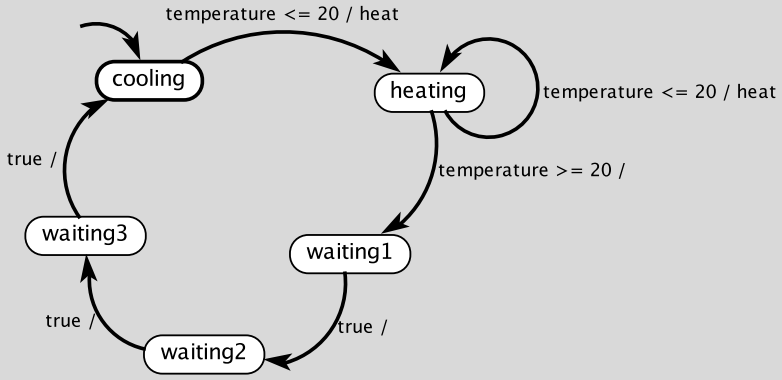
2. Consider a variant of the thermostat of example 3.5. In this variant, there is only one temperature threshold, and to avoid chattering the thermostat simply leaves the heat on or off for at least a fixed amount of time. In the initial state, if the temperature is less than or equal to 20 degrees Celsius, it turns the heater on, and leaves it on for at least 30 seconds. After that, if the temperature is greater than 20 degrees, it turns the heater off and leaves it off for at least 2 minutes. It turns it on again only if the temperature is less than or equal to 20 degrees.

   (a) Design an FSM that behaves as described, assuming it reacts exactly once every 30 seconds.

Solution: A solution is shown below:



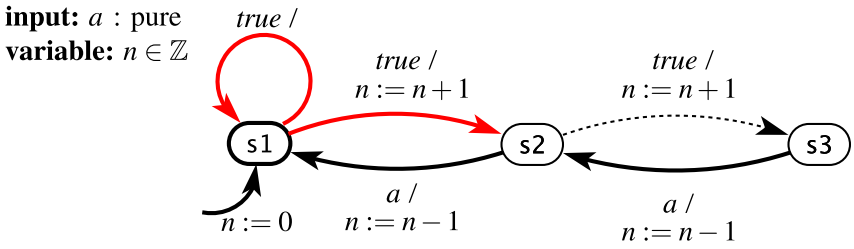
   (b) How many possible states does your thermostat have? Is this the smallest number of states possible?

Solution: The FSM has five states. Many alternative designs with more states exist.

   (c) Does this model thermostat have the time-scale invariance property?

Solution: The model does not have the hysteresis property because the timeout is a fixed amount of time, so varying the time scale of the input will yield distinctly different behavior.

4. How many reachable states does the following state machine have?



Solution: Three. The variable n is redundant, because it always has value 0 in state s1, 1 in s2, and 2 in s3.

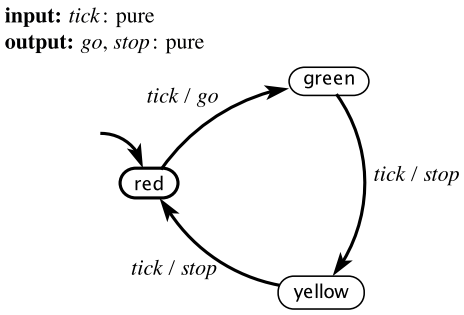


Figure 3.1: Deterministic finite-state machine for Exercise 5

5. Consider the deterministic finite-state machine in Figure 3.1 that models a simple traffic light.

   (a) Formally write down the description of this FSM as a 5-tuple:

$$(States, Inputs, Outputs, update, initialState) \ .$$

Solution: The FSM description is:

$$\begin{aligned} States &= \{\mathsf{red}, \mathsf{yellow}, \mathsf{green}\} \\ Inputs &= (\{tick\} \to \{present, absent\}) \\ Outputs &= (\{go, stop\} \to \{present, absent\}) \\ initialState &= \mathsf{red} \end{aligned}$$

The update function is defined as:

$$update(s,i) = \begin{cases} (\mathsf{green}, go) & \text{if } s = \mathsf{red} \wedge i(tick) = present \\ (\mathsf{yellow}, stop) & \text{if } s = \mathsf{green} \wedge i(tick) = present \\ (\mathsf{red}, stop) & \text{if } s = \mathsf{yellow} \wedge i(tick) = present \\ (s, absent) & \text{otherwise} \end{cases}$$

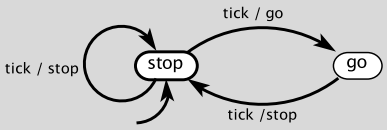
   (b) Give an execution trace of this FSM of length 4 assuming the input *tick* is *present* on each reaction.

