Studies in Systems, Decision and Control 89

Calin Belta
Boyan Yordanov
Ebru Aydin Gol

Formal Methods for Discrete-Time Dynamical Systems

Springer

Chapter 1
Transition Systems

In this book, we focus on transition systems as a modeling formalism for a wide range of processes. Although mathematically simple, transition systems are general enough to capture the behavior of systems defined over discrete (finite or infinite) or continuous (infinite) spaces. In subsequent chapters, this richness will allow us to use transition systems as a unifying framework for modeling both (infinite-state) discrete-time systems and their (finite-state) abstractions. In the following chapters we will also describe techniques for the analysis and control of finite and infinite transition systems, which are inspired by automata-theoretic model checking.

In this chapter, we define the syntax and semantics of transition systems, and provide several illustrative examples. In particular, we present different (deterministic, nondeterministic, finite, and infinite) transition system representations for discrete-time dynamical systems. We also introduce simulation and bisimulation relations, which are central for the construction of finite abstractions throughout the book.

1.1 Definitions and Examples

Definition 1.1 (*Transition system*) A transition system is a tuple $T = (X, \Sigma, \delta, O, o)$, where

- X is a (possibly infinite) set of states,
- Σ is a (possibly infinite) set of inputs (controls or actions),
- $\delta : X \times \Sigma \to 2^X$ is a transition function,
- O is a (possibly infinite) set of observations, and
- $o : X \to O$ is an observation map.

C. Belta et al., *Formal Methods for Discrete-Time Dynamical Systems*,
Studies in Systems, Decision and Control 89, DOI 10.1007/978-3-319-50763-7_1

A subset $X_r \subseteq X$ is called a *region* of T. A transition $\delta(x, \sigma) = X_r$ indicates that, while the system is in state x, it can make a transition to any state $x' \in X_r$ in region $X_r \subseteq X$ under input σ. We denote the set of inputs available at state $x \in X$ by

$$\Sigma^x = \{\sigma \in \Sigma \mid \delta(x, \sigma) \neq \emptyset\}. \tag{1.1}$$

A transition $\delta(x, \sigma)$ is *deterministic* if $|\delta(x, \sigma)| = 1$ and the transition system T is deterministic if for all states $x \in X$ and all inputs $\sigma \in \Sigma^x$, $\delta(x, \sigma)$ is deterministic. T is *non-blocking* if, for every state $x \in X$, $\Sigma^x \neq \emptyset$. Throughout this book, only non-blocking transition systems are considered. Transition system T is called *finite* if its sets of states X, inputs Σ, and observations O are all finite.

An *input word* of the system is defined as an infinite sequence $w_\Sigma = w_\Sigma(1)w_\Sigma(2)w_\Sigma(3)\ldots \in \Sigma^\omega$. A *trajectory* or *run* of T produced by input word w_Σ and originating at state $x_1 \in X$ is an infinite sequence $w_X = w_X(1)w_X(2)w_X(3)\ldots$ with the property that $w_X(k) \in X$, $w_X(1) = x_1$, and $w_X(k+1) \in \delta(w_X(k), w_\Sigma(k))$, for all $k \geq 1$. We denote the set of all trajectories of a transition system T originating at x by $T(x)$. We use $T(X_r) = \cup_{x' \in X_r} T(x')$ to denote the set of all trajectories of T originating in region $X_r \subseteq X$. As a consequence, $T(X)$ will denote the set of all trajectories of T.

A run $w_X = w_X(1)w_X(2)w_X(3)\ldots$ defines an *output word* (which we will refer to simply as *word*) $w_O = w_O(1)w_O(2)w_O(3)\ldots \in O^\omega$, where $w_O(k) = o(w_X(k))$ for all $k \geq 1$. The set of all words generated by the set of all trajectories starting at $x \in X$ is called the *language* of T originating at x and is denoted by $\mathscr{L}_T(x)$. The language of T originating at a region $X_r \subseteq X$ is $\mathscr{L}_T(X_r) = \bigcup_{x' \in X_r} \mathscr{L}_T(x')$. The language of T is defined as $\mathscr{L}_T(X)$, which for simplicity is also denoted as $\mathscr{L}_T$. We often represent an infinite word as a finite *prefix* followed by an infinite *suffix* as shown in Example 1.1.

