Interactive Transition Systems: Definitions and Examples

Venkatesh Choppella

2024-01-08 07:11:18+05:30

IIIT Hyderabad

Contents

The role of interaction

- An action is something that can influence to what a state changes.

- Systems in which actions drive a state are very common, e.g., user interfaces, mobile phones, etc.

- Iterative system: The rule for evolution is fixed:

$$x' = F(x)$$

- Interactive system: Set of actions U. Dynamics is now a relation $\rightarrow$: $X \times U \times X$.

- Our goal is to study systems in which actions play a prominent role.

What can we do with Transition Systems?

Transition systems

- are useful for modelling phenomena and mechanisms.
- provide a uniform notation for thinking and expressing change that is local. (From a state to the next state.)
- are to computing what differential equations are to science and engineering: a way to express incremental change.

Contents

Transition System: formal definition

Definition 1 (Transition System)
A transition System is a tuple $(X, X^0, U, \rightarrow, Y, h)$ where

1. X is a **state space**, a set of **states**.
2. X^0 is a subset of X and is the set of **initial states**.
3. U is an **action space**, a set of **actions**.
4. Y is an **observation space**, a set of **observations**.
5. $h : X \rightarrow Y$, called the **display** maps states to observations.
6. $\rightarrow \subseteq X \times U \times X$ is called the **transition relation** or **dynamics**. The transition (x, u, x') is written $x \xrightarrow{u} x'$.

Other names of Transition Systems

- (Labelled Transition) Systems
- State Machines
- Processes
- Automata

Transition system as a machine

1. X is the set of all possible **configurations** of the machine's '**moving parts**'.

2. X^0 is the set of all **initial configurations**.

3. $\to$ determines **how the parts move**.

4. The moving parts are 'under the hood'. The machine comes with a dashboard.

5. Y is the set of all possible **observed values** on the dashboard.

6. h determines how the configuration of the moving parts is **displayed** on the dashboard.

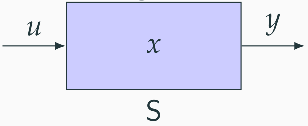
Transition system as a program

1. X is the set of all possible values of the **program variables**.
2. X^0 is the set of all possible **initial values** of the program variables.
3. F specifies how the program variables are **updated**.
4. Y is the set of all **possible values displayed** by the program.
5. h determines **how** the variable values are **converted** to displayed values.

States and Actions

1. S: System
2. x: State
3. u: Action
4. y: Observation

Block Diagram

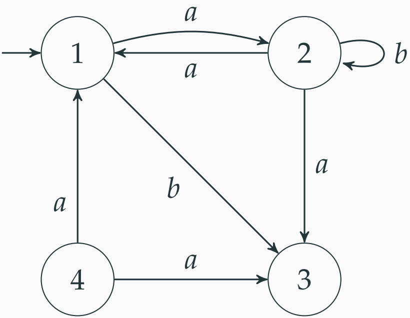


Example (NFA) and System Transition Graph

States and Actions

1. State space $X = \{1, 2, 3, 4\}$
2. Initial states: $X^0 = \{1\}$
3. Action space: $U = \{a, b\}$
4. Observation space: $Y = X$
5. Display: Identity function $h(x) = x$

Transition relation of the NFA



Successors of a state

$Post(x, u)$ is the set of u-successors of x.

$$Post(x, u) = \{x' \mid x \xrightarrow{u} x'\}$$

$Post(x)$ is the set of **successors** of x:

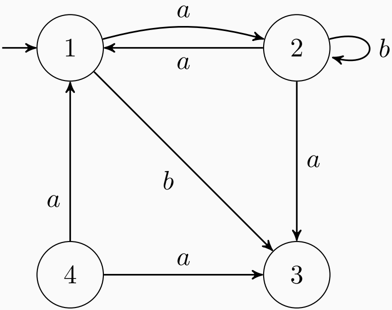
$$Post(x) = \bigcup_{u \in U} Post(x, u)$$

Actions enabled at a state

An action u is **enabled at** state x if $Post(x, u) \neq \emptyset$.

$En(x)$ is the set of all actions enabled at x.

Example NFA: Enabled actions and Successors



State x	$En(x)$	$Post(x, \mathsf{a})$	$Post(x, \mathsf{b})$	$Post(x)$
x_1	$\{a, b\}$	$\{x_2\}$	$\{x_3\}$	$\{x_2, x_3\}$
x_2	$\{a, b\}$	$\{x_1, x_3\}$	$\{x_2\}$	$\{x_1, x_2, x_3\}$
x_3	$\{\}$	$\{\}$	$\{\}$	$\{\}$
x_4	$\{a\}$	$\{x_1, x_3\}$	$\{\}$	$\{x_1, x_3\}$

State Properties

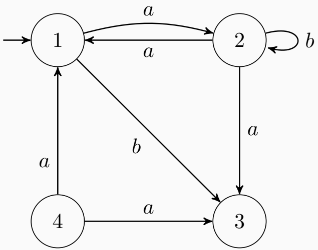
1. x is **terminal** if $Post(x) = \emptyset$.

