CS:4350 Logic in Computer Science

Transition Systems

Cesare Tinelli

Spring 2022

The University of Iowa

Credits

These slides are largely based on slides originally developed by **Andrei Voronkov** at the University of Manchester. Adapted by permission.

Outline

State-changing systems

Our main interest from now on is modeling *state-changing systems*

We assume a discrete notion of time, with each time corresponding to a *step* taken by the system

Informally	
At each step, the system is in a particular state	
The system state changes over time There are actions (controlled or not) that change the state	

State-changing systems

Our main interest from now on is modeling *state-changing systems*

We assume a discrete notion of time, with each time corresponding to a *step* taken by the system

Informally	Formally
At each step, the system is in a particular *state*	States can be characterized by values of a set of variables, called the *state variables*
The system state changes over time There are *actions* (controlled or not) that change the state	Actions *change values* of some state variables

State-changing systems

Our main interest from now on is modeling *state-changing systems*

We assume a discrete notion of time, with each time corresponding to a *step* taken by the system

Informally	Formally
At each step, the system is in a particular state	States can be characterized by values of a set of variables, called the *state variables*
The system state changes over time There are actions (controlled or not) that change the state	Actions change values of some state variables

Reasoning about state-changing systems

1. Build a formal model of this state-changing system describing
   - the behavior of the system, or
   - some abstraction of that behavior
2. Using a logic to specify and verify properties of the system

Reasoning about state-changing systems

1. Build a formal model of this state-changing system describing
   - the behavior of the system, or
   - some abstraction of that behavior
2. Using a logic to specify and verify properties of the system

Example, Vending machine

A state-changing system: vending machine dispensing drinks

- The machine has several components, including:
  - storage space for storing and preparing drinks,
  - a dispenser for the purchased drink, and
  - a coin slot to pay for the drink
- When the machine is operating, it goes through several states, depending on the behavior of the current customer
- Each action by the customer or the machine itself may change its state

  Ex: when the customer inserts a coin, the amount of money stored in the slot changes

Example, Vending machine

A state-changing system: vending machine dispensing drinks

- The machine has several components, including:
  - storage space for storing and preparing drinks,
  - a dispenser for the purchased drink, and
  - a coin slot to pay for the drink
- When the machine is operating, it goes through several states, depending on the behavior of the current customer
- Each action by the customer or the machine itself may change its state

  Ex: when the customer inserts a coin, the amount of money stored in the slot changes

Example, Vending machine

A state-changing system: vending machine dispensing drinks

- The machine has several components, including:
  - storage space for storing and preparing drinks,
  - a dispenser for the purchased drink, and
  - a coin slot to pay for the drink
- When the machine is operating, it goes through several states, depending on the behavior of the current customer
- Each action by the customer or the machine itself may change its state

  Ex: when the customer inserts a coin, the amount of money stored in the slot changes

State transition: action that may change the machine's state

Modeling state-changing systems

To build a formal model of a particular state-changing system, we specify its behavior in terms of

1. its state variables
2. the possible values for the state variables
3. the state transitions and how they change the values of the state variables

A state can be identified with

- the set of pairs (*variable*, *value*), or
- a function from variables to values

Formally, the system can be modeled as a transition system

Modeling state-changing systems

To build a formal model of a particular state-changing system, we specify its behavior in terms of

1. its state variables
2. the possible values for the state variables
3. the state transitions and how they change the values of the state variables

A state can be identified with

- the set of pairs $(\mathit{variable}, \mathit{value})$, or
- a function from variables to values

Formally, the system can be modeled as a transition system

Modeling state-changing systems

To build a formal model of a particular state-changing system, we specify its behavior in terms of

1. its state variables
2. the possible values for the state variables
3. the state transitions and how they change the values of the state variables

A state can be identified with

- the set of pairs $(variable, value)$, or
- a function from variables to values

Formally, the system can be modeled as a transition system

Transition systems

A *transition system* is a tuple $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$, where

1. S is a finite non-empty set, the set of *states* of $\mathbb{S}$
2. $In \subseteq S$ is a non-empty set of states, the set of *initial states* of $\mathbb{S}$
3. $T \subseteq S \times S$ is a set of state pairs, the transition relation of $\mathbb{S}$

Transition systems

A *transition system* is a tuple $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$, where

1. S is a finite non-empty set, the set of *states* of $\mathbb{S}$
2. $In \subseteq S$ is a non-empty set of states, the set of *initial states* of $\mathbb{S}$
3. $T \subseteq S \times S$ is a set of state pairs, the *transition relation* of $\mathbb{S}$

Transition systems

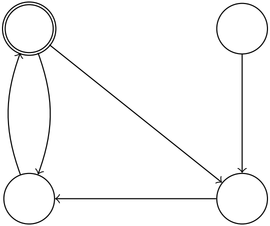
A *transition system* is a tuple $\mathbb{S} = (S, \mathit{In}, T, \mathcal{X}, \mathit{dom}, L)$, where

1. S is a finite non-empty set, the set of *states* of $\mathbb{S}$
2. $\mathit{In} \subseteq S$ is a non-empty set of states, the set of *initial states* of $\mathbb{S}$
3. $T \subseteq S \times S$ is a set of state pairs, the transition relation of $\mathbb{S}$

State Transition Graph

State Transition Graph of a transition system $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$:

- The nodes are the states of $\mathbb{S}$
- The arcs are elements of the transition relation:
  there is an edge from state s to state s' iff $(s, s') \in T$

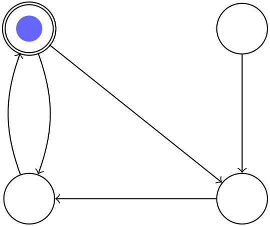


We denote the initial states with double circles

State Transition Graph

State Transition Graph of a transition system $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$:

- The nodes are the states of $\mathbb{S}$
- The arcs are elements of the transition relation: there is an edge from state s to state s' iff $(s, s') \in T$

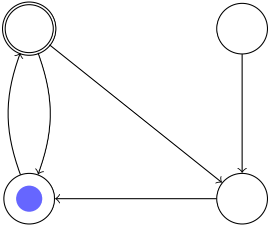


We denote the initial states with double circles

State Transition Graph

State Transition Graph of a transition system $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$:

- The nodes are the states of $\mathbb{S}$
- The arcs are elements of the transition relation: there is an edge from state s to state s' iff $(s, s') \in T$

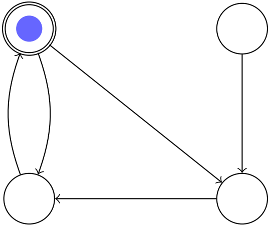


We denote the initial states with double circles

State Transition Graph

State Transition Graph of a transition system $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$:

- The nodes are the states of $\mathbb{S}$
- The arcs are elements of the transition relation:
  there is an edge from state s to state s' iff $(s, s') \in T$

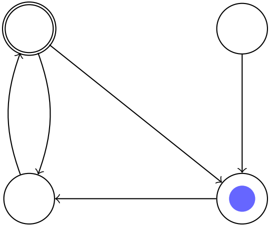


We denote the initial states with double circles

State Transition Graph

State Transition Graph of a transition system $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$:

- The nodes are the states of $\mathbb{S}$
- The arcs are elements of the transition relation: there is an edge from state s to state s' iff $(s, s') \in T$

