Language Equivalence is Undecidable in VASS with Restricted Nondeterminism

Wojciech Czerwiński[1], **Łukasz Orlikowski**[1,2]

[1]University of Warsaw
[2]Max Planck Institute for Software Systems (MPI-SWS), Kaiserslautern

23.10.2026

Outline

Introduction

Outline

Introduction

Main theorem

Outline

Introduction

Main theorem

Corollaries

Outline

Introduction

Main theorem

Corollaries

Conclusions

Equivalence

- A natural question for any pair of systems is whether they are equivalent

Equivalence

- A natural question for any pair of systems is whether they are equivalent
  - Language equivalence

Equivalence

- A natural question for any pair of systems is whether they are equivalent
  - Language equivalence
  - Two-sided simulation (whether both systems can simulate the other)

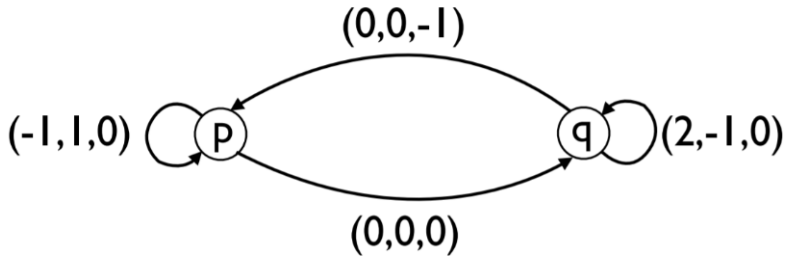
Equivalence

- A natural question for any pair of systems is whether they are equivalent
  - Language equivalence
  - Two-sided simulation (whether both systems can simulate the other)
  - Bisimulation

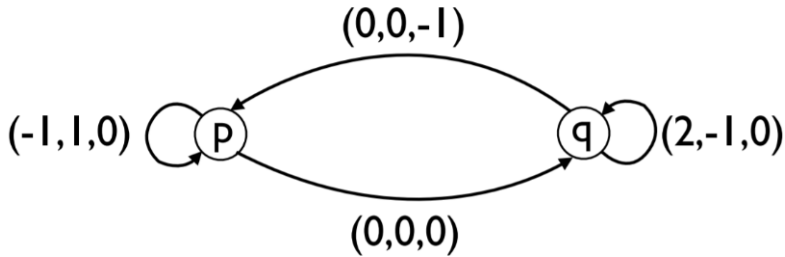
Equivalence

- A natural question for any pair of systems is whether they are equivalent
  - Language equivalence
  - Two-sided simulation (whether both systems can simulate the other)
  - Bisimulation
  - Isomorphism

Vector Addition Systems with States (VASS)

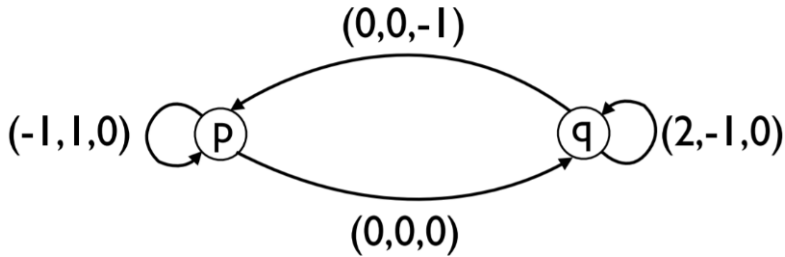


Vector Addition Systems with States (VASS)



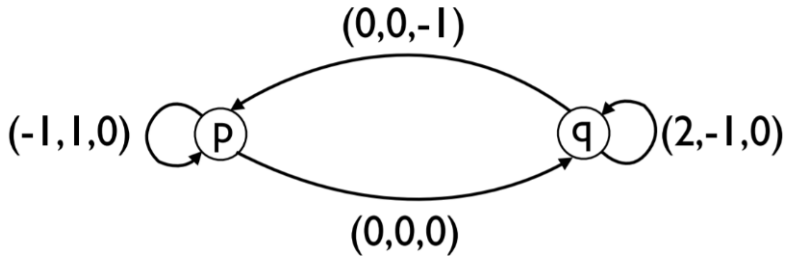
- We can equip a VASS with letters on transitions, initial and accepting configurations to recognize languages

Vector Addition Systems with States (VASS)



