A Fully Symbolic Bisimulation Algorithm

Department of Computer Science
University of California at Riverside

5th Workshop on Reachability Problems

Outline

Abstract

- Problem: bisimulation for asynchronous discrete-event models
- Limited to: systems with deterministic transitions
- Converted to reachability problem.
- Apply saturation heuristic
- For large quotient spaces: fastest symbolic bisimulation algorithm
- Bisimulation problem symbolically solvable for large quotient spaces, for deterministic systems with strong event locality.

Abstract

- Problem: bisimulation for asynchronous discrete-event models
- Limited to: systems with deterministic transitions
- Converted to reachability problem.
- Apply saturation heuristic
- For large quotient spaces: fastest symbolic bisimulation algorithm
- Bisimulation problem symbolically solvable for large quotient spaces, for deterministic systems with strong event locality.

Abstract

- Problem: bisimulation for asynchronous discrete-event models
- Limited to: systems with deterministic transitions
- Converted to reachability problem.
- Apply saturation heuristic
- For large quotient spaces: fastest symbolic bisimulation algorithm
- Bisimulation problem symbolically solvable for large quotient spaces, for deterministic systems with strong event locality.

Abstract

- Problem: bisimulation for asynchronous discrete-event models
- Limited to: systems with deterministic transitions
- Converted to reachability problem.
- Apply saturation heuristic
- For large quotient spaces: fastest symbolic bisimulation algorithm
- Bisimulation problem symbolically solvable for large quotient spaces, for deterministic systems with strong event locality.

Abstract

- Problem: bisimulation for asynchronous discrete-event models
- Limited to: systems with deterministic transitions
- Converted to reachability problem.
- Apply saturation heuristic
- For large quotient spaces: fastest symbolic bisimulation algorithm
- Bisimulation problem symbolically solvable for large quotient spaces, for deterministic systems with strong event locality.

Abstract

- Problem: bisimulation for asynchronous discrete-event models
- Limited to: systems with deterministic transitions
- Converted to reachability problem.
- Apply saturation heuristic
- For large quotient spaces: fastest symbolic bisimulation algorithm
- Bisimulation problem symbolically solvable for large quotient spaces, for deterministic systems with strong event locality.

Outline

Outline

Transition Systems deterministic, colored, labeled (discrete),(DCLTS)

Definition

- A DCLTS (*deterministic colored labeled transition system*) is a tuple $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$, where:
- $\mathcal{S}$ set of *states*
- $\mathcal{S}_{init} \subseteq \mathcal{S}$ *initial states*
- $\mathcal{E}$ set of *events* (transition labels)
- $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{E} \times \mathcal{S}$ labeled partial *transitions*
  $\mathcal{T}_{\alpha} : \mathcal{S} \nrightarrow \mathcal{S}$
  $\langle s_1, \alpha, s_2 \rangle \in \mathcal{T}_{\mathcal{E}}$ also written: $s_1 \xrightarrow{\alpha} s_2$
- $\mathcal{C}$ set of colors
- $c : \mathcal{S} \to \mathcal{C}$ state coloring

Transition Systems deterministic, colored, labeled (discrete),(DCLTS)

Definition

- A DCLTS (*deterministic colored labeled transition system*) is a tuple $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$, where:
- $\mathcal{S}$ — set of *states*
- $\mathcal{S}_{init} \subseteq \mathcal{S}$ — *initial states*
- $\mathcal{E}$ — set of *events* (transition labels)
- $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{E} \times \mathcal{S}$ — labeled partial *transitions*
  $\mathcal{T}_{\alpha} : \mathcal{S} \nrightarrow \mathcal{S}$
  $\langle s_1, \alpha, s_2 \rangle \in \mathcal{T}_{\mathcal{E}}$ also written: $s_1 \xrightarrow{\alpha} s_2$
- $\mathcal{C}$ — set of colors
- $c : \mathcal{S} \to \mathcal{C}$ — state coloring

Transition Systems deterministic, colored, labeled (discrete),(DCLTS)

Definition

- A DCLTS (*deterministic colored labeled transition system*) is a tuple $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$, where:
- $\mathcal{S}$ — set of *states*
- $\mathcal{S}_{init} \subseteq \mathcal{S}$ — *initial states*
- $\mathcal{E}$ — set of *events* (transition labels)
- $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{E} \times \mathcal{S}$ — labeled partial *transitions*
  $\mathcal{T}_\alpha : \mathcal{S} \nrightarrow \mathcal{S}$
  $\langle s_1, \alpha, s_2 \rangle \in \mathcal{T}_{\mathcal{E}}$ also written: $s_1 \xrightarrow{\alpha} s_2$
- $\mathcal{C}$ — set of colors
- $c : \mathcal{S} \to \mathcal{C}$ — state coloring

Transition Systems deterministic, colored, labeled (discrete),(DCLTS)

Definition

- A DCLTS (*deterministic colored labeled transition system*) is a tuple $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$, where:
- $\mathcal{S}$ set of *states*
- $\mathcal{S}_{init} \subseteq \mathcal{S}$ *initial states*
- $\mathcal{E}$ set of *events* (transition labels)
- $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{E} \times \mathcal{S}$ labeled partial *transitions*

  $\mathcal{T}_{\alpha} : \mathcal{S} \nrightarrow \mathcal{S}$

  $\langle s_1, \alpha, s_2 \rangle \in \mathcal{T}_{\mathcal{E}}$ also written: $s_1 \xrightarrow{\alpha} s_2$
- $\mathcal{C}$ set of colors
- $c : \mathcal{S} \to \mathcal{C}$ state coloring

Transition Systems deterministic, colored, labeled (discrete),(DCLTS)

Definition

- A DCLTS (*deterministic colored labeled transition system*) is a tuple $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$, where:
- $\mathcal{S}$ set of *states*
- $\mathcal{S}_{init} \subseteq \mathcal{S}$ *initial states*
- $\mathcal{E}$ set of *events* (transition labels)
- $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{E} \times \mathcal{S}$ labeled partial *transitions*
  $\mathcal{T}_{\alpha} : \mathcal{S} \nrightarrow \mathcal{S}$
  $\langle s_1, \alpha, s_2 \rangle \in \mathcal{T}_{\mathcal{E}}$ also written: $s_1 \xrightarrow{\alpha} s_2$
- $\mathcal{C}$ set of colors
- $c : \mathcal{S} \to \mathcal{C}$ state coloring

Transition Systems deterministic, colored, labeled (discrete),(DCLTS)

Definition

- A DCLTS (*deterministic colored labeled transition system*) is a tuple $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$, where:
- $\mathcal{S}$ — set of *states*
- $\mathcal{S}_{init} \subseteq \mathcal{S}$ — *initial states*
- $\mathcal{E}$ — set of *events* (transition labels)
- $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{E} \times \mathcal{S}$ — labeled partial *transitions*
  $\mathcal{T}_\alpha : \mathcal{S} \nrightarrow \mathcal{S}$
  $\langle s_1, \alpha, s_2 \rangle \in \mathcal{T}_{\mathcal{E}}$ also written: $s_1 \xrightarrow{\alpha} s_2$
- $\mathcal{C}$ — set of colors
- $c : \mathcal{S} \to \mathcal{C}$ — state coloring

Transition Systems deterministic, colored, labeled (discrete),(DCLTS)

Definition

- A DCLTS (*deterministic colored labeled transition system*) is a tuple $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$, where:
- $\mathcal{S}$ set of *states*
- $\mathcal{S}_{init} \subseteq \mathcal{S}$ *initial states*
- $\mathcal{E}$ set of *events* (transition labels)
- $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{E} \times \mathcal{S}$ labeled partial *transitions*
  $\mathcal{T}_\alpha : \mathcal{S} \nrightarrow \mathcal{S}$
  $\langle s_1, \alpha, s_2 \rangle \in \mathcal{T}_{\mathcal{E}}$ also written: $s_1 \xrightarrow{\alpha} s_2$
- $\mathcal{C}$ set of colors
- $c : \mathcal{S} \to \mathcal{C}$ state coloring

The (largest) bisimulation $\sim$

Definition

- Given a DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$, $\mathcal{S}_{init} \subseteq \mathcal{S}$, $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{E} \times \mathcal{S}$, $\mathcal{T}_{\alpha} : \mathcal{S} \nrightarrow \mathcal{S}$, $c : \mathcal{S} \to \mathcal{C}$
- $\sim$ largest equivalence relation $\mathcal{B} \subseteq \mathcal{S} \times \mathcal{S}$ where:
- Each pair in $\mathcal{B}$ has the same color, and $\forall \langle p, q \rangle \in \mathcal{B}$: $c(p) = c(q)$
- also has matching transitions to pairs in $\mathcal{B}$ $\forall \langle p, q \rangle \in \mathcal{B}$: $\forall \alpha \in \mathcal{E}, \forall p' \in \mathcal{S}, (p \xrightarrow{\alpha} p' \Rightarrow \exists q' \in \mathcal{S}, q \xrightarrow{\alpha} q' \wedge \langle p', q' \rangle \in \mathcal{B})$

The (largest) bisimulation $\sim$

Definition

- Given a DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$, $\mathcal{S}_{init} \subseteq \mathcal{S}$, $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{E} \times \mathcal{S}$, $\mathcal{T}_{\alpha} : \mathcal{S} \nrightarrow \mathcal{S}$, $c : \mathcal{S} \to \mathcal{C}$
- $\sim$ largest equivalence relation $\mathcal{B} \subseteq \mathcal{S} \times \mathcal{S}$ where:
- Each pair in $\mathcal{B}$ has the same color, and $\forall \langle p, q \rangle \in \mathcal{B}$: $c(p) = c(q)$
- also has matching transitions to pairs in $\mathcal{B}$ $\forall \langle p, q \rangle \in \mathcal{B}$: $\forall \alpha \in \mathcal{E}, \forall p' \in \mathcal{S}, (p \xrightarrow{\alpha} p' \Rightarrow \exists q' \in \mathcal{S}, q \xrightarrow{\alpha} q' \wedge \langle p', q' \rangle \in \mathcal{B})$

The (largest) bisimulation $\sim$

Definition

- Given a DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$, $\mathcal{S}_{init} \subseteq \mathcal{S}$, $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{E} \times \mathcal{S}$, $\mathcal{T}_{\alpha} : \mathcal{S} \nrightarrow \mathcal{S}$, $c : \mathcal{S} \to \mathcal{C}$
- $\sim$ largest equivalence relation $\mathcal{B} \subseteq \mathcal{S} \times \mathcal{S}$ where:
- Each pair in $\mathcal{B}$ has the same color, and $\forall \langle p, q \rangle \in \mathcal{B}$: $c(p) = c(q)$
- also has matching transitions to pairs in $\mathcal{B}$ $\forall \langle p, q \rangle \in \mathcal{B}$: $\forall \alpha \in \mathcal{E}, \forall p' \in \mathcal{S}, (p \xrightarrow{\alpha} p' \Rightarrow \exists q' \in \mathcal{S}, q \xrightarrow{\alpha} q' \wedge \langle p', q' \rangle \in \mathcal{B})$

The (largest) bisimulation $\sim$

Definition

- Given a DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$, $\mathcal{S}_{init} \subseteq \mathcal{S}$, $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{E} \times \mathcal{S}$, $\mathcal{T}_\alpha : \mathcal{S} \nrightarrow \mathcal{S}$, $c : \mathcal{S} \to \mathcal{C}$
- $\sim$ largest equivalence relation $\mathcal{B} \subseteq \mathcal{S} \times \mathcal{S}$ where:
- Each pair in $\mathcal{B}$ has the same color, and $\forall \langle p, q \rangle \in \mathcal{B}$: $c(p) = c(q)$
- also has matching transitions to pairs in $\mathcal{B}$ $\forall \langle p, q \rangle \in \mathcal{B}$: $\forall \alpha \in \mathcal{E}, \forall p' \in \mathcal{S}, (p \xrightarrow{\alpha} p' \Rightarrow \exists q' \in \mathcal{S}, q \xrightarrow{\alpha} q' \wedge \langle p', q' \rangle \in \mathcal{B})$

Matching Transitions to Pairs in $\mathcal{B}$.

