

State-changing systems

Our main interest from now on is modelling **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.

State-changing systems

Our main interest from now on is modelling **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.

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 in.
- ▶ When the machine is operating, it goes through several states depending on the behavior of the current **customer**.
- ▶ 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 stored in the slot changes.
- ▶ Actions which may change the state of the system are called **transitions**.

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 in.
- ▶ When the machine is operating, it goes through several states depending on the behavior of the current **customer**.
- ▶ 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 stored in the slot changes.
- ▶ Actions which may change the state of the system are called **transitions**.

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 in.
- ▶ When the machine is operating, it goes through several states depending on the behavior of the current **customer**.
- ▶ 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 stored 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.

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

1. S is a finite non-empty set, called the set of **states** of \mathbb{S} .
2. $In \subseteq S$ is a non-empty set of states, called the set of **initial states** of \mathbb{S} .
3. $T \subseteq S \times S$ is a set of pairs of states, called the **transition relation** of \mathbb{S} .
4. \mathcal{X} is a finite set, its members are called **state variables**.
5. dom is a mapping from \mathcal{X} such that for every state variable $v \in \mathcal{X}$, $dom(v)$ is a non-empty set, called the **domain for v** .
6. L is a function mapping states in S into interpretations, called the **labeling function** of \mathbb{S} . It will be explained later.

The transition system is said to be **finite-state** if for every state variable v , the domain $dom(v)$ for this variable is finite.

We will only study finite-state transition systems.

Transition systems

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

1. \mathcal{S} is a finite non-empty set, called the set of **states** of \mathbb{S} .
2. $In \subseteq \mathcal{S}$ is a non-empty set of states, called the set of **initial states** of \mathbb{S} .
3. $T \subseteq \mathcal{S} \times \mathcal{S}$ is a set of pairs of states, called the **transition relation** of \mathbb{S} .
4. \mathcal{X} is a finite set, its members are called **state variables**.
5. dom is a mapping from \mathcal{X} such that for every state variable $v \in \mathcal{X}$, $dom(v)$ is a non-empty set, called the **domain for v** .
6. L is a function mapping states in \mathcal{S} into interpretations, called the **labeling function** of \mathbb{S} . It will be explained later.

The transition system is said to be **finite-state** if for every state variable v , the domain $dom(v)$ for this variable is finite.

We will only study finite-state transition systems.

Transition systems

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

1. \mathcal{S} is a finite non-empty set, called the set of **states** of \mathbb{S} .
2. $In \subseteq \mathcal{S}$ is a non-empty set of states, called the set of **initial states** of \mathbb{S} .
3. $T \subseteq \mathcal{S} \times \mathcal{S}$ is a set of pairs of states, called the **transition relation** of \mathbb{S} .
4. \mathcal{X} is a finite set, its members are called **state variables**.
5. dom is a mapping from \mathcal{X} such that for every state variable $v \in \mathcal{X}$, $dom(v)$ is a non-empty set, called the **domain for v** .
6. L is a function mapping states in \mathcal{S} into interpretations, called the **labeling function** of \mathbb{S} . It will be explained later.

The transition system is said to be **finite-state** if for every state variable v , the domain $dom(v)$ for this variable is finite.

We will only study finite-state transition systems.

Transition systems

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

1. \mathcal{S} is a finite non-empty set, called the set of **states** of \mathbb{S} .
2. $In \subseteq \mathcal{S}$ is a non-empty set of states, called the set of **initial states** of \mathbb{S} .
3. $T \subseteq \mathcal{S} \times \mathcal{S}$ is a set of pairs of states, called the **transition relation** of \mathbb{S} .
4. \mathcal{X} is a finite set, its members are called **state variables**.
5. dom is a mapping from \mathcal{X} such that for every state variable $v \in \mathcal{X}$, $dom(v)$ is a non-empty set, called the **domain for v** .
6. L is a function mapping states in \mathcal{S} into interpretations, called the **labeling function** of \mathbb{S} . It will be explained later.

