Labelled Markov Processes
Lecture 1: Labelled Transition Systems

Prakash Panangaden\textsuperscript{1}

\textsuperscript{1}\textit{School of Computer Science}
\textit{McGill University}

January 2008, Winter School on Logic, IIT Kanpur
Outline

1. Introduction
2. Labelled transition systems
3. Bisimulation and Coinduction
4. Hennessy-Milner Logic
Outline

1. Introduction
2. Labelled transition systems
3. Bisimulation and Coinduction
4. Hennessy-Milner Logic
Outline

1. Introduction
2. Labelled transition systems
3. Bisimulation and Coinduction
4. Hennessy-Milner Logic
Overview

• Lecture 1: Labelled transition systems and bisimulation.
• Lecture 2: Labelled Markov processes.
• Lecture 3: Logical characterization of bisimulation.
• Lecture 4: The metric analogue of bisimulation.
Overview

- Lecture 1: Labelled transition systems and bisimulation.
- Lecture 2: Labelled Markov processes.
- Lecture 3: Logical characterization of bisimulation.
- Lecture 4: The metric analogue of bisimulation.
Overview

- Lecture 1: Labelled transition systems and bisimulation.
- Lecture 2: Labelled Markov processes.
- Lecture 3: Logical characterization of bisimulation.
- Lecture 4: The metric analogue of bisimulation.
Overview

- Lecture 1: Labelled transition systems and bisimulation.
- Lecture 2: Labelled Markov processes.
- Lecture 3: Logical characterization of bisimulation.
- Lecture 4: The metric analogue of bisimulation.
This lecture

- Labelled transition systems.
- Bisimulation.
- Making sense of coinduction.
- Games for bisimulation and simulation.
- Logical characterization.
This lecture

- Labelled transition systems.
- Bisimulation.
- Making sense of coinduction.
- Games for bisimulation and simulation.
- Logical characterization.
This lecture

- Labelled transition systems.
- Bisimulation.
- Making sense of coinduction.
- Games for bisimulation and simulation.
- Logical characterization.
This lecture

- Labelled transition systems.
- Bisimulation.
- Making sense of coinduction.
- Games for bisimulation and simulation.
- Logical characterization.
This lecture

- Labelled transition systems.
- Bisimulation.
- Making sense of coinduction.
- Games for bisimulation and simulation.
- Logical characterization.
Probabilistic bisimulation can be defined for continuous state-space systems. [LICS97]

- Logical characterization. [LICS98, Info and Comp 2002]
- Metric analogue of bisimulation. [CONCUR99, TCS2004]
- Approximation of LMPs. [LICS00, Info and Comp 2003]
- Weak bisimulation. [LICS02, CONCUR02]
- Real time. [QEST 2004, JLAP 2003, LMCS 2006]
Probabilistic bisimulation can be defined for continuous state-space systems. [LICS97]

Logical characterization. [LICS98, Info and Comp 2002]

Metric analogue of bisimulation. [CONCUR99, TCS2004]

Approximation of LMPs. [LICS00, Info and Comp 2003]

Weak bisimulation. [LICS02, CONCUR02]

Real time. [QEST 2004, JLAP 2003, LMCS 2006]
Summary of Results

- Probabilistic bisimulation can be defined for continuous state-space systems. [LICS97]
- Logical characterization. [LICS98, Info and Comp 2002]
- Metric analogue of bisimulation. [CONCUR99, TCS2004]
- Approximation of LMPs. [LICS00, Info and Comp 2003]
- Weak bisimulation. [LICS02, CONCUR02]
- Real time. [QEST 2004, JLAP 2003, LMCS 2006]
Probabilistic bisimulation can be defined for continuous state-space systems. [LICS97]

Logical characterization. [LICS98, Info and Comp 2002]

Metric analogue of bisimulation. [CONCUR99, TCS2004]

Approximation of LMPs. [LICS00, Info and Comp 2003]

Weak bisimulation. [LICS02, CONCUR02]