$\forall \langle p, q \rangle \in \mathcal{B}$:
$\forall \alpha \in \mathcal{E}, \forall p' \in \mathcal{S}, (p \xrightarrow{\alpha} p' \Rightarrow \exists q' \in \mathcal{S}, q \xrightarrow{\alpha} q' \wedge \langle p', q' \rangle \in \mathcal{B})$

$$
\begin{array}{ccc} p & \sim & q \\ \alpha \downarrow & & \\ p' & & \end{array}
\quad \Longrightarrow \quad
\begin{array}{ccc} p & \sim & q \\ \alpha \downarrow & & \downarrow \alpha \\ p' & \sim & q' \end{array}
$$

Bisimulation facts

- Extensional notion of equivalence of states in automata
- $O(m \log n)$ time algorithm (Paige & Tarjan '87)
- $O(m)$ time for single function coarsest partition (& Bonic '85)
- no cycles $\rightarrow O(m)$
- Explicit: data structures $\leftrightarrow$ edges, nodes
- Symbolic: Decision Diagrams $\leftrightarrow$ tuplesets
  - Bouali and Di Simone '92
  - Dovier, Piazza, and Policriti '04
  - Wimmer, Herbstritt, and Becker '07
  - This paper '11

Bisimulation facts

- Extensional notion of equivalence of states in automata
- O($m \log n$) time algorithm (Paige & Tarjan ’87)
- O(m) time for single function coarsest partition (& Bonic ’85)
- no cycles $\rightarrow$ O(m)
- Explicit: data structures $\leftrightarrow$ edges, nodes
- Symbolic: Decision Diagrams $\leftrightarrow$ tuplesets
  - Bouali and Di Simone ’92
  - Dovier, Piazza, and Policriti ’04
  - Wimmer, Herbstritt, and Becker ’07
  - This paper ’11

Bisimulation facts

- Extensional notion of equivalence of states in automata
- $O(m \log n)$ time algorithm (Paige & Tarjan '87)
- $O(m)$ time for single function coarsest partition (& Bonic '85)
- no cycles $\rightarrow O(m)$
- Explicit: data structures $\leftrightarrow$ edges, nodes
- Symbolic: Decision Diagrams $\leftrightarrow$ tuplesets
  - Bouali and Di Simone '92
  - Dovier, Piazza, and Policriti '04
  - Wimmer, Herbstritt, and Becker '07
  - This paper '11

Bisimulation facts

- Extensional notion of equivalence of states in automata
- $O(m \log n)$ time algorithm (Paige & Tarjan '87)
- $O(m)$ time for single function coarsest partition (& Bonic '85)
- no cycles $\rightarrow O(m)$
- Explicit: data structures $\leftrightarrow$ edges, nodes
- Symbolic: Decision Diagrams $\leftrightarrow$ tuplesets
  - Bouali and Di Simone '92
  - Dovier, Piazza, and Policriti '04
  - Wimmer, Herbstritt, and Becker '07
  - This paper '11

Bisimulation facts

- Extensional notion of equivalence of states in automata
- O($m \log n$) time algorithm (Paige & Tarjan '87)
- O(m) time for single function coarsest partition (& Bonic '85)
- no cycles $\rightarrow$ O(m)
- Explicit: data structures $\leftrightarrow$ edges, nodes
- Symbolic: Decision Diagrams $\leftrightarrow$ tuplesets
  - Bouali and Di Simone '92
  - Dovier, Piazza, and Policriti '04
  - Wimmer, Herbstritt, and Becker '07
  - This paper '11

Bisimulation facts

- Extensional notion of equivalence of states in automata
- $O(m \log n)$ time algorithm (Paige & Tarjan '87)
- $O(m)$ time for single function coarsest partition (& Bonic '85)
- no cycles $\rightarrow O(m)$
- Explicit: data structures $\leftrightarrow$ edges, nodes
- Symbolic: Decision Diagrams $\leftrightarrow$ tuplesets
  - Bouali and Di Simone '92
  - Dovier, Piazza, and Policriti '04
  - Wimmer, Herbstritt, and Becker '07
  - This paper '11

Bisimulation facts

- Extensional notion of equivalence of states in automata
- $O(m \log n)$ time algorithm (Paige & Tarjan '87)
- $O(m)$ time for single function coarsest partition (& Bonic '85)
- no cycles $\rightarrow O(m)$
- Explicit: data structures $\leftrightarrow$ edges, nodes
- Symbolic: Decision Diagrams $\leftrightarrow$ tuplesets
  - Bouali and Di Simone '92
  - Dovier, Piazza, and Policriti '04
  - Wimmer, Herbstritt, and Becker '07
  - This paper '11

Bisimulation facts

- Extensional notion of equivalence of states in automata
- O(m log n) time algorithm (Paige & Tarjan '87)
- O(m) time for single function coarsest partition (& Bonic '85)
- no cycles $\rightarrow$ O(m)
- Explicit: data structures $\leftrightarrow$ edges, nodes
- Symbolic: Decision Diagrams $\leftrightarrow$ tuplesets
    - Bouali and Di Simone '92
    - Dovier, Piazza, and Policriti '04
    - Wimmer, Herbstritt, and Becker '07
    - This paper '11

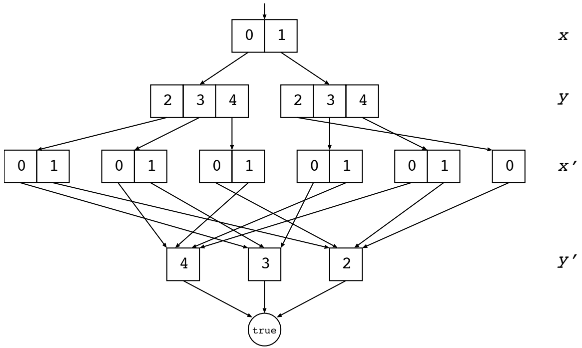
Bisimulation facts

- Extensional notion of equivalence of states in automata
- $O(m \log n)$ time algorithm (Paige & Tarjan '87)
- $O(m)$ time for single function coarsest partition (& Bonic '85)
- no cycles $\rightarrow O(m)$
- Explicit: data structures $\leftrightarrow$ edges, nodes
- Symbolic: Decision Diagrams $\leftrightarrow$ tuplesets
  - Bouali and Di Simone '92
  - Dovier, Piazza, and Policriti '04
  - Wimmer, Herbstritt, and Becker '07
  - This paper '11

Bisimulation facts

- Extensional notion of equivalence of states in automata
- $O(m \log n)$ time algorithm (Paige & Tarjan '87)
- $O(m)$ time for single function coarsest partition (& Bonic '85)
- no cycles $\rightarrow O(m)$
- Explicit: data structures $\leftrightarrow$ edges, nodes
- Symbolic: Decision Diagrams $\leftrightarrow$ tuplesets
  - Bouali and Di Simone '92
  - Dovier, Piazza, and Policriti '04
  - Wimmer, Herbstritt, and Becker '07
  - This paper '11

Quasi-Reduced Multi-Way Decision Diagrams QMDDs



x	y	x'	y'
0	2	0	3
0	2	1	2
0	3	0	4
0	3	1	3
0	4	0	2
0	4	1	4
1	2	0	2
1	3	0	3
1	3	1	4
1	4	0	4
1	4	1	2

QMDDs Represent Relations (Tuple-Sets)

- Each path in MDD (graph) corresponds to tuple in relation.
- Canonical: sharing $\leftrightarrow$ compression, comparison, *unique* table, non-mutable.
- Set operations implemented in SMART MDD library.
- Efficient memoized recursive algorithms for set operations: ($|()|, \cup, \cap, \setminus, \subseteq$).
- Efficient memoized recursive algorithms for functional operations: ($\exists, \forall, \times, \circ, ()^{-1}$).
- SMART Saturation-based state-space generation.
- Variable ordering matters.

QMDDs Represent Relations (Tuple-Sets)

- Each path in MDD (graph) corresponds to tuple in relation.
- Canonical: sharing $\leftrightarrow$ compression, comparison, *unique* table, non-mutable.
- Set operations implemented in SMART MDD library.
- Efficient memoized recursive algorithms for set operations: ($|()|$, $\cup$, $\cap$, $\setminus$, $\subseteq$).
- Efficient memoized recursive algorithms for functional operations: ($\exists$, $\forall$, $\times$, $\circ$, $()^{-1}$).
- SMART Saturation-based state-space generation.
- Variable ordering matters.

QMDDs Represent Relations (Tuple-Sets)

- Each path in MDD (graph) corresponds to tuple in relation.
- Canonical: sharing $\leftrightarrow$ compression, comparison, *unique* table, non-mutable.
- Set operations implemented in SMART MDD library.
- Efficient memoized recursive algorithms for set operations: ($|()|$, $\cup$, $\cap$, $\setminus$, $\subseteq$).
- Efficient memoized recursive algorithms for functional operations: ($\exists$, $\forall$, $\times$, $\circ$, $()^{-1}$).
- SMART Saturation-based state-space generation.
- Variable ordering matters.

QMDDs Represent Relations (Tuple-Sets)

- Each path in MDD (graph) corresponds to tuple in relation.
- Canonical: sharing $\leftrightarrow$ compression, comparison, *unique* table, non-mutable.
- Set operations implemented in SMART MDD library.
- Efficient memoized recursive algorithms for set operations: ($|()|$, $\cup$, $\cap$, $\setminus$, $\subseteq$).
- Efficient memoized recursive algorithms for functional operations: ($\exists$, $\forall$, $\times$, $\circ$, $()^{-1}$).
- SMART Saturation-based state-space generation.
- Variable ordering matters.

QMDDs Represent Relations (Tuple-Sets)

- Each path in MDD (graph) corresponds to tuple in relation.
- Canonical: sharing $\leftrightarrow$ compression, comparison, *unique* table, non-mutable.
- Set operations implemented in SMART MDD library.
- Efficient memoized recursive algorithms for set operations: ($|()|$, $\cup$, $\cap$, $\setminus$, $\subseteq$).
- Efficient memoized recursive algorithms for functional operations: ($\exists$, $\forall$, $\times$, $\circ$, $()^{-1}$).
- SMART Saturation-based state-space generation.
- Variable ordering matters.