2. x is **a fixed point for** u if

$$Post(x, u) = \{x\}$$

3. x is **transient for** u if
   3.1 u is enabled at x

   3.2 $x \notin Post(x, u)$

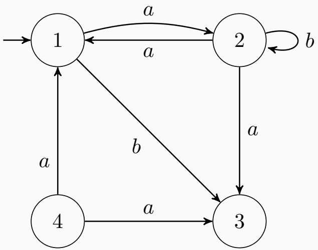


State Properties (Fixed and Transient)

1. x is **a fixed point** if it is fixed for each u enabled at x. Note that every terminal state is a fixed point.

2. x is **transient** if
   2.1 it is not terminal

   2.2 it is transient for each u enabled at x.



Example: Light Bulb

Demo

Light Bulb: states and interface

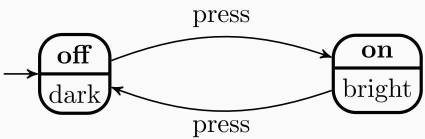
- **States:** $X = \{\mathsf{on}, \mathsf{off}\}$
- **Initial States:** $X^0 = \{\mathsf{off}\}$.
- **Actions:** $U = \{\mathsf{press}\}$
- **Observations:**: $Y = \{\mathsf{bright}, \mathsf{dark}\}$

Light Bulb: Display and Dynamics

- **Display:** $h(x) = \begin{cases} \text{bright} & \text{if } x = \text{on} \\ \text{dark} & \text{if } x = \text{off} \end{cases}$
- **Dynamics:** $F : X, U \to X$

$$F(\text{on}, \text{press}) = \text{off}$$
$$F(\text{off}, \text{press}) = \text{on}$$

Light Bulb: Transition Graph



Types of Dynamics

- $\rightarrow$ is **deterministic** if for each x, u, $x \xrightarrow{u} x_1$ and $x \xrightarrow{u} x_2$ implies $x_1 = x_2$. A state on an action goes to at most one state.
- $\rightarrow$ is **total** if every action is enabled at every state.

Exercise: Can you recall any examples of deterministic or total dynamics?

Contents

Runs

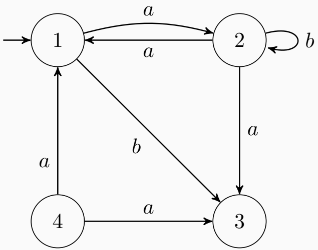
A **run** in a transition system is a labelled path in the transition graph of the system.

- **Finite Run**: $x_0 \xrightarrow{u_0} x_1 \xrightarrow{u_1} x_2 \ldots \xrightarrow{u_{n-1}} x_n$, $n \geq 0$. If $n = 0$, the run is **empty**. x_n is called the **destination** of the run.

- **Infinite Run**: $x_0 \xrightarrow{u_0} x_1 \xrightarrow{u_1} x_2 \ldots$. An infinite run has no destination.

- **Origin**: x_0 is the origin of the run.

We say that a state x_0 **reaches** x_d if there is a run with origin x_0 and destination x_d. We also say that the run *reaches* the destination x_d. Alternatively, x_d is **reachable from** x_0. Clearly, every state reaches itself.



Convergence

A state x is **converges to** x_d if x reaches x_d and x_d is a fixed point.

Sample Runs in the Light Bulb Example

- **Finite Run with origin** on**:** $\text{on} \xrightarrow{\text{press}} \text{off} \xrightarrow{\text{press}} \text{on} \xrightarrow{\text{press}} \text{off}$
- **Finite Run with origin** off**:** $\text{off} \xrightarrow{\text{press}} \text{on} \xrightarrow{\text{press}} \text{off}$
- **Infinite run with origin** on**:** $\text{on} \xrightarrow{\text{press}} \text{off} \xrightarrow{\text{press}} \text{on} \xrightarrow{\text{press}} \ldots.$

Complete runs

A **complete run**, or **completion** is a run that is either

- infinite, or
- finite, and the destination of the run is terminal

In the Light Bulb example, there are two completions:

$$\text{off} \xrightarrow{\text{press}} \text{on} \xrightarrow{\text{press}} \text{off} \xrightarrow{\text{press}} \text{on} \xrightarrow{\text{press}} \dots$$

$$\text{on} \xrightarrow{\text{press}} \text{off} \xrightarrow{\text{press}} \text{on} \xrightarrow{\text{press}} \dots$$

Executions

An **execution** is a completion whose origin is an initial state.

In the Light Bulb example, there is only one execution:

$$\text{off} \xrightarrow{\text{press}} \text{on} \xrightarrow{\text{press}} \text{off} \xrightarrow{\text{press}} \text{on} \xrightarrow{\text{press}} \ldots$$

Trajectory

A **trajectory** is a sequence of states obtained by projecting the states of a run.

