Model Checking Regular Safety Properties

Lecture #8 of Model Checking

Joost-Pieter Katoen

Lehrstuhl 2: Software Modeling & Verification

E-mail: `katoen@cs.rwth-aachen.de`

November 12, 2008

Overview Lecture #8

$\Rightarrow$ Regular Safety Properties

- Verifying Regular Safety Properties
  - Reduction to Invariant Checking
  - Proof of Correctness
  - The Algorithm

Safety properties

- LT property P_{safe} over *AP* is a *safety property* if
  - for all $\sigma \notin P_{safe}$ there exists a finite prefix $\widehat{\sigma}$ of σ such that:

$$P_{safe} \cap \left\{ \sigma' \in \left(2^{AP}\right)^{\omega} \mid \widehat{\sigma} \in \mathit{pref}(\sigma) \right\} \; = \; \varnothing$$

- The set *bp* of *bad prefixes* for P_{safe}:

$$\mathit{bp}(P_{safe}) \; = \; \left(2^{AP}\right)^{*} \setminus \mathit{pref}(P_{safe})$$

- The set *mbp* of *minimal bad prefixes* for P_{safe}:

$$\mathit{mbp}(P_{safe}) \; = \; \{ \, \sigma \in \left(2^{AP}\right)^{*} \mid \mathit{pref}(\sigma) \, \cap \, \mathit{bp}(P_{safe}) = \{ \, \sigma \, \} \, \}$$

Regular safety properties

- Definition:

  Safety property P_{safe} is regular if $bp(P_{safe})$ is a regular language

- Or, equivalently:

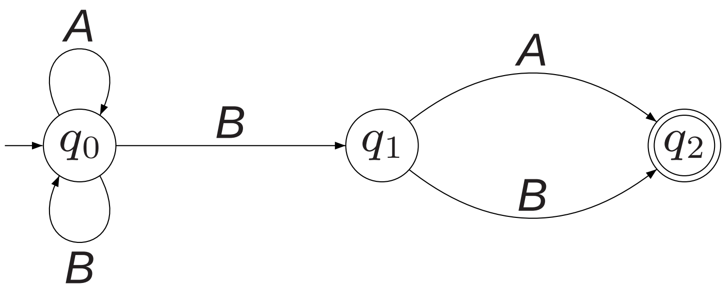
  Safety property P_{safe} is regular if there exists

  a finite automaton over the alphabet 2^{AP} recognizing $bp(P_{safe})$

Refresh your memory: Finite automata

A *nondeterministic finite automaton* (NFA) $\mathcal{A}$ is a tuple $(Q, \Sigma, \delta, Q_0, F)$ where:

- Q is a finite set of states
- Σ is an alphabet
- $\delta : Q \times \Sigma \to 2^Q$ is a transition function
- $Q_0 \subseteq Q$ a set of initial states
- $F \subseteq Q$ is a set of accept (or: final) states



Language of an automaton

- NFA $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$ and word $w = A_1 \ldots A_n \in \Sigma^*$
- A *run* for w in $\mathcal{A}$ is a finite sequence $q_0\, q_1\, \ldots\, q_n$ such that:
  - $q_0 \in Q_0$ and $q_i \xrightarrow{A_{i+1}} q_{i+1}$ for all $0 \leqslant i < n$
- Run $q_0\, q_1\, \ldots\, q_n$ is *accepting* if $q_n \in F$
- $w \in \Sigma^*$ is *accepted* by $\mathcal{A}$ if there exists an accepting run for w
- The *accepted language* of $\mathcal{A}$:

$$\mathcal{L}(\mathcal{A}) = \left\{ w \in \Sigma^* \mid \text{ there exists an accepting run for } w \text{ in } \mathcal{A} \right\}$$

- NFA $\mathcal{A}$ and $\mathcal{A}'$ are *equivalent* if $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$

Facts about finite automata

- They are as expressive as regular languages
- They are closed under $\cap$ and complementation
  - NFA $\mathcal{A} \otimes B$ (= cross product) accepts $\mathcal{L}(A) \cap \mathcal{L}(B)$
  - Total DFA $\overline{\mathcal{A}}$ (= swap all accept and normal states) accepts $\overline{\mathcal{L}(A)} = \Sigma^* \setminus \mathcal{L}(\mathcal{A})$
- They are closed under determinization (= removal of choice)
  - although at an exponential cost.....
- $\mathcal{L}(\mathcal{A}) = \varnothing$? = check for a reachable accept state in $\mathcal{A}$
  - this can be done using a simple depth-first search
- For regular language $\mathcal{L}$ there is a unique minimal DFA accepting $\mathcal{L}$

Regular safety properties

- Definition:

  Safety property P_{safe} is regular if $bp(P_{safe})$ is a regular language

- Or, equivalently:

  Safety property P_{safe} is regular if there exists
  an NFA $\mathcal{A}$ over the alphabet 2^{AP} with $\mathcal{L}(\mathcal{A}) = bp(P_{safe})$

Example regular safety properties

- Every invariant (over *AP*) is a regular safety property
  - traces of bad prefixes are of the form $\Phi^*(\neg\Phi)\text{true}^*$
  - where Φ is the invariant condition
  - symbol Φ stands for any $A \subseteq$ *AP* with $A \models \Phi$

- An example regular property which is not an invariant:

  "a red light is immediately preceded by a yellow light"

- An example non-regular safety property:

  "The number of inserted coins is at least the number of dispensed drinks"

Details

Property

Safety property P_{safe} is regular

if and only if

mbp(P_{safe}) is a regular language

Property

Safety property P_{safe} is regular

if and only if

mbp(P_{safe}) is a regular language

How to check whether a finite transition system satisfies a regular safety property?

Peterson's banking system

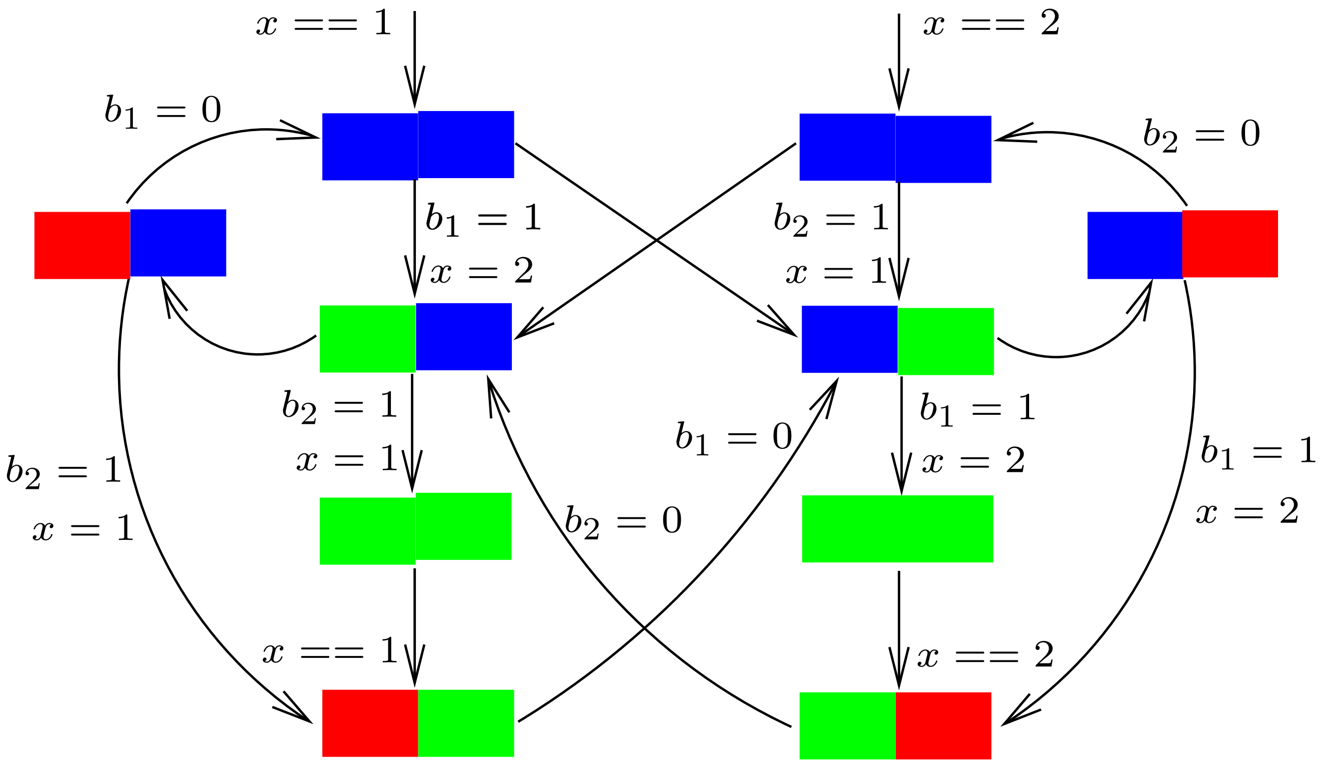
Person Left behaves as follows:

	while true $\{$
	$\ldots\ldots$
rq :	$b_1, x =$ true, 2;
wt :	$\textbf{wait until}(x == 1 \,\|\|\, \neg\, b_2)\ \{$
cs :	$\ldots$ @account$_L \ldots\}$
	$b_1 =$ false;
	$\ldots\ldots$
	$\}$

Person Right behaves as follows:

	while true $\{$
	$\ldots\ldots$
rq :	$b_2, x =$ true, 1;
wt :	$\textbf{wait until}(x == 2 \,\|\|\, \neg\, b_1)\ \{$
cs :	$\ldots$ @account$_R \ldots\}$
	$b_2 =$ false;
	$\ldots\ldots$
	$\}$

Is the banking system safe?