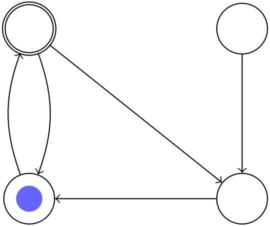


We denote the initial states with double circles

State Transition Graph

State Transition Graph of a transition system $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$:

- The nodes are the states of $\mathbb{S}$
- The arcs are elements of the transition relation:
  there is an edge from state s to state s' iff $(s, s') \in T$

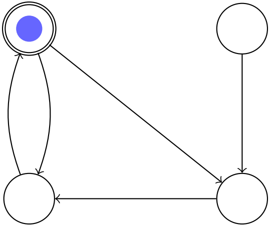


We denote the initial states with double circles

State Transition Graph

State Transition Graph of a transition system $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$:

- The nodes are the states of $\mathbb{S}$
- The arcs are elements of the transition relation: there is an edge from state s to state s' iff $(s, s') \in T$



We denote the initial states with double circles

Labelled transition systems

A *(labelled) transition system* is a tuple $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$, where

1. S is a finite non-empty set, the set of *states* of $\mathbb{S}$
2. $In \subseteq S$ is a non-empty set of states, the set of *initial states* of $\mathbb{S}$
3. $T \subseteq S \times S$ is a set of state pairs, the transition relation of $\mathbb{S}$
4. $\mathcal{X}$ is a finite set of *state variables*
5. dom is a mapping from $\mathcal{X}$ such that
   for every $x \in \mathcal{X}$, $dom(x)$ is a non-empty set of values, the *domain of x*
6. L is a function mapping states of S to interpretations, the *labelling function* of $\mathbb{S}$ (more on this later)

Labelled transition systems

A *(labelled) transition system* is a tuple $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$, where

1. S is a finite non-empty set, the set of *states* of $\mathbb{S}$
2. $In \subseteq S$ is a non-empty set of states, the set of *initial states* of $\mathbb{S}$
3. $T \subseteq S \times S$ is a set of state pairs, the transition relation of $\mathbb{S}$
4. $\mathcal{X}$ is a finite set of *state variables*
5. dom is a mapping from $\mathcal{X}$ such that
   for every $x \in \mathcal{X}$, $dom(x)$ is a non-empty set of values, the *domain of x*
6. L is a function mapping states of S to interpretations, the *labelling function* of $\mathbb{S}$ (more on this later)

Labelled transition systems

A *(labelled) transition system* is a tuple $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$, where

1. S is a finite non-empty set, the set of *states* of $\mathbb{S}$
2. $In \subseteq S$ is a non-empty set of states, the set of *initial states* of $\mathbb{S}$
3. $T \subseteq S \times S$ is a set of state pairs, the transition relation of $\mathbb{S}$
4. $\mathcal{X}$ is a finite set of *state variables*
5. dom is a mapping from $\mathcal{X}$ such that
   for every $x \in \mathcal{X}$, $dom(x)$ is a non-empty set of values, the *domain of* x
6. L is a function mapping states of S to interpretations, the *labelling function* of $\mathbb{S}$ (more on this later)

Labelled transition systems

A *(labelled) transition system* is a tuple $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$, where

1. S is a finite non-empty set, the set of *states* of $\mathbb{S}$
2. $In \subseteq S$ is a non-empty set of states, the set of *initial states* of $\mathbb{S}$
3. $T \subseteq S \times S$ is a set of state pairs, the transition relation of $\mathbb{S}$
4. $\mathcal{X}$ is a finite set of *state variables*
5. dom is a mapping from $\mathcal{X}$ such that
   for every $x \in \mathcal{X}$, $dom(x)$ is a non-empty set of values, the *domain of* x
6. L is a function mapping states of S to interpretations, the *labelling function* of $\mathbb{S}$ (more on this later)

Labelled transition systems

A *(labelled) transition system* is a tuple $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$, where

1. S is a finite non-empty set, the set of *states* of $\mathbb{S}$
2. $In \subseteq S$ is a non-empty set of states, the set of *initial states* of $\mathbb{S}$
3. $T \subseteq S \times S$ is a set of state pairs, the transition relation of $\mathbb{S}$
4. $\mathcal{X}$ is a finite set of *state variables*
5. dom is a mapping from $\mathcal{X}$ such that
   for every $x \in \mathcal{X}$, $dom(x)$ is a non-empty set of values, the *domain of* x
6. L is a function mapping states of S to interpretations, the *labelling function* of $\mathbb{S}$ (more on this later)

$\mathbb{S}$ is *finite-state* if $dom(x)$ is finite for all $x \in \mathcal{X}$

Labelled transition systems

A *(labelled) transition system* is a tuple $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$, where

1. S is a finite non-empty set, the set of *states* of $\mathbb{S}$
2. $In \subseteq S$ is a non-empty set of states, the set of *initial states* of $\mathbb{S}$
3. $T \subseteq S$
4. $\mathcal{X}$ is a
5. dom is
   for every $x \in \mathcal{X}$, $dom(x)$ is a non-empty set of values, the *domain of* x
6. L is a function mapping states of S to interpretations, the *labelling function* of $\mathbb{S}$ (more on this later)

We will only study finite-state transition systems

$\mathbb{S}$ is *finite-state* if $dom(x)$ is finite for all $x \in \mathcal{X}$

Systems in PLFD

$$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$$

Note this part of the definition transition system:

4. $\mathcal{X}$ is a finite set of *state variables*
5. dom is a mapping from $\mathcal{X}$ such that
   for every $x \in \mathcal{X}$, $dom(x)$ is a non-empty set of values, the *domain of* x

Systems in PLFD

$$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$$

Note this part of the definition transition system:

4. $\mathcal{X}$ is a finite set of *state variables*
5. dom is a mapping from $\mathcal{X}$ such that
   for every $x \in \mathcal{X}$, $dom(x)$ is a non-empty set of values, the *domain of* x

$\mathcal{X}$ and dom define an instance of PLFD!

Labeling function

$$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$$

Let $\mathbb{I}$ be the set of all interpretations for the instance of PLFD given by $\mathcal{X}$ and dom

1. for every $x \in \mathcal{X}$ and $s \in S$, we have $L(s)(x) \in dom(x)$
2. for every formula A over $\mathcal{X}$ and every state $s \in S$, either $L(s) \models A$ or $L(s) \not\models A$

Labeling function

$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$

Let $\mathbb{I}$ be the set of all interpretations for the instance of PLFD given by $\mathcal{X}$ and dom

L is a mapping from S to $\mathbb{I}$, associating every state to an interpretation of $\mathcal{X}$:

1. for every $x \in \mathcal{X}$ and $s \in S$, we have $L(s)(x) \in dom(x)$
2. for every formula A over $\mathcal{X}$ and every state $s \in S$, either $L(s) \models A$ or $L(s) \not\models A$

Labeling function

$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$

Let $\mathbb{I}$ be the set of all interpretations for the instance of PLFD given by $\mathcal{X}$ and dom

L is a mapping from S to $\mathbb{I}$, associating every state to an interpretation of $\mathcal{X}$:

1. for every $x \in \mathcal{X}$ and $s \in S$, we have $L(s)(x) \in dom(x)$
2. for every formula A over $\mathcal{X}$ and every state $s \in S$, either $L(s) \models A$ or $L(s) \not\models A$

Labeling function

$$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$$

Let $\mathbb{I}$ be the set of all interpretations for the instance of PLFD given by $\mathcal{X}$ and dom