- We can equip a VASS with letters on transitions, initial and accepting configurations to recognize languages
- We consider coverability VASS languages where configuration is accepting if it covers configuration from a finite set

Vector Addition Systems with States (VASS)



- We can equip a VASS with letters on transitions, initial and accepting configurations to recognize languages
- We consider coverability VASS languages where configuration is accepting if it covers configuration from a finite set
- Special case: trace languages, i.e., all configurations are accepting

Language equivalence for VASSs

- Language equivalence for VASSs is undecidable **(Jančar, 1995)**

Language equivalence for VASSs

- Language equivalence for VASSs is undecidable **(Jančar, 1995)**
- Checking equivalence for every relation between isomorphism and language equivalence is undecidable **(Jančar, 2001)**

Deterministic systems

- Deterministic systems are simple (we know what to do next)

Deterministic systems

- Deterministic systems are simple (we know what to do next)
- We often know efficient algorithms for deterministic systems

Deterministic systems

- Deterministic systems are simple (we know what to do next)
- We often know efficient algorithms for deterministic systems
- Language equivalence is decidable (by reduction to VASS reachability) for deterministic VASSs **(Araki and Kasami, 1975)**

Restricted nondeterminism

- Determinization is often costly (exponential blow-up)

Restricted nondeterminism

- Determinization is often costly (exponential blow-up)
- Hence we often look for a middle-ground (some restrictions of nondeterminism)

Restricted nondeterminism

- Determinization is often costly (exponential blow-up)
- Hence we often look for a middle-ground (some restrictions of nondeterminism)
- VASS is unambiguous if for each word w there is at most one accepting run over it

Restricted nondeterminism

- Determinization is often costly (exponential blow-up)
- Hence we often look for a middle-ground (some restrictions of nondeterminism)
- VASS is unambiguous if for each word w there is at most one accepting run over it
- Language equivalence is decidable if both VASSs are unambiguous **(Czerwiński and Hofman, 2022)**

Restricted nondeterminism

- Determinization is often costly (exponential blow-up)
- Hence we often look for a middle-ground (some restrictions of nondeterminism)
- VASS is unambiguous if for each word w there is at most one accepting run over it
- Language equivalence is decidable if both VASSs are unambiguous **(Czerwiński and Hofman, 2022)**
- VASS is history-deterministic if we can resolve nondeterministic choices on-the-fly

Restricted nondeterminism

- Determinization is often costly (exponential blow-up)
- Hence we often look for a middle-ground (some restrictions of nondeterminism)
- VASS is unambiguous if for each word w there is at most one accepting run over it
- Language equivalence is decidable if both VASSs are unambiguous **(Czerwiński and Hofman, 2022)**
- VASS is history-deterministic if we can resolve nondeterministic choices on-the-fly
- Language equivalence is decidable if both VASSs are history-deterministic and have one counter **(Prakash, Thejaswini, 2022)**

Main theorem

Theorem

It is undecidable to determine whether a given deterministic 2-VASS and a given history-deterministic 2-VASS have equal trace languages.

Proof sketch of the main theorem

- We reuse ideas from Jančar's proof

Proof sketch of the main theorem

- We reuse ideas from Jančar's proof
- We reduce from the reachability problem for two-counter automaton

Proof sketch of the main theorem

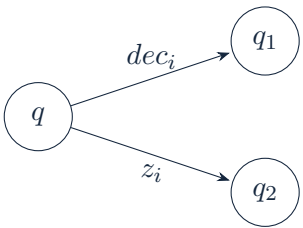
- We reuse ideas from Jančar's proof
- We reduce from the reachability problem for two-counter automaton
- The undecidability holds already when the two-counter automaton is deterministic

Proof sketch of the main theorem

- We reuse ideas from Jančar's proof
- We reduce from the reachability problem for two-counter automaton
- The undecidability holds already when the two-counter automaton is deterministic
  - Single initial and final state

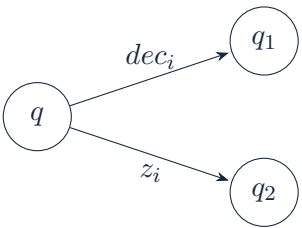
Proof sketch of the main theorem

- We reuse ideas from Jančar's proof
- We reduce from the reachability problem for two-counter automaton
- The undecidability holds already when the two-counter automaton is deterministic
  - Single initial and final state