The transition system is said to be **finite-state** if for every state variable v , the domain $dom(v)$ for this variable is finite.

We will only study finite-state transition systems.

Transition systems

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

1. \mathcal{S} is a finite non-empty set, called the set of **states** of \mathbb{S} .
2. $In \subseteq \mathcal{S}$ is a non-empty set of states, called the set of **initial states** of \mathbb{S} .
3. $T \subseteq \mathcal{S} \times \mathcal{S}$ is a set of pairs of states, called the **transition relation** of \mathbb{S} .
4. \mathcal{X} is a finite set, its members are called **state variables**.
5. dom is a mapping from \mathcal{X} such that for every state variable $v \in \mathcal{X}$, $dom(v)$ is a non-empty set, called the **domain for v** .
6. L is a function mapping states in \mathcal{S} into interpretations, called the **labeling function** of \mathbb{S} . It will be explained later.

The transition system is said to be **finite-state** if for every state variable v , the domain $dom(v)$ for this variable is finite.

We will only study finite-state transition systems.

Transition systems

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

1. \mathcal{S} is a finite non-empty set, called the set of **states** of \mathbb{S} .
2. $In \subseteq \mathcal{S}$ is a non-empty set of states, called the set of **initial states** of \mathbb{S} .
3. $T \subseteq \mathcal{S} \times \mathcal{S}$ is a set of pairs of states, called the **transition relation** of \mathbb{S} .
4. \mathcal{X} is a finite set, its members are called **state variables**.
5. dom is a mapping from \mathcal{X} such that for every state variable $v \in \mathcal{X}$, $dom(v)$ is a non-empty set, called the **domain for v** .
6. L is a function mapping states in \mathcal{S} into interpretations, called the **labeling function** of \mathbb{S} . It will be explained later.

The transition system is said to be **finite-state** if for every state variable v , the domain $dom(v)$ for this variable is finite.

We will only study finite-state transition systems.

Labeling function

Note this part of the definition:

4. \mathcal{X} is a finite set, its members are called **state variables**.
5. dom is a mapping from \mathcal{X} such that for every state variable $v \in \mathcal{X}$, $dom(v)$ is a non-empty set, called the **domain for v** .

That is, for a transition system $S = (S, In, T, \mathcal{X}, dom, L)$, the set of variables \mathcal{X} and the mapping dom defines an instance of propositional logic of finite domains.

Denote the set of all interpretations for this instance of PLFD by \mathbb{I} . Then the labelling function L is a mapping $L : S \rightarrow \mathbb{I}$, that is, it maps every state to an interpretation.

This means that

1. for every variable $v \in \mathcal{X}$ and every state $s \in S$, we have $L(s)(v) \in dom(v)$;
2. for every formula A of this instance of PLFD and every state $s \in S$, either $L(s) \models A$ or $L(s) \not\models A$.

Labeling function

Note this part of the definition:

4. \mathcal{X} is a finite set, its members are called **state variables**.
5. dom is a mapping from \mathcal{X} such that for every state variable $v \in \mathcal{X}$, $dom(v)$ is a non-empty set, called the **domain for v** .

That is, for a transition system $\mathcal{S} = (\mathcal{S}, In, T, \mathcal{X}, dom, L)$, the set of variables \mathcal{X} and the mapping dom defines an **instance of propositional logic of finite domains**.

Denote the set of all interpretations for this instance of PLFD by \mathbb{I} . Then the labelling function L is a mapping $L : \mathcal{S} \rightarrow \mathbb{I}$, that is, it maps every state to an interpretation.

This means that

1. for every variable $v \in \mathcal{X}$ and every state $s \in \mathcal{S}$, we have $L(s)(v) \in dom(v)$;
2. for every formula A of this instance of PLFD and every state $s \in \mathcal{S}$, either $L(s) \models A$ or $L(s) \not\models A$.