QMDDs Represent Relations (Tuple-Sets)

- Each path in MDD (graph) corresponds to tuple in relation.
- Canonical: sharing $\leftrightarrow$ compression, comparison, *unique* table, non-mutable.
- Set operations implemented in SMART MDD library.
- Efficient memoized recursive algorithms for set operations: ($|()|$, $\cup$, $\cap$, $\setminus$, $\subseteq$).
- Efficient memoized recursive algorithms for functional operations: ($\exists$, $\forall$, $\times$, $\circ$, $()^{-1}$).
- SMART Saturation-based state-space generation.
- Variable ordering matters.

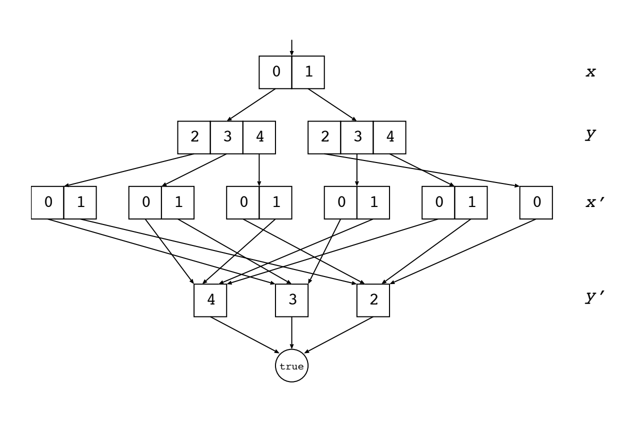
QMDDs Represent Relations (Tuple-Sets)

- Each path in MDD (graph) corresponds to tuple in relation.
- Canonical: sharing $\leftrightarrow$ compression, comparison, *unique* table, non-mutable.
- Set operations implemented in SMART MDD library.
- Efficient memoized recursive algorithms for set operations: ($|()|$, $\cup$, $\cap$, $\setminus$, $\subseteq$).
- Efficient memoized recursive algorithms for functional operations: ($\exists$, $\forall$, $\times$, $\circ$, $()^{-1}$).
- SMART Saturation-based state-space generation.
- Variable ordering matters.

QMDDs Represent Relations (Tuple-Sets)

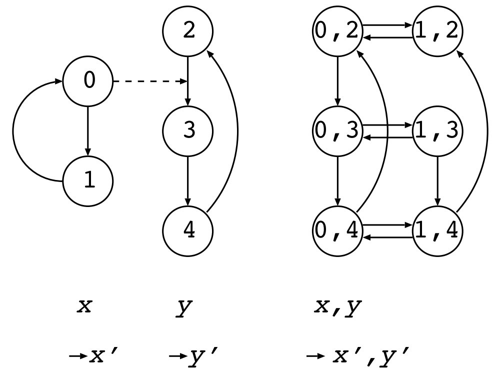
- Each path in MDD (graph) corresponds to tuple in relation.
- Canonical: sharing $\leftrightarrow$ compression, comparison, *unique* table, non-mutable.
- Set operations implemented in SMART MDD library.
- Efficient memoized recursive algorithms for set operations: ($|()|$, $\cup$, $\cap$, $\setminus$, $\subseteq$).
- Efficient memoized recursive algorithms for functional operations: ($\exists$, $\forall$, $\times$, $\circ$, $()^{-1}$).
- SMART Saturation-based state-space generation.
- Variable ordering matters.

Variable Ordering Matters (1)



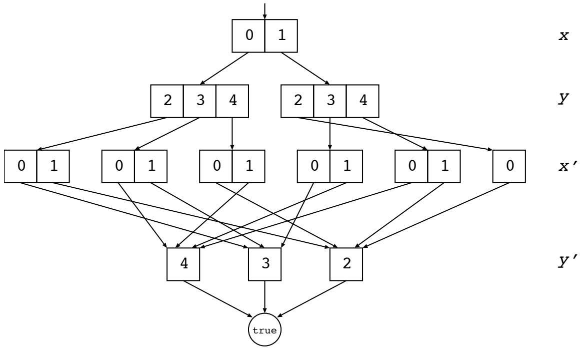
x	y	x'	y'
0	2	0	3
0	2	1	2
0	3	0	4
0	3	1	3
0	4	0	2
0	4	1	4
1	2	0	2
1	3	0	3
1	3	1	4
1	4	0	4
1	4	1	2

Variable Ordering Matters (2)



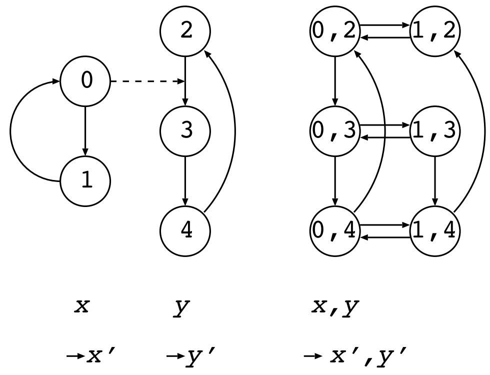
$\langle x, y\rangle \to \langle x', y'\rangle$
$\langle 0,2\rangle \to \langle 0,3\rangle$
$\langle 0,2\rangle \to \langle 1,2\rangle$
$\langle 0,3\rangle \to \langle 0,4\rangle$
$\langle 0,3\rangle \to \langle 1,3\rangle$
$\langle 0,4\rangle \to \langle 0,2\rangle$
$\langle 0,4\rangle \to \langle 1,4\rangle$
$\langle 1,2\rangle \to \langle 0,2\rangle$
$\langle 1,3\rangle \to \langle 0,3\rangle$
$\langle 1,3\rangle \to \langle 1,4\rangle$
$\langle 1,4\rangle \to \langle 0,4\rangle$
$\langle 1,4\rangle \to \langle 1,2\rangle$

Variable Ordering Matters (3)



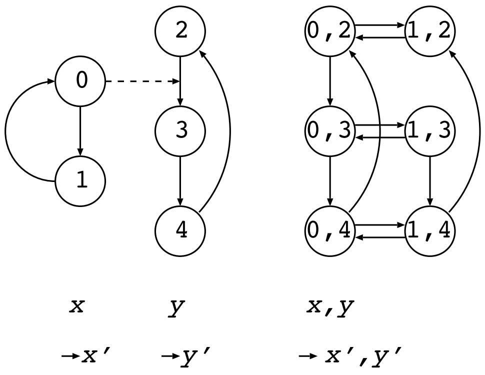
x	y	x'	y'
0	2	0	3
0	2	1	2
0	3	0	4
0	3	1	3
0	4	0	2
0	4	1	4
1	2	0	2
1	3	0	3
1	3	1	4
1	4	0	4
1	4	1	2

Variable Ordering Matters (4)



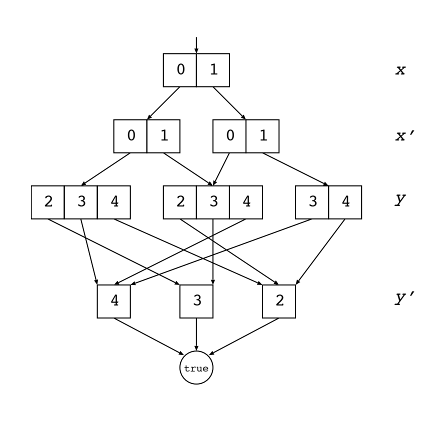
$\langle x\rangle \to \langle x'\rangle, \langle y\rangle \to \langle y'\rangle$
$\langle 0\rangle \to \langle 0\rangle, \langle 2\rangle \to \langle 3\rangle$
$\langle 0\rangle \to \langle 1\rangle, \langle 2\rangle \to \langle 2\rangle$
$\langle 0\rangle \to \langle 0\rangle, \langle 3\rangle \to \langle 4\rangle$
$\langle 0\rangle \to \langle 1\rangle, \langle 3\rangle \to \langle 3\rangle$
$\langle 0\rangle \to \langle 0\rangle, \langle 4\rangle \to \langle 2\rangle$
$\langle 0\rangle \to \langle 1\rangle, \langle 4\rangle \to \langle 4\rangle$
$\langle 1\rangle \to \langle 0\rangle, \langle 2\rangle \to \langle 2\rangle$
$\langle 1\rangle \to \langle 0\rangle, \langle 3\rangle \to \langle 3\rangle$
$\langle 1\rangle \to \langle 1\rangle, \langle 3\rangle \to \langle 4\rangle$
$\langle 1\rangle \to \langle 0\rangle, \langle 4\rangle \to \langle 4\rangle$
$\langle 1\rangle \to \langle 1\rangle, \langle 4\rangle \to \langle 2\rangle$

Variable Ordering Matters (5)



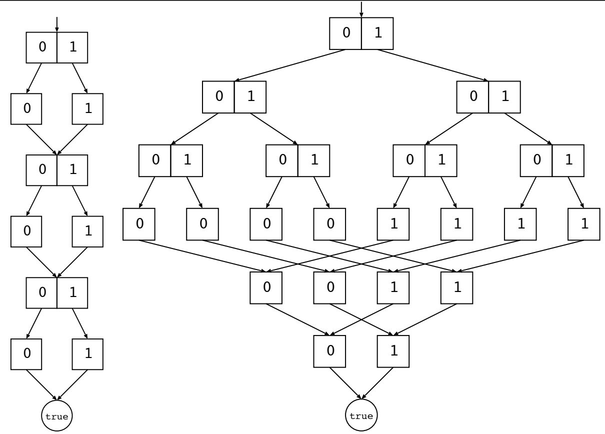
$\langle x\rangle \to \langle x'\rangle$,	$\langle y\rangle \to \langle y'\rangle$
$\langle 0\rangle \to \langle 0\rangle$,	$\langle 2\rangle \to \langle 3\rangle$
$\langle 0\rangle \to \langle 0\rangle$,	$\langle 3\rangle \to \langle 4\rangle$
$\langle 0\rangle \to \langle 0\rangle$,	$\langle 4\rangle \to \langle 2\rangle$
$\langle 0\rangle \to \langle 1\rangle$,	$\langle 2\rangle \to \langle 2\rangle$
$\langle 0\rangle \to \langle 1\rangle$,	$\langle 3\rangle \to \langle 3\rangle$
$\langle 0\rangle \to \langle 1\rangle$,	$\langle 4\rangle \to \langle 4\rangle$
$\langle 1\rangle \to \langle 0\rangle$,	$\langle 2\rangle \to \langle 2\rangle$
$\langle 1\rangle \to \langle 0\rangle$,	$\langle 3\rangle \to \langle 3\rangle$
$\langle 1\rangle \to \langle 0\rangle$,	$\langle 4\rangle \to \langle 4\rangle$
$\langle 1\rangle \to \langle 1\rangle$,	$\langle 3\rangle \to \langle 4\rangle$
$\langle 1\rangle \to \langle 1\rangle$,	$\langle 4\rangle \to \langle 2\rangle$

Variable Ordering Matters (6)