Solution:

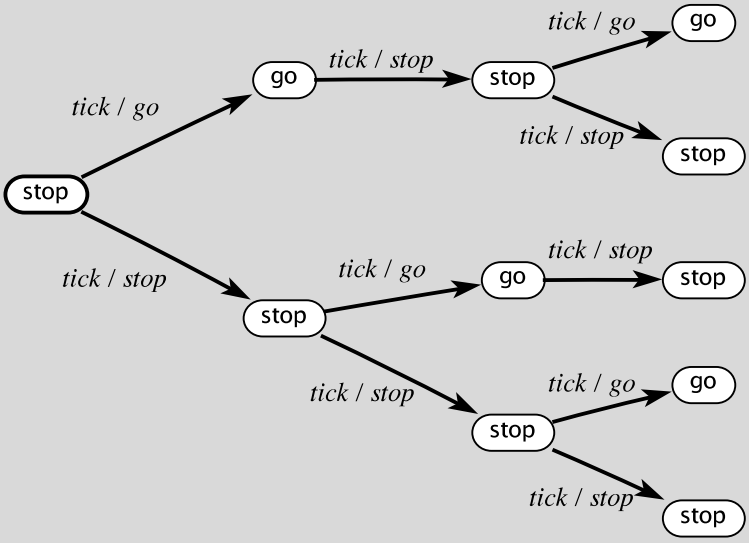
$$\mathsf{red} \xrightarrow{tick/go} \mathsf{green} \xrightarrow{tick/stop} \mathsf{yellow} \xrightarrow{tick/stop} \mathsf{red} \xrightarrow{tick/go} \cdots$$

   (c) Now consider merging the red and yellow states into a single stop state. Transitions that pointed into or out of those states are now directed into or out of the new stop state. Other transitions and the inputs and outputs stay the same. The new stop state is the new initial state. Is the resulting state machine deterministic? Why or why not? If it is deterministic, give a prefix of the trace of length 4. If it is non-deterministic, draw the computation tree up to depth 4.

Solution: The resulting state machine is given below. It is non-deterministic because there are two distinct transitions possible from state stop on input *tick*.

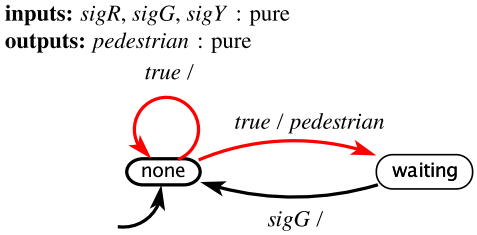


We have renamed the green state go. The computation tree for this FSM, up to depth 4, is given below:



6. This problem considers variants of the FSM in Figure 3.11, which models arrivals of pedestrians at a crosswalk. We assume that the traffic light at the crosswalk is controlled by the FSM in Figure 3.10. In all cases, assume a time triggered model, where both the pedestrian model and the traffic light model react once per second. Assume further that in each reaction, each machine sees as inputs the output produced by the other machine *in the same reaction* (this form of composition, which is called synchronous composition, is studied further in Chapter 6).

   (a) Suppose that instead of Figure 3.11, we use the following FSM to model the arrival of pedestrians:

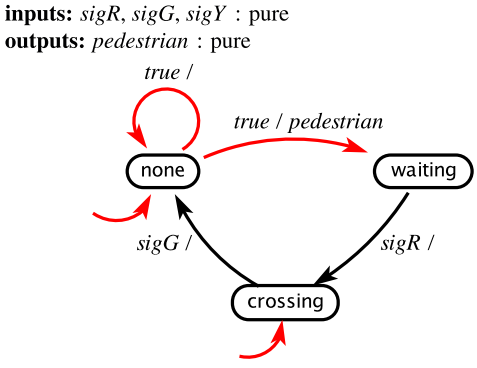


   Find a trace whereby a pedestrian arrives (the above machine transitions to waiting) but the pedestrian is never allowed to cross. That is, at no time after the pedestrian arrives is the traffic light in state red.

Solution: The traffic light begins in state red while the pedestrian model begins in state none. Suppose that the pedestrian model transitions to waiting in exactly the same reaction where the traffic light transitions to state green. The system will now perpetually remain in the same state, where the pedestrian model is in waiting and the traffic light is in state green.

Put another way, in the same reaction, we get red $\rightarrow$ green, which emits *sigG*, and none $\rightarrow$ waiting, which emits *pedestrian*. Once the composition is in state (green, none), all remaining reactions are stuttering transitions.

   (b) Suppose that instead of Figure 3.11, we use the following FSM to model the arrival of pedestrians:



   Here, the initial state is nondeterministically chosen to be one of none or crossing. Find a trace whereby a pedestrian arrives (the above machine transitions from none to waiting) but the pedestrian is never allowed to cross. That is, at no time after the pedestrian arrives is the traffic light in state red.

Solution: Suppose the initial state is chosen to be (red, none) and sometime in the first 60 reactions transitions to (red, waiting). Then eventually the composite machine will transition to (green, waiting), after which all reactions will stutter.

