Verification of temporal logics on infinite-state systems

Day 2, Lecture 1
Symbolic computation of reachability

Stéphane Demri and Valentin Goranko

ESSLLI 2007, Dublin, July 2007
Predecessors and successors

Given a transition system $\mathcal{T} = (S, R)$, $s \in S$ and $X \subseteq S$, denote:

- $R(s) := \{ t \in S \mid sRt \}$, $R(X) := \{ t \in S \mid \exists x \in X \ xRt \}$,
- $\text{pre}(X) = \langle R \rangle X := \{ t \in S \mid \exists x \in X \ tRx \} = R^{-1}(X)$: the set of immediate predecessors of $X$.
- $[R]X := \langle R \rangle \overline{X} = \{ t \in S \mid R(t) \subseteq X \}$.

Likewise, $\text{post}(X) = R(X)$: the set of immediate successors of $X$.

More notation: $R^0 := \text{Id}$; $R^{i+1} := R^i \circ R$; $R^{-i} := (R^{-1})^i$.

The set of all states from which $X$ can be reached within $n$ steps:

$$\text{pre}^n(X) := \bigcup_{i=0}^{n} \langle R^i \rangle X.$$ 

The set of all states which can be reached from $X$ within $n$ steps:

$$\text{post}^n(X) := \bigcup_{i=0}^{n} \langle R^{-i} \rangle X.$$
Transitive closure

Note that:

\[ X = \text{pre}^0(X) \subseteq \text{pre}^1(X) \subseteq \ldots; \quad X = \text{post}^0(X) \subseteq \text{post}^1(X) \subseteq \ldots \]

The reflexive and transitive closure of \( R \):

\[ R^* := \bigcup_{n=0}^{\infty} R^n. \]

The set of all predecessors of \( X \):

\[ \text{pre}^*(X) = \langle R^* \rangle X = \bigcup_{n=0}^{\infty} \text{pre}^n(X). \]

Likewise, the set of all successors of \( X \):

\[ \text{post}^*(X) = \bigcup_{n=0}^{\infty} \text{post}^n(X). \]
Reachability problems

Fix a transition system $\mathcal{T} = (S, R)$. Consider sets: $I \subseteq S$ of initial states, and $F \subseteq S$ of target states (or, bad states) in $\mathcal{T}$.

- **State-to-state reachability:**
  Is a state $q$ reachable from a state $p$ by a path in $\mathcal{T}$?

- **State-to-set reachability:**
  Backward: Is (a state in) the set $F$ reachable from a state $p$?
  Forwards: Is a state $p$ reachable from (a state in) the set $I$?

- **(Plain existential) Set-to-set reachability:** Is (a state in) the set $F$ reachable by some path in $\mathcal{T}$ starting from a state in $I$?

- **Recurrent (existential) (set-to-set) reachability:**
  Is (a state in) the set $F$ reachable infinitely often by some path in $\mathcal{T}$ starting from a state in $I$?

- **Universal (recurrent) reachability:**
  Does every path in $\mathcal{T}$ starting from $I$ reach (infinitely often) a state in $I$?

- **Constrained reachability:**
  Is the set $F$ reachable by some path in $\mathcal{T}$ starting from a state in $I$ and staying within a ‘safe’ region $C$?
Forward and backward reachability queries

- **Forward reachability query**: given $I$ compute the set $\text{post}^*(I)$ of all states in $\mathcal{T}$ reachable by a path from a state in $I$.

  In finite transition systems, $\text{post}^*(I) = \text{post}^n(I)$ for a large enough $n$, such that $\text{post}^{n+1}(I) = \text{post}^n(I)$.

  In infinite transition systems, the chain $I = \text{post}^0(I) \subseteq \text{post}^1(I) \subseteq \text{post}^2(I) \subseteq \ldots$ may never stabilize.

- **Backward reachability**: given $F$ compute the set $\text{pre}^*(F)$ of all states in $\mathcal{T}$ from which a state in $F$ is reachable.

