Unit-4: Regular properties

B. Srivathsan

Chennai Mathematical Institute

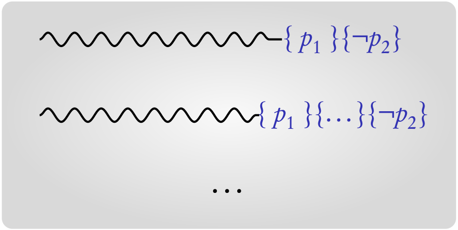
NPTEL-course

July - November 2015

Module 4:
Safety properties described by automata

AP-INF = set of **infinite words** over *PowerSet*(**AP**)

P: a property over AP



Bad-Prefixes

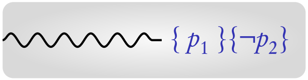
P is a safety property if there **exists** a set Bad-Prefixes such that

P is the set of **all words** that **do not start** with a Bad-Prefix

Property 1: if p_1 is true, then p_2 should be true in the next step

$\rightsquigarrow \{p_1\}\{\neg p_2\}$ BadPrefixes

Property 1: if p_1 is true, then p_2 should be true in the next step



BadPrefixes

$$\Sigma = \{ \{ \}, \{ p_1 \}, \{ p_2 \}, \{ p_1, p_2 \} \}$$

Property 1: if p_1 is true, then p_2 should be true in the next step

$\rightsquigarrow \{\, p_1 \,\}\{\neg p_2\}$

BadPrefixes

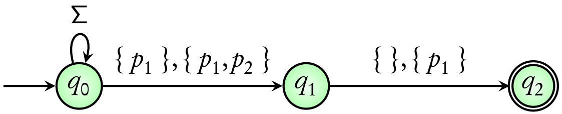
$$\Sigma \;=\; \{\,\{\,\},\,\{\,p_1\,\},\,\{\,p_2\,\},\,\{\,p_1,p_2\,\}\,\}$$

$$\begin{tikzcd}
{} \arrow[r] & q_0 \arrow[loop above, "\Sigma"] \arrow[r, "\{\,p_1\,\}{,}\{\,p_1{,}p_2\,\}"] & q_1 \arrow[r, "\{\,\}{,}\{\,p_1\,\}"] & q_2
\end{tikzcd}$$

Property 1: if p_1 is true, then p_2 should be true in the next step

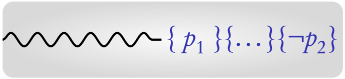
$\rightsquigarrow \{ p_1 \}\{\neg p_2\}$ BadPrefixes

$$\Sigma = \{ \{ \}, \{ p_1 \}, \{ p_2 \}, \{ p_1, p_2 \} \}$$



This **BadPrefixes** set is a **regular language**

Property 2: if p_1 is true, then p_2 should be true in the next to next step



"BadPrefixes"

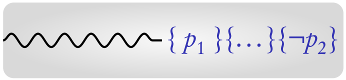
Property 2: if p_1 is true, then p_2 should be true in the next to next step

$$\rightsquigarrow \{\, p_1 \,\}\{\ldots\}\{\neg p_2\}$$

"BadPrefixes"

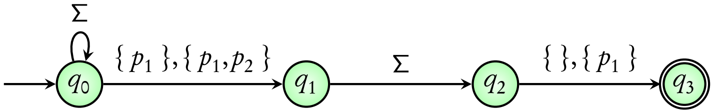
$$\Sigma \;=\; \{\, \{\,\},\ \{\, p_1 \,\},\ \{\, p_2 \,\},\ \{\, p_1, p_2 \,\} \,\}$$

Property 2: if p_1 is true, then p_2 should be true in the next to next step

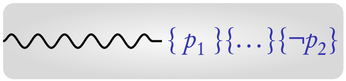


"BadPrefixes"

$$\Sigma = \{\ \{\ \},\ \{p_1\},\ \{p_2\},\ \{p_1, p_2\}\ \}$$


$$\begin{tikzcd}
\arrow[r] & q_0 \arrow[loop above, "\Sigma"] \arrow[r, "{\{p_1\},\{p_1,p_2\}}"] & q_1 \arrow[r, "\Sigma"] & q_2 \arrow[r, "{\{\ \},\{p_1\}}"] & q_3
\end{tikzcd}$$