L is a mapping from S to $\mathbb{I}$, associating every state to an interpretation of $\mathcal{X}$:

1. for every $x \in \mathcal{X}$ and $s \in S$, we have $L(s)(x) \in dom(x)$
2. for every formula A over $\mathcal{X}$ and every state $s \in S$, either $L(s) \models A$ or $L(s) \not\models A$

States as interpretations

If $L(s)(x) = v$, we say that x *has value* v *in state* s, and write $s(x) = v$

If $L(s) \models A$, we say that s *satisfies* A *or* A *is true in* s, and write $s \models A$

We will often identify s with $L(s)$

States as interpretations

If $L(s)(x) = v$, we say that x *has value* v *in state* s, and write $s(x) = v$

If $L(s) \models A$, we say that s *satisfies* A *or* A *is true in* s, and write $s \models A$

We will often *identify* s with $L(s)$

States as interpretations

If $L(s)(x) = v$, we say that x *has value* v *in state* s, and write $s(x) = v$

If $L(s) \models A$, we say that s *satisfies* A *or* A *is true in* s, and write $s \models A$

We will often identify s with $L(s)$

State Transition Graph

$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$

State transition graph of $\mathbb{S}$:

- The nodes are the states of $\mathbb{S}$
- The edges are the state pairs in T
- Nodes labeled by states

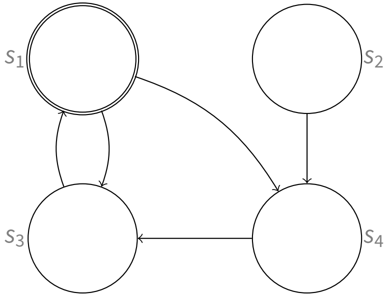
State Transition Graph

$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$

State transition graph of $\mathbb{S}$:

- The nodes are the states of $\mathbb{S}$
- The edges are the state pairs in T
- Nodes labeled by states

Example: $\mathcal{X} = \{x, y\}$, $dom(x) = dom(y) = \{0, 1\}$



$$
\begin{aligned}
S &= \{ s_1, s_2, s_3, s_4 \} \\
In &= \{ s_1 \} \\
T &= \{ (s_1, s_3), (s_1, s_4), (s_3, s_1), \\
&\quad (s_4, s_2), (s_4, s_3) \} \\
L &= \{ s_1 \mapsto \{ x \mapsto 1, y \mapsto 0 \}, \\
&\quad s_2 \mapsto \{ x \mapsto 1, y \mapsto 1 \}, \\
&\quad s_3 \mapsto \{ x \mapsto 0, y \mapsto 0 \}, \\
&\quad s_4 \mapsto \{ x \mapsto 0, y \mapsto 1 \} \}
\end{aligned}
$$

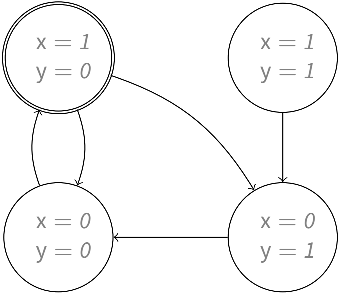
State Transition Graph with interpretations as states

$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$

State transition graph of $\mathbb{S}$:

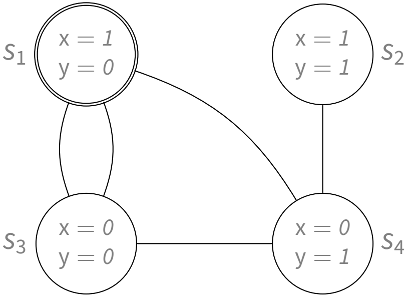
- The nodes are the states of $\mathbb{S}$
- The edges are the state pairs in T
- Nodes labeled by states

Example: $\mathcal{X} = \{x, y\}$, $dom(x) = dom(y) = \{0, 1\}$



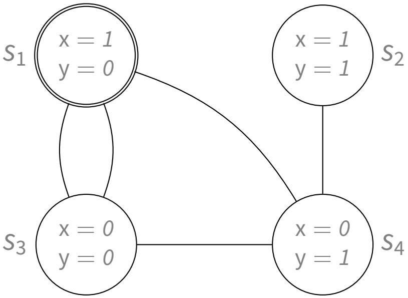
$$
\begin{array}{lcl}
 & & \quad x \;\; y \\
S & = & \{ (0,0), (0,1), (1,0), (1,1) \} \\
In & = & \{ (1,0) \} \\
T & = & \{ ((1,0),(0,1)), \\
 & & \;\; ((1,0),(0,0)), \\
 & & \;\; ((0,0),(1,0)), \\
 & & \;\; ((0,1),(0,0)), \\
 & & \;\; ((1,1),(0,1)) \}
\end{array}
$$

States as Interpretations



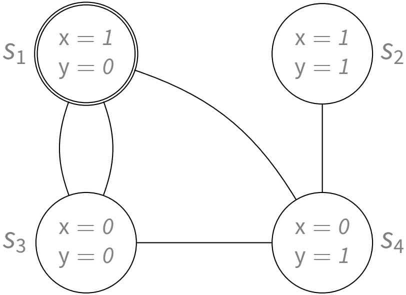
- $s_1 \models x$
- $s_2 \models x \land y$
- $s_3 \models x \leftrightarrow y$

States as Interpretations



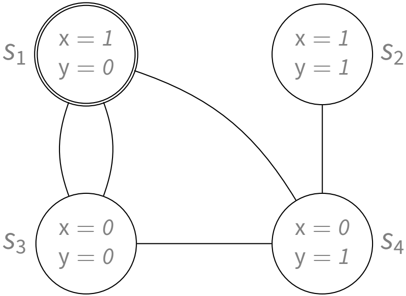
- $s_1 \models x$
- $s_2 \models x \wedge y$
- $s_3 \models x \leftrightarrow y$

States as Interpretations



- $s_1 \models x$
- $s_2 \models x \wedge y$
- $s_3 \models x \leftrightarrow y$

States as Interpretations



- $s_1 \models x$
- $s_2 \models x \wedge y$
- $s_3 \models x \leftrightarrow y$

Transitions

We will usually represent a transition relation as a union of transitions

Transition t: any set of state pairs

A transition t is

- *applicable* to a state s if there is a state s' such that $(s, s') \in t$
- *deterministic* if for every state s there is at most one state s' such that $(s, s') \in t$

Transitions

We will usually represent a transition relation as a union of transitions

Transition t: any set of state pairs

A transition t is

- *applicable* to a state s if there is a state s' such that $(s, s') \in t$
- *deterministic* if for every state s there is at most one state s' such that $(s, s') \in t$

Transitions

We will usually represent a transition relation as a union of transitions

Transition t: any set of state pairs

A transition t is

- *applicable* to a state s if there is a state s' such that $(s, s') \in t$
- *deterministic* if for every state s there is at most one state s' such that $(s, s') \in t$

Transitions

We will usually represent a transition relation as a union of transitions

Transition t: any set of state pairs

A transition t is

- *applicable* to a state s if there is a state s' such that $(s, s') \in t$
- *deterministic* if for every state s there is at most one state s' such that $(s, s') \in t$

Vending machine

1. The vending machine contains a drink storage, a coin slot, and a drink dispenser.

2. The drink storage stores drinks of two kinds: soda and coffee. We are only interested in whether a particular kind of drink is currently being stored or not (but not its amount).