$\langle x\ x'\ y\ y' \rangle$
$\langle 0\ 0\ 2\ 3 \rangle$
$\langle 0\ 0\ 3\ 4 \rangle$
$\langle 0\ 0\ 4\ 2 \rangle$
$\langle 0\ 1\ 2\ 2 \rangle$
$\langle 0\ 1\ 3\ 3 \rangle$
$\langle 0\ 1\ 4\ 4 \rangle$
$\langle 1\ 0\ 2\ 2 \rangle$
$\langle 1\ 0\ 3\ 3 \rangle$
$\langle 1\ 0\ 4\ 4 \rangle$
$\langle 1\ 1\ 3\ 4 \rangle$
$\langle 1\ 1\ 4\ 2 \rangle$

Variable Ordering Matters (7)



$\langle 000000 \rangle$
$\langle 001001 \rangle$
$\langle 010010 \rangle$
$\langle 011011 \rangle$
$\langle 100100 \rangle$
$\langle 101101 \rangle$
$\langle 110110 \rangle$
$\langle 111111 \rangle$

Set Closure (reachability in a finite space $\hat{\mathcal{S}}$)

Given: $\mathcal{T}_{\mathcal{E}} \subseteq \hat{\mathcal{S}} \times \hat{\mathcal{S}}$ — indexed set of transition relations
Given: $\mathcal{S}_{init} \subseteq \hat{\mathcal{S}}$ — set of initial states
Returns: $\mathcal{S} \subseteq \hat{\mathcal{S}} = (\bigcup_{\alpha \in \mathcal{E}} \mathcal{T}_{\alpha})^{*}(\mathcal{S}_{init})$ — states reachable from $\mathcal{S}_{init}$ by transitions $\mathcal{T}_{\mathcal{E}}$

Breadth-first algorithm:

Algorithm: IterativeTransitiveClosure

1. $\mathcal{T}_{temp} \leftarrow \bigcup_{\alpha \in \mathcal{E}} \mathcal{T}_{\alpha}$
2. $\mathcal{S}_{temp} \leftarrow \mathcal{S}_{init}$
3. Repeat:
   - $\mathcal{S}_{temp} \leftarrow \mathcal{S}_{temp} \cup \mathcal{T}_{temp}(\mathcal{S}_{temp})$
4. Until $\mathcal{S}_{temp}$ converges
5. Return: $\mathcal{S} = \mathcal{S}_{temp}$

Observations leading to Saturation Heuristic

- $(\bigcup_{\alpha\in\mathcal{E}}\mathcal{T}_\alpha)^*(\mathcal{S}_{init})$ (the final result) is often more compact than $(I\cup\bigcup_{\alpha\in\mathcal{E}}\mathcal{T}_\alpha)^n(\mathcal{S}_{init})$ (the n'th intermediate result), for most small n.
- Transition relations often have limited support (set of relevant variables).
  Top variable of transitions is widely distributed;
- Variable ordering matters.
- Saturation assumptions:
  - Symbolic representation of states with K variables $\hat{\mathcal{S}}=\mathcal{S}_K\times\cdots\times\mathcal{S}_1$
  - Partition $\mathcal{T}$ into $\bigcup_{\alpha\in\mathcal{E}}\mathcal{T}_\alpha$ so that $\mathcal{E}=K\ldots1$ and the support of $\mathcal{T}_\alpha$ is within variables $\alpha\ldots1$
  - $\mathcal{T}'_\alpha$ (with domain $\mathcal{S}_\alpha\times\cdots\times\mathcal{S}_1$) is available in interleaved form

Observations leading to Saturation Heuristic

- $(\bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha)^*(\mathcal{S}_{init})$ (the final result) is often more compact than $(I \cup \bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha)^n(\mathcal{S}_{init})$ (the n'th intermediate result), for most small n.
- Transition relations often have limited support (set of relevant variables).
  Top variable of transitions is widely distributed;
- Variable ordering matters.
- Saturation assumptions:
  - Symbolic representation of states with K variables $\hat{\mathcal{S}} = \mathcal{S}_K \times \cdots \times \mathcal{S}_1$
  - Partition $\mathcal{T}$ into $\bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha$ so that $\mathcal{E} = K \ldots 1$ and the support of $\mathcal{T}_\alpha$ is within variables $\alpha \ldots 1$
  - $\mathcal{T}'_\alpha$ (with domain $\mathcal{S}_\alpha \times \cdots \times \mathcal{S}_1$) is available in interleaved form

Observations leading to Saturation Heuristic

- $(\bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha)^*(\mathcal{S}_{init})$ (the final result) is often more compact than $(I \cup \bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha)^n(\mathcal{S}_{init})$ (the n'th intermediate result), for most small n.
- Transition relations often have limited support (set of relevant variables).
  Top variable of transitions is widely distributed;
- Variable ordering matters.
- Saturation assumptions:
  - Symbolic representation of states with K variables $\hat{\mathcal{S}} = \mathcal{S}_K \times \cdots \times \mathcal{S}_1$
  - Partition $\mathcal{T}$ into $\bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha$ so that $\mathcal{E} = K \ldots 1$ and the support of $\mathcal{T}_\alpha$ is within variables $\alpha \ldots 1$
  - $\mathcal{T}'_\alpha$ (with domain $\mathcal{S}_\alpha \times \cdots \times \mathcal{S}_1$) is available in interleaved form

Observations leading to Saturation Heuristic

- $(\bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha)^*(\mathcal{S}_{init})$ (the final result) is often more compact than $(I \cup \bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha)^n(\mathcal{S}_{init})$ (the n'th intermediate result), for most small n.
- Transition relations often have limited support (set of relevant variables).
  Top variable of transitions is widely distributed;
- Variable ordering matters.
- Saturation assumptions:
  - Symbolic representation of states with K variables $\hat{\mathcal{S}} = \mathcal{S}_K \times \cdots \times \mathcal{S}_1$
  - Partition $\mathcal{T}$ into $\bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha$ so that $\mathcal{E} = K \ldots 1$ and the support of $\mathcal{T}_\alpha$ is within variables $\alpha \ldots 1$
  - $\mathcal{T}'_\alpha$ (with domain $\mathcal{S}_\alpha \times \cdots \times \mathcal{S}_1$) is available in interleaved form

Observations leading to Saturation Heuristic

- $(\bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha)^*(\mathcal{S}_{init})$ (the final result) is often more compact than $(I \cup \bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha)^n(\mathcal{S}_{init})$ (the n'th intermediate result), for most small n.
- Transition relations often have limited support (set of relevant variables).
  Top variable of transitions is widely distributed;
- Variable ordering matters.
- Saturation assumptions:
  - Symbolic representation of states with K variables $\hat{\mathcal{S}} = \mathcal{S}_K \times \cdots \times \mathcal{S}_1$
  - Partition $\mathcal{T}$ into $\bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha$ so that $\mathcal{E} = K \ldots 1$ and the support of $\mathcal{T}_\alpha$ is within variables $\alpha \ldots 1$
  - $\mathcal{T}'_\alpha$ (with domain $\mathcal{S}_\alpha \times \cdots \times \mathcal{S}_1$) is available in interleaved form

Observations leading to Saturation Heuristic

- $(\bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha)^*(\mathcal{S}_{init})$ (the final result) is often more compact than $(I \cup \bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha)^n(\mathcal{S}_{init})$ (the n'th intermediate result), for most small n.
- Transition relations often have limited support (set of relevant variables).
  Top variable of transitions is widely distributed;
- Variable ordering matters.
- Saturation assumptions:
  - Symbolic representation of states with K variables $\hat{\mathcal{S}} = \mathcal{S}_K \times \cdots \times \mathcal{S}_1$
  - Partition $\mathcal{T}$ into $\bigcup_{\alpha\in\mathcal{E}} \mathcal{T}_\alpha$ so that $\mathcal{E} = K \ldots 1$ and the support of $\mathcal{T}_\alpha$ is within variables $\alpha \ldots 1$
  - $\mathcal{T}'_\alpha$ (with domain $\mathcal{S}_\alpha \times \cdots \times \mathcal{S}_1$) is available in interleaved form

Observations leading to Saturation Heuristic

- $(\bigcup_{\alpha\in\mathcal{E}}\mathcal{T}_\alpha)^*(\mathcal{S}_{init})$ (the final result) is often more compact than $(I\cup\bigcup_{\alpha\in\mathcal{E}}\mathcal{T}_\alpha)^n(\mathcal{S}_{init})$ (the n'th intermediate result), for most small n.
- Transition relations often have limited support (set of relevant variables).
  Top variable of transitions is widely distributed;
- Variable ordering matters.
- Saturation assumptions:
  - Symbolic representation of states with K variables $\hat{\mathcal{S}} = \mathcal{S}_K \times \cdots \times \mathcal{S}_1$
  - Partition $\mathcal{T}$ into $\bigcup_{\alpha\in\mathcal{E}}\mathcal{T}_\alpha$ so that $\mathcal{E} = K \ldots 1$ and the support of $\mathcal{T}_\alpha$ is within variables $\alpha \ldots 1$
  - $\mathcal{T}'_\alpha$ (with domain $\mathcal{S}_\alpha \times \cdots \times \mathcal{S}_1$) is available in interleaved form

Set Closure (reachability in a finite space $\hat{\mathcal{S}}$)

Given: $\mathcal{T}'_{K..1} \subseteq \hat{\mathcal{S}} \times \hat{\mathcal{S}}$ — indexed set of transition relations

Given: $\mathcal{S}_{init} \subseteq \hat{\mathcal{S}}$ — set of initial states

Returns: $\mathcal{S} \subseteq \hat{\mathcal{S}} = (\bigcup_{\alpha \in K..1} \mathcal{T}_\alpha)^*(\mathcal{S}_{init})$ — states reachable from $\mathcal{S}_{init}$ by transitions $\mathcal{T}_{\mathcal{E}}$

First call saturation based algorithm: SatClosure($\mathcal{T}'_{K:1}, \mathcal{S}_{init}$)

Saturation-based algorithm: (memoization code not shown)

Algorithm: SatClosure($\mathcal{T}'_{k:1}, \mathcal{S}_{init}$)

1. $\mathcal{S}_{temp} \leftarrow \mathcal{S}_{init}$
2. Repeat:
   - For each child s_i of $\mathcal{S}_{temp}$ do: $s'_i \leftarrow$ SatClosure($(\mathcal{T}'_{k-1:1}, s_i)$)
   - $\mathcal{S}_{temp} \leftarrow$ new node with children s'_*
   - $\mathcal{S}_{temp} \leftarrow \mathcal{S}_{temp} \cup \mathcal{T}'_k(\mathcal{S}_{temp})$
3. Until $\mathcal{S}_{temp}$ converges
4. Return: $\mathcal{S} = \mathcal{S}_{temp}$

Bisimulation algorithms

Generic iterative *splitting* algorithm:
Iterative update of some equivalence relation variable $\mathcal{B}$.