Real time. [QEST 2004, JLAP 2003, LMCS 2006]
Probabilistic bisimulation can be defined for continuous state-space systems. [LICS97]

Logical characterization. [LICS98, Info and Comp 2002]

Metric analogue of bisimulation. [CONCUR99, TCS2004]

Approximation of LMPs. [LICS00, Info and Comp 2003]

Weak bisimulation. [LICS02, CONCUR02]

Real time. [QEST 2004, JLAP 2003, LMCS 2006]
Probabilistic bisimulation can be defined for continuous state-space systems. [LICS97]

Logical characterization. [LICS98, Info and Comp 2002]

Metric analogue of bisimulation. [CONCUR99, TCS2004]

Approximation of LMPs. [LICS00, Info and Comp 2003]

Weak bisimulation. [LICS02, CONCUR02]

Real time. [QEST 2004, JLAP 2003, LMCS 2006]
Collaborators

- Josée Desharnais
- Radha Jagadeesan and Vineet Gupta
- Abbas Edalat
- Vincent Danos
Collaborators

- Josée Desharnais
- Radha Jagadeesan and Vineet Gupta
- Abbas Edalat
- Vincent Danos
Collaborators

- Josée Desharnais
- Radha Jagadeesan and Vineet Gupta
- Abbas Edalat
- Vincent Danos
Collaborators

- Josée Desharnais
- Radha Jagadeesan and Vineet Gupta
- Abbas Edalat
- Vincent Danos
The definition

- A set of states $S$,
- a set of labels or actions, $L$ or $A$ and
- a transition relation $\subseteq S \times A \times S$, usually written

$$\rightarrow_a \subseteq S \times S.$$ 

The transitions could be indeterminate (nondeterministic).
- We write $s \xrightarrow{a} s'$ for $(s, s') \in \rightarrow_a$. 

Panangaden  Labelled Markov Processes
The definition

- A set of states $S$,
- a set of labels or actions, $L$ or $A$ and
- a transition relation $\subseteq S \times A \times S$, usually written $\rightarrow_a \subseteq S \times S$.

The transitions could be indeterminate (nondeterministic).

- We write $s \xrightarrow{a} s'$ for $(s, s') \in \rightarrow_a$. 
A set of states $S$,
a set of *labels* or *actions*, $L$ or $A$ and
a transition relation $\subseteq S \times A \times S$, usually written

$$\rightarrow_a \subseteq S \times S.$$ 

The transitions could be indeterminate (nondeterministic).

We write $s \xrightarrow{a} s'$ for $(s, s') \in \rightarrow_a$. 
The definition

- A set of states $S$,
- a set of *labels* or *actions*, $L$ or $A$ and
- a transition relation $\subseteq S \times A \times S$, usually written

$$\rightarrow_a \subseteq S \times S.$$  

The transitions could be indeterminate (nondeterministic).
- We write $s \xrightarrow{a} s'$ for $(s, s') \in \rightarrow_a.$
A simple example

\[
\begin{align*}
A_1 & : S_0 & a & S_1 \\
& & a & S_2 \\
S_1 & b & S_3
\end{align*}
\]

\[
\begin{align*}
A_2 & : S_0 & a & S_1 \\
& & b & S_2 \\
& & c & S_4 \\
S_1 & c & S_3 \\
S_4 & a & S_3
\end{align*}
\]
A vending machine
Vending machine LTSs

![Diagram of a vending machine LTS with labels: Rs 5, Cof, Tea, Cup, and R.]
Another (?) vending machine LTSs
Are the two LTSs equivalent?

- One gives *us* the choice whereas the other makes the choice *internally*.
- The sequences that the machines can perform are identical: \([Rs.5; (Cof + Tea); Cup]^*\)
- We need to go beyond language equivalence.
Are the two LTSs equivalent?

- One gives us the choice whereas the other makes the choice internally.
- The sequences that the machines can perform are identical: \([Rs.5; (Cof + Tea); Cup]^*\)
- We need to go beyond language equivalence.
Are the two LTSs equivalent?

