rightarrow} \langle p, \gamma v \rangle$ for some $v \in \Gamma^*$.
The set of repeating heads of $\mathcal{BP}$ is denoted by R.

Characterization of configurations with accepting runs

Lemma

Let c be a configuration of a Büchi pushdown system $\mathcal{BP}$.
$\mathcal{BP}$ has an accepting run starting from $c \iff$ there exists a repeating head $\langle p, \gamma \rangle$ such that $c \Rightarrow \langle p, \gamma w \rangle$ for some $w \in \Gamma^*$.

The implication "$\Longleftarrow$" is obvious.
We prove "$\Longrightarrow$".

Proof

- assume that $\mathcal{BP}$ has an accepting run

$$\langle p_0, w_0\rangle, \langle p_1, w_1\rangle, \langle p_2, w_2\rangle, \ldots$$

  starting from from c
- let $i_0, i_1, \ldots$ be an increasing sequence of indices such that
  - $|w_{i_0}| = \min\{|w_j| \mid j \geq 0\}$
  - $|w_{i_k}| = \min\{|w_j| \mid j > i_{k-1}\}$ for $k > 0$
- once a configuration $\langle p_{i_k}, w_{i_k}\rangle$ is reached, the rest of the run never looks at or changes the bottom $|w_{i_k}| - 1$ stack symbols

Proof

- let γ_{i_k} be the topmost symbol of w_{i_k} for each $k \geq 0$
- as the number of pairs (p_{i_k}, γ_{i_k}) is bounded by $|P \times \Gamma|$, there has to be a pair (p, γ) repeated infinitely many times
- moreover, since some $g \in G$ becomes a control location infinitely often, we can select two indeces $j_1 < j_2$ out of $i_0, i_1, \ldots$ such that

  $$\langle p_{j_1}, w_{j_1} \rangle = \langle p, \gamma w \rangle \overset{r}{\Rightarrow} \langle p_{j_2}, w_{j_2} \rangle = \langle p, \gamma v w \rangle$$

  for some $w, v \in \Gamma^*$
- as w is never looked at or changed in the rest of the run, we have that $\langle p, \gamma \rangle \overset{r}{\Rightarrow} \langle p, \gamma v \rangle$
- this proves "$\Longrightarrow$"

Consequences

Lemma

Let c be a configuration of a Büchi pushdown system $\mathcal{BP}$. $\mathcal{BP}$ has an accepting run starting from c $\iff$ there exists a repeating head $\langle p, \gamma \rangle$ such that $c \Rightarrow \langle p, \gamma w \rangle$ for some $w \in \Gamma^*$.

- the set of all configurations violating the considered formula φ can be computed as $pre^*(R\Gamma^*)$, where $R\Gamma^* = \{\langle p, \gamma w \rangle \mid \langle p, \gamma \rangle \in R, w \in \Gamma^*\}$
- as R is finite, $R\Gamma^*$ is clearly regular
- $pre^*(C)$ can be easily computed for regular sets C
- the only remaining step to solve the model checking problem is the algorithm computing R

Computing R

Computing R is reduced to a graph-theoretic problem.

Given a $\mathcal{BP} = (P, \Gamma, \Delta, G)$, we construct a graph $\mathcal{G} = (P \times \Gamma, E)$ representing the reachability relation between heads, i.e.

- nodes are the heads of $\mathcal{BP}$,
- $E \subseteq (P \times \Gamma) \times \{0, 1\} \times (P \times \Gamma)$ is the smallest relation satisfying the following rule:

Rule

If $\langle p, \gamma \rangle \hookrightarrow \langle p'', v_1 \gamma' v_2 \rangle$ and $\langle p'', v_1 \rangle \Rightarrow \langle p', \varepsilon \rangle$ then

1. $((p, \gamma), 1, (p', \gamma')) \in E$ if $\langle p'', v_1 \rangle \stackrel{r}{\Rightarrow} \langle p', \varepsilon \rangle$ or $p \in G$
2. $((p, \gamma), 0, (p', \gamma')) \in E$ otherwise.

Computing R

Rule

If $\langle p, \gamma\rangle \hookrightarrow \langle p'', v_1\gamma' v_2\rangle$ and $\langle p'', v_1\rangle \Rightarrow \langle p', \varepsilon\rangle$ then

1. $((p, \gamma), 1, (p', \gamma')) \in E$ if $\langle p'', v_1\rangle \overset{r}{\Rightarrow} \langle p', \varepsilon\rangle$ or $p \in G$
2. $((p, \gamma), 0, (p', \gamma')) \in E$ otherwise.