- Start with $\mathcal{B}$ = coarsest partition of state space $\mathcal{S}$, $\mathcal{S} \times \mathcal{S}$ ($\sim \subseteq \mathcal{B}$)
- Initially *split* $\mathcal{B}$ based on state color
- Iteratively remove implausible members from $\mathcal{B}$ when required by definition of Bisimulation, by splitting $\mathcal{B}$ into smaller blocks.
  $\forall \alpha \in \mathcal{E}, \forall p' \in \mathcal{S}, (p \xrightarrow{\alpha} p' \Rightarrow \exists q' \in \mathcal{S}, q \xrightarrow{\alpha} q' \wedge \langle p', q' \rangle \in \mathcal{B})$
- Iteration continues until all equivalence classes have been used as splitters.
- $|\mathcal{B}|$ shrinks monotonically

Bisimulation algorithms

Generic iterative *splitting* algorithm:
Iterative update of some equivalence relation variable $\mathcal{B}$.

- Start with $\mathcal{B} =$ coarsest partition of state space $\mathcal{S}$, $\mathcal{S} \times \mathcal{S}$ ($\sim \subseteq \mathcal{B}$)
- Initially *split* $\mathcal{B}$ based on state color
- Iteratively remove implausible members from $\mathcal{B}$ when required by definition of Bisimulation, by splitting $\mathcal{B}$ into smaller blocks. $\forall \alpha \in \mathcal{E}, \forall p' \in \mathcal{S}, (p \xrightarrow{\alpha} p' \Rightarrow \exists q' \in \mathcal{S}, q \xrightarrow{\alpha} q' \wedge \langle p', q' \rangle \in \mathcal{B})$
- Iteration continues until all equivalence classes have been used as splitters.
- $|\mathcal{B}|$ shrinks monotonically

Bisimulation algorithms

Generic iterative *splitting* algorithm:
Iterative update of some equivalence relation variable $\mathcal{B}$.

- Start with $\mathcal{B} =$ coarsest partition of state space $\mathcal{S}$, $\mathcal{S} \times \mathcal{S}$ $(\sim \subseteq \mathcal{B})$
- Initially *split* $\mathcal{B}$ based on state color
- Iteratively remove implausible members from $\mathcal{B}$ when required by definition of Bisimulation, by splitting $\mathcal{B}$ into smaller blocks.
$\forall \alpha \in \mathcal{E}, \forall p' \in \mathcal{S}, (p \xrightarrow{\alpha} p' \Rightarrow \exists q' \in \mathcal{S}, q \xrightarrow{\alpha} q' \wedge \langle p', q' \rangle \in \mathcal{B})$
- Iteration continues until all equivalence classes have been used as splitters.
- $|\mathcal{B}|$ shrinks monotonically

Bisimulation algorithms

Generic iterative *splitting* algorithm:
Iterative update of some equivalence relation variable $\mathcal{B}$.

- Start with $\mathcal{B} =$ coarsest partition of state space $\mathcal{S}$, $\mathcal{S} \times \mathcal{S}$ ($\sim \subseteq \mathcal{B}$)
- Initially *split* $\mathcal{B}$ based on state color
- Iteratively remove implausible members from $\mathcal{B}$ when required by definition of Bisimulation, by splitting $\mathcal{B}$ into smaller blocks.
  $\forall \alpha \in \mathcal{E}, \forall p' \in \mathcal{S}, (p \stackrel{\alpha}{\to} p' \Rightarrow \exists q' \in \mathcal{S}, q \stackrel{\alpha}{\to} q' \wedge \langle p', q' \rangle \in \mathcal{B})$
- Iteration continues until all equivalence classes have been used as splitters.
- $|\mathcal{B}|$ shrinks monotonically

Matching Transitions to Pairs in $\mathcal{B}$.

$\forall \langle p, q \rangle \in \mathcal{B}$:
$\forall \alpha \in \mathcal{E}, \forall p' \in \mathcal{S}, (p \xrightarrow{\alpha} p' \Rightarrow \exists q' \in \mathcal{S}, q \xrightarrow{\alpha} q' \wedge \langle p', q' \rangle \in \mathcal{B})$

$$
\begin{array}{ccc} p & \sim & q \\ \alpha\downarrow & & \\ p' & & \end{array}
\quad \Longrightarrow \quad
\begin{array}{ccc} p & \sim & q \\ \alpha\downarrow & & \downarrow\alpha \\ p' & \sim & q' \end{array}
$$

Splitting (Contrapositive).

$\forall \langle p, q \rangle \in \mathcal{B}$:
$\forall \alpha \in \mathcal{E}, \forall p' \in \mathcal{S}, (p \xrightarrow{\alpha} p' \Rightarrow \exists q' \in \mathcal{S}, q \xrightarrow{\alpha} q' \wedge \langle p', q' \rangle \in \mathcal{B})$

$$\begin{array}{ccc} p & \not\sim & q \\ \alpha\downarrow & & \\ p' & & \end{array} \quad \Longleftarrow \quad \begin{array}{ccc} p & \sim ? & q \\ \alpha\downarrow & & \alpha\not\downarrow \\ p' & \sim & q' \end{array}$$

Outline

Our Contribution

- Interleaved partition representation
  necessary for large partitions
- Deterministic transitions
  temporary focus $\leftrightarrow$ key realization
- Negative space ($\overline{\approx}$)
  Closure/Reachability problem
- Saturation $\leftrightarrow$ Efficient
  Symbolic Heuristic

Our Contribution

- Interleaved partition representation necessary for large partitions
- Deterministic transitions
  temporary focus $\leftrightarrow$ key realization
- Negative space ($\overline{\approx}$)
  Closure/Reachability problem
- Saturation $\leftrightarrow$ Efficient
  Symbolic Heuristic

Our Contribution

- Interleaved partition representation necessary for large partitions
- Deterministic transitions temporary focus $\leftrightarrow$ key realization
- Negative space ($\eqsim$) Closure/Reachability problem
- Saturation $\leftrightarrow$ Efficient Symbolic Heuristic

Our Contribution

- Interleaved partition representation necessary for large partitions
- Deterministic transitions
  temporary focus $\leftrightarrow$ key realization
- Negative space ($\eqsim$)
  Closure/Reachability problem
- Saturation $\leftrightarrow$ Efficient
  Symbolic Heuristic

Outline

Splitting.

Example

$$
\begin{tikzcd}
p \arrow[d, "\alpha"'] & \not\sim & q \\
p' & &
\end{tikzcd}
\quad \Longleftarrow \quad
\begin{tikzcd}
p \arrow[d, "\alpha"'] & \sim ? & q \arrow[d, "\alpha", "/" marking] \\
p' & \sim & q'
\end{tikzcd}
$$

Splitting.

Example

$$\begin{tikzcd} p \arrow[d, "\alpha"'] & \sim ? & q \arrow[d, "\alpha\!\!/"'] \\ p' & \sim & q' \end{tikzcd} \implies \begin{tikzcd} p \arrow[d, "\alpha"'] & \not\sim & q \\ p' & & \end{tikzcd}$$

Splitting with Deterministic Transitions (main idea).

Example

$$
\begin{tikzcd}
p \arrow[r, phantom, "\sim ?"] \arrow[d, "\alpha"'] & q \arrow[d, "\alpha"'] \\
p' \arrow[r, phantom, "\not\sim"] & q'
\end{tikzcd}
\Longrightarrow
\begin{tikzcd}
p \arrow[r, phantom, "\sim ?"] \arrow[d, "\alpha"'] & q \arrow[d, "\alpha\not"'] \\
p' \arrow[r, phantom, "\sim"] & q''
\end{tikzcd}
\Longrightarrow
\begin{tikzcd}
p \arrow[r, phantom, "\not\sim"] \arrow[d, "\alpha"'] & q \arrow[d, "\alpha"'] \\
p' \arrow[r, phantom, "\not\sim"] & q'
\end{tikzcd}
$$

Splitting with Deterministic Transitions (simplified).

Example

$$
\begin{tikzcd}
p \arrow[d, "\alpha"'] & \sim ? & q \arrow[d, "\alpha"'] & \Longrightarrow & p \arrow[d, "\alpha"'] & \color{red}{\not\sim} & q \arrow[d, "\alpha"'] \\
p' & \color{red}{\not\sim} & q' & & p' & \color{red}{\not\sim} & q'
\end{tikzcd}
$$

Splitting with Deterministic Transitions (BPRR).

Example

$$\begin{array}{ccccccc} p & & q & & \langle p & & q \rangle \\ \alpha \big\downarrow & \wedge & \alpha \big\downarrow & \Longleftrightarrow & & \alpha \Big\Uparrow & \\ p' & & q' & & \langle p' & & q' \rangle \end{array}$$

$\mathcal{P}_\alpha = (\mathcal{T}_\alpha \times \mathcal{T}_\alpha)^{-1}$ $(\forall \alpha \in \mathcal{E})$

Splitting with Deterministic Transitions (BPRR).

Example

$$\begin{array}{ccc} \langle p \quad \sim ? \quad q\rangle & & \langle p \quad \nsim \quad q\rangle \\ \alpha \Big\Uparrow & \Longrightarrow & \alpha \Big\Uparrow \\ \langle p' \quad \nsim \quad q'\rangle & & \langle p' \quad \nsim \quad q'\rangle \end{array}$$

$$\mathcal{P}_\alpha = (\mathcal{T}_\alpha \times \mathcal{T}_\alpha)^{-1} \qquad (\forall \alpha \in \mathcal{E})$$

Splitting with Deterministic Transitions (BPRR).

Example

$$
\begin{array}{ccccccccccc}
\langle s_0,s_0\rangle & \Rightarrow & \langle s_0,s_1\rangle & \Leftarrow & \langle s_0,s_2\rangle & \Leftarrow & \langle s_0,s_3\rangle & \Leftarrow & \langle s_0,s_4\rangle & & \langle s_0,s_5\rangle \\
\Downarrow & & & & & & \Downarrow & & \Uparrow & & \Uparrow \\
\langle s_1,s_0\rangle & & \langle s_1,s_1\rangle & \Leftarrow & \langle s_1,s_2\rangle & & \langle s_1,s_3\rangle & \Rightarrow & \langle s_1,s_4\rangle & \Leftarrow & \langle s_1,s_5\rangle \\
\Uparrow & & \Uparrow & & \Downarrow & & \Updownarrow & & \Updownarrow & & \\
\langle s_2,s_0\rangle & & \langle s_2,s_1\rangle & \Rightarrow & \langle s_2,s_2\rangle & & \langle s_2,s_3\rangle & & \langle s_2,s_4\rangle & \Rightarrow & \langle s_2,s_5\rangle \\
\Uparrow & & & & & & & & & & \Downarrow \\
\langle s_3,s_0\rangle & \Rightarrow & \langle s_3,s_1\rangle & \Leftrightarrow & \langle s_3,s_2\rangle & & \langle s_3,s_3\rangle & \Rightarrow & \langle s_3,s_4\rangle & \Rightarrow & \langle s_3,s_5\rangle \\
\Uparrow & & \Downarrow & & & & \Downarrow & & \Uparrow & & \Updownarrow \\
\langle s_4,s_0\rangle & \Leftarrow & \langle s_4,s_1\rangle & \Leftrightarrow & \langle s_4,s_2\rangle & & \langle s_4,s_3\rangle & \Leftarrow & \langle s_4,s_4\rangle & & \langle s_4,s_5\rangle \\
 & & \Uparrow & & \Downarrow & & \Downarrow & & & & \\
\langle s_5,s_0\rangle & \Leftarrow & \langle s_5,s_1\rangle & & \langle s_5,s_2\rangle & \Rightarrow & \langle s_5,s_3\rangle & \Leftrightarrow & \langle s_5,s_4\rangle & & \langle s_5,s_5\rangle
\end{array}
$$

$\mathcal{P} = \bigcup_{\alpha\in\mathcal{E}} \mathcal{P}_\alpha$

Splitting with Deterministic Transitions (BPRR).