Now: $F$ is reachable from $I$

iff $\text{post}^*(I) \cap F \neq \emptyset$ (forward reachability analysis)

iff $\text{pre}^*(F) \cap I \neq \emptyset$ (backward reachability analysis).
Deciding reachability

On finite systems all reachability problems are easily decidable. E.g., state-to-state reachability can be decided by breadth-first search; likewise for plain reachability from a set to a set.

Exercise: find simple algorithms deciding the various reachability problems in a finite transition system. Estimate their complexity and try to argue their optimality. Then consult the literature.

On infinite systems deciding reachability is a non-trivial problem.

For instance, even the plain reachability problem is undecidable for transition graphs of Turing machines given as input, by reduction from the Halting problem.

Moreover, the plain reachability problem is undecidable for transition graph of the universal Turing machine.

Central problem in verification of infinite state systems is to identify important cases of decidable reachability and develop efficient algorithms computing reachability queries.
A ‘simple’ universal reachability problem

Consider the Presburger transition system $C$:

$$(\exists y (x = y + y) \land 2x' = x) \lor (\neg \exists y (x = y + y) \land x' = 3x + 1)$$

Is it true that

$$C \models x = 1 \to \forall F(x = 1)?$$
Pushdown systems

Pushdown system:

\[ \mathcal{P} = (P, \Gamma, \Delta), \]

where:

- \( P \) is a finite set of control states (locations);
- \( \Gamma \) is stack alphabet;
- \( \Delta \subseteq P \times \Gamma \times \Gamma^* \times P \) is transition relation.

Configuration of \( \mathcal{P} \) is a word in \( P\Gamma^* \):

\((\langle \text{current control state} \rangle \ p, \langle \text{stack content} \rangle \ w)\).

Sometimes a restriction is imposed on the transitions, to only add (push) or remove (pop) at most one symbol at the top of the stack.

Also, sometimes a special stack bottom symbol \( \bot \in \Gamma \) is assumed, which can never be popped.

Pushdown automaton: a pushdown system with input alphabet, designated initial state and initial stack symbol.

Pushdown automata accept precisely the context-free languages.
Pushdown graphs

Given a pushdown system $\mathcal{P} = (P, \Gamma, \Delta)$, its configuration (transition) graph is

$$G_{\mathcal{P}} = (V, \leftarrow),$$

where $V$ is the set of all configurations of $\mathcal{P}$ and $\leftarrow$ is the transition relation between configurations defined by:

$$(p, \gamma w) \leftarrow (p', uw) \text{ iff } (p, \gamma, p', u) \in \Delta;$$

**Pushdown graph**: the configuration graph of a pushdown system.

Labelled transition systems can be modelled, too, as labelled pushdown graphs, by adding an input alphabet $A$ with the labels of the transition relations, and defining the configuration graph as

$$G_{\mathcal{P}} = (V, \{\leftarrow_a\}_{a \in A}),$$

where each $E_a$ is a transition relation defined by:

$$(p, \gamma w) \leftarrow_a (p', uw) \text{ iff } (p, \gamma, a, p', u) \in \Delta.$$
Consider the labelled pushdown system $\mathcal{P} = (Q, \Gamma, A, \Delta)$, where

- $Q = \{q_0, q_1\}$
- $\Gamma = \{z\}$
- $A = \{a, b, c, d\}$
- $\Delta$ consists of the transitions 
  \[(q_0, z) \xrightarrow{a} (q_0, zz), (q_0, z) \xrightarrow{b} (q_0, \varepsilon), (q_0, z) \xrightarrow{c} (q_1, z), (q_1, z) \xrightarrow{d} (q_1, \varepsilon)\]

The transition graph generated by $(q_0, z)$ is depicted below:
Muller and Schupp’s characterization of pushdown graphs

Consider a transition graph \( G = (V, E, v_0) \) with an designated initial vertex \( v_0 \).

\( V_n \): the set of vertices in \( G \) reachable from \( v_0 \) by a non-oriented path of length \( \leq n \).

