Simulation-based Inclusion Checking Algorithms for ω-Languages

Francesco Parolini

23 July, 2020

Università degli Studi di Padova

Presentation

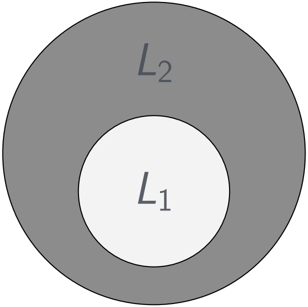
Università degli Studi di Padova

- **Candidate:** Francesco Parolini
- **Supervisor:** Prof. Francesco Ranzato
- **Co-supervisor:** Prof. Pierre Ganty, IMDEA Software Institute, Madrid
- **PhD. Student:** Kyveli Doveri, IMDEA Software Institute, Madrid

The Language Inclusion Problem

Definition (Language Inclusion Problem)

Let L_1 and L_2 be two languages. The **language inclusion problem** consists in deciding whether $L_1 \subseteq L_2$ holds or not.



Characteristics

- Whether the problem is computable or not depends on the class of the languages
- Also if it turns out to be computable, it is usually an hard problem

Applications

- Model checking
- Compilers construction
- Automata-based Verification

Definition (ω-language)

An ω-**language** L is a set of strings of *infinite length*.

Examples of words of **infinite** length:

$$abbb\cdots = ab^{\omega}$$

$$babbaababab\cdots = babba(ab)^{\omega}$$

ω-languages

Università degli Studi di Padova

Definition (ω-language)

An ω-**language** L is a set of strings of *infinite length*.

Examples of words of **infinite** length:

$$abbb\cdots = ab^\omega$$

$$babbaababab\cdots = babba(ab)^\omega$$

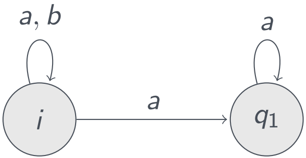
Our focus is on ω-**regular languages.**

Büchi automata

A **Büchi automaton** is a tuple $\mathcal{B} = \langle Q, \delta, \{i\}, F \rangle$

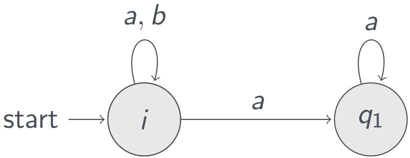
Büchi automata

A **Büchi automaton** is a tuple $\mathcal{B} = \langle Q, \delta, \{i\}, F \rangle$

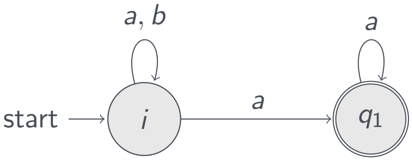


Büchi automata

A **Büchi automaton** is a tuple $\mathcal{B} = \langle Q, \delta, \{i\}, F \rangle$



A **Büchi automaton** is a tuple $\mathcal{B} = \langle Q, \delta, \{i\}, F \rangle$



A **trace** over the word $a_1 a_2 a_3 \ldots$:

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots$$

A **trace** over the word $a_1 a_2 a_3 \ldots$:

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots$$

An **initial** trace over the word $a_1 a_2 a_3 \ldots$:

$$\color{red}{i} \color{black}{} \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots$$

Traces

A **trace** over the word $a_1 a_2 a_3 \ldots$:

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots$$

An **initial** trace over the word $a_1 a_2 a_3 \ldots$:

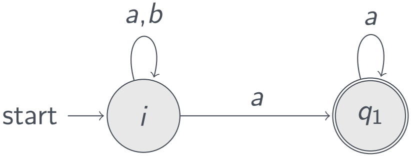
$$i \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots$$

A **fair** trace over the word $a_1 a_2 a_3 \ldots$:

$$q_0 \xrightarrow{a_1} q_f \xrightarrow{a_2} q_3 \xrightarrow{a_3} \cdots \xrightarrow{a_i} q_f \xrightarrow{a_{i+1}} \cdots \xrightarrow{a_j} q_f \xrightarrow{a_{j+1}} \cdots$$

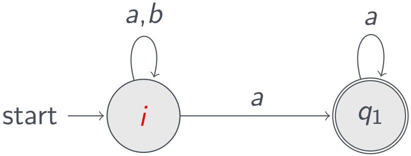
Traces

An **initial** and **fair** trace over aba^{ω}:



Traces

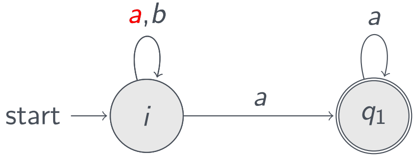
An **initial** and **fair** trace over aba^ω:



i

Traces

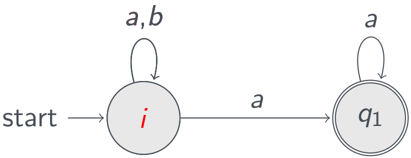
An **initial** and **fair** trace over aba^ω:



$$i \xrightarrow{a}$$

Traces

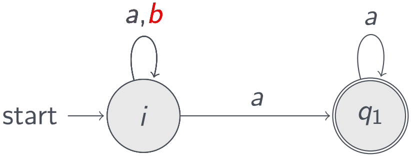
An **initial** and **fair** trace over aba^ω:



$$i \xrightarrow{a} i$$

Traces

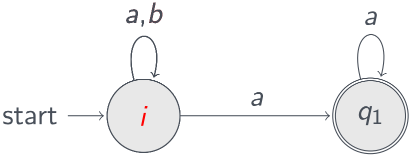
An **initial** and **fair** trace over aba^ω:



$$i \xrightarrow{a} i \xrightarrow{b}$$

Traces

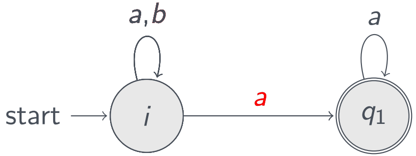
An **initial** and **fair** trace over aba^ω:



$$i \xrightarrow{a} i \xrightarrow{b} i$$

Traces

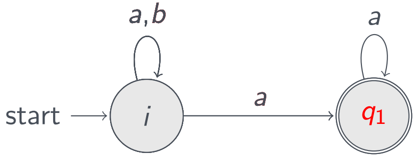
An **initial** and **fair** trace over aba^ω:



$$i \xrightarrow{a} i \xrightarrow{b} i \xrightarrow{a}$$

Traces

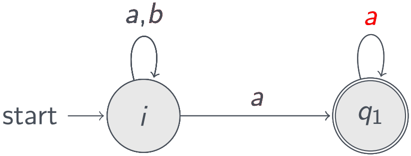
An **initial** and **fair** trace over aba^ω:



$$i \xrightarrow{a} i \xrightarrow{b} i \xrightarrow{a} q_1$$

Traces

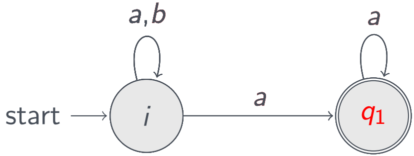
An **initial** and **fair** trace over aba^ω:



$$i \xrightarrow{a} i \xrightarrow{b} i \xrightarrow{a} q_1 \xrightarrow{a}$$

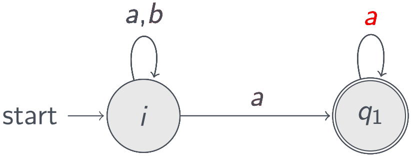
Traces

An **initial** and **fair** trace over aba^ω:



$$i \xrightarrow{a} i \xrightarrow{b} i \xrightarrow{a} q_1 \xrightarrow{a} q_1$$

An **initial** and **fair** trace over aba^ω:



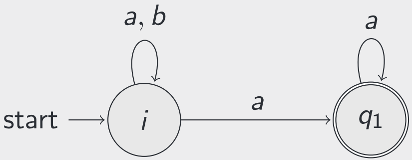
$$i \xrightarrow{a} i \xrightarrow{b} i \xrightarrow{a} q_1 \xrightarrow{a} q_1 \xrightarrow{a} \cdots$$

The language of a Büchi automaton

The *language recognized by a Büchi automaton* $\mathcal{B}$ is:

$$\mathcal{L}(\mathcal{B}) = \{w \mid \text{there is an initial and fair trace over } w\}$$

Example



$$\mathcal{L}(\mathcal{B}) = \{a^\omega, ba^\omega, aba^\omega, bba^\omega, \dots\} = (a+b)^* a^\omega$$

$\omega-$regular languages

Definition (ω-regular language)

The class of languages recognized by Büchi automata is called ω-**regular languages**.