Example

$$
\begin{array}{ccccccccccc}
\langle s_0 \sim s_0 \rangle & \Rightarrow & \langle s_0 ? s_1 \rangle & \Leftarrow & \langle s_0 ? s_2 \rangle & \Leftarrow & \langle s_0 ? s_3 \rangle & \Leftarrow & \langle s_0 ? s_4 \rangle & & \langle s_0 ? s_5 \rangle \\
\Downarrow & & & & & & \Downarrow & & \Uparrow & & \Uparrow \\
\langle s_1 ? s_0 \rangle & & \langle s_1 \sim s_1 \rangle & \Leftarrow & \langle s_1 ? s_2 \rangle & & \langle s_1 ? s_3 \rangle & \Rightarrow & \langle s_1 ? s_4 \rangle & \Leftarrow & \langle s_1 \nsim s_5 \rangle \\
\Uparrow & & \Uparrow & & \Downarrow & & \Updownarrow & & \Updownarrow & & \\
\langle s_2 ? s_0 \rangle & & \langle s_2 ? s_1 \rangle & \Rightarrow & \langle s_2 \sim s_2 \rangle & & \langle s_2 ? s_3 \rangle & & \langle s_2 ? s_4 \rangle & \Rightarrow & \langle s_2 ? s_5 \rangle \\
\Uparrow & & & & & & & & & & \Downarrow \\
\langle s_3 ? s_0 \rangle & \Rightarrow & \langle s_3 ? s_1 \rangle & \Leftrightarrow & \langle s_3 ? s_2 \rangle & & \langle s_3 \sim s_3 \rangle & \Rightarrow & \langle s_3 ? s_4 \rangle & \Rightarrow & \langle s_3 ? s_5 \rangle \\
\Uparrow & & \Downarrow & & & & \Downarrow & & \Uparrow & & \Updownarrow \\
\langle s_4 ? s_0 \rangle & \Leftarrow & \langle s_4 ? s_1 \rangle & \Leftrightarrow & \langle s_4 ? s_2 \rangle & & \langle s_4 ? s_3 \rangle & \Leftarrow & \langle s_4 \sim s_4 \rangle & & \langle s_4 ? s_5 \rangle \\
& & \Uparrow & & \Downarrow & & \Downarrow & & & & \\
\langle s_5 ? s_0 \rangle & \Leftarrow & \langle s_5 \nsim s_1 \rangle & & \langle s_5 ? s_2 \rangle & \Rightarrow & \langle s_5 ? s_3 \rangle & \Leftrightarrow & \langle s_5 ? s_4 \rangle & & \langle s_5 \sim s_5 \rangle
\end{array}
$$

$\mathcal{P} = \bigcup_{\alpha \in \mathcal{E}} \mathcal{P}_\alpha$

Splitting with Deterministic Transitions (BPRR).

Example

$$
\begin{array}{ccccccccccc}
\langle s_0 \sim s_0 \rangle & \Rightarrow & \langle s_0 \not\sim s_1 \rangle & \Leftarrow & \langle s_0 \not\sim s_2 \rangle & \Leftarrow & \langle s_0 \not\sim s_3 \rangle & \Leftarrow & \langle s_0 \not\sim s_4 \rangle & & \langle s_0 \not\sim s_5 \rangle \\
\Downarrow & & & & & & \Downarrow & & \Uparrow & & \Uparrow \\
\langle s_1 \not\sim s_0 \rangle & & \langle s_1 \sim s_1 \rangle & \Leftarrow & \langle s_1 \,?\, s_2 \rangle & & \langle s_1 \not\sim s_3 \rangle & \Rightarrow & \langle s_1 \not\sim s_4 \rangle & \Leftarrow & \langle s_1 \not\sim s_5 \rangle \\
\Uparrow & & \Uparrow & & \Downarrow & & \Updownarrow & & \Updownarrow & & \\
\langle s_2 \not\sim s_0 \rangle & & \langle s_2 \,?\, s_1 \rangle & \Rightarrow & \langle s_2 \sim s_2 \rangle & & \langle s_2 \not\sim s_3 \rangle & & \langle s_2 \not\sim s_4 \rangle & \Rightarrow & \langle s_2 \not\sim s_5 \rangle \\
\Uparrow & & & & & & & & & & \Downarrow \\
\langle s_3 \not\sim s_0 \rangle & \Rightarrow & \langle s_3 \not\sim s_1 \rangle & \Leftrightarrow & \langle s_3 \not\sim s_2 \rangle & & \langle s_3 \sim s_3 \rangle & \Rightarrow & \langle s_3 \,?\, s_4 \rangle & \Rightarrow & \langle s_3 \not\sim s_5 \rangle \\
\Uparrow & & \Downarrow & & & & \Downarrow & & \Uparrow & & \Updownarrow \\
\langle s_4 \not\sim s_0 \rangle & \Leftarrow & \langle s_4 \not\sim s_1 \rangle & \Leftrightarrow & \langle s_4 \not\sim s_2 \rangle & & \langle s_4 \,?\, s_3 \rangle & \Leftarrow & \langle s_4 \sim s_4 \rangle & & \langle s_4 \not\sim s_5 \rangle \\
& & \Uparrow & & \Downarrow & & \Downarrow & & & & \\
\langle s_5 \not\sim s_0 \rangle & \Leftarrow & \langle s_5 \not\sim s_1 \rangle & & \langle s_5 \not\sim s_2 \rangle & \Rightarrow & \langle s_5 \not\sim s_3 \rangle & \Leftrightarrow & \langle s_5 \not\sim s_4 \rangle & & \langle s_5 \sim s_5 \rangle
\end{array}
$$

Use Saturation-based set closure.

Splitting with Deterministic Transitions (BPRR).

Example

$$
\begin{array}{ccccccccccc}
\langle s_0 \sim s_0 \rangle & \Rightarrow & \langle s_0 \not\sim s_1 \rangle & \Leftarrow & \langle s_0 \not\sim s_2 \rangle & \Leftarrow & \langle s_0 \not\sim s_3 \rangle & \Leftarrow & \langle s_0 \not\sim s_4 \rangle & & \langle s_0 \not\sim s_5 \rangle \\
\Downarrow & & & & & & \Downarrow & & \Uparrow & & \Uparrow \\
\langle s_1 \not\sim s_0 \rangle & & \langle s_1 \sim s_1 \rangle & \Leftarrow & \langle s_1 \sim s_2 \rangle & & \langle s_1 \not\sim s_3 \rangle & \Rightarrow & \langle s_1 \not\sim s_4 \rangle & \Leftarrow & \langle s_1 \not\sim s_5 \rangle \\
\Uparrow & & \Uparrow & & \Downarrow & & \Updownarrow & & \Updownarrow & & \\
\langle s_2 \not\sim s_0 \rangle & & \langle s_2 \sim s_1 \rangle & \Rightarrow & \langle s_2 \sim s_2 \rangle & & \langle s_2 \not\sim s_3 \rangle & & \langle s_2 \not\sim s_4 \rangle & \Rightarrow & \langle s_2 \not\sim s_5 \rangle \\
\Uparrow & & & & & & & & & & \Downarrow \\
\langle s_3 \not\sim s_0 \rangle & \Rightarrow & \langle s_3 \not\sim s_1 \rangle & \Leftrightarrow & \langle s_3 \not\sim s_2 \rangle & & \langle s_3 \sim s_3 \rangle & \Rightarrow & \langle s_3 \sim s_4 \rangle & \Rightarrow & \langle s_3 \not\sim s_5 \rangle \\
\Uparrow & & \Downarrow & & & & \Downarrow & & \Uparrow & & \Updownarrow \\
\langle s_4 \not\sim s_0 \rangle & \Leftarrow & \langle s_4 \not\sim s_1 \rangle & \Leftrightarrow & \langle s_4 \not\sim s_2 \rangle & & \langle s_4 \sim s_3 \rangle & \Leftarrow & \langle s_4 \sim s_4 \rangle & & \langle s_4 \not\sim s_5 \rangle \\
 & & \Uparrow & & \Downarrow & & \Downarrow & & & & \\
\langle s_5 \not\sim s_0 \rangle & \Leftarrow & \langle s_5 \not\sim s_1 \rangle & & \langle s_5 \not\sim s_2 \rangle & \Rightarrow & \langle s_5 \not\sim s_3 \rangle & \Leftrightarrow & \langle s_5 \not\sim s_4 \rangle & & \langle s_5 \sim s_5 \rangle
\end{array}
$$

Only bisimilar pairs remain.

Splitting with Deterministic Transitions (BPRR).

Example

$$
\begin{array}{ccccccccccc}
\langle s_0 \sim s_0 \rangle & \Rightarrow & & \Leftarrow & & \Leftarrow & & \Leftarrow & & & \\
\Downarrow & & & & & & \Downarrow & & \Uparrow & & \Uparrow \\
& & \langle s_1 \sim s_1 \rangle & \Leftarrow & \langle s_1 \sim s_2 \rangle & & & \Rightarrow & & \Leftarrow & \\
\Uparrow & & \Uparrow & & \Downarrow & & \Updownarrow & & \Updownarrow & & \\
& & \langle s_2 \sim s_1 \rangle & \Rightarrow & \langle s_2 \sim s_2 \rangle & & & & & \Rightarrow & \\
\Uparrow & & & & & & & & & & \Downarrow \\
& \Rightarrow & & \Leftrightarrow & & & \langle s_3 \sim s_3 \rangle & \Rightarrow & \langle s_3 \sim s_4 \rangle & \Rightarrow & \\
\Uparrow & & \Downarrow & & & & \Downarrow & & \Uparrow & & \Updownarrow \\
& \Leftarrow & & \Leftrightarrow & & & \langle s_4 \sim s_3 \rangle & \Leftarrow & \langle s_4 \sim s_4 \rangle & & \\
& & \Uparrow & & \Downarrow & & \Downarrow & & & & \\
& \Leftarrow & & & & \Rightarrow & & \Leftrightarrow & & & \langle s_5 \sim s_5 \rangle
\end{array}
$$

Only bisimilar pairs remain.

Outline

Alteration of generic splitting algorithm.

- Initial partition, then iterative splitting based on pair relations.
- Two ways a pair may be non-bisimilar:
- (initial)
- $$\begin{tikzcd} p \arrow[d, "\alpha"'] & \sim ? & q \arrow[d, "\alpha", "/" marking] \\ p' & & {} \end{tikzcd} \Longrightarrow \begin{tikzcd} p \arrow[d, "\alpha"'] & \not\sim & q \arrow[d, "\alpha", "/" marking] \\ p' & & {} \end{tikzcd}$$
- (iterative)
- $$\begin{tikzcd} p \arrow[d, "\alpha"'] & \sim ? & q \arrow[d, "\alpha"] \\ p' & \not\sim & q' \end{tikzcd} \Longrightarrow \begin{tikzcd} p \arrow[d, "\alpha"'] & \not\sim & q \arrow[d, "\alpha"] \\ p' & \not\sim & q' \end{tikzcd}$$

Alteration of generic splitting algorithm.