For an arbitrary region $X_r \subseteq X$ and set of inputs $\Sigma' \subseteq \Sigma$, we define the set of states $Post_T(X_r, \Sigma')$ that can be reached from X_r in one step by applying an input in Σ' (called *successors* of X_r under Σ') as

$$Post_T(X_r, \Sigma') = \{x \in X \mid \exists x' \in X_r,\ \exists \sigma \in \Sigma',\ x \in \delta(x', \sigma)\} \tag{1.2}$$

Similarly, the set of states that reach some $X_r \subseteq X$ in one step under the application of some input from $\Sigma' \subseteq \Sigma$ (called *predecessors* of X_r under Σ') can be defined as

$$Pre_T(X_r, \Sigma') = \{x \in X \mid \exists x' \in X_r,\ \exists \sigma \in \Sigma',\ x' \in \delta(x, \sigma)\} \tag{1.3}$$

For a deterministic T, each state x has a single successor under a given input σ, i.e. $Post_T(x, \sigma) = \delta(x, \sigma)$[1] is a singleton, but, in general, can have multiple predecessors, i.e., $Pre_T(x, \sigma)$ is a region of T. However, for a nondeterministic T it is possible that both $Post_T(x, \sigma)$ and $Pre_T(x, \sigma)$ are regions of T.

[1] Since the $Post$ operator was defined for a set of inputs, the correct notation here is $Post_T(x, \{\sigma\})$. For simplicity, and with a slight abuse of notation, we omit the set notation when only a singleton input is considered. The same observation applies to the Pre operator.

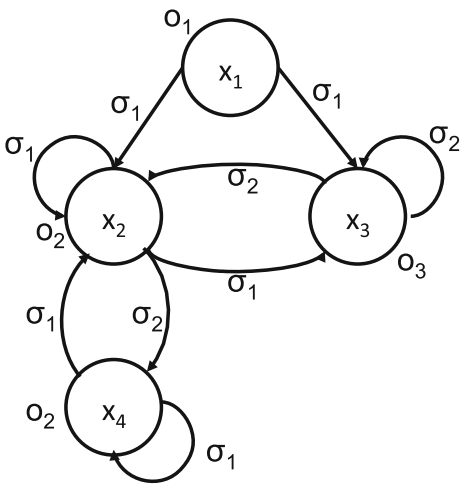


Fig. 1.1 Graphical representation of the transition system defined in Example 1.1. Each state is represented by a *circle* containing the state label. The observation of each state is shown close to its *circle* and transitions are represented by *arrows* between states. The input enabling a transition is shown on *top* of the corresponding *arrow*

Example 1.1 The finite, non-deterministic transition system $T = (X, \Sigma, \delta, O, o)$ shown in Fig. 1.1 is defined formally by

- $X = \{x_1, x_2, x_3, x_4\}$,
- $\Sigma = \{\sigma_1, \sigma_2\}$,
- $\delta(x_1, \sigma_1) = \{x_2, x_3\}$, $\delta(x_2, \sigma_1) = \{x_2, x_3\}$, $\delta(x_2, \sigma_2) = \{x_4\}$, $\delta(x_3, \sigma_2) = \{x_2, x_3\}$, $\delta(x_4, \sigma_1) = \{x_2, x_4\}$,
- $O = \{o_1, o_2, o_3\}$,
- $o(x_1) = o_1$, $o(x_2) = o(x_4) = o_2$, $o(x_3) = o_3$.

The set of inputs available at the states of the system are $\Sigma^{x_1} = \{\sigma_1\}$, $\Sigma^{x_2} = \{\sigma_1, \sigma_2\}$, $\Sigma^{x_3} = \{\sigma_2\}$, and $\Sigma^{x_4} = \{\sigma_1\}$. States x_1 and x_3 form a region $X_r = \{x_1, x_3\}$. Region X_r can be reached from states x_1 and x_2 under σ_1, and, therefore, $Pre_T(X_r, \sigma_1) = \{x_1, x_2\}$. Similarly, states x_2 and x_3 are reachable from region X_r under σ_1, and therefore $Post_T(X_r, \sigma_1) = \{x_2, x_3\}$.