Applications

- LTL as Büchi automata
- Automata-based model checking

Deciding the Language Inclusion

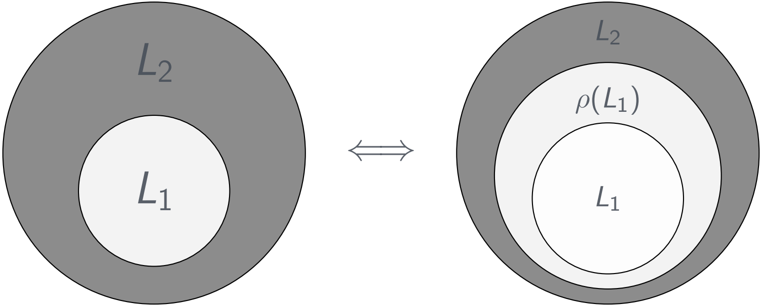
- Languages are **not finite**, we can't just compare them

Deciding the Language Inclusion

- Languages are **not finite**, we can't just compare them
- **Abstract Interpretation**
  - Static program analysis
  - Giving up precision for computability

Deciding the Language Inclusion

We started from the "Doveri-Ganty" framework for checking the language inclusion, which relies on *Abstract Interpretation* techniques.



Details

A **ultimately periodic** word:

$$abc(de)^\omega$$

Details

A **ultimately periodic** word:

$$abc(de)^\omega$$

We define:

$$I_L \triangleq \{(u, v) \mid uv^\omega \in L\}$$

A **ultimately periodic** word:

$$abc(de)^\omega$$

We define:

$$I_L \triangleq \{(u, v) \mid uv^\omega \in L\}$$

Then, one key observation is:

$$L_1 \subseteq L_2 \iff I_{L_1} \subseteq I_{L_2}$$

A **ultimately periodic** word:

$$abc(de)^{\omega}$$

We define:

$$I_L \triangleq \{(u, v) \mid uv^{\omega} \in L\}$$

Then, one key observation is:

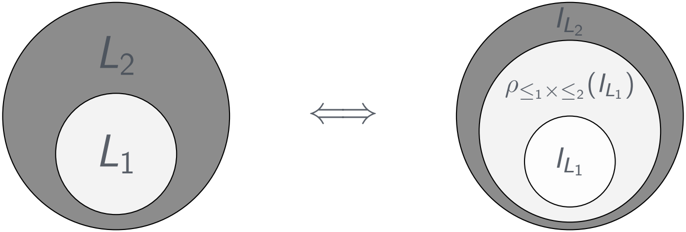
$$L_1 \subseteq L_2 \iff I_{L_1} \subseteq I_{L_2}$$

Let $\leq_1, \leq_2$ be two **preorders** on words.

$$\rho_{\leq_1 \times \leq_2}(I_L) \triangleq \{(s, t) \mid \exists (u, v) \in I_L, u \leq_1 s \ \wedge \ v \leq_2 t\}$$

Let $\leq_1, \leq_2$ be two preorders on words that meet a list of requirements related to **computability** and **completeness**.

$$L_1 \subseteq L_2 \iff \rho_{\leq_1 \times \leq_2}(I_{L_1}) \subseteq I_{L_2}$$

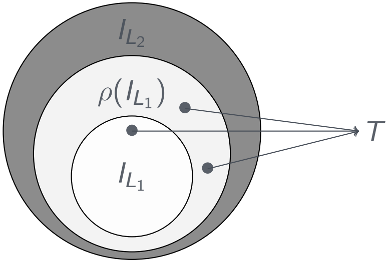


Observation: usually when abstracting one object we gain decidability, but here the abstraction goes from one infinite set (I_{L_1}) to another infinite set... Why?

Gaining decidability

We can extract from the abstraction $\rho_{\leq_1 \times \leq_2}(I_{L_1})$ a **finite** set, say T, such that:

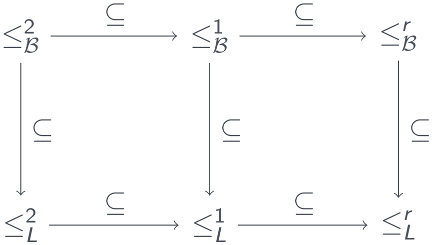
$$L_1 \subseteq L_2 \iff \forall (u, v) \in T, uv^\omega \in L_2$$