One gives *us* the choice whereas the other makes the choice *internally*.

The sequences that the machines can perform are identical: \([Rs.5; (Cof + Tea); Cup]^*\)

We need to go beyond language equivalence.
s and t are states of a labelled transition system. We say s is **bisimilar** to t – written $s \sim t$ – if

$$s \xrightarrow{a} s' \Rightarrow \exists t' \text{ such that } t \xrightarrow{a} t' \text{ and } s' \sim t'$$

and

$$t \xrightarrow{a} t' \Rightarrow \exists s' \text{ such that } s \xrightarrow{a} s' \text{ and } s' \sim t'.$$
Does it make sense?

- The definition of bisimilarity seems circular.
- In fact, it is perfectly well defined.
- There are three or four ways of explaining it.
Does it make sense?

- The definition of bisimilarity seems circular.
- In fact, it is perfectly well defined.
- There are three or four ways of explaining it.
Does it make sense?

- The definition of bisimilarity seems circular.
- In fact, it is perfectly well defined.
- There are three or four ways of explaining it.
Define a family of equivalence relations $\sim_n$ indexed by the natural numbers.

$\sim_0$ is the universal relation: $\forall s, t \ s \sim_0 t$.

$s \sim_{n+1} t$ if

$$\forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ and } s' \sim_n t'$$

and vice versa.

$s \sim t$ if and only if $\forall n, s \sim_n t$. 

Panangaden
Labelled Markov Processes
Coinduction via induction

- Define a *family* of equivalence relations $\sim_n$ indexed by the natural numbers.
- $\sim_0$ is the universal relation: $\forall s, t \ s \sim_0 t$.
- $s \sim_{n+1} t$ if
  
  $$\forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ and } s' \sim_n t'$$

  and vice versa.
- $s \sim t$ if and only if $\forall n, s \sim_n t$. 

Panangaden Labelled Markov Processes
Coinduction via induction

- Define a *family* of equivalence relations $\sim_n$ indexed by the natural numbers.
- $\sim_0$ is the universal relation: $\forall s, t \ s \sim_0 t$.
- $s \sim_{n+1} t$ if
  \[ \forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ and } s' \sim_n t' \]
  and vice versa.
- $s \sim t$ if and only if $\forall n, s \sim_n t$. 
Coinduction via induction

- Define a *family* of equivalence relations $\sim_n$ indexed by the natural numbers.
- $\sim_0$ is the universal relation: $\forall s, t \ s \sim_0 t$.
- $s \sim_{n+1} t$ if

\[\forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ and } s' \sim_n t'\]

and vice versa.
- $s \sim t$ if and only if $\forall n, s \sim_n t$. 

Panangaden

Labelled Markov Processes
Fix a labelled transition system with state space $S$.

Let $\mathcal{R}$ be the collection of equivalence relations on $S$ ordered by inclusion.

Define $\mathcal{F}: \mathcal{R} \rightarrow \mathcal{R}$ by

$$s\mathcal{F}(R)t \text{ means } \forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ and } s'Rt'$$

and vice versa.

$\mathcal{R}$ is a complete lattice partially ordered by inclusion and $\mathcal{F}$ is a monotone function.

It is a (moderately) easy exercise to show that $\mathcal{F}$ has a greatest fixed point: this is bisimulation.
Coinduction as a greatest fixed point

- Fix a labelled transition system with state space $S$.
- Let $\mathcal{R}$ be the collection of equivalence relations on $S$ ordered by inclusion.
- Define $\mathcal{F} : \mathcal{R} \rightarrow \mathcal{R}$ by

$$s \mathcal{F}(R)t \text{ means } \forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ and } s'Rt'$$

and vice versa.
- $\mathcal{R}$ is a complete lattice partially ordered by inclusion and $\mathcal{F}$ is a monotone function.
- It is a (moderately) easy exercise to show that $\mathcal{F}$ has a greatest fixed point: this is bisimulation.
Coinduction as a greatest fixed point

- Fix a labelled transition system with state space $S$.
- Let $\mathcal{R}$ be the collection of equivalence relations on $S$ ordered by inclusion.
- Define $\mathcal{F} : \mathcal{R} \to \mathcal{R}$ by

$$s \mathcal{F}(R) t \text{ means } \forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ and } s' R t'$$

and vice versa.