\( G_n \): the subgraph of \( G \) induced by the set of vertices \( V \setminus V_n \). The vertices in \( V_{n+1} \setminus V_n \): boundary vertices of \( G_n \).

The ends of the graph \( G \): the connected (via non-oriented paths) components of \( G_n \), for \( n \leq 0 \).

**Theorem** [Muller and Schupp, 1985]:
A graph of bounded in- and out- degrees is a pushdown graph iff the number of distinct isomorphism types of its ends is finite.

**Example:** the complete binary tree \( T_2 \).
**Non-example:** the infinite grid \( \mathbb{N} \times \mathbb{N} \).

**Exercise:** describe the pushdown trees.
Decidability of MSO of pushdown graphs

**Theorem [Muller and Schupp, 1985]:** Every pushdown graph is MSO-interprettable in the infinite binary tree $\mathcal{T}_2$.

**Corollary:** Every pushdown graph has a decidable MSO-theory.

However, the complexity of MSO over $\mathcal{T}_2$ is non-elementary.

On the other hand, reachability queries are embedded into a small fragment of MSO. Thus, one can hope for a better complexity of computing reachability.
Symbolic computation of reachability queries in pushdown systems

The idea [Büchi’64; Bouajjani, Esparza, Maler 97; Finkel, Willems, Wolper ’97]: introduce special automata on the stack alphabet that recognize sets of configurations of pushdown systems, thus providing *symbolic representation* of such configuration sets as regular languages.

These are finite automata reading words from the stack alphabet of the pushdown system and simulating the transitions in it.

Then, we show that if a set of target configurations $C$ is regular, then the set $\text{pre}^*(C)$ of all configurations from which $C$ is reachable is regular, too, and is moreover effectively computable from $C$, by constructing a multi-automaton recognizing $\text{pre}^*(C)$ from one recognizing $C$.

By modifying the construction, we can also show that if a set of initial configurations $I$ is regular, then the set of all reachable configurations $\text{post}^*(I)$ is regular, too.
**$P$-automata**

Let $\mathcal{P} = (P, \Gamma, \Delta)$ be a pushdown system with $P = \{p_1, \ldots, p_m\}$.

$\mathcal{P}$-automaton: 

$$A = (\Gamma, Q, \delta, I, F),$$

where:

- $Q$ is a finite set of control states;
- $P = \{p_1, \ldots, p_m\} \subseteq Q$ is the set of initial states;
- $F \subseteq Q$ is a set of final states;
- $\delta \subseteq Q \times \Gamma \times Q$ is a transition relation.

Derivation relation in $A$: $\xhookrightarrow{} \subseteq Q \times \Gamma^* \times Q$ is the smallest relation such that:

- $\triangleright q \xhookrightarrow{\varepsilon} q$ for each $q \in Q$;
- $\triangleright$ if $(q, \gamma, q') \in \delta$ then $q \xhookrightarrow{\gamma} q'$;
- $\triangleright$ if $q \xhookrightarrow{w} q'$ and $q' \xhookrightarrow{\gamma} q''$ then $q \xhookrightarrow{w\gamma} q''$.

$A$ accepts a configuration $(p_i, w)$ iff $p_i \xhookrightarrow{w} q$ for some $q \in F$.

A set of configurations in $\mathcal{P}$ is regular if it is the set of all configurations $Conf(A)$ accepted by some $\mathcal{P}$-automaton $A$. 
Constructing the reachability automaton for $pre^*(C)$

The idea: given a pushdown system $\mathcal{P} = (P, \Gamma, \delta)$ and a $\mathcal{P}$-automaton $A$ recognizing a set of target configurations $C$, we build a sequence of $\mathcal{P}$-automata $A = A_0, A_1, \ldots, A_n, \ldots$ on the same set of states but with more transitions, accepting the configurations respectively in $pre^0(C), pre^1(C), \ldots, pre^n(C), \ldots$.

Simultaneously with the construction, we will prove by induction on $n$ that $A_n$ accepts precisely the configurations in $pre^n(C)$.