Property 2: if p_1 is true, then p_2 should be true in the next to next step



“BadPrefixes”

$$\Sigma = \{ \{ \}, \{ p_1 \}, \{ p_2 \}, \{ p_1, p_2 \} \}$$

$$\begin{tikzcd} \arrow[r] & q_0 \arrow[loop above, "\Sigma"] \arrow[r, "\{ p_1 \},\{ p_1,p_2 \}"] & q_1 \arrow[r, "\Sigma"] & q_2 \arrow[r, "\{ \},\{ p_1 \}"] & q_3 \end{tikzcd}$$

This **BadPrefixes** set is a **regular language**

Property 3: at any point, the number of times p_1 has occured should be less than the number of times p_2 has occured

Property 3: at any point, the number of times p_1 has occured should be less than the number of times p_2 has occured

$$\Sigma = \{ \{ \}, \{ p_1 \}, \{ p_2 \}, \{ p_1, p_2 \} \}$$

Property 3: at any point, the number of times p_1 has occured should be less than the number of times p_2 has occured

$$\Sigma = \{ \{ \}, \{ p_1 \}, \{ p_2 \}, \{ p_1, p_2 \} \}$$

BadPrefixes = words where number of times p_1 occurs is more than that of p_2

Property 3: at any point, the number of times p_1 has occured should be less than the number of times p_2 has occured

$$\Sigma = \{ \{ \}, \{ p_1 \}, \{ p_2 \}, \{ p_1, p_2 \} \}$$

BadPrefixes = words where number of times p_1 occurs is more than that of p_2

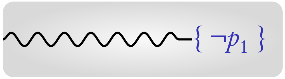
This **BadPrefixes** set is **not a regular language**

Regular safety properties

A safety property is **regular** if the associated **BadPrefixes** set is a **regular language**

Invariants are **regular safety properties**

Property: Always p_1 is true



"Bad-Prefixes"

$$\Sigma^* \{\neg p_1\}$$

BadPrefixes set for invariant properties is a **regular language**

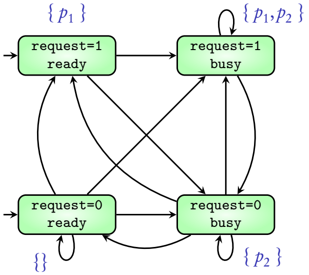
Coming next: An algorithm to model-check safety properties