- $\mathcal{R}$ is a complete lattice partially ordered by inclusion and $\mathcal{F}$ is a monotone function.
- It is a (moderately) easy exercise to show that $\mathcal{F}$ has a greatest fixed point: this is bisimulation.
Coinduction as a greatest fixed point

- Fix a labelled transition system with state space $S$.
- Let $\mathcal{R}$ be the collection of equivalence relations on $S$ ordered by inclusion.
- Define $\mathcal{F} : \mathcal{R} \rightarrow \mathcal{R}$ by

$$s\mathcal{F}(R)t \text{ means } \forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ and } s'Rt'$$

and vice versa.
- $\mathcal{R}$ is a complete lattice partially ordered by inclusion and $\mathcal{F}$ is a monotone function.
- It is a (moderately) easy exercise to show that $\mathcal{F}$ has a greatest fixed point: this is bisimulation.
Coinduction as a greatest fixed point

- Fix a labelled transition system with state space $S$.
- Let $\mathcal{R}$ be the collection of equivalence relations on $S$ ordered by inclusion.
- Define $\mathcal{F}: \mathcal{R} \rightarrow \mathcal{R}$ by
  
  $$s \mathcal{F}(R) t \text{ means } \forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ and } s' R t'$$

  and vice versa.
- $\mathcal{R}$ is a complete lattice partially ordered by inclusion and $\mathcal{F}$ is a monotone function.
- It is a (moderately) easy exercise to show that $\mathcal{F}$ has a greatest fixed point: this is bisimulation.
Bisimulation relations

● Define a (note the indefinite article) bisimulation relation $R$ to be an equivalence relation on $S$ such that

$$sRt \text{ means } \forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ with } s'Rt'$$

and vice versa.

● This is not circular; it is a condition on $R$.

● We define $s \sim t$ if there is some bisimulation relation $R$ with $sRt$.

● This is the version that is used most often.
Define a (note the indefinite article) bisimulation relation $R$ to be an equivalence relation on $S$ such that

$sRt$ means $\forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t'$ with $s'Rt'$

and vice versa.

This is not circular; it is a condition on $R$.

We define $s \sim t$ if there is some bisimulation relation $R$ with $sRt$.

This is the version that is used most often.
Define a (note the indefinite article) bisimulation relation $R$ to be an equivalence relation on $S$ such that

$$sRt \text{ means } \forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ with } s' Rt'$$

and vice versa.

This is not circular; it is a condition on $R$.

We define $s \sim t$ if there is some bisimulation relation $R$ with $sRt$.

This is the version that is used most often.
Define a (note the indefinite article) bisimulation relation $R$ to be an equivalence relation on $S$ such that:

$sRt$ means $\forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t'$ with $s' Rt'$

and vice versa.

This is not circular; it is a condition on $R$.

We define $s \sim t$ if there is some bisimulation relation $R$ with $sRt$.

This is the version that is used most often.
Define a (note the indefinite article) bisimulation relation $R$ to be an equivalence relation on $S$ such that

$$sRt \text{ means } \forall a, s \xrightarrow{a} s' \Rightarrow \exists t', t \xrightarrow{a} t' \text{ with } s'Rt'$$

and vice versa.

This is not circular; it is a condition on $R$.

We define $s \sim t$ if there is some bisimulation relation $R$ with $sRt$.

This is the version that is used most often.
An example

Here $s_0$ and $t_0$ are not bisimilar.
However $s_0$ and $t_0$ can simulate each other!
An example

Here $s_0$ and $t_0$ are not bisimilar.

