

State-changing systems

Informally	Formally
At each time moment, the system is in a particular state .	This state can be characterized by values of some variables, called the state variables .
The system state is changing in time. There are actions (controlled or not) that change the state.	Actions change values of some state variables.

Examples

1. **Reactive systems**. These are the systems that interact with their environment.
2. **Concurrent systems**. These are the systems which consist of a set of components functioning together. Usually, functioning of these components can be described independently, but they communicate through **shared variables** or some kind of **communication channels**, for example queues.

Reasoning about state-changing systems

1. Build a **formal model** of this state-changing system which describes, in particular, functioning of the system, or some abstraction thereof.
2. Using a **logic to specify and verify properties** of the system.

Vending machine example

Consider an example state-changing system: a **vending machine** which dispenses drinks in a university department. The machine has several components, including at least the following: a **storage space** for storing and preparing drinks, a **box** for dispensing drinks and a **coin slot** for putting coins. When the machine is operating, it comes through several states depending on the behavior of the current **customer**.

Vending machine example

Each action undertaken by the customer or by the machine itself may **change the state** of the machine. For example, when the customer inserts a coin in the coin slot, the amount of money collected in the slot changes.

Actions which may change the state of the system are called **transitions**.

Modeling state-changing systems

To build a **formal model** of a particular state-changing system, we should define

1. What are the **state variables**.
2. What are the possible **values** of the state variables.
3. What are the **transitions** and how they change the values of the state variables.

A **state** can be identified with the set of pairs *(variable, value)*, or with a **function from variables to values**.

Transition systems

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

1. \mathcal{X} is a finite set of **state variables**.
2. D is a non-empty set, called the **domain**. Elements of D are called **values**.
3. dom is a mapping from \mathcal{X} to the set of non-empty subsets of D . For each state variable $v \in \mathcal{X}$, the set $dom(v)$ is called the **domain for v** .
4. I and T will be explained later.

State and Transition

A **state** of a transition system \mathbb{S} is a function $s : \mathcal{X} \rightarrow D$ such that for every $x \in \mathcal{X}$ we have $s(x) \in \text{dom}(x)$.

A **transition** is a set of pairs of states.

Transition systems

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

1. \mathcal{X} is a finite set of **state variables**.
2. D is a non-empty set, called the **domain**. Elements of D are called **values**.
3. dom is a mapping from \mathcal{X} to the set of non-empty subsets of D . For each state variable $v \in \mathcal{X}$, the set $dom(v)$ is called the **domain for v** .
4. I is a set of states, called **initial states**.
5. T is a set of transitions.

Transitions

- ▶ A transition t is **applicable** to a state s if there exists a state s' such that $(s, s') \in t$.
- ▶ A transition t is **deterministic** if for every state s there exists at most one state s' such that $(s, s') \in t$.
- ▶ The **transition relation of S** , denoted by Tr_S , is the set of pairs of states $\bigcup_{t \in T} t$, i.e., it is the union of all transitions in the system.
- ▶ A transition system S is said to be **finite-state** if its domain D is finite, and infinite-state otherwise.

Vending Machine

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

Vending machine

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

Formalization: Variables and Domains

variable	domain	explanation
st_coffee	{0, 1}	drink storage contains coffee
st_beer	{0, 1}	drink storage contains beer
disp	{ <i>none, beer, coffee</i> }	content of drink dispenser
coins	{0, 1, 2, 3}	number of coins in the slot
customer	{ <i>none, student, prof</i> }	customer

Transitions

1. *Recharge* which results in the drink storage having both beer and coffee.
2. *Customer_arrives*, after which a customer appears at the machine.
3. *Customer_leaves*, after which the customer leaves.
4. *Coin_insert*, when the customer inserts a coin in the machine.
5. *Dispense_beer*, when the customer presses the button to get a can of beer.
6. *Dispense_coffee*, when the customer presses the button to get a cup of coffee.
7. *Take_drink*, when the customer removes a drink from the dispenser.

Symbolic Representation of Sets of States

Let $\mathbb{S} = (\mathcal{X}, D, dom, I, T)$ be a finite-state transition system.

- ▶ Then dom defines an instance of PLFD. Denote it by $\mathcal{L}(\mathbb{S})$.
- ▶ Every state of \mathbb{S} is an interpretation for $\mathcal{L}(\mathbb{S})$ and vice versa.
- ▶ Therefore, every formula F defines a set states:

$$\{s \mid s \models F\}.$$

Example

Let us represent the set of states in which the machine is ready to dispense a drink. In every such state, **a drink should be available**, **the drink dispenser empty**, and **the coin slot contain enough coins**. This can be expressed by:

$$\begin{aligned} & (\text{st_coffee} \vee \text{st_beer}) \wedge \text{disp} = \text{none} \wedge \\ & ((\text{coins} = 1 \wedge \text{st_coffee}) \vee \text{coins} = 2 \vee \text{coins} = 3). \end{aligned}$$

Symbolic Representation of Transitions

- ▷ In addition to the set of propositional variables

$\mathcal{X} = \{x_1, \dots, x_n\}$, introduce a set of **next state variables**

$\mathcal{X}' = \{x'_1, \dots, x'_n\}$.

