# Abstract Regular Model Checking

*by*

Ahmed Bouajjani, Peter Habermehl and Tomáš Vojnar

presented by

Joakim Byg

# Introduction

- Problem
  - Reachability
    - Safety
    - Liveness
    - UPPAAL
  - State Space Space explosion
- Abstract/Over Approximate

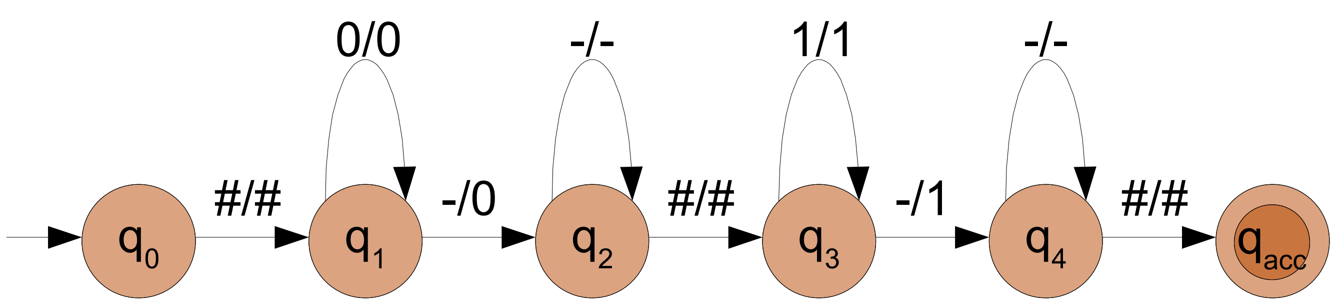
# Motivation

- Regular Model Checking (RMC) is Turing Complete
  - even though:

    *A set of initial strings, **I**, is regular and a given Transducer, **T**, is a (nondeterministic) Finite Automaton, the computation of **T*(I)** is not necessary regular! Where **T*(I)** means that **T** is used none or several times on **I***

# Example

- Let *I* be described by the regular expression:
  #0-*#1-*# , and
  Let $\Sigma_T$= {#,0,1,-} and $Q$,$q_0$,$F$ and $\delta$ $(\subseteq Q \times \Sigma_{T\varepsilon} \times \Sigma_{T\varepsilon} \times Q)$ of *T* is defined as the graph implies:




- Now *T*(I)* actually describes the language:
  $\#0^n\#1^n\#$, which is context free (not regular)

# Agenda

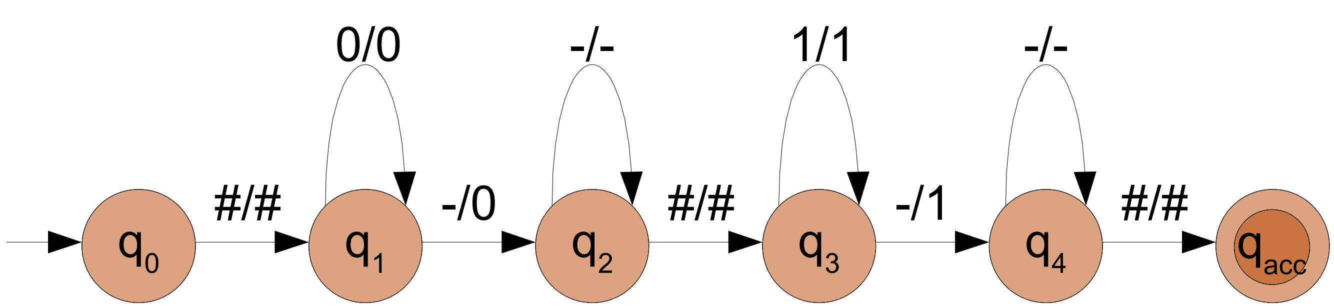
- Concerning Abstractions
- How to Abstract
- Experiments
- Conclusion