However $s_0$ and $t_0$ can simulate each other!
Here $s_0$ and $t_0$ are not bisimilar.
However $s_0$ and $t_0$ can *simulate* each other!
The bisimulation game

- Two players: maker (M) and spoiler (S). M wants to establish a bisimulation and S wants to spoil the bisimulation.
- S chooses a process with which to play and makes a move.
- M must match S’s move.
- S chooses again which process she wants to play and makes a move which M must match.
- If M has a winning strategy then the processes are bisimilar.
- If we did not allow S to switch after the first move then a winning strategy for M implies two-way simulation: much weaker than bisimulation.
Two players: maker (M) and spoiler (S). M wants to establish a bisimulation and S wants to spoil the bisimulation.

S chooses a process with which to play and makes a move.

M must match S’s move.

S chooses again which process she wants to play and makes a move which M must match.

If M has a winning strategy then the processes are bisimilar.

If we did not allow S to switch after the first move then a winning strategy for M implies two-way simulation: much weaker than bisimulation.
The bisimulation game

- Two players: maker (M) and spoiler (S). M wants to establish a bisimulation and S wants to spoil the bisimulation.
- S chooses a process with which to play and makes a move.
- M must match S’s move.
- S chooses again which process she wants to play and makes a move which M must match.
- If M has a winning strategy then the processes are bisimilar.
- If we did not allow S to switch after the first move then a winning strategy for M implies two-way simulation: much weaker than bisimulation.
The bisimulation game

- Two players: maker (M) and spoiler (S). M wants to establish a bisimulation and S wants to spoil the bisimulation.
- S chooses a process with which to play and makes a move.
- M must match S’s move.
- S chooses again which process she wants to play and makes a move which M must match.
- If M has a winning strategy then the processes are bisimilar.
- If we did not allow S to switch after the first move then a winning strategy for M implies two-way simulation: much weaker than bisimulation.
The bisimulation game

- Two players: maker (M) and spoiler (S). M wants to establish a bisimulation and S wants to spoil the bisimulation.
- S chooses a process with which to play and makes a move.
- M must match S’s move.
- S chooses again which process she wants to play and makes a move which M must match.
- If M has a winning strategy then the processes are bisimilar.
- If we did not allow S to switch after the first move then a winning strategy for M implies two-way simulation: much weaker than bisimulation.
The bisimulation game

- Two players: maker (M) and spoiler (S). M wants to establish a bisimulation and S wants to spoil the bisimulation.
- S chooses a process with which to play and makes a move.
- M must match S’s move.
- S chooses again which process she wants to play and makes a move which M must match.
- If M has a winning strategy then the processes are bisimilar.
- If we did not allow S to switch after the first move then a winning strategy for M implies two-way simulation: much weaker than bisimulation.
Define a logic as follows:

\[
\phi ::= T | \neg \phi | \phi_1 \land \phi_2 | \langle a \rangle \phi
\]

- \( s \models \langle a \rangle \phi \) means that \( s \xrightarrow{a} s' \) and \( t \models \phi \).
- We can define a dual to \( \langle \rangle \) (written \( [] \)) by using negation.
- \( s \models [a] \phi \) means that if \( s \) can do an \( a \) the resulting state must satisfy \( \phi \).

How do we know that two processes are **not** bisimilar?
How do we know that two processes are **not** bisimilar?

Define a logic as follows:

\[ \phi ::= T | \neg \phi | \phi_1 \land \phi_2 | \langle a \rangle \phi \]

- \( s \models \langle a \rangle \phi \) means that \( s \xrightarrow{a} s' \) and \( t \models \phi \).
- We can define a dual to \( \langle \rangle \) (written \( [] \)) by using negation.
- \( s \models [a] \phi \) means that if \( s \) can do an \( a \) the resulting state must satisfy \( \phi \).
How do we know that two processes are not bisimilar?