Algorithm to solve $L_1 \subseteq L_2$

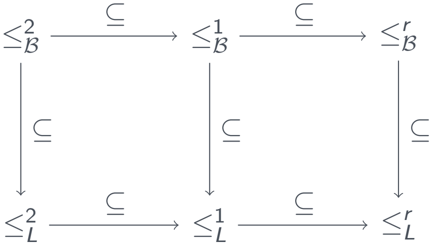
Università degli Studi di Padova

- They give `BAInc`, algorithm to solve $L_1 \subseteq L_2$
  1. Computes T
  2. Checks if $\forall (u, v) \in T, uv^\omega \in L_2$
- `BAInc` is parametrized by $\leq_1, \leq_2$



Algorithm to solve $L_1 \subseteq L_2$

- They give `BAInc`, algorithm to solve $L_1 \subseteq L_2$
  1. Computes T
  2. Checks if $\forall (u, v) \in T, uv^\omega \in L_2$
- `BAInc` is parametrized by $\leq_1, \leq_2$

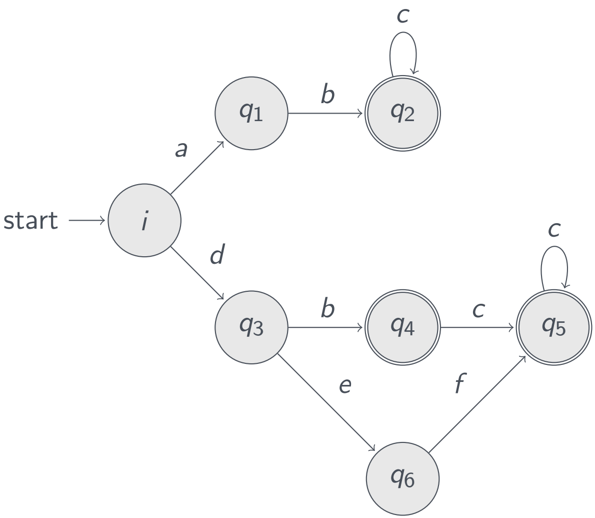


My task: to define new preorders $\leq_1, \leq_2$

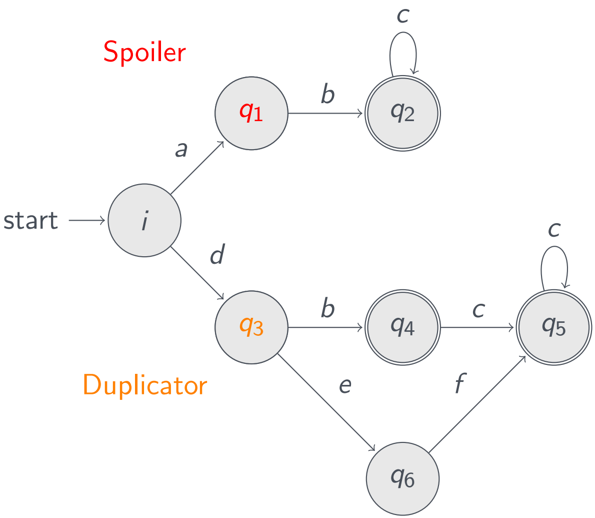
Simulations

- Behavioural relations
- Intuitively, one state is simulated by another if the second can match all the moves of the first
- Fundamental in Process Calculi
- There are many known algorithms to compute simulations