Labeling function

Note this part of the definition:

4. \mathcal{X} is a finite set, its members are called **state variables**.
5. dom is a mapping from \mathcal{X} such that for every state variable $v \in \mathcal{X}$, $dom(v)$ is a non-empty set, called the **domain for v** .

That is, for a transition system $\mathcal{S} = (\mathcal{S}, In, T, \mathcal{X}, dom, L)$, the set of variables \mathcal{X} and the mapping dom defines an **instance of propositional logic of finite domains**.

Denote the set of all interpretations for this instance of PLFD by \mathbb{I} . Then the labelling function L is a mapping $L : \mathcal{S} \rightarrow \mathbb{I}$, that is, it maps every state to an interpretation.

This means that

1. for every variable $v \in \mathcal{X}$ and every state $s \in \mathcal{S}$, we have $L(s)(v) \in dom(v)$;
2. for every formula A of this instance of PLFD and every state $s \in \mathcal{S}$, either $L(s) \models A$ or $L(s) \not\models A$.

Labeling function

Note this part of the definition:

4. \mathcal{X} is a finite set, its members are called **state variables**.
5. dom is a mapping from \mathcal{X} such that for every state variable $v \in \mathcal{X}$, $dom(v)$ is a non-empty set, called the **domain for v** .

That is, for a transition system $\mathcal{S} = (\mathcal{S}, In, T, \mathcal{X}, dom, L)$, the set of variables \mathcal{X} and the mapping dom defines an **instance of propositional logic of finite domains**.

Denote the set of all interpretations for this instance of PLFD by \mathbb{I} . Then the labelling function L is a mapping $L : \mathcal{S} \rightarrow \mathbb{I}$, that is, it maps every state to an interpretation.

This means that

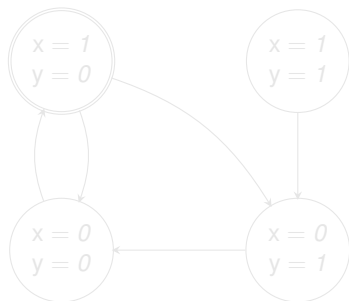
1. for every variable $v \in \mathcal{X}$ and every state $s \in \mathcal{S}$, we have $L(s)(v) \in dom(v)$;
2. for every formula A of this instance of PLFD and every state $s \in \mathcal{S}$, either $L(s) \models A$ or $L(s) \not\models A$.

State Transition Graph

State Transition Graph of a transition system \mathcal{S} :

- ▶ The **nodes** are the states of \mathcal{S} .
- ▶ The **arcs** are elements of the transition relation: there is an arc from a state s to a state s' if and only if $(s, s') \in T$.

Assume two boolean-valued variables x, y .



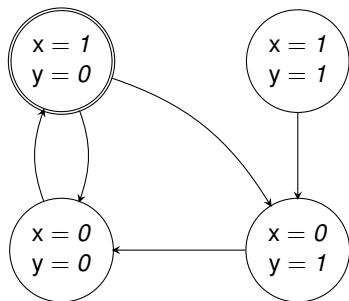
We denote the initial state(s) using double lines.

State Transition Graph

State Transition Graph of a transition system \mathcal{S} :

- ▶ The **nodes** are the states of \mathcal{S} .
- ▶ The **arcs** are elements of the transition relation: there is an arc from a state s to a state s' if and only if $(s, s') \in T$.

Assume two boolean-valued variables x, y .



We denote the initial state(s) using double lines.

States as interpretations

- ▶ If $L(s)(x) = v$ then we say that x has the value v in s and write $s(x) = v$.
- ▶ If $L(s) \models A$ then we say that s satisfies A or A is true in s write $s \models A$.

In both cases, we identify s with $L(s)$.

Transitions

When we model systems, we will usually represent the transition relation as a union of so-called transitions.

- ▶ A **transition** t is any set of pairs of states.
- ▶ 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 $(s, s') \in t$.

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.