Can we guarantee that only one person at a time has access to the bank account?

"always $\neg$ (@account$_L$ $\wedge$ @account$_R$)"

Is the banking system safe?

- Safe = at most one person may have access to the account
- Unsafe: two have access to the account simultaneously
  - unsafe behaviour can be characterized by bad prefix
  - alternatively (in this case) by the finite automaton:

- Checking safety: $\mathit{Traces}(\mathit{TS}_{Pet}) \cap \mathit{BadPref}(P_{safe}) = \varnothing$?
  - intersection, complementation and emptiness of languages . . .

Problem statement

Let

- P_{safe} be a *regular* safety property over *AP*
- $\mathcal{A}$ be an NFA recognizing the bad prefixes of P_{safe}
  - assume that $\varepsilon \notin \mathcal{L}(\mathcal{A})$
  - $\Rightarrow$ otherwise all finite words over 2^{AP} are bad prefixes and $P_{safe} = \varnothing$
- *TS* be a *finite* transition system (over *AP*) without terminal states

How to establish whether $TS \models P_{safe}$?

Basic idea of the algorithm

$$TS \models P_{safe} \quad \text{if and only if} \quad \mathit{Traces}_{fin}(TS) \cap bp(P_{safe}) = \varnothing$$

$$\text{if and only if} \quad \mathit{Traces}_{fin}(TS) \cap \mathcal{L}(\mathcal{A}) = \varnothing$$

$$\text{if and only if} \quad TS \otimes \mathcal{A} \models \text{“always”}\ \Phi$$

But . . . . . . this amounts to invariant checking on $TS \otimes \mathcal{A}$

$\Rightarrow$ *checking regular safety properties can be done by depth-first search!*

Synchronous product (revisited)

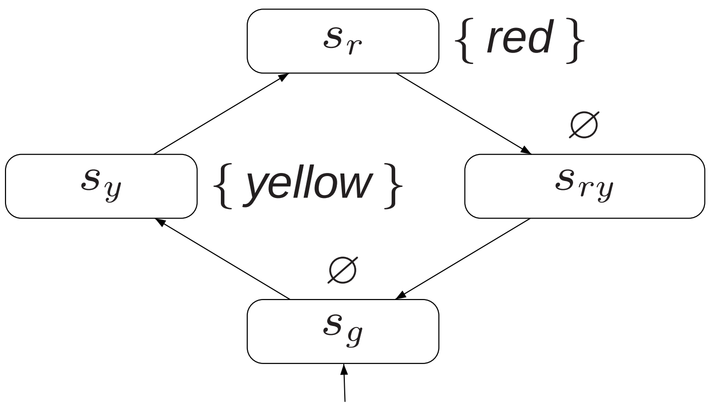
For transition system $TS = (S, Act, \rightarrow, I, AP, L)$ without terminal states and $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$ an NFA with $\Sigma = 2^{AP}$ and $Q_0 \cap F = \varnothing$, let:

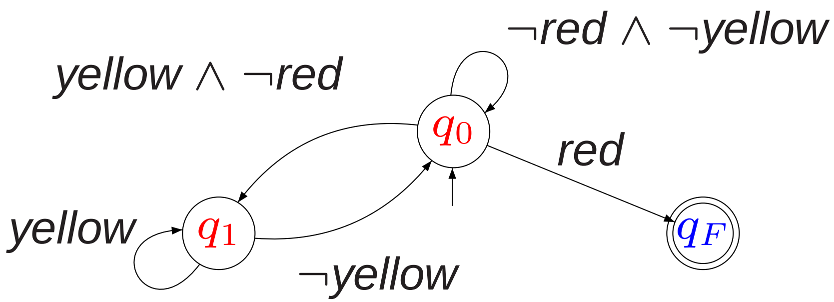
$$TS \otimes \mathcal{A} \ = \ (S', Act, \rightarrow', I', AP', L') \qquad \text{where}$$