# Concerning Abstractions

State Space Reduction

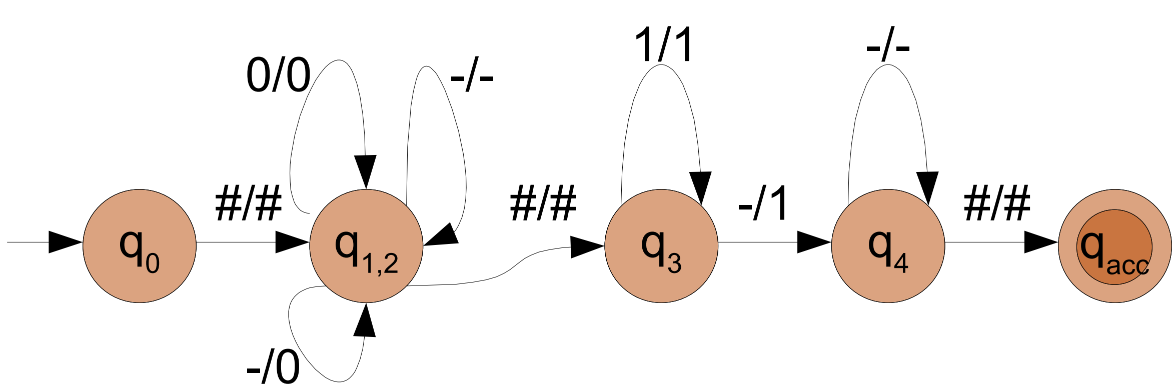
# Turing Completeness

- Problem:
  - Infinitely many reachable states (variables)
    - State Space Explosion
- Methods of Reduction of States
  - length preserving of strings
  - Over Approximation



# Turing Completeness

- Problem:
  - Infinitely many reachable states (variables)
    - State Space Explosion
- Methods of Reduction of States
  - length preserving of strings
  - Over Approximation



# Over Approximating

- We want an over approximation of $T^*(I)$ with less states.
  If the over approximation result in a positive result with respect to $\alpha(T)^*(I) \cap L(B)=\emptyset$,
  $T^*(I) \cap L(B)=\emptyset$ also holds.

# Over Approximating

- Let $\boldsymbol{M}_{\Sigma}$ denote all FA over the finite alphabet, $\Sigma$, and Let $\boldsymbol{A}_{\Sigma}$ be a FA, st. $\boldsymbol{A}_{\Sigma} \subseteq \boldsymbol{M}_{\Sigma}$, then $\alpha$ is a function:

$$\alpha : \boldsymbol{M}_{\Sigma} \rightarrow \boldsymbol{A}_{\Sigma}, \text{ st. } \forall M \in \boldsymbol{M}_{\Sigma} : L(M) \subseteq L(\alpha(M))$$

which is *finitary* $\Leftrightarrow$ $\boldsymbol{A}_{\Sigma}$ is finite.

# Over Approximating

- Now we use this idea on transducers
  - Let $\tau_s$ denote the smallest deterministic automaton of $\tau(L(M))$ and $\tau_\alpha(M)= \alpha(\tau_s(M))$
  - Because $\alpha$ is finitary, we will when computing $\tau_\alpha(M)$ iteratively reach a situation: $\tau_\alpha^{k+1}(M) = \tau_\alpha^{k}(M)$
    - What does this imply, with respect to $L(\tau_\alpha^{k}(M))$?