- Initial partition, then iterative splitting based on pair relations.
- Two ways a pair may be non-bisimilar:
- (initial)
- $$\begin{tikzcd} p \arrow[d, "\alpha"'] \arrow[r, phantom, "\sim ?"] & q \arrow[d, "\alpha", "/" marking] \\ p' & {} \end{tikzcd} \Longrightarrow \begin{tikzcd} p \arrow[d, "\alpha"'] \arrow[r, phantom, "\not\sim"] & q \arrow[d, "\alpha", "/" marking] \\ p' & {} \end{tikzcd}$$
- (iterative)
- $$\begin{tikzcd} p \arrow[d, "\alpha"'] \arrow[r, phantom, "\sim ?"] & q \arrow[d, "\alpha"] \\ p' \arrow[r, phantom, "\not\sim"] & q' \end{tikzcd} \Longrightarrow \begin{tikzcd} p \arrow[d, "\alpha"'] \arrow[r, phantom, "\not\sim"] & q \arrow[d, "\alpha"] \\ p' \arrow[r, phantom, "\not\sim"] & q' \end{tikzcd}$$

Alteration of generic splitting algorithm.

- Initial partition, then iterative splitting based on pair relations.
- Two ways a pair may be non-bisimilar:
- (initial)
- $$\begin{tikzcd} p \arrow[d, "\alpha"'] & \sim ? & q \arrow[d, "\alpha"', "/" marking] \\ p' & & {} \end{tikzcd} \Longrightarrow \begin{tikzcd} p \arrow[d, "\alpha"'] & \not\sim & q \arrow[d, "\alpha"', "/" marking] \\ p' & & {} \end{tikzcd}$$
- (iterative)
- $$\begin{tikzcd} p \arrow[d, "\alpha"'] & \sim ? & q \arrow[d, "\alpha"'] \\ p' & \not\sim & q' \end{tikzcd} \Longrightarrow \begin{tikzcd} p \arrow[d, "\alpha"'] & \not\sim & q \arrow[d, "\alpha"'] \\ p' & \not\sim & q' \end{tikzcd}$$

Alteration of generic splitting algorithm.

- Initial partition, then iterative splitting based on pair relations.
- Two ways a pair may be non-bisimilar:
- (initial)
- $$\begin{tikzcd} p \arrow[d, "\alpha"'] & \sim ? & q \arrow[d, "\alpha"', "/"{marking}] \\ p' & & {} \end{tikzcd} \quad \Longrightarrow \quad \begin{tikzcd} p \arrow[d, "\alpha"'] & \nsim & q \arrow[d, "\alpha"', "/"{marking}] \\ p' & & {} \end{tikzcd}$$
- (iterative)
- $$\begin{tikzcd} p \arrow[d, "\alpha"'] & \sim ? & q \arrow[d, "\alpha"'] \\ p' & \nsim & q' \end{tikzcd} \quad \Longrightarrow \quad \begin{tikzcd} p \arrow[d, "\alpha"'] & \nsim & q \arrow[d, "\alpha"'] \\ p' & \nsim & q' \end{tikzcd}$$

Constructing BPRRs.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{S}$ (and the corresponding $\mathcal{T}'_{\mathcal{E}}$)
- Construct new domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Create BPRRs: $\mathcal{P}_{\mathcal{E}} \subseteq \hat{\mathcal{B}} \times \hat{\mathcal{B}}$.
- $\mathcal{P}_\alpha(\langle s_1, s_2 \rangle) =$ pairs $\langle s_3, s_4 \rangle$
  where $s_1 = \mathcal{T}_\alpha(s_3) \wedge s_2 = \mathcal{T}_\alpha(s_4)$ $(\forall \alpha \in \mathcal{E})$
- $\mathcal{P}_\alpha = (\mathcal{T}'_\alpha \times \mathcal{T}'_\alpha)^{-1}$ $(\forall \alpha \in \mathcal{E})$

Constructing BPRRs.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{S}$ (and the corresponding $\mathcal{T}'_{\mathcal{E}}$)
- Construct new domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Create BPRRs: $\mathcal{P}_{\mathcal{E}} \subseteq \hat{\mathcal{B}} \times \hat{\mathcal{B}}$.
- $\mathcal{P}_\alpha(\langle s_1, s_2 \rangle) =$ pairs $\langle s_3, s_4 \rangle$ where $s_1 = \mathcal{T}_\alpha(s_3) \wedge s_2 = \mathcal{T}_\alpha(s_4)$ $(\forall \alpha \in \mathcal{E})$
- $\mathcal{P}_\alpha = (\mathcal{T}'_\alpha \times \mathcal{T}'_\alpha)^{-1}$ $(\forall \alpha \in \mathcal{E})$

Constructing BPRRs.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{S}$ (and the corresponding $\mathcal{T}'_{\mathcal{E}}$)
- Construct new domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Create BPRRs: $\mathcal{P}_{\mathcal{E}} \subseteq \hat{\mathcal{B}} \times \hat{\mathcal{B}}$.
- $\mathcal{P}_{\alpha}(\langle s_1, s_2 \rangle) =$ pairs $\langle s_3, s_4 \rangle$
  where $s_1 = \mathcal{T}_{\alpha}(s_3) \wedge s_2 = \mathcal{T}_{\alpha}(s_4)$ $(\forall \alpha \in \mathcal{E})$
- $\mathcal{P}_{\alpha} = (\mathcal{T}'_{\alpha} \times \mathcal{T}'_{\alpha})^{-1}$ $(\forall \alpha \in \mathcal{E})$

Constructing BPRRs.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{S}$ (and the corresponding $\mathcal{T}'_{\mathcal{E}}$)
- Construct new domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Create BPRRs: $\mathcal{P}_{\mathcal{E}} \subseteq \hat{\mathcal{B}} \times \hat{\mathcal{B}}$.
- $\mathcal{P}_\alpha(\langle s_1, s_2 \rangle) =$ pairs $\langle s_3, s_4 \rangle$
  where $s_1 = \mathcal{T}_\alpha(s_3) \wedge s_2 = \mathcal{T}_\alpha(s_4)$ $(\forall \alpha \in \mathcal{E})$
- $\mathcal{P}_\alpha = (\mathcal{T}'_\alpha \times \mathcal{T}'_\alpha)^{-1}$ $(\forall \alpha \in \mathcal{E})$

Constructing BPRRs.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_\mathcal{E}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_\mathcal{E} \subseteq \mathcal{S} \times \mathcal{S}$ (and the corresponding $\mathcal{T}'_\mathcal{E}$)
- Construct new domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Create BPRRs: $\mathcal{P}_\mathcal{E} \subseteq \hat{\mathcal{B}} \times \hat{\mathcal{B}}$.
- $\mathcal{P}_\alpha(\langle s_1, s_2 \rangle) =$ pairs $\langle s_3, s_4 \rangle$
  where $s_1 = \mathcal{T}_\alpha(s_3) \wedge s_2 = \mathcal{T}_\alpha(s_4)$ $(\forall \alpha \in \mathcal{E})$
- $\mathcal{P}_\alpha = (\mathcal{T}'_\alpha \times \mathcal{T}'_\alpha)^{-1}$ $(\forall \alpha \in \mathcal{E})$

Constructing BPRRs.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_\mathcal{E}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_\mathcal{E} \subseteq \mathcal{S} \times \mathcal{S}$ (and the corresponding $\mathcal{T}'_\mathcal{E}$)
- Construct new domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Create BPRRs: $\mathcal{P}_\mathcal{E} \subseteq \hat{\mathcal{B}} \times \hat{\mathcal{B}}$.
- $\mathcal{P}_\alpha(\langle s_1, s_2 \rangle) =$ pairs $\langle s_3, s_4 \rangle$
  where $s_1 = \mathcal{T}_\alpha(s_3) \wedge s_2 = \mathcal{T}_\alpha(s_4)$ $(\forall \alpha \in \mathcal{E})$
- $\mathcal{P}_\alpha = (\mathcal{T}'_\alpha \times \mathcal{T}'_\alpha)^{-1}$ $(\forall \alpha \in \mathcal{E})$

Initial Partition.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{S}$, Pair domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Partition according to coloring function $c(s) = \{\alpha \in \mathcal{E} | \exists s' \in \mathcal{S} : \langle s, s' \rangle \in \mathcal{T}_\alpha\}$
- Initial Partition: $\overline{\mathcal{B}}_{init} \subseteq \hat{\mathcal{B}}$, where only 1 member of each pair enables $\mathcal{T}_\alpha$, for some $\alpha \in \mathcal{E}$.
- States where $\mathcal{T}_\alpha$ is enabled: $\mathcal{S}_{[\alpha]} = \{s \in \mathcal{S} | \exists s' : \langle s, s' \rangle \in \mathcal{T}_\alpha\}$.
- Initial Partition: $\overline{\mathcal{B}}_{init} = \bigcup_{\alpha \in \mathcal{E}} \left(\mathcal{S}_{[\alpha]} \times (\mathcal{S} \setminus \mathcal{S}_{[\alpha]}) \right) \cup \left((\mathcal{S} \setminus \mathcal{S}_{[\alpha]}) \times \mathcal{S}_{[\alpha]} \right)$.

Initial Partition.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{S}$, Pair domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Partition according to coloring function $c(s) = \{\alpha \in \mathcal{E} | \exists s' \in \mathcal{S} : \langle s, s' \rangle \in \mathcal{T}_\alpha\}$
- Initial Partition: $\overline{\mathcal{B}}_{init} \subseteq \hat{\mathcal{B}}$, where only 1 member of each pair enables $\mathcal{T}_\alpha$, for some $\alpha \in \mathcal{E}$.
- States where $\mathcal{T}_\alpha$ is enabled: $\mathcal{S}_{[\alpha]} = \{s \in \mathcal{S} | \exists s' : \langle s, s' \rangle \in \mathcal{T}_\alpha\}$.
- Initial Partition: $\overline{\mathcal{B}}_{init} = \bigcup_{\alpha \in \mathcal{E}} (\mathcal{S}_{[\alpha]} \times (\mathcal{S} \setminus \mathcal{S}_{[\alpha]})) \cup ((\mathcal{S} \setminus \mathcal{S}_{[\alpha]}) \times \mathcal{S}_{[\alpha]})$.