Edges are labelled with 1 if an accepting control state is passed between the heads, by 0 otherwise.

Conditions $\langle p'', v_1\rangle \Rightarrow \langle p', \varepsilon\rangle$ or $\langle p'', v_1\rangle \overset{r}{\Rightarrow} \langle p', \varepsilon\rangle$ can be checked by the algorithm for $pre^*(\{\langle p', \varepsilon\rangle\})$ or its small modification, respectively.

Computing R

Once $\mathcal{G}$ is constructed, R can be computed using the fact that:

a head $\langle p, \gamma \rangle$ is repeating $\iff$ (p, γ) is in a strongly connected component of $\mathcal{G}$ which has an internal edge labelled with 1

Example

The graph $\mathcal{G}$ for $\mathcal{BP} = (\{p_0, p_1, p_2\}, \{\gamma_0, \gamma_1, \gamma_2\}, \Delta, \{p_2\})$, where

$$\begin{aligned}\Delta = \{ \quad & \langle p_0, \gamma_0\rangle \hookrightarrow \langle p_1, \gamma_1\gamma_0\rangle, \quad \langle p_2, \gamma_2\rangle \hookrightarrow \langle p_0, \gamma_1\rangle, \\ & \langle p_1, \gamma_1\rangle \hookrightarrow \langle p_2, \gamma_2\gamma_0\rangle, \quad \langle p_0, \gamma_1\rangle \hookrightarrow \langle p_0, \varepsilon\rangle \quad \}.\end{aligned}$$

Rule

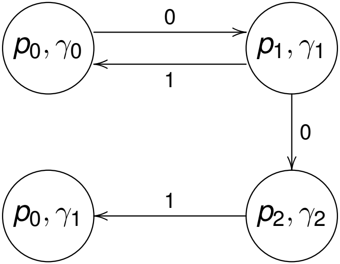
If $\langle p, \gamma\rangle \hookrightarrow \langle p'', v_1\gamma' v_2\rangle$ and $\langle p'', v_1\rangle \Rightarrow \langle p', \varepsilon\rangle$ then

1. $((p, \gamma), 1, (p', \gamma')) \in E$ if $\langle p'', v_1\rangle \overset{r}{\Rightarrow} \langle p', \varepsilon\rangle$ or $p \in G$
2. $((p, \gamma), 0, (p', \gamma')) \in E$ otherwise.

Example

The graph $\mathcal{G}$ for $\mathcal{BP} = (\{p_0, p_1, p_2\}, \{\gamma_0, \gamma_1, \gamma_2\}, \Delta, \{p_2\})$, where

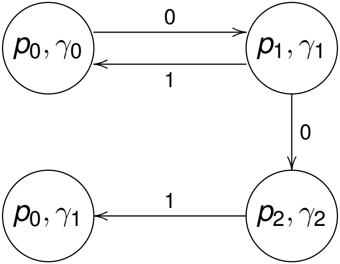
$$\Delta = \{ \quad \langle p_0, \gamma_0 \rangle \hookrightarrow \langle p_1, \gamma_1 \gamma_0 \rangle, \quad \langle p_2, \gamma_2 \rangle \hookrightarrow \langle p_0, \gamma_1 \rangle, \\ \langle p_1, \gamma_1 \rangle \hookrightarrow \langle p_2, \gamma_2 \gamma_0 \rangle, \quad \langle p_0, \gamma_1 \rangle \hookrightarrow \langle p_0, \varepsilon \rangle \quad \}.$$



Example

The graph $\mathcal{G}$ for $\mathcal{BP} = (\{p_0, p_1, p_2\}, \{\gamma_0, \gamma_1, \gamma_2\}, \Delta, \{p_2\})$, where

$$\begin{aligned}\Delta = \{ \quad & \langle p_0, \gamma_0\rangle \hookrightarrow \langle p_1, \gamma_1\gamma_0\rangle, \quad \langle p_2, \gamma_2\rangle \hookrightarrow \langle p_0, \gamma_1\rangle, \\ & \langle p_1, \gamma_1\rangle \hookrightarrow \langle p_2, \gamma_2\gamma_0\rangle, \quad \langle p_0, \gamma_1\rangle \hookrightarrow \langle p_0, \varepsilon\rangle \quad \}.\end{aligned}$$



Repeating heads: $\langle p_0, \gamma_0\rangle$, $\langle p_1, \gamma_1\rangle$

Algorithm: notes

We give an algorithm computing R for a given $\mathcal{BP}$ in normal form.

The algorithm runs in two phases.