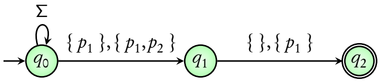
Model

Safety property

Atomic propositions AP $= \{ p_1, p_2 \}$

p_1: `request=1` p_2: `status=busy`





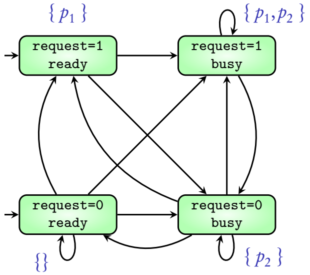
BadPrefixes

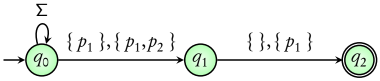
Model

Safety property

Atomic propositions AP $= \{ p_1, p_2 \}$

p_1: `request=1` p_2: `status=busy`

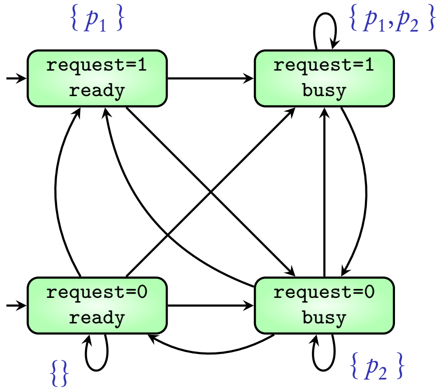


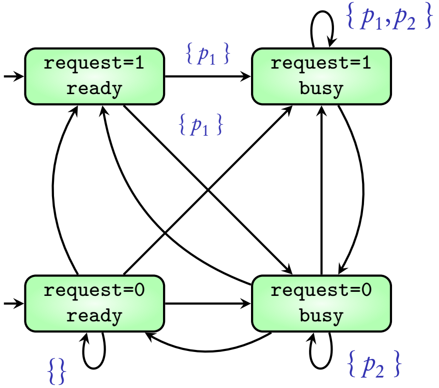


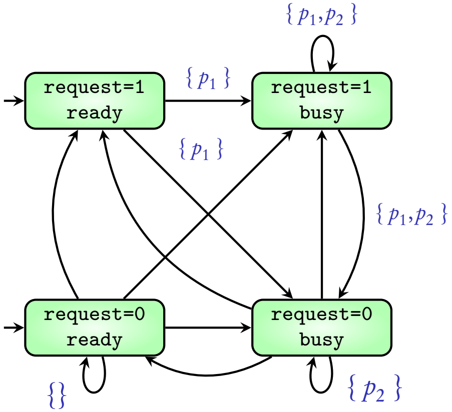
BadPrefixes

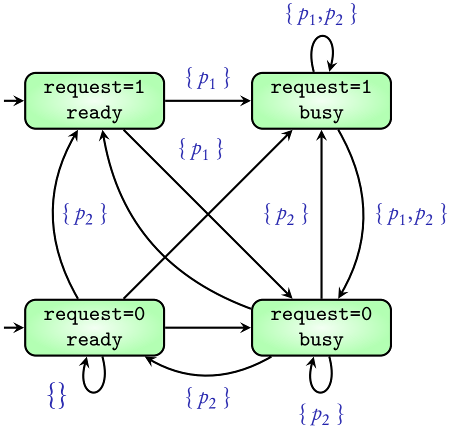
Does the model satisfy the safety property?

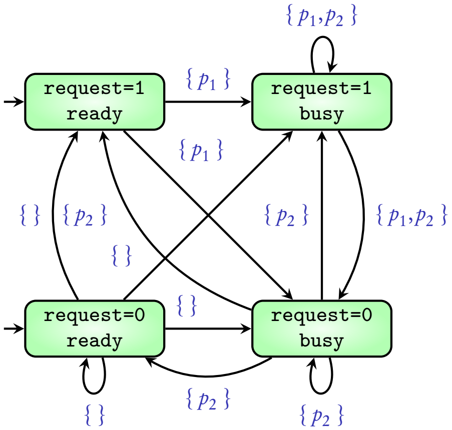
Step 1: Transition system $\longrightarrow$ automaton

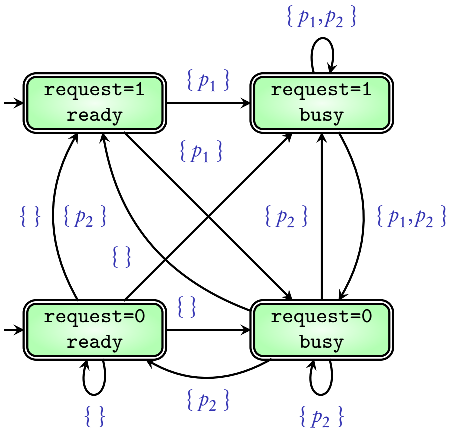

$\{ p_1 \}$
$\{ p_1, p_2 \}$
request=1
ready
request=1
busy
request=0
ready
request=0
busy
$\{\}$
$\{ p_2 \}$


request=1
ready
request=1
busy
request=0
ready
request=0
busy
$\{ p_1 \}$
$\{ p_1 \}$
$\{ p_1, p_2 \}$
$\{\}$
$\{ p_2 \}$