Proof sketch of the main theorem

- We reuse ideas from Jančar's proof
- We reduce from the reachability problem for two-counter automaton
- The undecidability holds already when the two-counter automaton is deterministic
  - Single initial and final state





Proof sketch of the main theorem

Goal: From two-counter automaton M with initial state q_I and final state q_F we want to construct two 2-VASSs A and B such that

Proof sketch of the main theorem

Goal: From two-counter automaton M with initial state q_I and final state q_F we want to construct two 2-VASSs A and B such that

- A is deterministic and B is history-deterministic

Proof sketch of the main theorem

Goal: From two-counter automaton M with initial state q_I and final state q_F we want to construct two 2-VASSs A and B such that

- A is deterministic and B is history-deterministic
- if $q_I(0,0) \xrightarrow{*} q_F(v)$ in M then $L_T(A) \neq L_T(B)$

Proof sketch of the main theorem

Goal: From two-counter automaton M with initial state q_I and final state q_F we want to construct two 2-VASSs A and B such that

- A is deterministic and B is history-deterministic
- if $q_I(0,0) \xrightarrow{*} q_F(v)$ in M then $L_T(A) \neq L_T(B)$
- otherwise $L_T(A) = L_T(B)$

Proof sketch of the main theorem

Step 1: Construct auxiliary VASS N

Proof sketch of the main theorem

Step 1: Construct auxiliary VASS N

- Treat M as a 2-VASS, replacing each zero-test transition by two consecutive transitions with zero counter effect.

Proof sketch of the main theorem

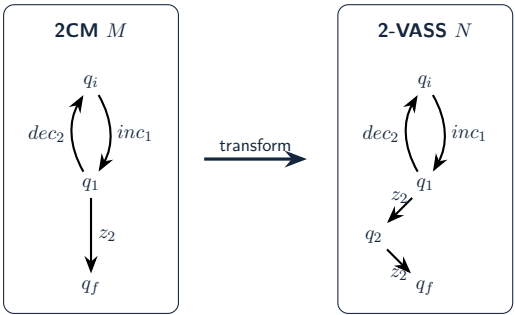
Step 1: Construct auxiliary VASS N

- Treat M as a 2-VASS, replacing each zero-test transition by two consecutive transitions with zero counter effect.
- Transition labels $\{inc_1, inc_2, z_1, z_2, dec_1, dec_2\}$ represent operations of M.

Proof sketch of the main theorem

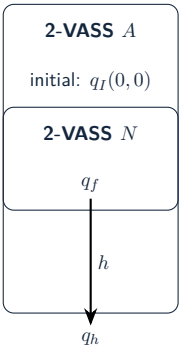
Step 1: Construct auxiliary VASS N

- Treat M as a 2-VASS, replacing each zero-test transition by two consecutive transitions with zero counter effect.
- Transition labels $\{inc_1, inc_2, z_1, z_2, dec_1, dec_2\}$ represent operations of M.