Starting from x_1, under input word $w_\Sigma = \sigma_1\sigma_1\sigma_2(\sigma_1)^\omega$, one possible run of the system is $w_X = x_1x_2x_2(x_4)^\omega$. Sequences $\sigma_1\sigma_1\sigma_2$, $x_1x_2x_2$ are prefixes (repeated once) and σ_1, x_4 are suffixes (repeated infinitely many times) for input word and run, respectively. This run originates in region X_r and defines an infinite (output) word $w_O = o_1(o_2)^\omega \in \mathscr{L}_T(X_r)$. There is an infinite number of infinite words in the language $\mathscr{L}_T(X_r) = \{o_1(o_2)^\omega, o_1o_2(o_3)^\omega, o_1(o_3)^\omega, (o_3)^\omega, o_3o_3(o_2)^\omega, \ldots\}$.

A transition system $T = (X, \Sigma, \delta, O, o)$ is usually referred to as a *control transition system*. It is used as a model in control problems, where the goal is to generate a control strategy from a control specification, possibly given as a temporal logic statement over the observations of the system. This topic will be addressed in detail later in the book. For example, a control strategy enforcing the specification "if o_1 is satisfied initially, then eventually reach and remain in a region where either o_1 or o_2 are satisfied, or o_3 is satisfied" is derived in Chap. 5 for the transition system from Example 1.1. In analysis problems, we are usually interested in checking whether all the trajectories of a system satisfy such a specification, under arbitrary input choices. Since the input is irrelevant, the input set Σ is a singleton, where the only input enables all transitions available at all states. For simplicity of notation, this input set can be omitted completely, leading to a simpler (autonomous, uncontrolled) transition system $T = (X, \delta, O, o)$, where X, O, and o are as given in Definition 1.1 and the transition function assumes the simpler form $\delta : X \to 2^X$. In the uncontrolled case, the definitions of the successor and predecessor sets (Eqs. (1.2) and (1.3)) also assume simpler forms:

$$Post_T(X_r) = \{x' \in X \mid \exists x \in X_r,\ x' \in \delta(x)\} = \bigcup_{x \in X_r} \delta(x), \tag{1.4}$$

$$Pre_T(X_r) = \{x \in X \mid \exists x' \in X_r,\ x' \in \delta(x)\}, \tag{1.5}$$

where $X_r \subseteq X$ is a region. We will use both control transition systems and (autonomous) transition systems throughout the book. We will refer to both simply as *transition systems* and we will use the same symbols to denote their states, transitions, outputs, and output maps. The exact meaning of the transition function and of the $Post$ and Pre operators will be clear from the context. In particular, a transition system with no inputs is deterministic if it has at most one transition from each state.

Example 1.2 An example of a finite, non-deterministic transition system (without inputs) $T = (X, \delta, O, o)$ is given in Fig. 1.2. It is characterized by

- $X = \{x_1, x_2, x_3, x_4\}$,
- $\delta(x_1) = \{x_1, x_2\}$, $\delta(x_2) = \{x_4\}$, $\delta(x_4) = \{x_3\}$, $\delta(x_3) = \{x_3, x_1\}$,
- $O = \{o_1, o_2, o_3\}$,
- $o(x_1) = o(x_2) = o_1$, $o(x_3) = o_3$, $o(x_4) = o_2$.

For region $X_r = \{x_1, x_3\}$, we have $Pre_T(X_r) = \{x_1, x_3, x_4\}$ and $Post_T(X_r) = \{x_1, x_2, x_3\}$.

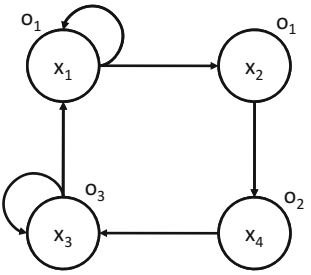


Fig. 1.2 Graphical representation of the transition system from Example 1.2. Unlike the control transition system from Fig. 1.1, there are no inputs labeling the transitions between the states

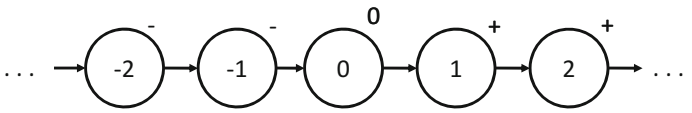


Fig. 1.3 Graphical representation of the transition system defined in Example 1.3

Example 1.3 The infinite, deterministic transition system $T = (X, \delta, O, o)$ shown in Fig. 1.3 is defined formally by

- $X = \mathbb{Z}$,
- $\delta(x) = x + 1, \ \forall x \in \mathbb{Z}$
- $O = \{-, 0, +\}$,
- $o(x) = \begin{cases} + & \text{if } x > 0 \\ - & \text{if } x < 0 \\ 0 & \text{otherwise} \end{cases}$

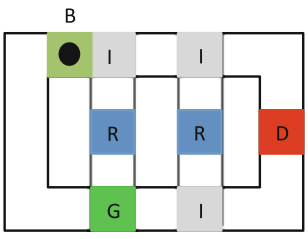
At each state, only the sign of the integer labeling the state is observed.