3. The coin slot can accommodate up to three coins.

4. The drink dispenser can hold at most one drink. Any drink in it must be removed before the next one can be dispensed.

5. A can of soda costs two coins. A cup of coffee costs one coin.

6. There are two kinds of customers: students and professors. Students drink only soda, professors drink only coffee.

7. From time to time the drink storage can be restocked.

Vending machine

1. The vending machine contains a drink storage, a coin slot, and a drink dispenser.

2. The drink storage stores drinks of two kinds: soda and coffee. We are only interested in whether a particular kind of drink is currently being stored or not (but not its amount).

3. The coin slot can accommodate up to three coins.

4. The drink dispenser can hold at most one drink. Any drink in it must be removed before the next one can be dispensed.

5. A can of soda costs two coins. A cup of coffee costs one coin.

6. There are two kinds of customers: students and professors. Students drink only soda, professors drink only coffee.

7. From time to time the drink storage can be restocked.

Vending machine

1. The vending machine contains a drink storage, a coin slot, and a drink dispenser.
2. The drink storage stores drinks of two kinds: soda and coffee. We are only interested in whether a particular kind of drink is currently being stored or not (but not its amount).
3. The coin slot can accommodate up to three coins.
4. The drink dispenser can hold at most one drink. Any drink in it must be removed before the next one can be dispensed.
5. A can of soda costs two coins. A cup of coffee costs one coin.
6. There are two kinds of customers: students and professors. Students drink only soda, professors drink only coffee.
7. From time to time the drink storage can be restocked.

Vending machine

1. The vending machine contains a drink storage, a coin slot, and a drink dispenser.
2. The drink storage stores drinks of two kinds: soda and coffee. We are only interested in whether a particular kind of drink is currently being stored or not (but not its amount).
3. The coin slot can accommodate up to three coins.
4. The drink dispenser can hold at most one drink. Any drink in it must be removed before the next one can be dispensed.
5. A can of soda costs two coins. A cup of coffee costs one coin.
6. There are two kinds of customers: students and professors. Students drink only soda, professors drink only coffee.
7. From time to time the drink storage can be restocked.

Vending machine

1. The vending machine contains a drink storage, a coin slot, and a drink dispenser.
2. The drink storage stores drinks of two kinds: soda and coffee. We are only interested in whether a particular kind of drink is currently being stored or not (but not its amount).
3. The coin slot can accommodate up to three coins.
4. The drink dispenser can hold at most one drink. Any drink in it must be removed before the next one can be dispensed.
5. A can of soda costs two coins. A cup of coffee costs one coin.
6. There are two kinds of customers: students and professors. Students drink only soda, professors drink only coffee.
7. From time to time the drink storage can be restocked.

Formalization: Variables and Domains

variable	domain	explanation
st_coffee	$\{0, 1\}$	drink storage contains coffee
st_soda	$\{0, 1\}$	drink storage contains soda
disp	$\{\mathit{none}, \mathit{soda}, \mathit{coffee}\}$	content of drink dispenser
coins	$\{\mathit{0}, \mathit{1}, \mathit{2}, \mathit{3}\}$	number of coins in the slot
customer	$\{\mathit{none}, \mathit{student}, \mathit{prof}\}$	customer

Transitions for the Vending Machine

1. *Restock*, results in the drink storage having both soda and coffee
2. *Customer_arrives*, corresponds to a customer arriving at the machine
3. *Customer_leaves*, corresponds to the customer's leaving
4. *Coin_insert*, corresponds to the customer's inserting a coin in the machine
5. *Dispense_soda*, results in the customer's getting a can of soda
6. *Dispense_coffee*, results in the customer's getting a cup of coffee
7. *Take_drink*, corresponds to the customer's removing a drink from the dispenser

Symbolic Representation of Sets of States

Let $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$ be a (finite-state) labelled transition system

Every PLFD formula F over the variables in $\mathcal{X}$ defines a set states:

$$\{ s \in S \mid s \models F \}$$

Symbolic Representation of Sets of States

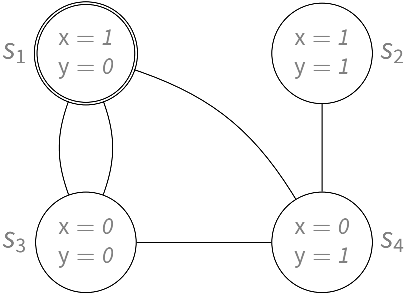
Let $\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$ be a (finite-state) labelled transition system

Every PLFD formula F over the variables in $\mathcal{X}$ defines a set states:

$$\{\, s \in S \mid s \models F \,\}$$

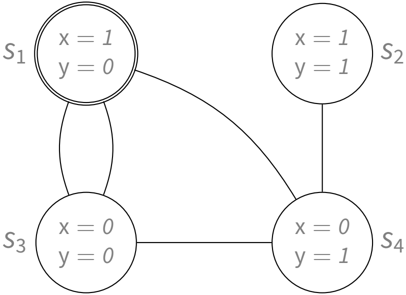
We say that F *(symbolically) represents* this set of states

Symbolic Representation of Sets of States



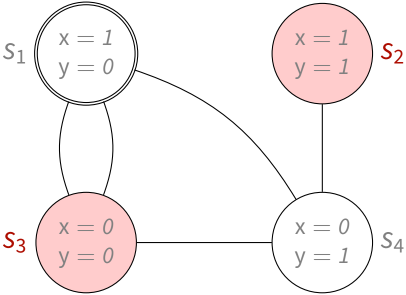
- $x \leftrightarrow y$
- $x \wedge y$
- $\neg x$

Symbolic Representation of Sets of States



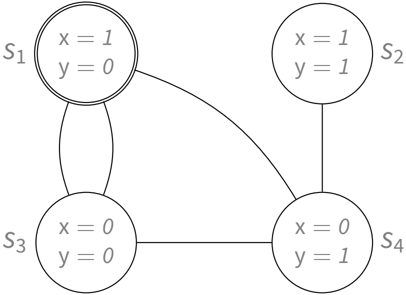
- $x \leftrightarrow y$
- $x \wedge y$
- $\neg x$

Symbolic Representation of Sets of States



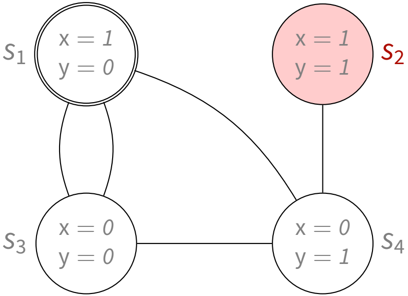
- $x \leftrightarrow y$ represents $\{ s_2, s_3 \}$
- $x \wedge y$
- $\neg x$

Symbolic Representation of Sets of States



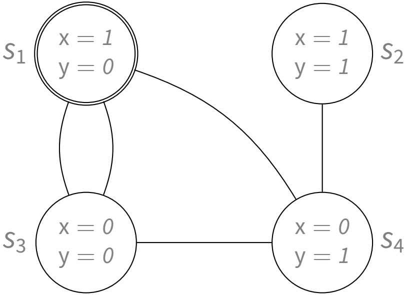
- $x \leftrightarrow y$ represents $\{ s_2, s_3 \}$
- $x \land y$
- $\neg x$