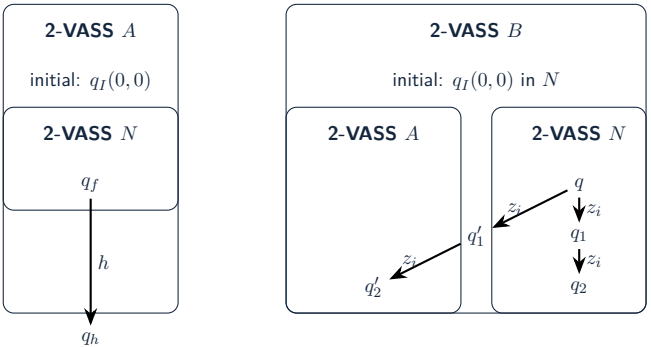
Proof sketch of the main theorem

Step 2: Construct VASSs A and B



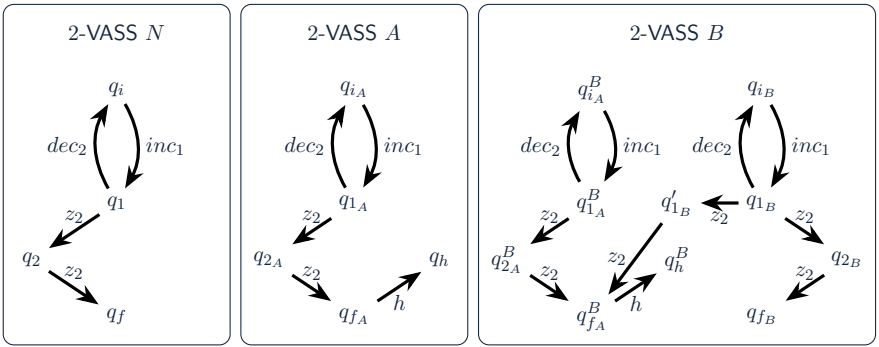
Proof sketch of the main theorem

Step 2: Construct VASSs A and B



- For every transition over z_i in B in the right part we allow to go to the left part by executing two transitions: first decrementing counter i, and the second incrementing

Proof sketch of the main theorem

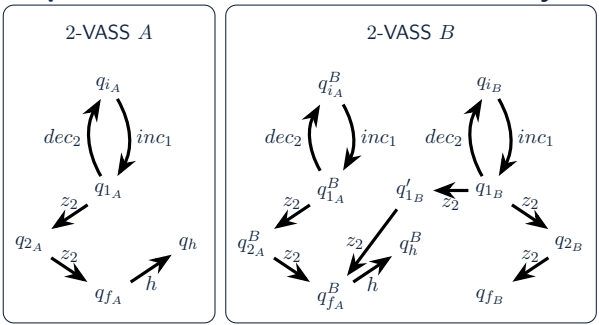


Proof sketch of the main theorem

Step 3: A is **deterministic** and B is **history-deterministic**

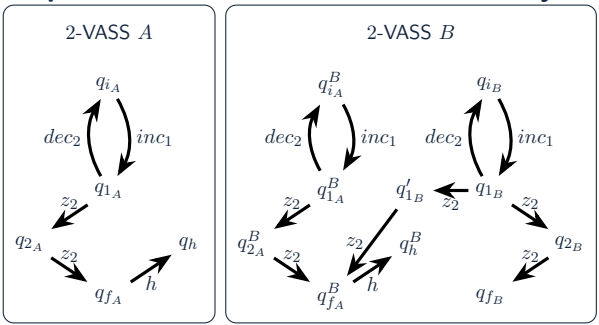
Proof sketch of the main theorem

Step 3: A is **deterministic** and B is **history-deterministic**



Proof sketch of the main theorem

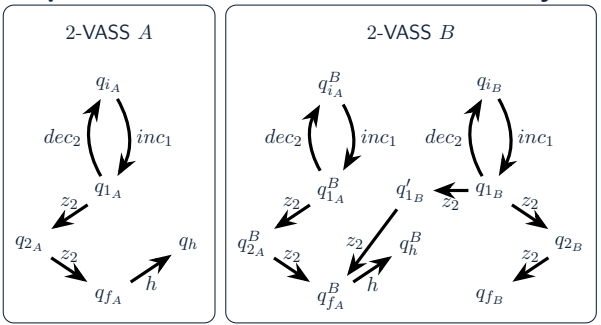
Step 3: A is **deterministic** and B is **history-deterministic**



- A is deterministic: for every state q and every letter a there is at most one outgoing transition from q over a in A

Proof sketch of the main theorem

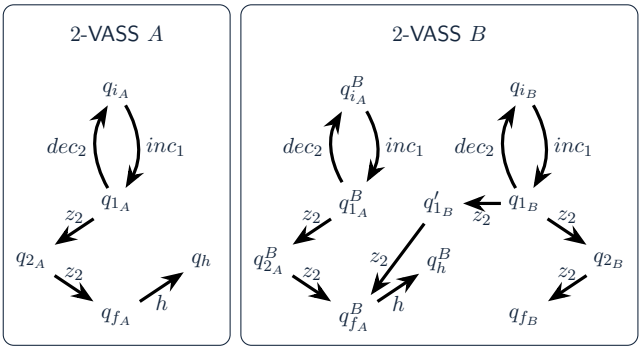
Step 3: A is **deterministic** and B is **history-deterministic**



- A is deterministic: for every state q and every letter a there is at most one outgoing transition from q over a in A
- B is history-deterministic: to resolve nondeterminism it is enough to cross from the right part to the left part as soon as possible