Example 1.4 Consider a messenger robot moving in the environment depicted in Fig. 1.4a. The robot (black disk) starts at the base B. When it detects a current region of interest (base B, data gather region G, recharge regions R, dangerous region D) or an intersection I, the robot is required to stop and choose the next immediate direction of motion (West, East, South, North). Then the robot turns towards the chosen direction and keeps following the road. We assume that the robot's actuators and sensors are unreliable, and as a result, the transition to a next intersection is not guaranteed. For example, while driving East from

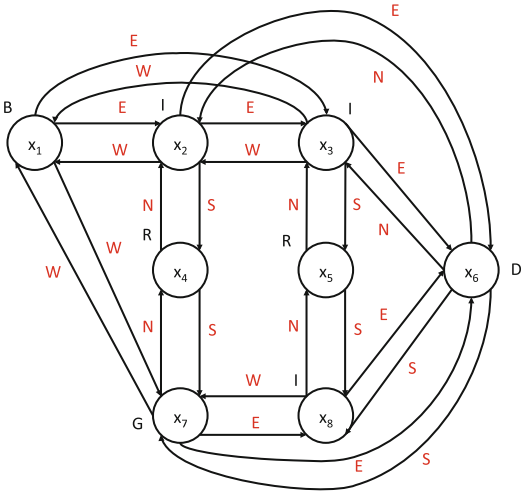
B, the robot can end up at either of the two intersections to the right of B in Fig. 1.4a, i.e., it can fail to stop at the first intersection. The motion of the robot in this environment can be modeled as the nondeterministic transition system T shown in Fig. 1.4b, which is described by

- $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}$;
- $\Sigma = \{W, E, S, N\}$;
- $O = \{B, G, R, D, I\}$; and
- δ and o as shown in Fig. 1.4b.

Example 1.5 A simple gene network is represented schematically in Fig. 1.5a. The network consist of three genes A, B and C expressed from separate promoters. When expressed, each gene produces a different protein (also denoted respectively by A, B and C). Protein C acts as a repressor and binds to the promoters of genes A and B. Therefore, whenever protein C is present, proteins A and B are not produced. Similarly, proteins A and B act as activators for the promoter of gene C, which can be expressed only if either protein A or B is present.



(a) A robot in an indoor environment

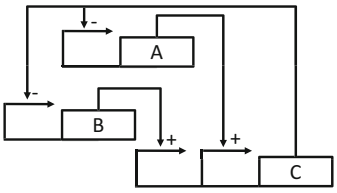


(b) The corresponding transition system

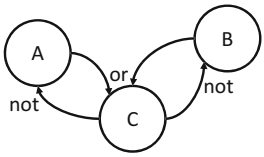
Fig. 1.4 A finite nondeterministic transition system representation for the motion of a robot in an indoor environment (see Example 1.4)

The regulatory interactions of this simple gene network can be captured using the Boolean network model represented schematically in Fig. 1.5b. The model consists of 3 Boolean variables signifying that each gene can be in one of two possible states (i.e., expressed (on) or not expressed (off)). At each step, the model evolves dynamically according to the following rules: the next state of variable A (denoted by A') is given by the Boolean formula $A' = \neg C$ and, similarly, $B' = \neg C$ and $C' = A \vee B$. At each discrete time step, all the variables are updated simultaneously.