Symbolic Representation of Sets of States



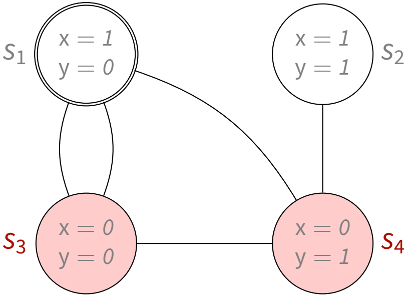
- $x \leftrightarrow y$ represents $\{ s_2, s_3 \}$
- $x \wedge y$ represents $\{ s_2 \}$
- $\neg x$

Symbolic Representation of Sets of States



- $x \leftrightarrow y$ represents $\{ s_2, s_3 \}$
- $x \wedge y$ represents $\{ s_2 \}$
- $\neg x$

Symbolic Representation of Sets of States



- $x \leftrightarrow y$ represents $\{ s_2, s_3 \}$
- $x \wedge y$ represents $\{ s_2 \}$
- $\neg x$ represents $\{ s_3, s_4 \}$

Example

Let us represent the set of states in which the machine is ready to dispense a drink

In every such state, it must be the case that

- a drink is available
- the drink dispenser is empty, and
- the coin slot contains enough coins

Example

Let us represent the set of states in which the machine is ready to dispense a drink

In every such state, it must be the case that

- a drink is available
- the drink dispenser is empty, and
- the coin slot contains enough coins

This can be expressed by:

$$
\begin{aligned}
&(\mathrm{st\_coffee} \lor \mathrm{st\_soda}) \land \\
&\mathrm{disp} = \mathit{none} \land \\
&((\mathrm{coins} = \mathit{1} \land \mathrm{st\_coffee}) \lor \mathrm{coins} = \mathit{2} \lor \mathrm{coins} = \mathit{3})
\end{aligned}
$$

Symbolic Representation of Transitions

$$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$$

A *transition* t in $\mathbb{S}$ is a binary relation on s, i.e., a set of state pairs:

$$t = \{ (s, s') \mid s, s' \in S \}$$

It takes the system from some *current state* or *pre-state* s
to some *next state* or *post-state* s'

Can we represent transitions symbolically using PLFD formulas?

Not immediately.

PLFD formulas over $\mathcal{X}$ can only express properties of a single state

Symbolic Representation of Transitions

$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$

A *transition* t in $\mathbb{S}$ is a binary relation on s, i.e., a set of state pairs:

$$t = \{\, (s, s') \mid s, s' \in S \,\}$$

It takes the system from some *current state* or *pre-state* s
to some *next state* or *post-state* s'

Can we represent transitions symbolically using PLFD formulas?

Not immediately.

PLFD formulas over $\mathcal{X}$ can only express properties of a single state

Symbolic Representation of Transitions

$$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$$

A *transition* t in $\mathbb{S}$ is a binary relation on s, i.e., a set of state pairs:

$$t = \{ (s, s') \mid s, s' \in S \}$$

It takes the system from some *current state* or *pre-state* s
to some *next state* or *post-state* s'

Can we represent transitions symbolically using PLFD formulas?

Not immediately.

PLFD formulas over $\mathcal{X}$ can only express properties of a single state

Symbolic Representation of Transitions

How can we represent transitions using formulas?

- Introduce a new set of *next-state variables* $\mathcal{X}' = \{ x' \mid x \in \mathcal{X} \}$
- Treat pairs of states as interpretations of formulas over $\mathcal{X} \cup \mathcal{X}'$

$$\text{For all } x \in \mathcal{X}, \quad (s, s')(x) \stackrel{\text{def}}{=} s(x)$$
$$\text{For all } x \in \mathcal{X}, \quad (s, s')(x') \stackrel{\text{def}}{=} s'(x)$$

- A formula F over variables $\mathcal{X} \cup \mathcal{X}'$ represents symbolically a transition t if

$$t = \{ (s, s') \in S^2 \mid (s, s') \models F \}$$

Symbolic Representation of Transitions

How can we represent transitions using formulas?

- Introduce a new set of *next-state variables* $\mathcal{X}' = \{\, x' \mid x \in \mathcal{X} \,\}$
- Treat pairs of states as interpretations of formulas over $\mathcal{X} \cup \mathcal{X}'$

$$\text{For all } x \in \mathcal{X}, \quad (s, s')(x) \stackrel{\text{def}}{=} s(x)$$
$$\text{For all } x \in \mathcal{X}, \quad (s, s')(x') \stackrel{\text{def}}{=} s'(x)$$

- A formula F over variables $\mathcal{X} \cup \mathcal{X}'$ represents symbolically a transition t if

$$t = \{\, (s, s') \in S^2 \mid (s, s') \models F \,\}$$

Symbolic Representation of Transitions

How can we represent transitions using formulas?

- Introduce a new set of *next-state variables* $\mathcal{X}' = \{ x' \mid x \in \mathcal{X} \}$
- Treat pairs of states as interpretations of formulas over $\mathcal{X} \cup \mathcal{X}'$

$$\text{For all } x \in \mathcal{X}, \quad (s, s')(x) \stackrel{\text{def}}{=} s(x)$$
$$\text{For all } x \in \mathcal{X}, \quad (s, s')(x') \stackrel{\text{def}}{=} s'(x)$$

- A formula F over variables $\mathcal{X} \cup \mathcal{X}'$ represents symbolically a transition t if

$$t = \{ (s, s') \in S^2 \mid (s, s') \models F \}$$

Symbolic Representation of Transitions

How can we represent transitions using formulas?

- Introduce a new set of *next-state variables* $\mathcal{X}' = \{ x' \mid x \in \mathcal{X} \}$
- Treat pairs of states as interpretations of formulas over $\mathcal{X} \cup \mathcal{X}'$

$$\begin{aligned} \text{For all } x \in \mathcal{X}, \quad (s, s')(x) &\stackrel{\text{def}}{=} s(x) \\ \text{For all } x \in \mathcal{X}, \quad (s, s')(x') &\stackrel{\text{def}}{=} s'(x) \end{aligned}$$

- A formula F over variables $\mathcal{X} \cup \mathcal{X}'$ represents symbolically a transition t if

$$t = \{ (s, s') \in S^2 \mid (s, s') \models F \}$$

Example

The transition *Restock*:

$$\text{customer} = \textit{none} \land \text{st\_coffee}' \land \text{st\_soda}'$$

Example

The transition *Restock*:

$$\underbrace{\text{customer} = \textit{none}}_{\text{current state description}} \;\wedge\; \underbrace{\text{st\_coffee}' \wedge \text{st\_soda}'}_{\text{next state description}}$$

Example

The transition *Restock*:

$$\underbrace{\mathsf{customer} = \mathit{none}}_{\text{current state description}} \;\wedge\; \underbrace{\mathsf{st\_coffee}' \wedge \mathsf{st\_soda}'}_{\text{next state description}}$$

Note: This formula describes a strange transition after which, for example

- coins may appear in and disappear from the slot
- drinks may appear in and disappear from the dispenser
- ...