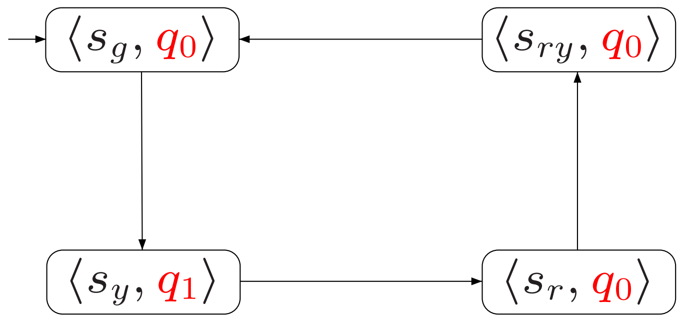
- $S' = S \times Q$, $AP' = Q$ and $L'(\langle s, q \rangle) \ = \ \{ q \}$
- $\rightarrow'$ is the smallest relation defined by: $\dfrac{s \xrightarrow{\alpha} t \ \wedge \ q \xrightarrow{L(t)} p}{\langle s, q \rangle \xrightarrow{\alpha}{}' \langle t, p \rangle}$
- $I' = \{ \langle s_0, q \rangle \mid s_0 \in I \ \wedge \ \exists q_0 \in Q_0.\ q_0 \xrightarrow{L(s_0)} q \}$

without loss of generality it may be assumed that $TS \otimes \mathcal{A}$ has no terminal states

Example product







A note on terminal states

- Although *TS* has no terminal state *TS* $\otimes \mathcal{A}$ may have one
- This can only occur if $\delta(q, A) = \varnothing$ for some $A \subseteq$ *AP*
- Let NFA $\mathcal{A}$ with some reachable state q with $\delta(q, A) = \varnothing$
- Obtain an equivalent NFA $\mathcal{A}'$ as follows:
  - introduce new state $q_{trap} \notin Q$
  - if $\delta(q, A) = \varnothing$ let $\delta'(q, A) = \{ q_{trap} \}$
  - set $\delta'(q_{trap}, A) = \{ q_{trap} \}$ for all $A \subseteq$ *AP*
  - keep all other transitions that are present in $\mathcal{A}$

$\Rightarrow$ Assume that *TS* $\otimes \mathcal{A}$ has no terminal states

Verification of regular safety properties

Let *TS* over *AP*, NFA $\mathcal{A}$, and P a regular safety property with $\mathcal{L}(\mathcal{A}) = bp(P)$

The following statements are equivalent:

(a) $TS \models P$

(b) $Traces_{fin}(TS) \cap \mathcal{L}(\mathcal{A}) = \varnothing$

(c) $TS \otimes \mathcal{A} \models P_{inv(A)} = \bigwedge_{q \in F} \neg q$

Proof

Counterexamples

For each initial path fragment $\langle s_0, q_1 \rangle \ldots \langle s_n, q_{n+1} \rangle$ of *TS* $\otimes \mathcal{A}$:

$$q_1, \ldots, q_n \not\in F \text{ and } q_{n+1} \in F \quad \Rightarrow \quad \underbrace{\textit{trace}(s_0\, s_1 \ldots s_n)}_{\text{bad prefix for } P_{safe}} \in \mathcal{L}(\mathcal{A})$$

Verification algorithm

Input: finite transition system *TS* and regular safety property P_{safe}
Output: true if *TS* $\models P_{safe}$. Otherwise false plus a counterexample for P_{safe}.

Let NFA $\mathcal{A}$ (with accept states F) be such that $\mathcal{L}(\mathcal{A}) = bp(P_{safe})$;
Construct the product transition system *TS* $\otimes\, \mathcal{A}$;
Check the invariant $P_{inv(\mathcal{A})}$ with proposition $\neg F = \bigwedge_{q \in F} \neg q$ on *TS* $\otimes\, \mathcal{A}$

if *TS* $\otimes\, \mathcal{A} \models P_{inv(\mathcal{A})}$ **then**
  return true
else
  Determine initial path fragment $\langle s_0, q_1 \rangle \ldots \langle s_n, q_{n+1} \rangle$ of *TS* $\otimes\, \mathcal{A}$ with $q_{n+1} \in F$
  return (false, $s_0\, s_1 \ldots s_n$)
fi

Example

Time complexity

The time and space complexity of checking $TS \models P_{safe}$ is in:

$$\mathcal{O}(|TS| \cdot |\mathcal{A}|)$$

where $\mathcal{A}$ is an NFA with $\mathcal{L}(\mathcal{A}) = mbp(P_{safe})$

The size of NFA $\mathcal{A}$, denoted $|\mathcal{A}|$, is the number of states and transitions in $\mathcal{A}$:

$$|\mathcal{A}| \;=\; |Q| + \sum_{q \in Q} \sum_{A \in \Sigma} |\,\delta(q, A)\,|$$