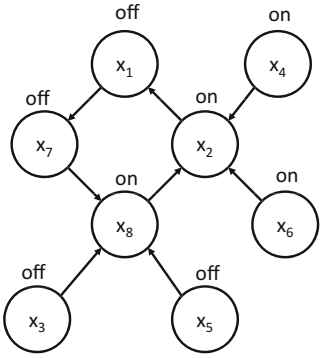
The Boolean network model from Fig. 1.5b can be translated into the finite, deterministic transition system $T = (X, \delta, O, o)$ shown in Fig. 1.5c. The set of states of the system $X = \{x_1, x_2, \ldots, x_8\}$ captures all possible valuation of the Boolean variables from the model (i.e., $x_1 := (A = \bot, B = \bot, C = \bot)$, $x_2 := (A = \bot, B = \bot, C = \top)$, etc., where $\bot$ and $\top$ denotes the Boolean "false" and "true", respectively). Transitions between states of T are assigned using the dynamics of the Boolean network (i.e., $\delta(x_1) = x_7$ since at state x_1



(a) Schematic representation of a simple gene network consisting of genes A,B and C.



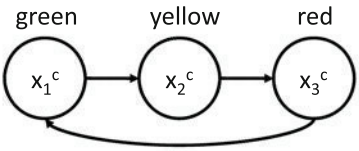
(b) Boolean network model of the gene network from (a).



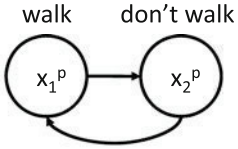
(c) Transition system representation of the Boolean network from (b) where only the state of geneC (on/off) is observed.

Fig. 1.5 Gene networks can be modeled as Boolean networks, which in turn can be described by finite, deterministic transition systems (see Example 1.5 for additional details)

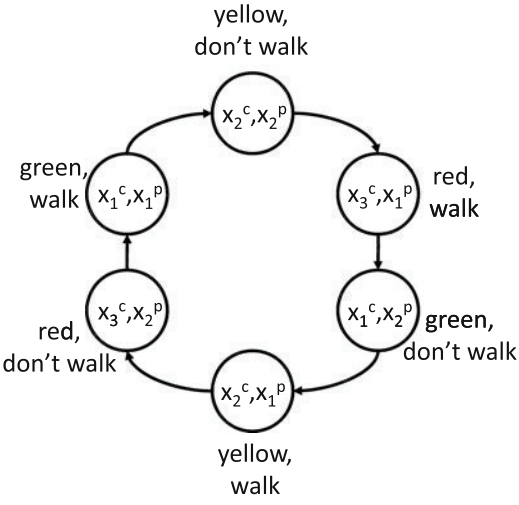
we have ($A = \bot$, $B = \bot$, $C = \bot$) and therefore at the next step ($A' = \top$, $B' = \top$, $C' = \bot$), which defines state x_7). Assuming that we can only observe whether gene C is expressed or not, the set of observations can be given by $O = \{$"on", "off"$\}$ where the observation map of T is defined for all states $x \in X$ by $o(x) =$"on" if and only if $C = \top$ at state x and $o(x) =$"off" otherwise. Throughout this book, we use quotes around observations defined as strings (such as "on","off") but not when the observations are symbols (such as o_1, o_2, o_3 in Example 1.2 or $-, 0, +$ in Example 1.3).



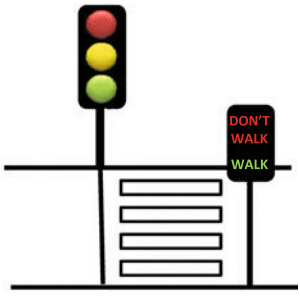
(a) Car traffic light system (T_c)



(b) Pedestrian traffic light system (T_p)



(c) Overall crossing traffic light system ($T = T_c \otimes T_p$)



(d) Graphical representation of the pedestrian crossing

Fig. 1.6 A finite transition system representation of the pedestrian intersection traffic light described in Example 1.6

Example 1.6 Consider the model of a traffic light at a pedestrian crossing (Fig. 1.6) shown graphically in Fig. 1.6d.

The system consists of two components:

(i) a car traffic light with a "red", "yellow" and a "green" signal, and
(ii) a pedestrian light with a "walk" and "don't walk" signal.

Initially, the car and pedestrian traffic light components are modeled as separate transition systems T_c and T_p (shown respectively in Fig. 1.6a, b), where each signal is captured by an observation (i.e., $O_c = \{$"red", "yellow", "green"$\}$ and $O_p = \{$"walk", "don't walk"$\}$). For both transition systems, each state has a unique observation (i.e., in each state, each traffic light is displaying a unique signal). To distinguish between states of T_c and T_p, we denote them by $X_c = \{x_1^c, x_2^c, x_3^c\}$ and $X_p = \{x_1^p, x_2^p\}$, respectively.