Frame problem

We must express explicitly, possibly for a large number of state variables, that

the values of certain variables do not change after a transition

Example

$$(\text{coins} = 0 \leftrightarrow \text{coins}' = 0) \land$$
$$(\text{coins} = 1 \leftrightarrow \text{coins}' = 1) \land$$
$$(\text{coins} = 2 \leftrightarrow \text{coins}' = 2) \land$$
$$(\text{coins} = 3 \leftrightarrow \text{coins}' = 3)$$

This is known as the *frame problem*

It arises in artificial intelligence, knowledge representation, planning, and in reasoning about actions in general

Frame problem

We must express explicitly, possibly for a large number of state variables, that

the values of certain variables do not change after a transition

Example

$$
\begin{aligned}
(\text{coins} = 0 \leftrightarrow \text{coins}' = 0) \ \wedge \\
(\text{coins} = 1 \leftrightarrow \text{coins}' = 1) \ \wedge \\
(\text{coins} = 2 \leftrightarrow \text{coins}' = 2) \ \wedge \\
(\text{coins} = 3 \leftrightarrow \text{coins}' = 3)
\end{aligned}
$$

This is known as the *frame problem*

It arises in artificial intelligence, knowledge representation, planning, and in reasoning about actions in general

Frame problem

We must express explicitly, possibly for a large number of state variables, that

the values of certain variables do not change after a transition

Example

$$
\begin{aligned}
&(\text{coins} = 0 \leftrightarrow \text{coins}' = 0)\ \wedge\\
&(\text{coins} = 1 \leftrightarrow \text{coins}' = 1)\ \wedge\\
&(\text{coins} = 2 \leftrightarrow \text{coins}' = 2)\ \wedge\\
&(\text{coins} = 3 \leftrightarrow \text{coins}' = 3)
\end{aligned}
$$

This is known as the *frame problem*

It arises in artificial intelligence, knowledge representation, planning, and in reasoning about actions in general

Frame problem

We must express explicitly, possibly for a large number of state variables, that

the values of certain variables do not change after a transition

Example

$$
\begin{aligned}
&(\text{coins} = 0 \leftrightarrow \text{coins}' = 0)\ \land \\
&(\text{coins} = 1 \leftrightarrow \text{coins}' = 1)\ \land \\
&(\text{coins} = 2 \leftrightarrow \text{coins}' = 2)\ \land \\
&(\text{coins} = 3 \leftrightarrow \text{coins}' = 3)
\end{aligned}
$$

This is known as the *frame problem*

It arises in artificial intelligence, knowledge representation, planning, and in reasoning about actions in general

The frame formula

$$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$$

Notation:

When $dom(x) = dom(y)$,

$$x \neq v \stackrel{\text{def}}{=} \neg(x = v)$$
$$x = y \stackrel{\text{def}}{=} \bigwedge_{v \in dom(x)} (x = v \leftrightarrow y = v)$$

For $\{ x_1, \ldots, x_n \} \subseteq \mathcal{X}$,

$$only(x_1, \ldots, x_n) \stackrel{\text{def}}{=} \bigwedge_{x \in \mathcal{X} \setminus \{ x_1, \ldots, x_n \}} x = x'$$

$only(x_1, \ldots, x_n)$ can be used in symbolic transitions to state that $x_1, \ldots, x_n$ are the only variables whose values may change in the transition

The frame formula

$$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$$

Notation:

When $dom(x) = dom(y)$,

$$x \neq v \stackrel{\text{def}}{=} \neg(x = v)$$

$$x = y \stackrel{\text{def}}{=} \bigwedge_{v \in dom(x)} (x = v \leftrightarrow y = v)$$

For $\{ x_1, \ldots, x_n \} \subseteq \mathcal{X}$,

$$only(x_1, \ldots, x_n) \stackrel{\text{def}}{=} \bigwedge_{x \in \mathcal{X} \setminus \{ x_1, \ldots, x_n \}} x = x'$$

$only(x_1, \ldots, x_n)$ can be used in symbolic transitions to state that $x_1, \ldots, x_n$ are the only variables whose values may change in the transition

The frame formula

$$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$$

Notation:

When $dom(x) = dom(y)$,

$$x \neq v \stackrel{\text{def}}{=} \neg(x = v)$$
$$x = y \stackrel{\text{def}}{=} \bigwedge_{v \in dom(x)} (x = v \leftrightarrow y = v)$$

For $\{ x_1, \ldots, x_n \} \subseteq \mathcal{X}$,

$$only(x_1, \ldots, x_n) \stackrel{\text{def}}{=} \bigwedge_{x \in \mathcal{X} \setminus \{ x_1, \ldots, x_n \}} x = x'$$

$only(x_1, \ldots, x_n)$ can be used in symbolic transitions to state that $x_1, \ldots, x_n$ are the only variables whose values may change in the transition

Preconditions and postconditions

$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$

Typical symbolic representation of a transition t in $\mathbb{S}$:

A PLFD formula $F_1 \wedge F_2$ where

1. F_1 is a formula over variables $\mathcal{X}$ expressing t's precondition
2. F_2 is a formula over $\mathcal{X} \cup \mathcal{X}'$ expressing t's postcondition

Preconditions and postconditions

$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$

Typical symbolic representation of a transition t in $\mathbb{S}$:

A PLFD formula $F_1 \wedge F_2$ where

1. F_1 is a formula over variables $\mathcal{X}$ expressing t's precondition
2. F_2 is a formula over $\mathcal{X} \cup \mathcal{X}'$ expressing t's postcondition

Preconditions and postconditions

$$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$$

Typical symbolic representation of a transition t in $\mathbb{S}$:

A PLFD formula $F_1 \wedge F_2$ where

1. F_1 is a formula over variables $\mathcal{X}$ expressing t's precondition
2. F_2 is a formula over $\mathcal{X} \cup \mathcal{X}'$ expressing t's postcondition

Preconditions and postconditions

$\mathbb{S} = (S, In, T, \mathcal{X}, dom, L)$

Typical symbolic representation of a transition t in $\mathbb{S}$:

A PLFD formula $F_1 \wedge F_2$ where

1. F_1 is a formula over variables $\mathcal{X}$ expressing t's precondition
2. F_2 is a formula over $\mathcal{X} \cup \mathcal{X}'$ expressing t's postcondition

precondition: a necessary condition for a state of $\mathbb{S}$ of to be a pre-state of t
postcondition: a condition relating t's post-states to their corresponding pre-state

Transitions for the Vending Machine

1. *Restock*, results in the drink storage having both soda and coffee
2. *Customer_arrives*, corresponds to a customer arriving at the machine
3. *Customer_leaves*, corresponds to the customer’s leaving
4. *Coin_insert*, corresponds to the customer’s inserting a coin in the machine
5. *Dispense_soda*, results in the customer’s getting a can of soda
6. *Dispense_coffee*, results in the customer’s getting a cup of coffee
7. *Take_drink*, corresponds to the customer’s removing a drink from the dispenser

Transitions: Symbolic Representation

Restock
Customer_arrives
Customer_leaves
Coin_insert

Transitions: Symbolic Representation

Restock
Customer_arrives
Customer_leaves
Coin_insert

precondition *postcondition*

$$\mathit{Restock} \stackrel{\text{def}}{=} \text{customer} = \mathit{none} \wedge \text{st\_coffee}' \wedge \text{st\_soda}' \wedge \mathit{only}(\text{st\_coffee}, \text{st\_soda})$$

Transitions: Symbolic Representation

Restock
Customer_arrives
Customer_leaves
Coin_insert

$$
\begin{aligned}
& && \textit{precondition} && \textit{postcondition} \\
\textit{Restock} &\stackrel{\text{def}}{=} && \text{customer} = \textit{none} && \wedge\ \text{st\_coffee}' \wedge \text{st\_soda}' \\
& && && \wedge\ \textit{only}(\text{st\_coffee}, \text{st\_soda})
\end{aligned}
$$