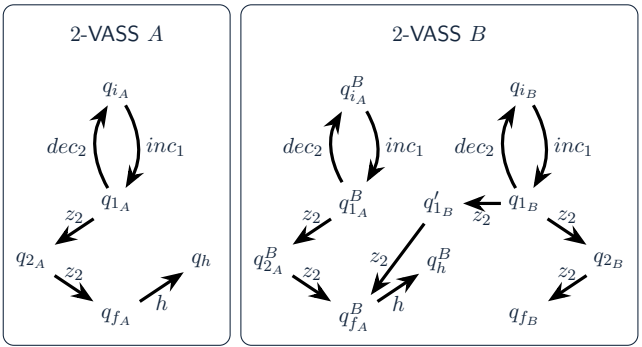
Proof sketch of the main theorem

Step 4: if $q_I(0,0) \xrightarrow{*} q_F(v)$ in M then $L_T(A) \neq L_T(B)$



Proof sketch of the main theorem

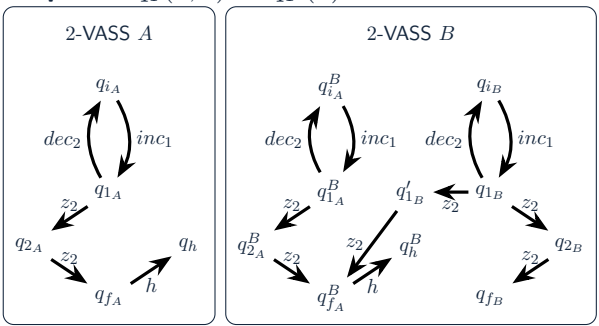
Step 4: if $q_I(0,0) \xrightarrow{*} q_F(v)$ in M then $L_T(A) \neq L_T(B)$



- Let ρ be the run witnessing reachability in M. Then ρh is read by A but not by B

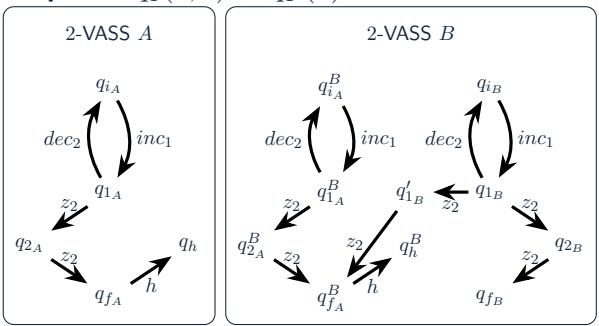
Proof sketch of the main theorem

Step 5: if $q_I(0,0) \xrightarrow{*} q_F(v)$ does **not** hold then $L_T(A) = L_T(B)$



Proof sketch of the main theorem

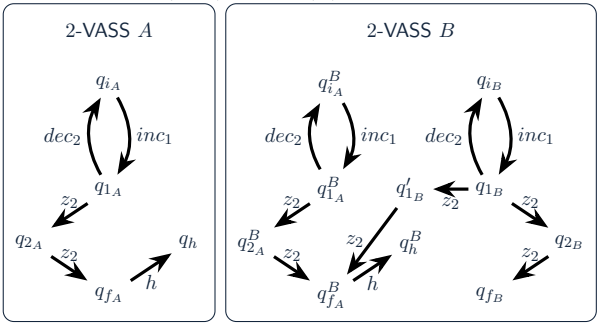
Step 5: if $q_I(0,0) \xrightarrow{*} q_F(v)$ does **not** hold then $L_T(A) = L_T(B)$



- Until the word we are reading follows the run of M it can be read in both A and B

Proof sketch of the main theorem

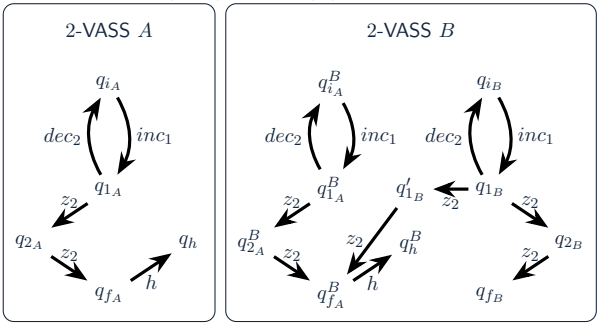
Step 5: if $q_I(0,0) \xrightarrow{*} q_F(v)$ does **not** hold then $L_T(A) = L_T(B)$