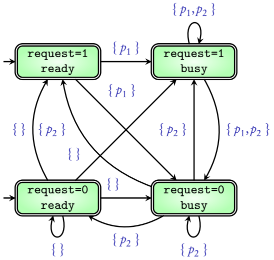

{ p_1,p_2 }
request=1
ready
{ p_1 }
request=1
busy
{ p_1 }
{ p_1,p_2 }
request=0
ready
request=0
busy
{}
{ p_2 }

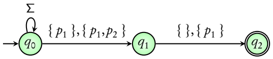

request=1
ready
request=1
busy
request=0
ready
request=0
busy
$\{p_1,p_2\}$
$\{p_1\}$
$\{p_1\}$
$\{p_2\}$
$\{p_2\}$
$\{p_1,p_2\}$
$\{\}$
$\{p_2\}$
$\{p_2\}$

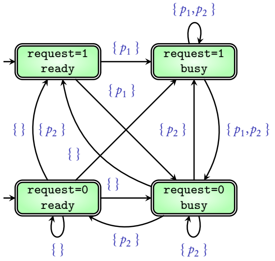

request=1
ready
request=1
busy
request=0
ready
request=0
busy
$\{ p_1 \}$
$\{ p_1, p_2 \}$
$\{ p_1 \}$
$\{\}$
$\{ p_2 \}$
$\{ p_2 \}$
$\{ p_1, p_2 \}$
$\{\}$
$\{\}$
$\{\}$
$\{ p_2 \}$
$\{ p_2 \}$


$\{p_1,p_2\}$
$\{p_1\}$
request=1 ready
request=1 busy
$\{p_1\}$
$\{\}$
$\{p_2\}$
$\{p_2\}$
$\{p_1,p_2\}$
$\{\}$
$\{\}$
request=0 ready
request=0 busy
$\{p_2\}$
$\{\}$
$\{p_2\}$

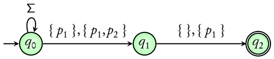
Step 2: Take a synchronous product with property automaton


request=1
ready
request=1
busy
request=0
ready
request=0
busy
$\{p_1\}$
$\{p_1, p_2\}$
$\{p_1\}$
$\{\}$
$\{p_2\}$
$\{\}$
$\{p_2\}$
$\{p_1, p_2\}$
$\{\}$
$\{\}$
$\{p_2\}$
$\{p_2\}$

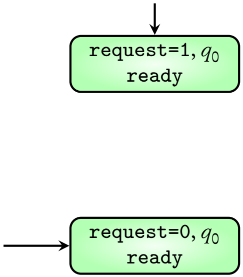

Σ
q_0
$\{p_1\}, \{p_1, p_2\}$
q_1
$\{\}, \{p_1\}$
q_2



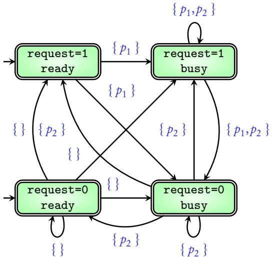
request=1 ready
request=1 busy
request=0 ready
request=0 busy
$\{p_1,p_2\}$
$\{p_1\}$
$\{p_1\}$
$\{\}$
$\{p_2\}$
$\{\}$
$\{p_2\}$
$\{p_1,p_2\}$
$\{\}$
$\{\}$
$\{p_2\}$
$\{p_2\}$



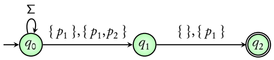
Σ
q_0
$\{p_1\},\{p_1,p_2\}$
q_1
$\{\},\{p_1\}$
q_2



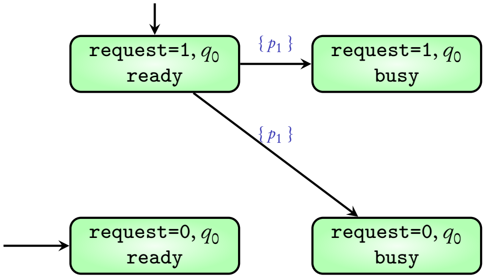
request=1, q_0 ready
request=0, q_0 ready