Transitions: Symbolic Representation

Restock
Customer_arrives
Customer_leaves
Coin_insert

$$\begin{aligned} & & \textit{precondition} & & \textit{postcondition} \\ \textit{Restock} & \stackrel{\text{def}}{=} & \text{customer} = \textit{none} & \wedge & \text{st\_coffee}' \wedge \text{st\_soda}' \\ & & & \wedge & \textit{only}(\text{st\_coffee}, \text{st\_soda}) \\ \textit{Customer\_arrives} & \stackrel{\text{def}}{=} & \text{customer} = \textit{none} & \wedge & \text{customer}' \neq \textit{none} \\ & & & \wedge & \textit{only}(\text{customer}) \end{aligned}$$

Transitions: Symbolic Representation

$$
\begin{array}{rcll}
 & & \textit{precondition} & \textit{postcondition} \\
\textit{Restock} & \stackrel{\text{def}}{=} & \text{customer} = \textit{none} & \wedge\ \text{st\_coffee}' \wedge \text{st\_soda}' \\
 & & & \wedge\ \textit{only}(\text{st\_coffee}, \text{st\_soda}) \\
\textit{Customer\_arrives} & \stackrel{\text{def}}{=} & \text{customer} = \textit{none} & \wedge\ \text{customer}' \neq \textit{none} \\
 & & & \wedge\ \textit{only}(\text{customer})
\end{array}
$$

Transitions: Symbolic Representation

Restock
Customer_arrives
Customer_leaves
Coin_insert

$$
\begin{array}{rcll}
 & & \textit{precondition} & \quad \textit{postcondition} \\
\textit{Restock} & \stackrel{\text{def}}{=} & \text{customer} = \textit{none} & \wedge\ \text{st\_coffee}' \wedge \text{st\_soda}' \\
 & & & \wedge\ \textit{only}(\text{st\_coffee}, \text{st\_soda}) \\
\textit{Customer\_arrives} & \stackrel{\text{def}}{=} & \text{customer} = \textit{none} & \wedge\ \text{customer}' \neq \textit{none} \\
 & & & \wedge\ \textit{only}(\text{customer}) \\
\textit{Customer\_leaves} & \stackrel{\text{def}}{=} & \text{customer} \neq \textit{none} & \wedge\ \text{customer}' = \textit{none} \\
 & & & \wedge\ \textit{only}(\text{customer})
\end{array}
$$

Transitions: Symbolic Representation

Restock
Customer_arrives
Customer_leaves
Coin_insert

$$
\begin{array}{rcll}
 & & \textit{precondition} & \quad \textit{postcondition} \\
\textit{Restock} & \stackrel{\text{def}}{=} & \text{customer} = \textit{none} & \wedge\ \text{st\_coffee}' \wedge \text{st\_soda}' \\
 & & & \wedge\ \textit{only}(\text{st\_coffee}, \text{st\_soda}) \\
\textit{Customer\_arrives} & \stackrel{\text{def}}{=} & \text{customer} = \textit{none} & \wedge\ \text{customer}' \neq \textit{none} \\
 & & & \wedge\ \textit{only}(\text{customer}) \\
\textit{Customer\_leaves} & \stackrel{\text{def}}{=} & \text{customer} \neq \textit{none} & \wedge\ \text{customer}' = \textit{none} \\
 & & & \wedge\ \textit{only}(\text{customer})
\end{array}
$$

Transitions: Symbolic Representation

Restock
Customer_arrives
Customer_leaves
Coin_insert

$$
\begin{aligned}
& && \textit{precondition} && \textit{postcondition} \\
\textit{Restock} &\overset{\text{def}}{=} && \text{customer} = \textit{none} && \wedge\ \text{st\_coffee}' \wedge \text{st\_soda}' \\
& && && \wedge\ \textit{only}(\text{st\_coffee}, \text{st\_soda}) \\
\textit{Customer\_arrives} &\overset{\text{def}}{=} && \text{customer} = \textit{none} && \wedge\ \text{customer}' \neq \textit{none} \\
& && && \wedge\ \textit{only}(\text{customer}) \\
\textit{Customer\_leaves} &\overset{\text{def}}{=} && \text{customer} \neq \textit{none} && \wedge\ \text{customer}' = \textit{none} \\
& && && \wedge\ \textit{only}(\text{customer}) \\
\textit{Coin\_insert} &\overset{\text{def}}{=} && \text{customer} \neq \textit{none} && \wedge\ (\text{coins} = \mathit{0} \rightarrow \text{coins}' = \mathit{1}) \\
& \wedge && \text{coins} \neq \mathit{3} && \wedge\ (\text{coins} = \mathit{1} \rightarrow \text{coins}' = \mathit{2}) \\
& && && \wedge\ (\text{coins} = \mathit{2} \rightarrow \text{coins}' = \mathit{3}) \\
& && && \wedge\ \textit{only}(\text{coins})
\end{aligned}
$$

Transitions

Dispense_soda
Dispense_coffee
Take_drink

Transitions

Dispense_soda
Dispense_coffee
Take_drink

$$\mathit{Dispense\_soda} \stackrel{\text{def}}{=} \begin{array}{l} \text{customer} = \mathit{student} \land \text{st\_soda} \land \\ \text{disp} = \mathit{none} \land (\text{coins} = \mathit{2} \lor \text{coins} = \mathit{3}) \land \\ (\text{coins} = \mathit{2} \rightarrow \text{coins}' = \mathit{0}) \land \\ (\text{coins} = \mathit{3} \rightarrow \text{coins}' = \mathit{1}) \land \\ \text{disp}' = \mathit{soda} \land \mathit{only}(\text{st\_soda}, \text{disp}, \text{coins}) \end{array}$$

Transitions

Dispense_soda
Dispense_coffee
Take_drink

$$\begin{aligned} \textit{Dispense\_soda} \;\stackrel{\text{def}}{=}\; & \text{customer} = \textit{student} \land \text{st\_soda} \land \\ & \text{disp} = \textit{none} \land (\text{coins} = 2 \lor \text{coins} = 3) \land \\ & (\text{coins} = 2 \rightarrow \text{coins}' = 0) \land \\ & (\text{coins} = 3 \rightarrow \text{coins}' = 1) \land \\ & \text{disp}' = \textit{soda} \land \textit{only}(\text{st\_soda}, \text{disp}, \text{coins}) \end{aligned}$$

Transitions

Dispense_soda
Dispense_coffee
Take_drink

$$\begin{aligned}
\textit{Dispense_soda} \stackrel{\text{def}}{=}\ & \text{customer} = \textit{student} \wedge \text{st_soda} \wedge \\
& \text{disp} = \textit{none} \wedge (\text{coins} = 2 \vee \text{coins} = 3) \wedge \\
& (\text{coins} = 2 \rightarrow \text{coins}' = 0) \wedge \\
& (\text{coins} = 3 \rightarrow \text{coins}' = 1) \wedge \\
& \text{disp}' = \textit{soda} \wedge \textit{only}(\text{st_soda}, \text{disp}, \text{coins})
\end{aligned}$$

$$\begin{aligned}
\textit{Dispense_coffee} \stackrel{\text{def}}{=}\ & \text{customer} = \textit{prof} \wedge \text{st_coffee} \wedge \\
& \text{disp} = \textit{none} \wedge \text{coins} \neq 0 \wedge \\
& (\text{coins} = 1 \rightarrow \text{coins}' = 0) \wedge \\
& (\text{coins} = 2 \rightarrow \text{coins}' = 1) \wedge \\
& (\text{coins} = 3 \rightarrow \text{coins}' = 2) \wedge \\
& \text{disp}' = \textit{coffee} \wedge \textit{only}(\text{st_coffee}, \text{disp}, \text{coins})
\end{aligned}$$