Initial Partition.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{S}$, Pair domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Partition according to coloring function $c(s) = \{\alpha \in \mathcal{E} | \exists s' \in \mathcal{S} : \langle s, s' \rangle \in \mathcal{T}_\alpha\}$
- Initial Partition: $\overline{\mathcal{B}}_{init} \subseteq \hat{\mathcal{B}}$, where only 1 member of each pair enables $\mathcal{T}_\alpha$, for some $\alpha \in \mathcal{E}$.
- States where $\mathcal{T}_\alpha$ is enabled: $\mathcal{S}_{[\alpha]} = \{s \in \mathcal{S} | \exists s' : \langle s, s' \rangle \in \mathcal{T}_\alpha\}$.
- Initial Partition: $\overline{\mathcal{B}}_{init} = \bigcup_{\alpha \in \mathcal{E}} (\mathcal{S}_{[\alpha]} \times (\mathcal{S} \setminus \mathcal{S}_{[\alpha]})) \cup ((\mathcal{S} \setminus \mathcal{S}_{[\alpha]}) \times \mathcal{S}_{[\alpha]})$.

Initial Partition.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{S}$, Pair domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Partition according to coloring function $c(s) = \{\alpha \in \mathcal{E} | \exists s' \in \mathcal{S} : \langle s, s' \rangle \in \mathcal{T}_{\alpha}\}$
- Initial Partition: $\overline{\mathcal{B}}_{init} \subseteq \hat{\mathcal{B}}$, where only 1 member of each pair enables $\mathcal{T}_{\alpha}$, for some $\alpha \in \mathcal{E}$.
- States where $\mathcal{T}_{\alpha}$ is enabled: $\mathcal{S}_{[\alpha]} = \{s \in \mathcal{S} | \exists s' : \langle s, s' \rangle \in \mathcal{T}_{\alpha}\}$.
- Initial Partition: $\overline{\mathcal{B}}_{init} = \bigcup_{\alpha \in \mathcal{E}} \left(\mathcal{S}_{[\alpha]} \times (\mathcal{S} \setminus \mathcal{S}_{[\alpha]}) \right) \cup \left((\mathcal{S} \setminus \mathcal{S}_{[\alpha]}) \times \mathcal{S}_{[\alpha]} \right)$.

Initial Partition.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{S}$, Pair domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Partition according to coloring function $c(s) = \{\alpha \in \mathcal{E} | \exists s' \in \mathcal{S} : \langle s, s' \rangle \in \mathcal{T}_\alpha\}$
- Initial Partition: $\overline{\mathcal{B}}_{init} \subseteq \hat{\mathcal{B}}$, where only 1 member of each pair enables $\mathcal{T}_\alpha$, for some $\alpha \in \mathcal{E}$.
- States where $\mathcal{T}_\alpha$ is enabled: $\mathcal{S}_{[\alpha]} = \{s \in \mathcal{S} | \exists s' : \langle s, s' \rangle \in \mathcal{T}_\alpha\}$.
- Initial Partition: $\overline{\mathcal{B}}_{init} = \bigcup_{\alpha \in \mathcal{E}} (\mathcal{S}_{[\alpha]} \times (\mathcal{S} \setminus \mathcal{S}_{[\alpha]})) \cup ((\mathcal{S} \setminus \mathcal{S}_{[\alpha]}) \times \mathcal{S}_{[\alpha]})$.

Initial Partition.

Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$:

- Transitions: $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{S}$, Pair domain: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- Partition according to coloring function $c(s) = \{\alpha \in \mathcal{E} | \exists s' \in \mathcal{S} : \langle s, s' \rangle \in \mathcal{T}_\alpha\}$
- Initial Partition: $\overline{\mathcal{B}}_{init} \subseteq \hat{\mathcal{B}}$, where only 1 member of each pair enables $\mathcal{T}_\alpha$, for some $\alpha \in \mathcal{E}$.
- States where $\mathcal{T}_\alpha$ is enabled: $\mathcal{S}_{[\alpha]} = \{s \in \mathcal{S} | \exists s' : \langle s, s' \rangle \in \mathcal{T}_\alpha\}$.
- Initial Partition: $\overline{\mathcal{B}}_{init} = \bigcup_{\alpha \in \mathcal{E}} \left(\mathcal{S}_{[\alpha]} \times (\mathcal{S} \setminus \mathcal{S}_{[\alpha]})\right) \cup \left((\mathcal{S} \setminus \mathcal{S}_{[\alpha]}) \times \mathcal{S}_{[\alpha]}\right)$.

Algorithm

- Given bisimulation problem for DCLTS $\langle \mathcal{S}, \mathcal{S}_{init}, \mathcal{E}, \mathcal{T}_{\mathcal{E}}, \mathcal{C}, c \rangle$:
- Transitions: $\mathcal{T}_{\mathcal{E}} \subseteq \mathcal{S} \times \mathcal{S}$ (and the corresponding $\mathcal{T}'_{\mathcal{E}}$)

Algorithm: *SaturationBisimulation*$(\mathcal{S}, \mathcal{T}'_{\mathcal{E}})$

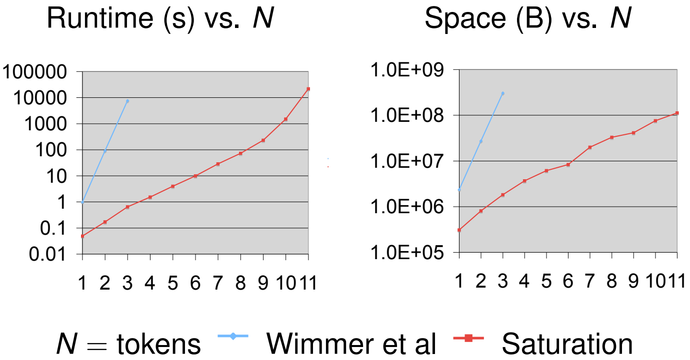
- Define: $\hat{\mathcal{B}} = \mathcal{S} \times \mathcal{S}$
- For $(\alpha \in \mathcal{E})$ loop: $\mathcal{P}_\alpha \leftarrow (\mathcal{T}'_\alpha \times \mathcal{T}'_\alpha)^{-1}$
- For $(\beta \in K : 1)$ loop: $\mathcal{P}'_{2\beta} \leftarrow \bigcup_{\alpha | Top(\mathcal{T}_\alpha)=\beta} \mathcal{P}_\alpha$ merge by level
- Construct: $\overline{\mathcal{B}}_{init} \leftarrow \bigcup_{\alpha\in\mathcal{E}} (\mathcal{S}_{[\alpha]} \times (\mathcal{S} \setminus \mathcal{S}_{[\alpha]})) \cup ((\mathcal{S} \setminus \mathcal{S}_{[\alpha]}) \times \mathcal{S}_{[\alpha]})$ where $\mathcal{S}_{[\alpha]} = \{s \in \mathcal{S} | \exists s' : \langle s, s' \rangle \in \mathcal{T}_\alpha\}$.
- $\overline{\approx} \leftarrow SatClosure(\mathcal{P}'_{[2K:1]}, \overline{\mathcal{B}}_{init})$
- Return: $\hat{\mathcal{B}} \setminus \overline{\approx}$ $=\sim$

Outline

1. Preliminaries
   - Abstract
   - Background
2. Our Algorithm
   - Our Contribution
   - Main Idea
   - Building the algorithm
3. Results and Future Work
   - Performance Results
   - Discussion
   - Future Work

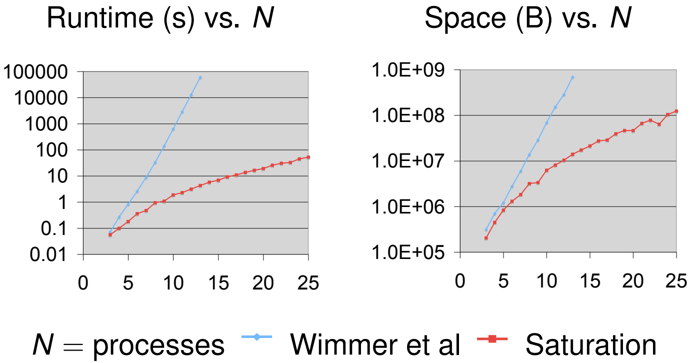
Combined N token "Kanban" Model Results

Performance for kanban model bisimulation



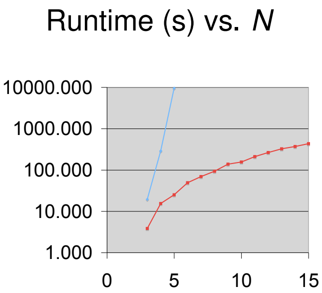
Combined N process "Robin" Model Results

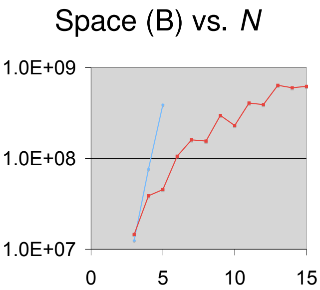
Performance for round-robin scheduler model bisimulation



Combined N process "Leader" Model Results

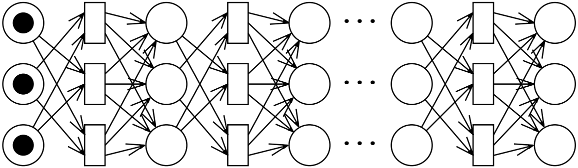
Performance for leader model bisimulation





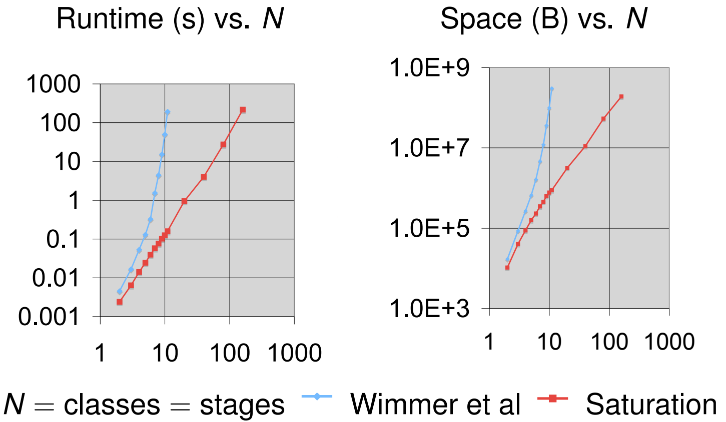
N = processes — Wimmer et al — Saturation

Cascade model



Combined 3 $\times$ N stage "Cascade" Model Results

Performance for cascade model bisimulation



Outline

1. Preliminaries
   - Abstract
   - Background
2. Our Algorithm
   - Our Contribution
   - Main Idea
   - Building the algorithm
3. Results and Future Work
   - Performance Results
   - Discussion
   - Future Work

Discussion of Results

Qualitative evaluation of Quantitative results:

- Saturation performed well in all cases (especially Robin.).
- Inflated resource usage for our implementation of Wimmer's algorithm.
- Saturation is not necessarily always fastest.

Additional Thoughts:

- Additional optimizations are possible for our algorithm.
- Our implementation of Wimmer's algorithm needs adjustment.

Outline

Future Work

Improvements to current work:

- Extend to non-deterministic transitions.
- “Weak” bisimulation (invisible transitions).
- SMART Additional optimization and tuning.

Summary

- Implementation of bisimulation algorithms in SMART
- Comparison using Petri net models.
- Obtained fully-symbolic algorithm with good performance even with many classes

Acknowledgement

This work was supported in part by the National Science Foundation under Grant CCF-1018057.

The End

fin