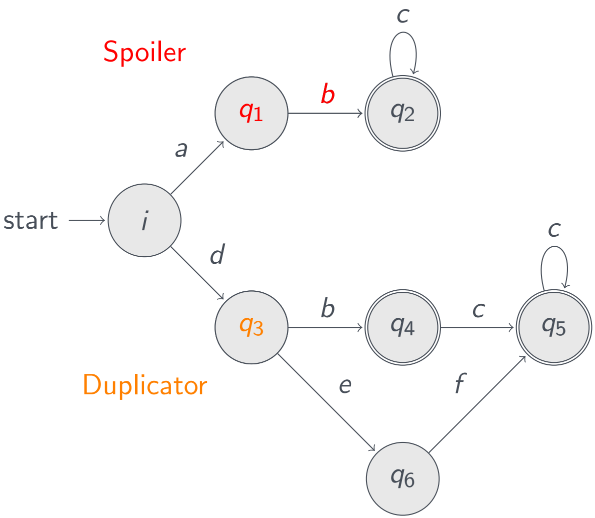
The Game of Simulation



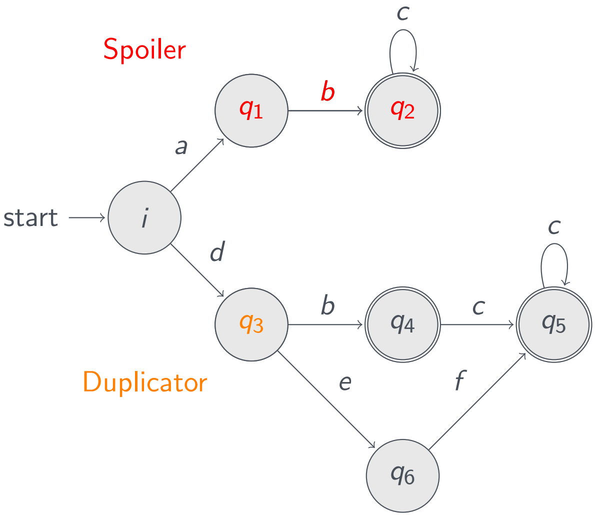
The Game of Simulation



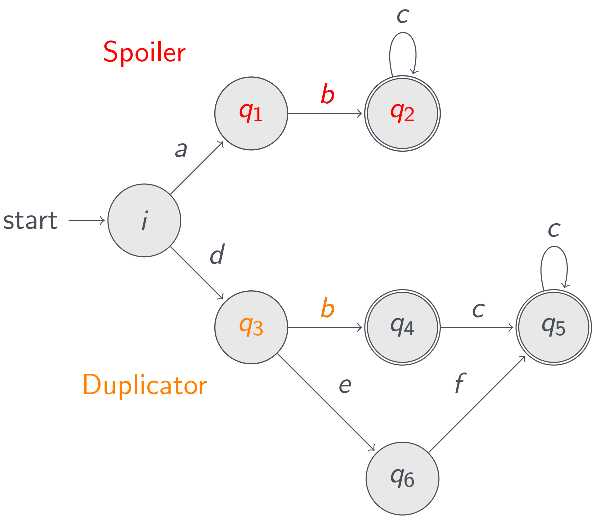
The Game of Simulation



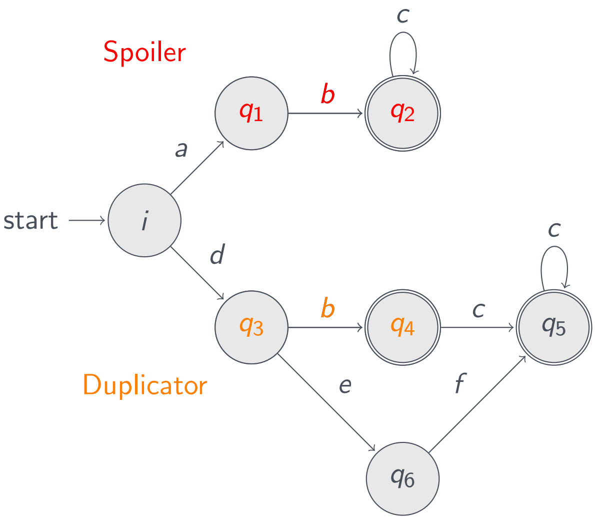
The Game of Simulation



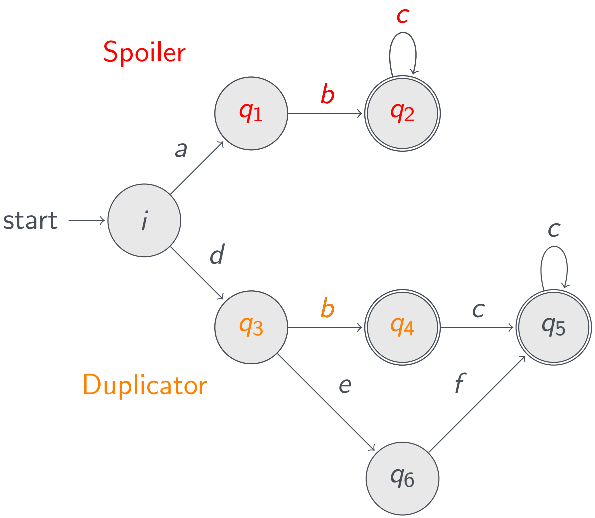
The Game of Simulation



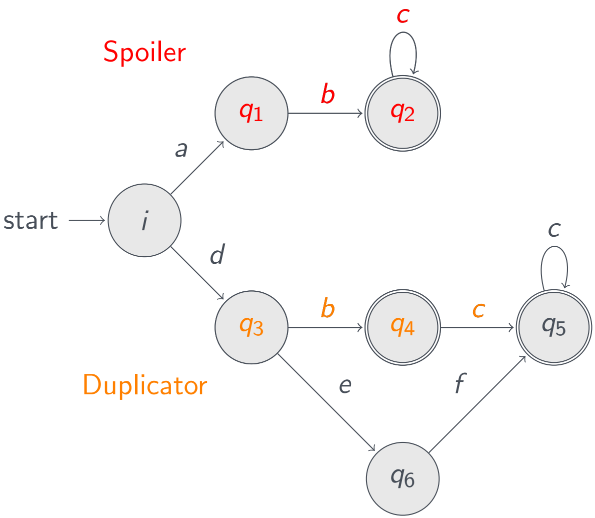