Transitions

Dispense_soda
Dispense_coffee
Take_drink

$$\begin{aligned}
\textit{Dispense_soda} \stackrel{\text{def}}{=}\ & \text{customer} = \textit{student} \land \text{st_soda} \land \\
& \text{disp} = \textit{none} \land (\text{coins} = \textit{2} \lor \text{coins} = \textit{3}) \land \\
& (\text{coins} = \textit{2} \rightarrow \text{coins}' = \textit{0}) \land \\
& (\text{coins} = \textit{3} \rightarrow \text{coins}' = \textit{1}) \land \\
& \text{disp}' = \textit{soda} \land \textit{only}(\text{st_soda}, \text{disp}, \text{coins})
\end{aligned}$$

$$\begin{aligned}
\textit{Dispense_coffee} \stackrel{\text{def}}{=}\ & \text{customer} = \textit{prof} \land \text{st_coffee} \land \\
& \text{disp} = \textit{none} \land \text{coins} \neq \textit{0} \land \\
& (\text{coins} = \textit{1} \rightarrow \text{coins}' = \textit{0}) \land \\
& (\text{coins} = \textit{2} \rightarrow \text{coins}' = \textit{1}) \land \\
& (\text{coins} = \textit{3} \rightarrow \text{coins}' = \textit{2}) \land \\
& \text{disp}' = \textit{coffee} \land \textit{only}(\text{st_coffee}, \text{disp}, \text{coins})
\end{aligned}$$

Transitions

Dispense_soda
Dispense_coffee
Take_drink

$$\begin{aligned}
\mathit{Dispense_soda} \;\stackrel{\text{def}}{=}\;\; & \text{customer} = \mathit{student} \wedge \text{st_soda} \wedge \\
& \text{disp} = \mathit{none} \wedge (\text{coins} = \mathit{2} \vee \text{coins} = \mathit{3}) \wedge \\
& (\text{coins} = \mathit{2} \rightarrow \text{coins}' = \mathit{0}) \wedge \\
& (\text{coins} = \mathit{3} \rightarrow \text{coins}' = \mathit{1}) \wedge \\
& \text{disp}' = \mathit{soda} \wedge \mathit{only}(\text{st_soda}, \text{disp}, \text{coins})
\end{aligned}$$

$$\begin{aligned}
\mathit{Dispense_coffee} \;\stackrel{\text{def}}{=}\;\; & \text{customer} = \mathit{prof} \wedge \text{st_coffee} \wedge \\
& \text{disp} = \mathit{none} \wedge \text{coins} \neq \mathit{0} \wedge \\
& (\text{coins} = \mathit{1} \rightarrow \text{coins}' = \mathit{0}) \wedge \\
& (\text{coins} = \mathit{2} \rightarrow \text{coins}' = \mathit{1}) \wedge \\
& (\text{coins} = \mathit{3} \rightarrow \text{coins}' = \mathit{2}) \wedge \\
& \text{disp}' = \mathit{coffee} \wedge \mathit{only}(\text{st_coffee}, \text{disp}, \text{coins})
\end{aligned}$$

$$\begin{aligned}
\mathit{Take_drink} \;\stackrel{\text{def}}{=}\;\; & \text{customer} \neq \mathit{none} \wedge \text{disp} \neq \mathit{none} \wedge \\
& \text{disp}' = \mathit{none} \wedge \mathit{only}(\text{disp})
\end{aligned}$$

Temporal properties of transition systems

1. There is no state in which professors and students are both customers.
2. Students always get soda.
3. The machine cannot dispense drinks forever without recharging.
4. Eventually, the machine runs out of soda.
5. If coffee has just been dispensed, the machine must have had coins right before.
6. If the machine is never restocked it will never dispense drinks.
7. The machine never dispenses drinks at a discount or for free.
8. …

Temporal properties of transition systems

1. There is no state in which professors and students are both customers.
2. Students always get soda.
3. The machine cannot dispense drinks forever without recharging.
4. Eventually, the machine runs out of soda.
5. If coffee has just been dispensed, the machine must have had coins right before.
6. If the machine is never restocked it will never dispense drinks.
7. The machine never dispenses drinks at a discount or for free.
8. …

Temporal properties of transition systems

1. There is no state in which professors and students are both customers.
2. Students always get soda.
3. The machine cannot dispense drinks forever without recharging.
4. Eventually, the machine runs out of soda.
5. If coffee has just been dispensed, the machine must have had coins right before.
6. If the machine is never restocked it will never dispense drinks.
7. The machine never dispenses drinks at a discount or for free.
8. …

Temporal properties of transition systems

1. There is no state in which professors and students are both customers.
2. Students always get soda.
3. The machine cannot dispense drinks forever without recharging.
4. Eventually, the machine runs out of soda.
5. If coffee has just been dispensed, the machine must have had coins right before.
6. If the machine is never restocked it will never dispense drinks.
7. The machine never dispenses drinks at a discount or for free.
8. …

Temporal properties of transition systems

1. There is no state in which professors and students are both customers.
2. Students always get soda.
3. The machine cannot dispense drinks forever without recharging.
4. Eventually, the machine runs out of soda.
5. If coffee has just been dispensed, the machine must have had coins right before.
6. If the machine is never restocked it will never dispense drinks.
7. The machine never dispenses drinks at a discount or for free.
8. …

Temporal properties of transition systems

1. There is no state in which professors and students are both customers.
2. Students always get soda.
3. The machine cannot dispense drinks forever without recharging.
4. Eventually, the machine runs out of soda.
5. If coffee has just been dispensed, the machine must have had coins right before.
6. If the machine is never restocked it will never dispense drinks.
7. The machine never dispenses drinks at a discount or for free.
8. …

Temporal properties of transition systems

1. There is no state in which professors and students are both customers.
2. Students always get soda.
3. The machine cannot dispense drinks forever without recharging.
4. Eventually, the machine runs out of soda.
5. If coffee has just been dispensed, the machine must have had coins right before.
6. If the machine is never restocked it will never dispense drinks.
7. The machine never dispenses drinks at a discount or for free.
8. …

Temporal properties of transition systems

1. There is no state in which professors and students are both customers.
2. Students always get soda.
3. The machine cannot dispense drinks forever without recharging.
4. Eventually, the machine runs out of soda.
5. If coffee has just been dispensed, the machine must have had coins right before.
6. If the machine is never restocked it will never dispense drinks.
7. The machine never dispenses drinks at a discount or for free.
8. …

Temporal properties of transition systems

1. There is no state in which professors and students are both customers.
2. Students always get soda.
3. The machine cannot dispense drinks forever without recharging.
4. Eventually, the machine runs out of soda.
5. If coffee has just been dispensed, the machine must have had coins right before.
6. If the machine is never restocked it will never dispense drinks.
7. The machine never dispenses drinks at a discount or for free.
8. …

These properties, which talk about the systems behavior over time, cannot be expressed in PLFD. We need a more expressive logic!