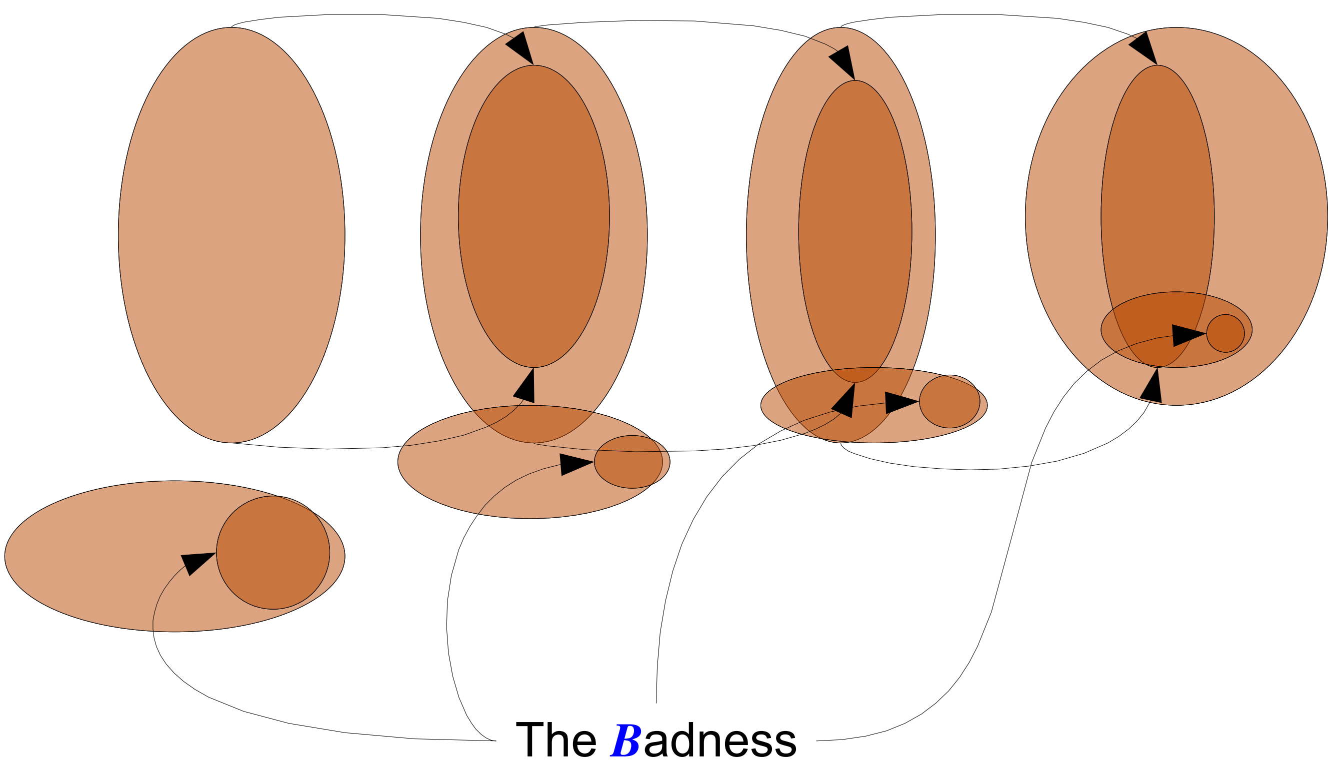
# Over Approximating

- Now we use this idea on transducers
  - Let $\tau_s$ denote the smallest deterministic automaton of $\tau(L(M))$ and $\tau_\alpha(M) = \alpha(\tau_s(M))$
  - Because $\alpha$ is finitary, we will when computing $\tau_\alpha(M)$ iteratively reach a situation: $\tau_\alpha^{k+1}(M) = \tau_\alpha^{k}(M)$
    - What does this imply, with respect to $L(\tau_\alpha^{k}(M))$?

      $\tau^*(L(M)) \subseteq L(\tau_\alpha^{k}(M))$

# Over Approximating

- What if $\tau^*(I) \cap L(B) = \emptyset$ suddenly is false?



# Over Approximating

- Then Refining is necessary.
  - Maybe we will have to refine back to the initial transducer.
  - This way we will get a “maybe” answer from the computation.

# How to Abstract

Collapsing of States

# Two Ways of Collapsing

- Predicate Languages
- Bounded Length Behaviours

# Predicate Languages

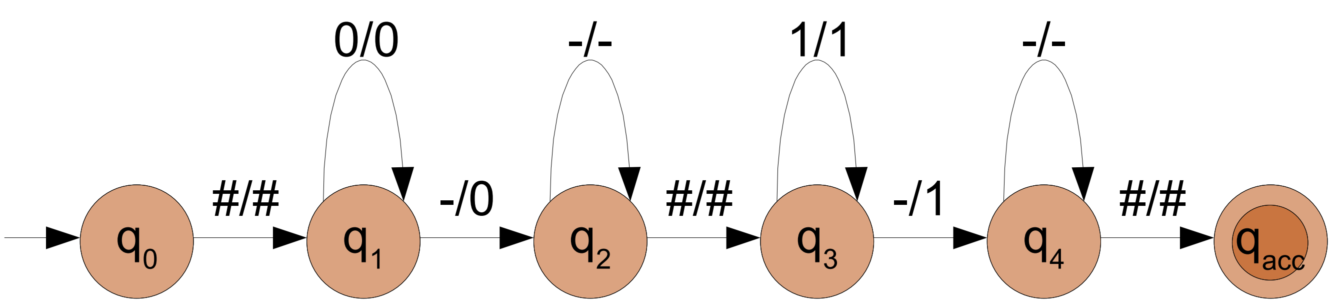
- Define: Backwards Language, $L^{\leftarrow}$, is the set of words that can be reached from some state, $q$, of a FA, $M$, to $q_0$ of $M$:

  $$L^{\leftarrow}(M, q) = \{w \mid q_0 \rightarrow^{w} q\}$$

- Define: Forward Predicate Language, $F_{\Upsilon}$, is the language of a given predicate automaton, $\Upsilon$.
- Define: Backwards Predicate Language, $B_{\Upsilon}$, is the backwards language of a predicate automaton, $\Upsilon$.