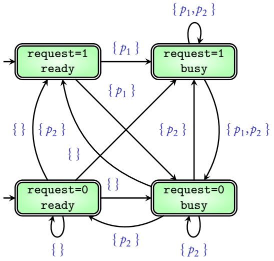
request=1 ready
request=1 busy
request=0 ready
request=0 busy
$\{p_1\}$
$\{p_1,p_2\}$
$\{p_1\}$
$\{\}$
$\{p_2\}$
$\{p_2\}$
$\{p_1,p_2\}$
$\{\}$
$\{\}$
$\{\}$
$\{p_2\}$
$\{p_2\}$

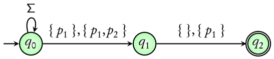


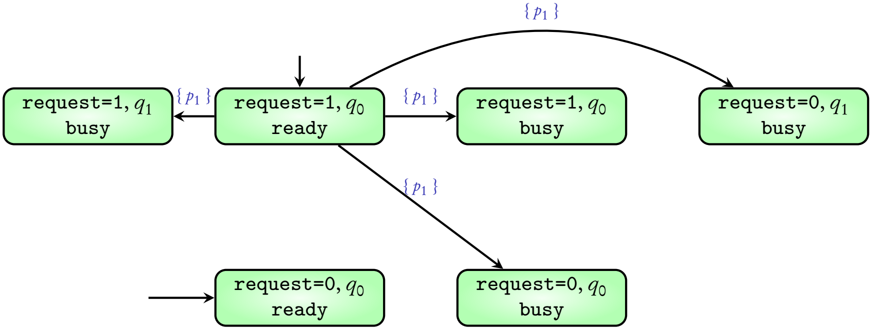
Σ
q_0
$\{p_1\},\{p_1,p_2\}$
q_1
$\{\},\{p_1\}$
q_2

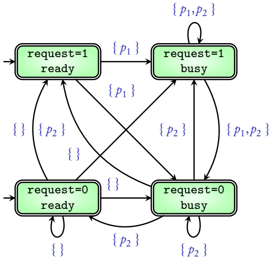


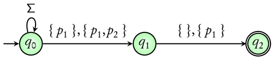
request=1, q_0 ready
$\{p_1\}$
request=1, q_0 busy
$\{p_1\}$
request=0, q_0 ready
request=0, q_0 busy

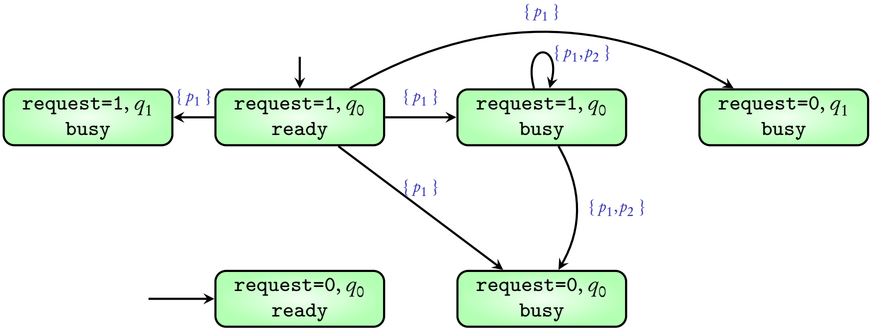

request=1
ready
request=1
busy
request=0
ready
request=0
busy
$\{p_1\}$
$\{p_1,p_2\}$
$\{p_1\}$
$\{\}$
$\{p_2\}$
$\{p_2\}$
$\{p_1,p_2\}$
$\{\}$
$\{\}$
$\{\}$
$\{p_2\}$
$\{p_2\}$

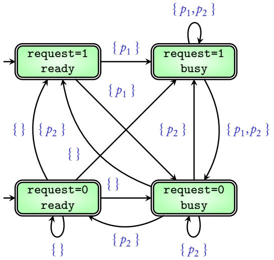

Σ
q_0
$\{p_1\},\{p_1,p_2\}$
q_1
$\{\},\{p_1\}$
q_2

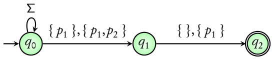

request=1, q_1
busy
request=1, q_0
ready
request=1, q_0
busy
request=0, q_1
busy
request=0, q_0
ready
request=0, q_0
busy
$\{p_1\}$
$\{p_1\}$
$\{p_1\}$
$\{p_1\}$