By assuming that the transitions of the two systems are perfectly synchronized (i.e., transitions are taken at the same time), the overall model of the crossing is obtained by constructing the product $T = T_c \otimes T_p$ (Fig. 1.6c) of the car and pedestrian traffic light components. Each state of the product T specifies the entire state of the crossing. The observations are obtained by collecting the individual observations and the transitions are obtained by matching the transitions of the individual systems. For example, in state (x_3^c, x_1^p) the car traffic light displays the "red" signal (i.e., transition system T_c is in state x_3^c) and the pedestrian traffic light displays the "walk" signal (i.e., transition system T_p is in state x_1^p).

Note that the pedestrian crossing is modeled as an autonomous, deterministic transition system. In other words, it is assumed that this system is not influenced by the environment or any external inputs (i.e., the system is autonomous) and, instead, it keeps changing the signals according to a predetermined program, always following the same sequence (i.e., the system is deterministic).

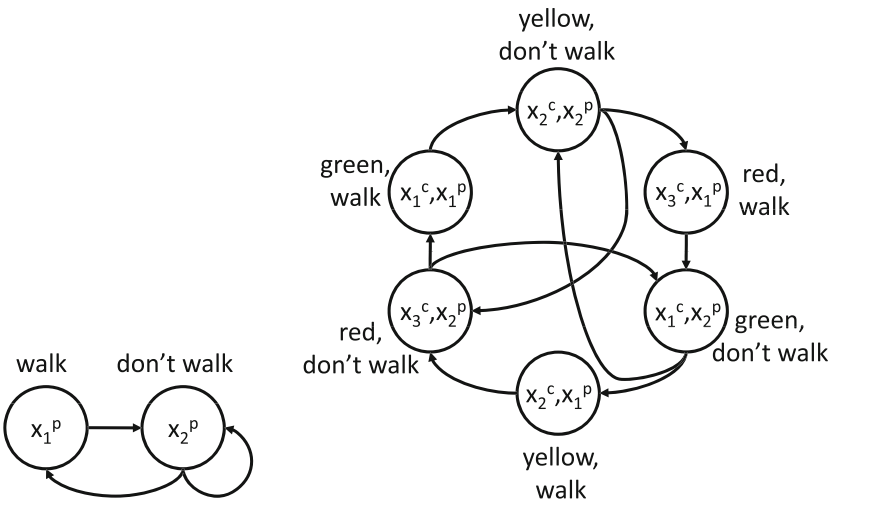
The traffic light model from this example includes a dangerous state (x_1^c, x_1^p) with the observation "green", "walk". Approaches for identifying such unsafe behaviors will be presented in Part II of this book (Examples 3.1 and 4.3).

Example 1.7 We further complicate Example 1.6 by considering a crossing where a pedestrian must push a button to request the "walk" signal. As in Example 1.6, we model the overall crossing as the composition of the car and pedestrian traffic light components, where the car traffic light component T_c remains unchanged as in Fig. 1.6a.

To capture the behavior of the pedestrian traffic light, we use the nondeterministic transition system T_p' shown in Fig. 1.7a. Whenever this system is in state x_2^p (showing the "don't walk" signal), a nondeterministic transition to either states x_1^p ("walk") or x_2^p ("don't walk") can be taken, depending if a pedestrian pushes the button. An explicit input is not considered in this case to represent whether the button is pressed. Instead, this action is considered as a nondeterministic event.

The pedestrian and car traffic light components are synchronized as in Example 1.6 by constructing the product $T' = T_c \otimes T_p'$ shown in Fig. 1.7b, which is nondeterministic since T_p' is nondeterministic. The product T' captures all possible behavior of the traffic light system — at any discrete time step, a "walk" request might or might not occur depending on the behavior of a pedestrian, which results in the nondeterministic transitions in T'.

Note that the dangerous state (x_1^c, x_1^p) with observation "green", "walk" from the initial traffic light model described in Example 1.6 is not eliminated by considering a "walk" request button. In fact, the set of states remains unchanged and only additional (nondeterministic) transitions are introduced.



(a) Pedestrian controlled traffic light system (T_p')

(b) Overall crossing traffic light system ($T' = T_c \otimes T_p'$)

Fig. 1.7 A finite transition system representation of the pedestrian intersection traffic light described in Example 1.7