A trajectory may be finite or infinite (depending on the run).

Here are some trajectories in the Light Bulb example, trajectories of the system.

- on, off, on, off
- $on, off, on, off, \ldots$
- $off, on, off, \ldots$

Trace

A **trace** is a sequence of observations obtained by mapping the display function on the states of a trajectory.

A trace may be finite or infinite (depending on the trajectory).

In the Light Bulb example, here are some traces of the system.

- $\text{bright}, \text{dark}, \text{bright}, \text{dark}$
- $\text{bright}, \text{dark}, \text{bright}, \text{dark}, \ldots$
- $\text{dark}, \text{bright}, \text{dark}, \ldots$

Example: Run, Trajectory and Trace

Run $\quad \mathsf{off} \xrightarrow{\mathsf{press}} \mathsf{on} \xrightarrow{\mathsf{press}} \mathsf{off} \xrightarrow{\mathsf{press}} \mathsf{on}$

Trajectory $\quad \mathsf{off} \longrightarrow \mathsf{on} \longrightarrow \mathsf{off} \longrightarrow \mathsf{on}$

$$\downarrow h \qquad \downarrow h \qquad \downarrow h \qquad \downarrow h$$

Trace $\quad \mathsf{dark} \qquad \mathsf{light} \qquad \mathsf{dark} \qquad \mathsf{light}$

Execution Traces and Behaviour

- An **execution trace** is a trace obtained from an execution of the system.
- The **Behaviour** of a system is the set of all its execution traces.

In the Light Bulb example, the behaviour is a set consisting of just one execution trace:

$$\text{dark}, \text{bright}, \text{dark}, \text{bright}, \ldots$$

Contents

Transition System Types

- **Finite:** X and U are finite.
- **Autonomous:** U is a singleton.
- **Controlled:** $|U| > 1$.
- **Transparent:** if $Y = X$ and h is the identity map.
- **Agile:** $X^0 = X$.
- **Deterministic:** $\rightarrow$ is deterministic.
- **Total:** $\rightarrow$ is total.

Transition System Types (contd)

- **Iterative:** autonomous, deterministic and total.
- **Convergent:** every execution reaches a fixed point.

Discrete Flows as agile, transparent iterative Systems

A discrete flow is a pair $D = (X, F : X \to X)$.

It may be seen as an iterative system that is

- Agile: $X^0 = X$
- Autonomous: $|U| = 1$
- Deterministic and total: the transition relation is isomorphic to F.
- Transparent: $Y = X$ and $h = Id_X$

Contents

ATM Demo

Demo

Modelling the ATM application: States

1. **Mode**: $m : M = \{main, bal(History), dep, wth\}$
2. **History** : $History = \{main, dep, wth\}$
3. **Balance**: $b : \mathbb{Z}$
4. **State** : $(m, b) : X = M \times \mathbb{Z}$

Modelling the ATM application: Initial States

$X^0 = \{(main, 20000)\}$

Modelling the ATM application: Actions

$$U = \{show, deposit, withdraw, confirm(\mathbb{Z}), main\}$$

Modelling the ATM application: Observables

1. $Button = \{show, deposit, withdraw, confirm, goto - main\}$
2. $Y = (msg : String, buttons : set[Button])$

Modelling the ATM app: Transitions

Action	main b	bal(m) b	dep b	wth b
show	bal(main) b			
deposit	dep b			
withdraw	wth b			
confirm(v)			bal(dep) b+v	bal(wth) b-v
goto-main		main b	main b	main b

Modelling the ATM: Display map

State	msg	buttons
(main,b)	"Welcome"	show, deposit, withdraw
(bal(main), b)	"Your a/c has" + b	goto-main
(bal(dep), b)	"Deposit successful: Your a/c has " + b	goto-main
(bal(wth),b)	"Withdrawal successful: Your a/c has " + b	goto-main
(dep, b)	"Enter the amt to deposit"	confirm, goto-main
(wth, b)	"Enter the amt to withdraw"	confirm, goto-main

Properties of the ATM App

1. Modes, actions are specified using enumerated types.
2. The system has no fixed or terminal states.
3. All states are transient
4. Some modes and actions are parameterised.

Correctness

What is an accurate statement about the correctness of the ATM app?

1.

2.

Contents

Key concepts

- Transition System
- Successors of a state
- Runs, Completions, Executions
- Trajectories, Traces
- Execution Traces and Behaviour

Key ideas

- Transition Systems are mathematical models of machines and physical processes.
- Transition systems consists of states, actions, and observations.
- Display maps states to observations. $h(x) = y$
- Dynamics relates a state and an action to an action. $x \xrightarrow{u} x'$.
- Transition systems have behaviour.

Contents

Tic-tac-toe

- Model the mechanics of the tic-tac-toe game as a system.