- Until the word we are reading follows the run of M it can be read in both A and B
- If it deviates from the run of M it must happen on reading z_i when ith counter is not zero

Proof sketch of the main theorem

Step 5: if $q_I(0,0) \xrightarrow{*} q_F(v)$ does **not** hold then $L_T(A) = L_T(B)$



- Until the word we are reading follows the run of M it can be read in both A and B
- If it deviates from the run of M it must happen on reading z_i when ith counter is not zero
- Then B can cross from the right part to the left part and behave later as A

Corollaries

Undecidability applies also for:

- coverability languages

Corollaries

Undecidability applies also for:

- coverability languages
- unambiguous vs nondeterministic case (unlike unambiguous vs unambiguous case)

Corollaries

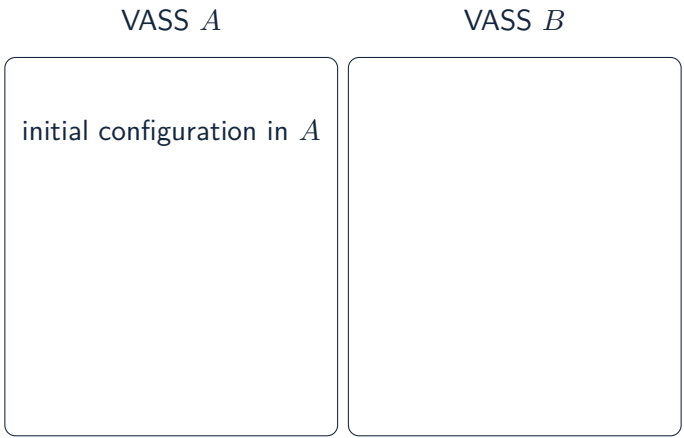
Undecidability applies also for:

- coverability languages
- unambiguous vs nondeterministic case (unlike unambiguous vs unambiguous case)
- containment of a language of a deterministic 2-VASS in a language of a history-deterministic 2-VASS