Define a logic as follows:

$$\phi ::= T | \neg \phi | \phi_1 \land \phi_2 | \langle a \rangle \phi$$

- $s \models \langle a \rangle \phi$ means that $s \xrightarrow{a} s'$ and $t \models \phi$.
- We can define a dual to $\langle \rangle$ (written $[]$) by using negation.
- $s \models [a] \phi$ means that if $s$ can do an $a$ the resulting state must satisfy $\phi$. 
Define a logic as follows:

\[ \phi ::= T \mid \neg \phi \mid \phi_1 \land \phi_2 \mid \langle a \rangle \phi \]

- \( s \models \langle a \rangle \phi \) means that \( s \xrightarrow{a} s' \) and \( t \models \phi \).
- We can define a dual to \( \langle \rangle \) (written \( [] \)) by using negation.
- \( s \models [a] \phi \) means that if \( s \) can do an \( a \) the resulting state must satisfy \( \phi \).
How do we know that two processes are not bisimilar?

Define a logic as follows:

\[ \phi ::= T | \neg \phi | \phi_1 \land \phi_2 | \langle a \rangle \phi \]

- \( s \models \langle a \rangle \phi \) means that \( s \xrightarrow{a} s' \) and \( t \models \phi \).
- We can define a dual to \( \langle \rangle \) (written \([\!]\)) by using negation.
- \( s \models [a] \phi \) means that if \( s \) can do an \( a \) the resulting state must satisfy \( \phi \).
Examples of HM Logic

- $T$ is satisfied by any process, $F$ is not satisfied by any process.
- $s \models \langle a \rangle T$ means $s$ can do an $a$ action.
- $s \models \neg \langle a \rangle \phi$ or $s \models [a]F$ means $s$ cannot do an $a$ action.
- $s \models \langle a \rangle (\langle b \rangle T)$ means that $s$ can do an $a$ and then do a $b$. 
Examples of HM Logic

- $T$ is satisfied by any process, $F$ is not satisfied by any process.
- $s \models ⟨a⟩T$ means $s$ can do an $a$ action.
- $s \models \neg ⟨a⟩\phi$ or $s \models [a]F$ means $s$ cannot do an $a$ action.
- $s \models ⟨a⟩(⟨b⟩T)$ means that $s$ can do an $a$ and then do a $b$. 
Examples of HM Logic

- \( T \) is satisfied by any process, \( F \) is not satisfied by any process.
- \( s \models \langle a \rangle T \) means \( s \) can do an \( a \) action.
- \( s \models \neg\langle a \rangle \phi \) or \( s \models [a]F \) means \( s \) cannot do an \( a \) action.
- \( s \models \langle a \rangle (\langle b \rangle T) \) means that \( s \) can do an \( a \) and then do a \( b \).
Examples of HM Logic

- $T$ is satisfied by any process, $F$ is not satisfied by any process.
- $s \models \langle a \rangle T$ means $s$ can do an $a$ action.
- $s \models \neg \langle a \rangle \phi$ or $s \models [a]F$ means $s$ cannot do an $a$ action.
- $s \models \langle a \rangle (\langle b \rangle T)$ means that $s$ can do an $a$ and then do a $b$. 
The Hennessy-Milner theorem

- Two processes are bisimilar if and only if they satisfy the same formulas of HM logic.
- Basic assumption: the processes are finitely-branching (otherwise you need infinitary conjunctions).
- To show that two processes are not bisimilar find a formula on which they disagree.
The Hennessy-Milner theorem

- Two processes are bisimilar if and only if they satisfy the same formulas of HM logic.
- Basic assumption: the processes are finitely-branching (otherwise you need infinitary conjunctions).
- To show that two processes are not bisimilar find a formula on which they disagree.
The Hennessy-Milner theorem

- Two processes are bisimilar if and only if they satisfy the same formulas of HM logic.
- Basic assumption: the processes are finitely-branching (otherwise you need infinitary conjunctions).
- To show that two processes are not bisimilar find a formula on which they disagree.
Our first example

Here \( s_0 \) and \( t_0 \) are not bisimilar.

\( s_0 \models \langle a \rangle (\neg \langle b \rangle T) \) but \( t_0 \) does not satisfy this formula.