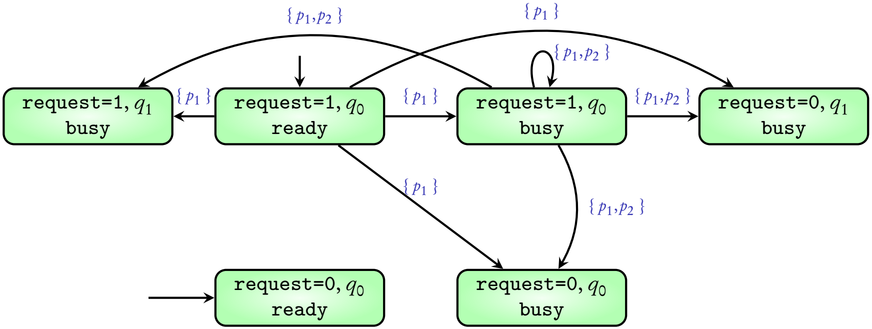

request=1 ready
request=1 busy
request=0 ready
request=0 busy
{ p1 }
{ p1,p2 }
{ p1 }
{ } { p2 }
{ p2 }
{ p1,p2 }
{ }
{ }
{ }
{ p2 }
{ p2 }

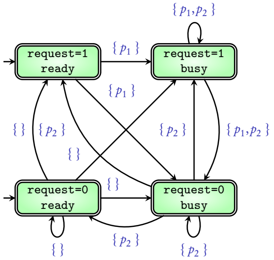

Σ
q0
{ p1 },{ p1,p2 }
q1
{ },{ p1 }
q2

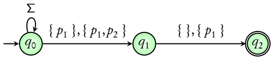

request=1, q1 busy
request=1, q0 ready
request=1, q0 busy
request=0, q1 busy
request=0, q0 ready
request=0, q0 busy
{ p1 }
{ p1,p2 }
{ p1 }
{ p1 }
{ p1 }
{ p1,p2 }

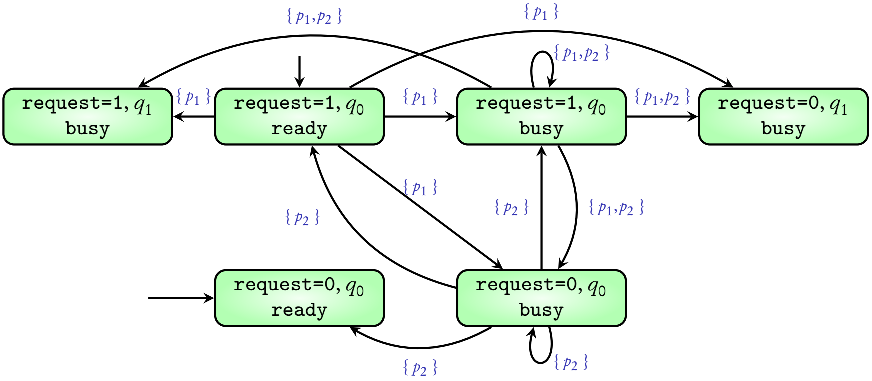

request=1 ready
request=1 busy
request=0 ready
request=0 busy
$\{p_1,p_2\}$
$\{p_1\}$
$\{p_1\}$
$\{\}$
$\{p_2\}$
$\{\}$
$\{p_2\}$
$\{p_1,p_2\}$
$\{\}$
$\{\}$
$\{p_2\}$
$\{p_2\}$