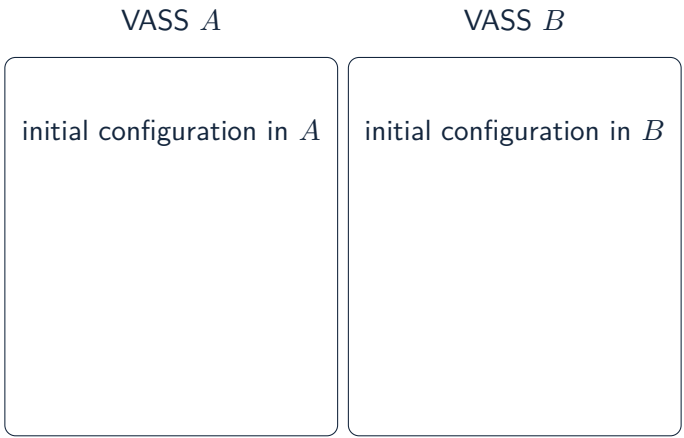
Simulation



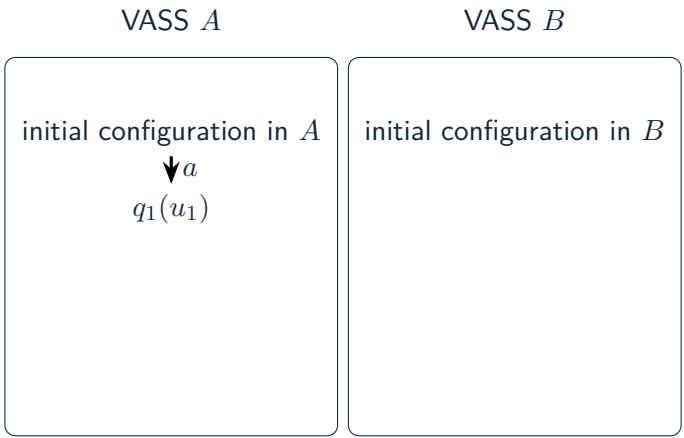
Simulation



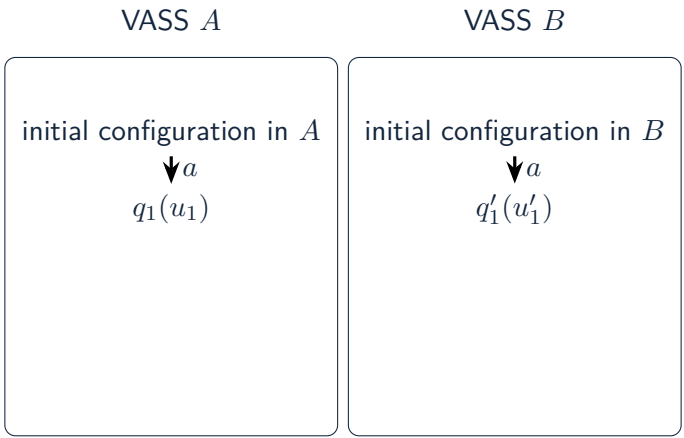
Simulation


VASS A
VASS B
initial configuration in A
initial configuration in B

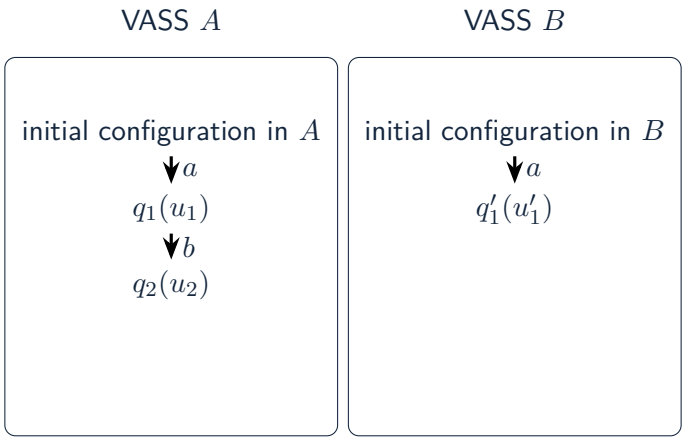
Simulation


VASS A
VASS B
initial configuration in A
a
$q_1(u_1)$
initial configuration in B

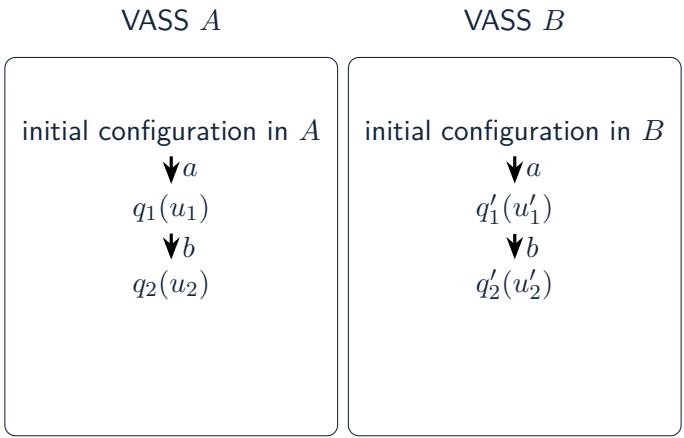
Simulation


VASS A
VASS B
initial configuration in A
a
$q_1(u_1)$
initial configuration in B
a
$q'_1(u'_1)$

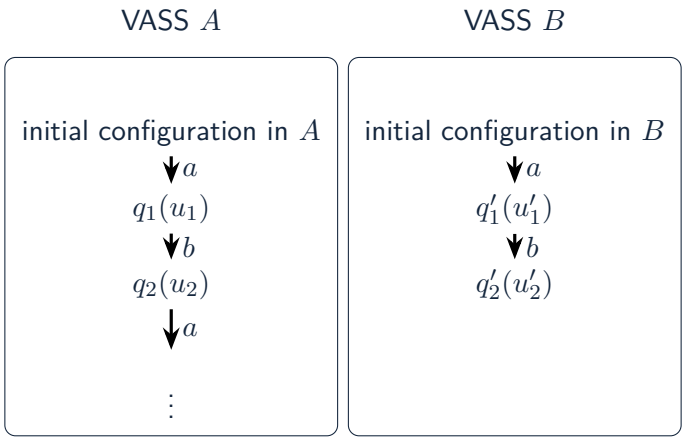
Simulation


VASS A
VASS B
initial configuration in A
a
$q_1(u_1)$
b
$q_2(u_2)$
initial configuration in B
a
$q_1'(u_1')$