\( t_0 \models \langle a \rangle (\langle b \rangle T \land \langle c \rangle T) \) but \( s_0 \) does not satisfy this.

The conjunction captures branching structure.
Our first example

Here $s_0$ and $t_0$ are not bisimilar.

- $s_0 \models \langle a \rangle (\neg \langle b \rangle T)$ but $t_0$ does not satisfy this formula.
- $t_0 \models \langle a \rangle (\langle b \rangle T \land \langle c \rangle T)$ but $s_0$ does not satisfy this.
- The conjunction captures branching structure.
Our first example

Here $s_0$ and $t_0$ are not bisimilar.
- $s_0 \models \langle a \rangle (\neg \langle b \rangle T)$ but $t_0$ does not satisfy this formula.
- $t_0 \models \langle a \rangle (\langle b \rangle T \land \langle c \rangle T)$ but $s_0$ does not satisfy this.
- The conjunction captures branching structure.
Our first example

Here $s_0$ and $t_0$ are not bisimilar.

$s_0 \models \langle a \rangle (\neg \langle b \rangle T)$ but $t_0$ does not satisfy this formula.

$t_0 \models \langle a \rangle (\langle b \rangle T \land \langle c \rangle T)$ but $s_0$ does not satisfy this.

The conjunction captures branching structure.
Our first example

Here $s_0$ and $t_0$ are not bisimilar.

$s_0 \models \langle a \rangle (\neg \langle b \rangle T)$ but $t_0$ does not satisfy this formula.

$t_0 \models \langle a \rangle (\langle b \rangle T \land \langle c \rangle T)$ but $s_0$ does not satisfy this.

The conjunction captures branching structure.
The role of negation

Consider the processes below:

\[ s_0 \models \langle a \rangle \neg \langle b \rangle \quad \text{but} \quad t_0 \quad \text{does not.} \]

\[ s_0 \quad \text{and} \quad t_0 \quad \text{agree on all formulas without negation.} \]

\[ \text{Note that} \quad [a] \text{ has an implicit negation.} \]
The role of negation

Consider the processes below:

\[ s_0 \models \langle a \rangle \neg \langle b \rangle T \] but \( t_0 \) does not.

\( s_0 \) and \( t_0 \) agree on all formulas without negation.

Note that \([a]\) has an implicit negation.
The role of negation

Consider the processes below:

\[ s_0 \models \langle a \rangle \neg \langle b \rangle T \] but \( t_0 \) does not.

\( s_0 \) and \( t_0 \) agree on all formulas \textit{without negation}.

Note that \([a]\) has an implicit negation.
The role of negation

Consider the processes below:

\[
\begin{align*}
  s_0 & \xrightarrow{a} s_1 \xrightarrow{a} s_2 \xrightarrow{b} s_3 \\
  t_0 & \xrightarrow{a} t_2 \xrightarrow{b} t_3
\end{align*}
\]

- \( s_0 \models \langle a \rangle \neg \langle b \rangle T \) but \( t_0 \) does not.
- \( s_0 \) and \( t_0 \) agree on all formulas \textit{without negation}.
- Note that [\( a \)] has an implicit negation.
Simulation can be defined by dropping the “vice versas” in the definition of bisimulation.

We would like a theorem of the form: if $s$ simulates $t$ then every formula that $t$ satisfies is also satisfied by $s$.

There cannot be a logical characterization of simulation as long as there is negation.
Simulation can be defined by dropping the “vice versas” in the definition of bisimulation.

We would like a theorem of the form: if $s$ simulates $t$ then every formula that $t$ satisfies is also satisfied by $s$.

There cannot be a logical characterization of simulation as long as there is negation.
Simulation can be defined by dropping the “vice versas” in the definition of bisimulation.

We would like a theorem of the form: if \( s \) simulates \( t \) then every formula that \( t \) satisfies is also satisfied by \( s \).

There cannot be a logical characterization of simulation as long as there is negation.
We do everything probabilistically.