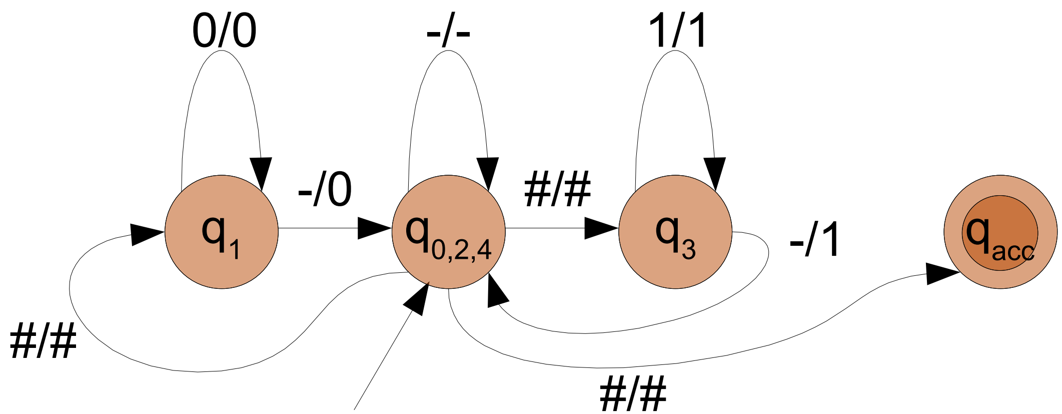
# Predicate Languages

- Define: Two states, $q_x$,$q_z$, of FA are state-equivalent, when the intersection of their predicate languages is nonempty:
  $L_\wp(M, q_x) \cap L_\wp(M, q_z) = S$, where $S$ is nonempty
- Example:



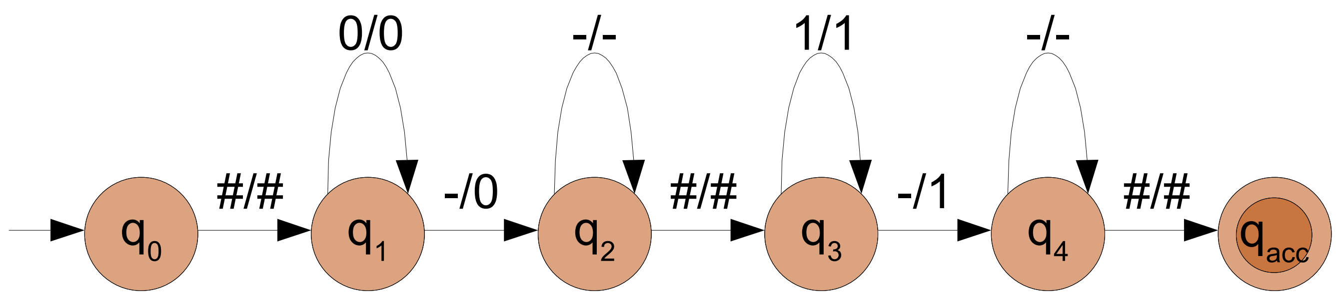
# Predicate Languages

- Define: Two states, $q_x$,$q_z$, of FA are state-equivalent, when the intersection of their predicate languages is nonempty:
  $L_{\wp}(M, q_x) \cap L_{\wp}(M, q_z) = S$, where $S$ is nonempty
- Example:



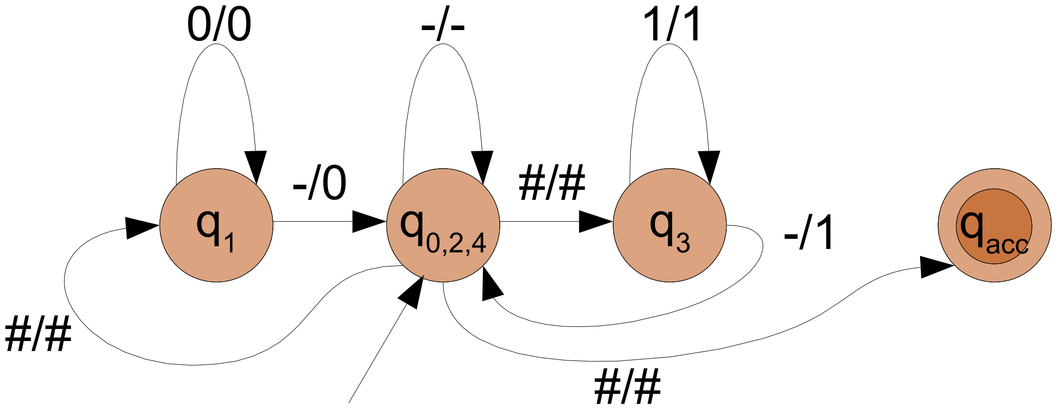
# Bounded Length Behaviours

- Think of Predicate Languages, but words must have a certain length!
- Example:
  Length ≤1