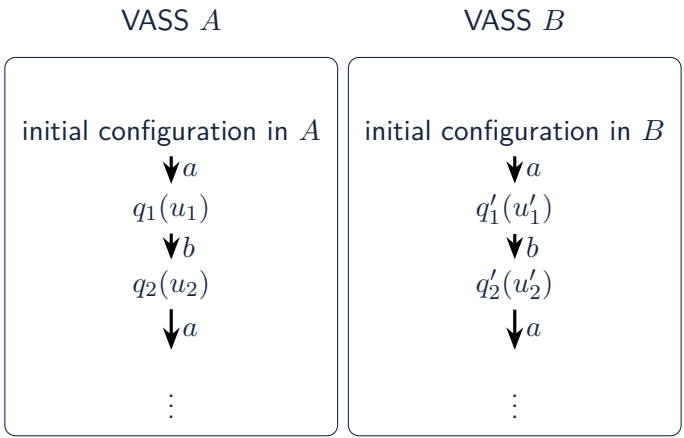
Simulation


VASS A
VASS B
initial configuration in A
a
$q_1(u_1)$
b
$q_2(u_2)$
initial configuration in B
a
$q_1'(u_1')$
b
$q_2'(u_2')$

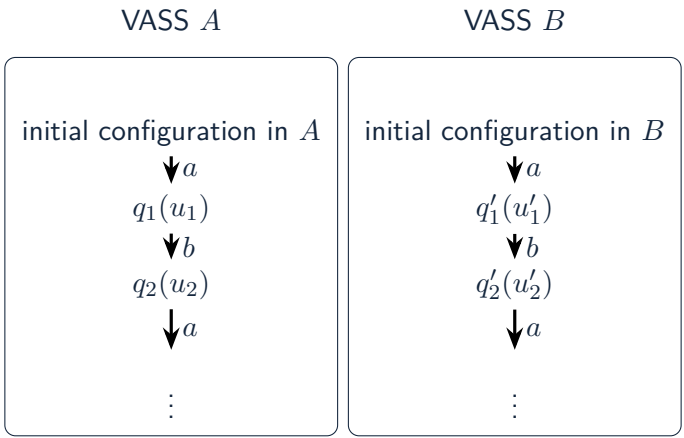
Simulation


VASS A
VASS B
initial configuration in A
a
$q_1(u_1)$
b
$q_2(u_2)$
a
$\vdots$
initial configuration in B
a
$q_1'(u_1')$
b
$q_2'(u_2')$

Simulation


VASS A
VASS B
initial configuration in A
a
$q_1(u_1)$
b
$q_2(u_2)$
a
initial configuration in B
a
$q_1'(u_1')$
b
$q_2'(u_2')$
a

Simulation



- VASSs A and B are in two-sided simulation if A simulates B and B simulates A

Corollary

- Observation: For all history-deterministic VASSs A and B, it holds that A and B are in two-sided simulation if and only if $L_T(A) = L_T(B)$.

Corollary

- Observation: For all history-deterministic VASSs A and B, it holds that A and B are in two-sided simulation if and only if $L_T(A) = L_T(B)$.

Theorem

For any equivalence relation R defined on set all of VASSs over alphabet Σ between two-sided simulation and trace language equivalence given deterministic 2-VASS A and history-deterministic 2-VASS B it is undecidable to determine whether $(A, B) \in R$.

Conclusions

- Language equivalence/two-sided simulation is undecidable for deterministic vs. history-deterministic 2-VASSs.

Conclusions

- Language equivalence/two-sided simulation is undecidable for deterministic vs. history-deterministic 2-VASSs.
- We cannot extend undecidability to bisimulation, as it is decidable when one of the VASSs is deterministic **(Jančar, 1995)**.

Conclusions

- Language equivalence/two-sided simulation is undecidable for deterministic vs. history-deterministic 2-VASSs.
- We cannot extend undecidability to bisimulation, as it is decidable when one of the VASSs is deterministic **(Jančar, 1995)**.
- Open question: what about bisimulation when one/both VASSs are history-deterministic?

Conclusions

- Language equivalence/two-sided simulation is undecidable for deterministic vs. history-deterministic 2-VASSs.
- We cannot extend undecidability to bisimulation, as it is decidable when one of the VASSs is deterministic **(Jančar, 1995)**.
- Open question: what about bisimulation when one/both VASSs are history-deterministic?

Thank you!

Questions?