- ▷ **Pairs of states as interpretations.** For every variable $x \in \mathcal{X}$ define

$$\begin{aligned}(s, s')(x) &\stackrel{\text{def}}{=} s(x); \\(s, s')(x') &\stackrel{\text{def}}{=} s'(x).\end{aligned}$$

- ▷ **Symbolic representation.** Formula F of variables $\mathcal{X} \cup \mathcal{X}'$ represents a transition t if $t = \{(s, s') \mid (s, s') \models F\}$.

Example

The transition *Recharge*:

$$\text{customer} = \text{none} \wedge \text{st_coffee}' \wedge \text{st_beer}'.$$

Example

The transition *Recharge*:

$$\text{customer} = \text{none} \wedge \text{st_coffee}' \wedge \text{st_beer}'.$$

But this formula includes describes a **very strange transition** after which, for example

- ▶ coins may appear in and disappear from the slot;
- ▶ dfrinks may appear in and disappear from the dispenser.
- ▶ ...

Frame problem

One has to express explicitly, maybe for a large number of state variables, that the values of these variables do not change after a transition. For 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 **frame problem** arises in artificial intelligence, knowledge representation, and reasoning about actions.

Notation for the frame formula

Abbreviations (we assume $dom(x) = dom(y)$):

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

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

Let \mathbb{S} be a transition system and $\{x_1, \dots, x_n\} \subseteq \mathcal{X}$ be a set of state variables of $\mathcal{L}(\mathbb{S})$. Define

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

This formula expresses that x_1, \dots, x_n are **the only** variables whose values can be changed by the transition.

Preconditions and postconditions

When we represent a transition symbolically using a formula F of variables $\mathcal{X} \cup \mathcal{X}'$, the formula F is usually represented as the conjunction $F_1 \wedge F_2$ of two formulas:

1. F_1 expresses some conditions on the variables \mathcal{X} which are necessary to execute the transition (**precondition**);
2. F_2 expresses some conditions relating variables in \mathcal{X} to those in \mathcal{X}' , i.e., conditions which show how the values of the variables after the transition relate to their values before the transition (**postcondition**).

Transitions

Recharge $\stackrel{\text{def}}{=} \text{customer} = \text{none} \wedge \text{st_coffee}' \wedge \text{st_beer}' \wedge \text{only}(\text{st_coffee}, \text{st_beer}).$

Customer_arrives $\stackrel{\text{def}}{=} \text{customer} = \text{none} \wedge \text{customer}' \neq \text{none} \wedge \text{only}(\text{customer})$

Customer_leaves $\stackrel{\text{def}}{=} \text{customer} \neq \text{none} \wedge \text{customer}' = \text{none} \wedge \text{only}(\text{customer}).$

Coin_insert $\stackrel{\text{def}}{=} \text{customer} \neq \text{none} \wedge \text{coins} \neq 3 \wedge (\text{coins} = 0 \rightarrow \text{coins}' = 1) \wedge (\text{coins} = 1 \rightarrow \text{coins}' = 2) \wedge (\text{coins} = 2 \rightarrow \text{coins}' = 3) \wedge \text{only}(\text{coins}).$

Transitions

Dispense_beer $\stackrel{\text{def}}{=}$ $\text{customer} = \text{student} \wedge \text{st_beer} \wedge$
 $\text{disp} = \text{none} \wedge (\text{coins} = 2 \vee \text{coins} = 3) \wedge$
 $\text{disp}' = \text{beer} \wedge$
 $(\text{coins} = 2 \rightarrow \text{coins}' = 0) \wedge (\text{coins} = 3 \rightarrow \text{coins}' = 1) \wedge$
 $\text{only}(\text{st_beer}, \text{disp}, \text{coins}).$

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

Take_drink $\stackrel{\text{def}}{=}$ $\text{customer} \neq \text{none} \wedge \text{disp} \neq \text{none} \wedge$
 $\text{disp}' = \text{none} \wedge$
 $\text{only}(\text{disp}).$

Transitions

Model checkers often use a convention that the variables that can change are those variables x such that x' occurs in the problem.

Under this convention we can remove *only(...)* from all transitions and change *Dispense_beer* and *Dispense_coffee* as follows:

Dispense_beer $\stackrel{\text{def}}{=}$ $\text{customer} = \text{student} \wedge \text{st_beer} \wedge$
 $\text{disp} = \text{none} \wedge (\text{coins} = 2 \vee \text{coins} = 3) \wedge$
 $\text{disp}' = \text{beer} \wedge$
 $(\text{coins} = 2 \rightarrow \text{coins}' = 0) \wedge (\text{coins} = 3 \rightarrow \text{coins}' = 1) \wedge$
 $\text{st_beer}' = \text{st_beer}'.$

Dispense_coffee $\stackrel{\text{def}}{=}$ $\text{customer} = \text{prof} \wedge \text{st_coffee} \wedge$
 $\text{disp} = \text{none} \wedge \text{coins} \neq 0 \wedge$
 $\text{disp}' = \text{coffee} \wedge$
 $(\text{coins} = 1 \rightarrow \text{coins}' = 0) \wedge (\text{coins} = 2 \rightarrow \text{coins}' = 1) \wedge$
 $(\text{coins} = 3 \rightarrow \text{coins}' = 2) \wedge$
 $\text{st_coffee}' = \text{st_coffee}'.$

Temporal properties of transition systems

1. There is **no state** in which professor and student are both customers.
2. Students **never** drink coffee.
3. The machine cannot dispense drinks **forever** without recharging.