1. It computes $\mathcal{A}_{pre^*}$ recognizing $pre^*(\{\langle p, \varepsilon\rangle \mid p \in P\})$. Every transition (p, γ, p') of $\mathcal{A}_{pre^*}$ signifies that $\langle p, \gamma\rangle \Rightarrow \langle p', \varepsilon\rangle$.

   We enrich the transitions of $\mathcal{A}_{pre^*}$: transitions (p, γ, p') are replaced by $(p, [\gamma, b], p')$ where b is a boolean. The meaning of $(p, [\gamma, 1], p')$ should be that $\langle p, \gamma\rangle \overset{r}{\Rightarrow} \langle p', \varepsilon\rangle$.
2. It constructs the graph $\mathcal{G}$, identifies its strongly conected components (e.g. using Tarjan's algorithm), and determines the set of repeating heads.

We define $G(p) = 1$ if $p \in G$ and $G(p) = 0$ otherwise.

Algorithm

Input: $\mathcal{BP} = (P, \Gamma, \Delta, G)$ in normal form Output: the set of repeating heads in $\mathcal{BP}$

$rel := \emptyset$; $trans := \emptyset$; $\Delta' := \emptyset$;
forall $\langle p, \gamma \rangle \hookrightarrow \langle p', \varepsilon \rangle \in \Delta$ do $trans := trans \cup \{(p, [\gamma, G(p)], p')\}$;
while $trans \neq \emptyset$ do
  pop $t = (p, [\gamma, b], p')$ from $trans$;
  if $t \notin rel$ then
    $rel := rel \cup \{t\}$;
    forall $\langle p_1, \gamma_1 \rangle \hookrightarrow \langle p, \gamma \rangle \in \Delta$ do $trans := trans \cup \{(p_1, [\gamma_1, b \vee G(p_1)], p')\}$;
    forall $\langle p_1, \gamma_1 \rangle \xrightarrow{b'} \langle p, \gamma \rangle \in \Delta'$ do $trans := trans \cup \{(p_1, [\gamma_1, b \vee b'], p')\}$;
    forall $\langle p_1, \gamma_1 \rangle \hookrightarrow \langle p, \gamma\gamma_2 \rangle \in \Delta$ do
      $\Delta' := \Delta' \cup \{\langle p_1, \gamma_1 \rangle \xrightarrow{b \vee G(p_1)} \langle p', \gamma_2 \rangle\}$;
      forall $(p', [\gamma_2, b'], p'') \in rel$ do
        $trans := trans \cup \{(p_1, [\gamma_1, b \vee b' \vee G(p_1)], p'')\}$; % end of part 1
$R := \emptyset$; $E := \emptyset$; % beginning of part 2
forall $\langle p, \gamma \rangle \hookrightarrow \langle p', \gamma' \rangle \in \Delta$ do $E := E \cup \{((p, \gamma), G(p), (p', \gamma'))\}$;
forall $\langle p, \gamma \rangle \xrightarrow{b} \langle p', \gamma' \rangle \in \Delta'$ do $E := E \cup \{((p, \gamma), b, (p', \gamma'))\}$;
forall $\langle p, \gamma \rangle \hookrightarrow \langle p', \gamma'\gamma'' \rangle \in \Delta$ do $E := E \cup \{((p, \gamma), G(p), (p', \gamma'))\}$;
find all strongly connected components in $\mathcal{G} = ((P \times \Gamma), E)$;
forall components C do
  if C has a 1-edge then $R := R \cup C$;
return R

Theorem

Theorem

Let $\mathcal{BP} = (P, \Gamma, \Delta, G)$ be a Büchi pushdown system. The set of repeating heads R can be computed in $\mathcal{O}(|P|^2 \cdot |\Delta|)$ time and $\mathcal{O}(|P| \cdot |\Delta|)$ space.

The first part is similar to the algorithm computing $\mathcal{A}_{pre^*}$. The size of $\mathcal{G}$ is in $\mathcal{O}(|P| \cdot |\Delta|)$. Determining the strongly connected components takes linear time in the size of the graph *[Tarjan1972]*. The same holds for searching each component for an internal 1-edge.

Theorem

Theorem

Let $\mathcal{P}$ be a pushdown system and φ be an LTL formula. The global model checking problem can be solved in $\mathcal{O}(|\mathcal{P}|^3 \cdot |\mathcal{B}|^3)$ time and $\mathcal{O}(|\mathcal{P}|^2 \cdot |\mathcal{B}|^2)$ space, where $\mathcal{B}$ is a Büchi automaton corresponding to $\neg\varphi$.

Coming next week

Partial order reduction

- When can a state/transition be safely removed from a Kripke structure?
- What is a stuttering principle?
- Can we effectively compute the reduction?