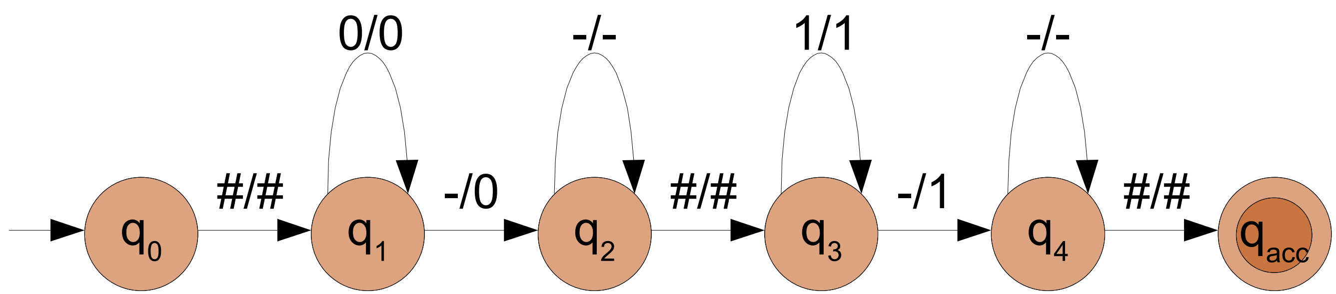
# Bounded Length Behaviours

- Think of Predicate Languages, but words must have a certain length!
- Example:
  Length ≤1



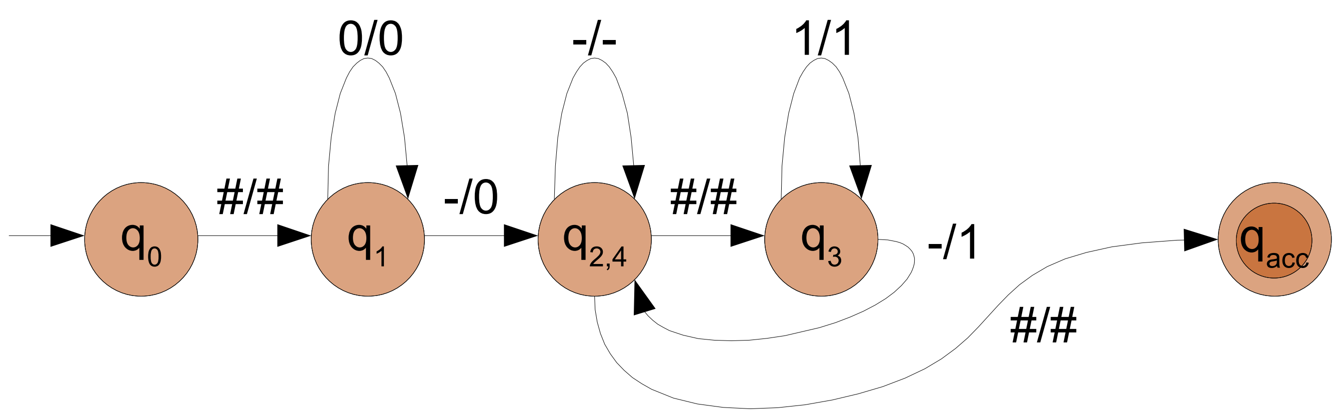
# Bounded Length Behaviours

- Think of Predicate Languages, but words must have a certain length!
- Example:
  Length ≥2



# Bounded Length Behaviours

- Think of Predicate Languages, but words must have a certain length!
- Example:
  Length ≥2



# Experiments

- Examples of application:
  - Alternating Bit Protocol
  - Petri Nets (Systems with unbounded counters)
  - Dynamic Linked Data Structures

# Dynamic Linked Data Structures

- Reversing a linear list
- String encoding of memory and pointers

# Result

- Quite Fast verifications at max 22 sec on low end PC (1,7GHz P4)!
- 5 Gossiping Girls 20min on a 2,0GHz P Core Duo in UPPAAL

# Summery

- What is an Abstraction/Over Approximation
  - Computable when $\alpha$ is finitary
  - Sneaking Bad Configurations
- How is it done
  - Predicate Language
  - Bounded Length Behaviour
- Quite Effective

# Last Words

- Future work
  - Will some classes of problems be guaranteed to terminate?
  - Lower Bound in Bounded Length Behaviour equivalence :)
- My opinion
  - Promising