The Game of Simulation



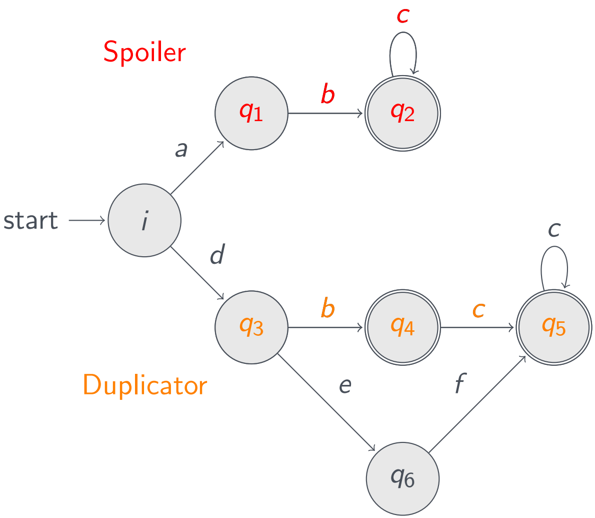
The Game of Simulation



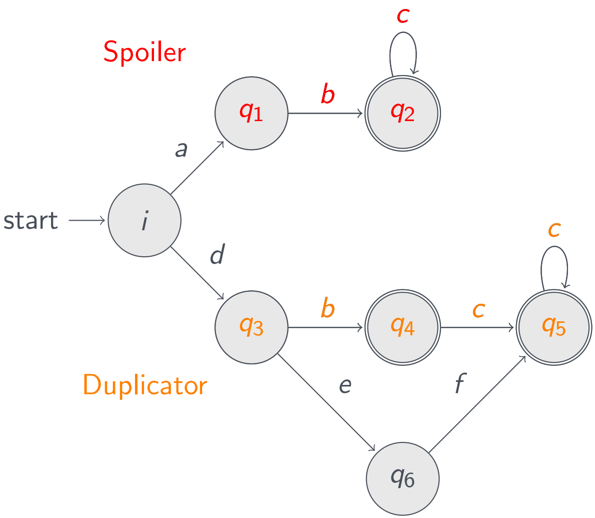
The Game of Simulation

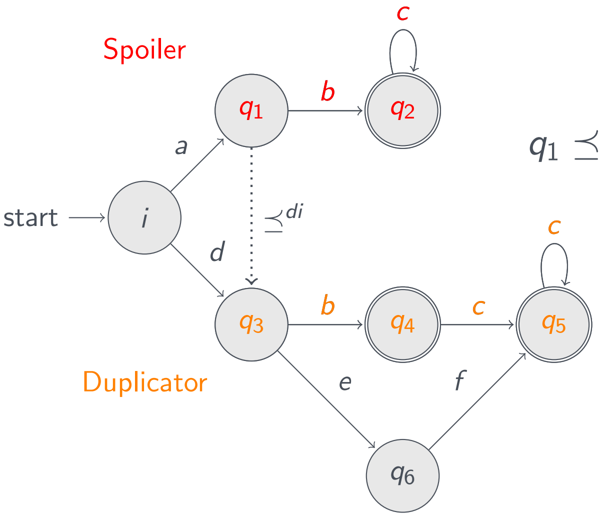


The Game of Simulation



The Game of Simulation



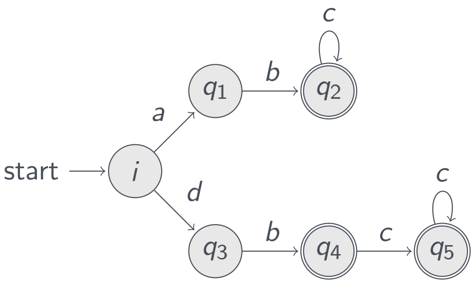


$$q_1 \preceq^{di} q_3$$

My work

I started from:

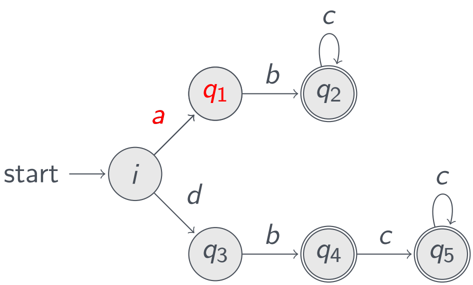
$$u \sqsubseteq^r_{\mathcal{B}} v \iff \textbf{for each } \text{state } p \text{ such that } i \overset{u}{\rightsquigarrow} p,$$
$$\textbf{exists } \text{a state } q \text{ such that } i \overset{v}{\rightsquigarrow} q \text{ and } p \preceq^{di} q$$



My work

I started from:

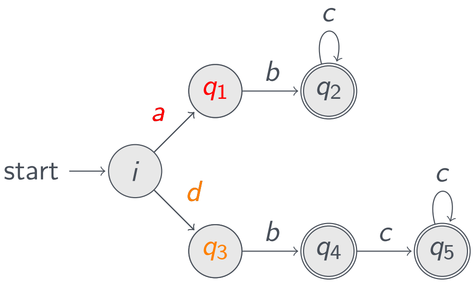
$$u \sqsubseteq^r_{\mathcal{B}} v \iff \textbf{for each } \text{state } p \text{ such that } i \overset{u}{\rightsquigarrow} p,$$
$$\textbf{exists } \text{a state } q \text{ such that } i \overset{v}{\rightsquigarrow} q \text{ and } p \preceq^{di} q$$



My work

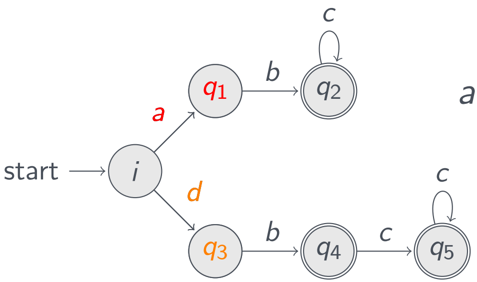
I started from:

$$u \sqsubseteq^r_{\mathcal{B}} v \iff \textbf{for each } \text{state } p \text{ such that } i \overset{u}{\rightsquigarrow} p,$$
$$\textbf{exists } \text{a state } q \text{ such that } i \overset{v}{\rightsquigarrow} q \text{ and } p \preceq^{di} q$$



I started from:

$$u \sqsubseteq^r_{\mathcal{B}} v \iff \textbf{for each} \text{ state } p \text{ such that } i \overset{u}{\rightsquigarrow} p, \textbf{ exists} \text{ a state } q \text{ such that } i \overset{v}{\rightsquigarrow} q \text{ and } p \preceq^{di} q$$

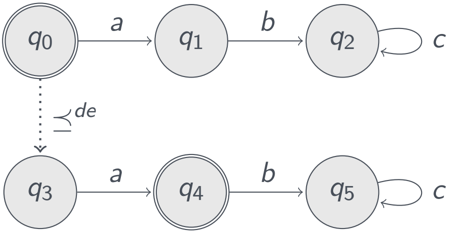


$a \sqsubseteq^r_{\mathcal{B}} d$

New preorders

Generalization using **different simulations**:

- $\sqsubseteq_{\mathcal{B}}^{de,r}$

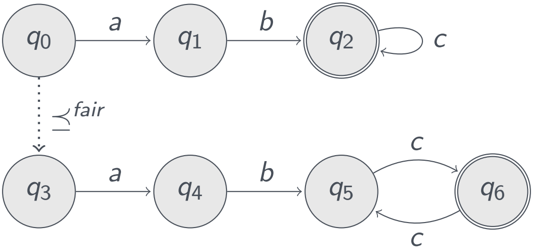


New preorders

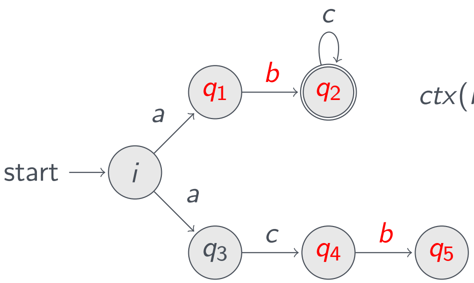
Università degli Studi di Padova

Generalization using **different simulations**:

- $\sqsubseteq_{\mathcal{B}}^{de,r}$
- $\sqsubseteq_{\mathcal{B}}^{fair,r}$



The **context** of a word:



$ctx(b) = \{(q_1, q_2), (q_4, q_5)\}$

Generalization using **pairs** of states:

- $\sqsubseteq^1_\mathcal{B}$
- $\sqsubseteq^2_\mathcal{B}$

My work

- Proved a list of requirements related to **computability** and **completeness**
  1. computability
  2. right-monotonicity ($u \leq v \implies uw \leq vw$)
  3. being a well-quasiorder (for each infinite sequence $\{x_i\}_{i \in \mathbb{N}}$, $\exists i, j : i < j \ \wedge \ x_i \leq x_j$)
  4. $\rho_{\leq_1 \times \leq_2}(I_{L_2}) = I_{L_2}$
- Identified which pairs are suitable for the framework

$$\sqsubseteq^1_{\mathcal{B}}, \sqsubseteq^2_{\mathcal{B}}$$

$$\sqsubseteq^r_{\mathcal{B}}, \sqsubseteq^2_{\mathcal{B}}$$

$$\sqsubseteq^{de,r}_{\mathcal{B}}, \sqsubseteq^2_{\mathcal{B}}$$

$$\sqsubseteq^{fair,r}_{\mathcal{B}}, \sqsubseteq^2_{\mathcal{B}}$$

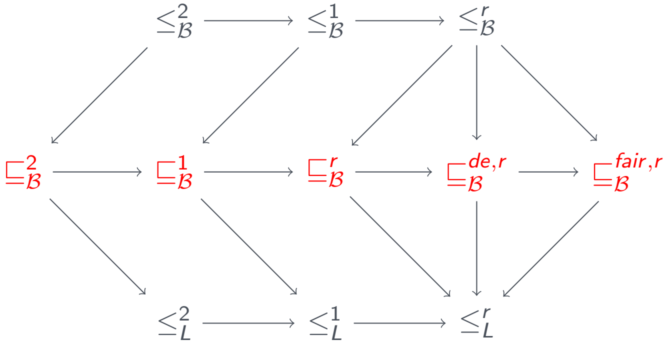
Other considered simulations

- K-lookahead simulations
- Trace inclusions
- "K-delayed" simulations

Other considered simulations

- K-lookahead simulations
- Trace inclusions
- "K-delayed" simulations

The relations on words derived from these do not meet the requirements.



$\lesssim^2_{\mathcal{B}}$
$\lesssim^1_{\mathcal{B}}$
$\lesssim^r_{\mathcal{B}}$
$\sqsubseteq^2_{\mathcal{B}}$
$\sqsubseteq^1_{\mathcal{B}}$
$\sqsubseteq^r_{\mathcal{B}}$
$\sqsubseteq^{de,r}_{\mathcal{B}}$
$\sqsubseteq^{fair,r}_{\mathcal{B}}$
$\lesssim^2_{L}$
$\lesssim^1_{L}$
$\lesssim^r_{L}$

Why bother?

Simulations and the language inclusion problem:

Why bother?

Simulations and the language inclusion problem:

- **2010**: Abdulla, P.A. et al. *When simulation meets antichains.*

Why bother?

Simulations and the language inclusion problem:

- **2010**: Abdulla, P.A. et al. *When simulation meets antichains.*
- **2011**: Abdulla, P.A. et al. *Advanced Ramsey-based Büchi automata inclusion testing.*

Why bother?

Simulations and the language inclusion problem:

- **2010**: Abdulla, P.A. et al. *When simulation meets antichains.*
- **2011**: Abdulla, P.A. et al. *Advanced Ramsey-based Büchi automata inclusion testing.*
- **2013**: Bonchi, F. and Pous, D. *Checking NFA equivalence with bisimulations up to congruence.*

Why bother?

Simulations and the language inclusion problem:

- **2010**: Abdulla, P.A. et al. *When simulation meets antichains.*
- **2011**: Abdulla, P.A. et al. *Advanced Ramsey-based Büchi automata inclusion testing.*
- **2013**: Bonchi, F. and Pous, D. *Checking NFA equivalence with bisimulations up to congruence.*
- **2017**: Mayr, R. and Clemente, L. *Efficient reduction of nondeterministic automata with application to language inclusion testing.*

What's next

Thanks for your attention