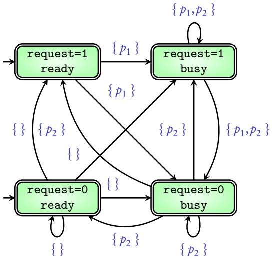

Σ
q_0
$\{p_1\},\{p_1,p_2\}$
q_1
$\{\},\{p_1\}$
q_2


request=1, q_1 busy
request=1, q_0 ready
request=1, q_0 busy
request=0, q_1 busy
request=0, q_0 ready
request=0, q_0 busy
$\{p_1,p_2\}$
$\{p_1\}$
$\{p_1,p_2\}$
$\{p_1\}$
$\{p_1\}$
$\{p_1,p_2\}$
$\{p_1\}$
$\{p_1,p_2\}$

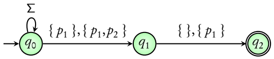

request=1 ready
request=1 busy
request=0 ready
request=0 busy
$\{p_1,p_2\}$
$\{p_1\}$
$\{p_1\}$
$\{\}$
$\{p_2\}$
$\{\}$
$\{p_2\}$
$\{p_1,p_2\}$
$\{\}$
$\{\}$
$\{p_2\}$
$\{p_2\}$


Σ
q_0
$\{p_1\},\{p_1,p_2\}$
q_1
$\{\},\{p_1\}$
q_2

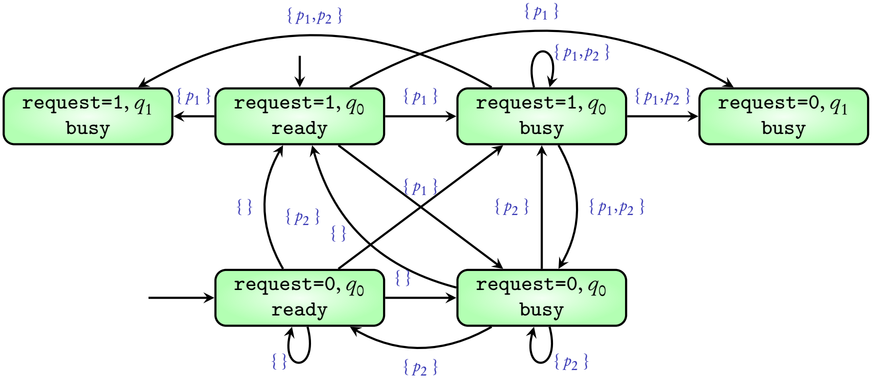

request=1, q_1 busy
request=1, q_0 ready
request=1, q_0 busy
request=0, q_1 busy
request=0, q_0 ready
request=0, q_0 busy
$\{p_1,p_2\}$
$\{p_1\}$
$\{p_1,p_2\}$
$\{p_1\}$
$\{p_1\}$
$\{p_1,p_2\}$
$\{p_1\}$
$\{p_2\}$
$\{p_2\}$
$\{p_1,p_2\}$
$\{p_2\}$
$\{p_2\}$



request=1 ready
request=1 busy
request=0 ready
request=0 busy
{ p_1 }
{ p_1,p_2 }
{ }
{ p_2 }



Σ
q_0
q_1
q_2
{ p_1 },{ p_1,p_2 }
{ },{ p_1 }



request=1, q_1 busy
request=1, q_0 ready
request=1, q_0 busy
request=0, q_1 busy
request=0, q_0 ready
request=0, q_0 busy
{ p_1 }
{ p_1,p_2 }
{ }
{ p_2 }

Step 3: Check if the language of the product automaton is empty

Step 3: Check if the language of the product automaton is empty

If language is empty, there are **no bad prefixes**

Step 3: Check if the language of the product automaton is empty

If language is empty, there are **no bad prefixes**

- Language empty $\longrightarrow$ model satisfies safety property
- Language non-empty $\longrightarrow$ model does not satisfy safety property

- Step 1: Convert model to automaton
- Step 2: Take synchronous product with **BadPrefixes** automaton
- Step 3: Check if language of product is empty

- Language empty $\longrightarrow$ model satisfies safety property
- Language non-empty $\longrightarrow$ model does not satisfy safety property

Regular safety properties

BadPrefixes is regular

Algorithm