This sequence is bound to stabilize at some $n$, and then, by the claim above, the resulting $\mathcal{P}$-automaton $A_{pre^*}$ accepts precisely $pre^*(C)$.

For technical reasons, assume $A$ has no transitions leading to an initial state.
The inductive construction of $A_{pre^*}$

For $n = 0$: $A_0 = A$ and the claim is immediate.

Suppose $A_i$ is constructed and satisfies the inductive claim. Let $\rightarrow_i$ denote the transition relation in $A_i$.

Note that: if $(p_k, w) \in \text{pre}^i(C)$ and $(p_j, \gamma) \rightarrow (p_k, w) \in \delta$, then $(p_j, \gamma) \in \text{pre}^{i+1}(C)$.

That suggests the inductive step of the construction:

if $p_k \xrightarrow{w}_i q$ and $(p_j, \gamma) \rightarrow (p_k, w) \in \delta$,
then a transition $p_j \xrightarrow{\gamma}_i q$ is added to $A_{i+1}$.

The idea: if a configuration $(p_k, ww')$ is accepted by $A_i$ because $p_k \xrightarrow{w}_i q \xrightarrow{w'}_i q' \in F$, then the new transition $p_j \xrightarrow{\gamma}_i q$ enables $A_{i+1}$ to accept $(p_j, \gamma w')$ because $p_j \xrightarrow{\gamma}_{i+1} q \xrightarrow{w'}_i q' \in F$.

Exercise: now prove the inductive claim for $A_{i+1}$.

Complexity: in (low) PSPACE in the sizes of both the PDS $\mathcal{P}$ and the $\mathcal{P}$-automaton.
Constructing the reachability automaton for $\text{pre}^*(C) : \text{example}$

Run animation.
Addendum: prefix recognizable graphs

$\Sigma$: a fixed finite alphabet.

Prefix-rewriting system on a language $V \subseteq \Sigma^*$: a finite set of rewrite rules of the type $U_1 \rightarrow_a U_2$ where $U_1$, $U_2$ are regular subsets of $V$.

There is a transition of type $a$ from a word $v_1$ to a word $v_2$ iff there is a rule $U_1 \rightarrow_a U_2$ and words $v \in V$, $u_1 \in U_1$, $u_2 \in U_2$ such that $v_1 = u_1 v$ and $v_2 = u_2 v$.

A (transition) graph $G = (V, \{E_a\}_{a \in A})$ is prefix-recognizable over $\Sigma$ if $V \subseteq \Sigma^*$ is regular and the transitions in $G$ are determined by some prefix-rewriting system on $V$. Thus, for each $a \in A$,

$$E_a = \bigcup_{i=1}^{m} (U_i^1 \times U_i^2) V_i = \bigcup_{i=1}^{m} \{(u_1 v, u_2 v) \mid v \in V_i, u_1 \in U_i^1, u_2 \in U_i^2\}$$

for some regular languages $V_i, U_i^1, U_i^2, i = 1, \ldots, m.$
Prefix recognizable graphs: examples

Every pushdown graph is prefix recognizable: every transition can be written as a rewriting rule on the configuration graph.

In fact, the pushdown graphs are (up to isomorphism) precisely the prefix-recognizable graphs with finite in- and out-degrees.

But, there are prefix-recognizable graphs in with infinite out-degree.

The reachability analysis and decidability results for pushdown systems discussed here can be extended to prefix-recognizable graphs.
Other important cases of decidable reachability in infinite state systems

- **Well quasi-orderings.** Important example: configuration graphs of Petri nets. ([Mayr'81]).
- **Ground tree-rewriting systems.** ([Dauchet, Tison, ’90]: these have decidable MSO).
- **Prefix-recognizable graphs** ([Caucal’96]: these have decidable MSO).
- **Rational graphs with length-preserving or length-monotone transition relations.** ([Morvan’00])
- **More cases and results of decidable reachability will be discussed further in the course.**