8. (NOTE: This exercise is rather advanced.) This exercise studies properties of discrete signals as formally defined in the sidebar on page 44. Specifically, we will show that discreteness is not a compositional property. That is, when combining two discrete behaviors in a single system, the resulting combination is not necessarily discrete.

   (a) Consider a pure signal $x: \mathbb{R} \to \{present, absent\}$ given by

   $$x(t) = \begin{cases} present & \text{if } t \text{ is a non-negative integer} \\ absent & \text{otherwise} \end{cases}$$

   for all $t \in \mathbb{R}$. Show that this signal is discrete.

   Solution: We need only to give an order-preserving one-to-one function of the form $f: T \to \mathbb{N}$, where T is defined as in the sidebar. In this case, the set of times when the signal is present is $T = \mathbb{N}$, so we can choose the identity function for f, which is trivial order preserving and one-to-one.

   (b) Consider a pure signal $y: \mathbb{R} \to \{present, absent\}$ given by

   $$y(t) = \begin{cases} present & \text{if } t = 1 - 1/n \text{ for any positive integer } n \\ absent & \text{otherwise} \end{cases}$$

   for all $t \in \mathbb{R}$. Show that this signal is discrete.

   Solution: We need only to give an order-preserving one-to-one function of the form $f: T \to \mathbb{N}$. In this case, the set of times when the signal is present is

   $$T = \{1 - 1/1, 1 - 1/2, 1 - 1/3, \cdots, 1 - 1/n, \cdots\}$$

   or

   $$T = \{0, 1/2, 2/3, \cdots\}$$

   Thus, we can define f as

   $$\forall\, t \in T, \quad f(t) = n \text{ where } t = 1 - 1/n\ .$$

   This is clearly one-to-one and order preserving. Hence, y is discrete.

   (c) Consider a signal w that is the merge of x and y in the previous two parts. That is, $w(t) = present$ if either $x(t) = present$ or $y(t) = present$, and is *absent* otherwise. Show that w is not discrete.

   Solution: Assume to the contrary that w is discrete. Then there exists an order-preserving one-to-one function $f: T \to \mathbb{N}$. Note that $1 \in T$ and that $1 - 1/n \in T$ for all $n \in \mathbb{N}$, $n > 0$. Moreover, $1 > 1 - 1/n$ for all $n \in \mathbb{N}$, $n > 0$. Since f is one-to-one and order preserving, it must be true that

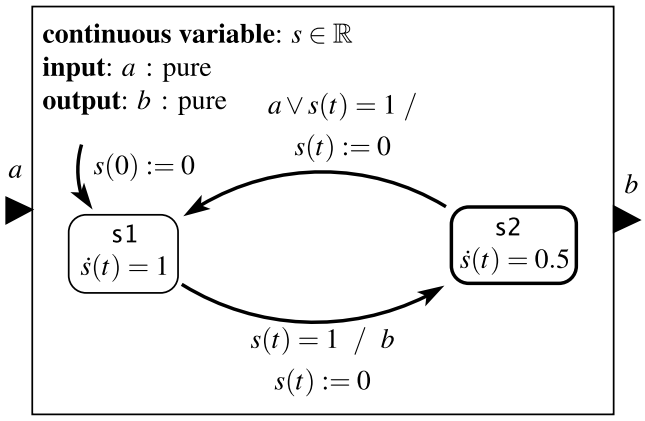
   $$f(1) > f(1 - 1/n)\ .$$

   However, $1 - 1/n$ has no upper bound for $n \in \mathbb{N}$, $n > 0$, so $f(1 - 1/n)$ has no upper bound in $\mathbb{N}$, a contradiction. Hence, w must not be discrete.

   (d) Consider the example shown in Figure 3.1. Assume that each of the two signals *arrival* and *departure* is discrete. Show that this does not imply that the output *count* is a discrete signal.

Solution: Let *arrival* $= x$ from part (a) and *departure* $= y$ from part (b). Then the set T of times when *count* is present is the same as the set T in part (c). Therefore, *count* is not discrete.

4. Consider the following timed automaton:



Assume that the input signals a and b are discrete continuous-time signals, meaning that each can be given as a function of form $a: \mathbb{R} \to \{present, absent\}$, where at almost all times $t \in \mathbb{R}$, $a(t) = absent$. Assume that the state machine can take at most one transition at each distinct time t, and that machine begins executing at time $t = 0$.

(a) Sketch the output b if the input a is present only at times

$$t = 0.75, 1.5, 2.25, 3, 3.75, 4.5, \cdots$$

Include at least times from $t = 0$ to $t = 5$.

Solution: The output is present at times $t = 1, 2.5, 4, \cdots$.

(b) Sketch the output b if the input a is present only at times $t = 0, 1, 2, 3, \cdots$.

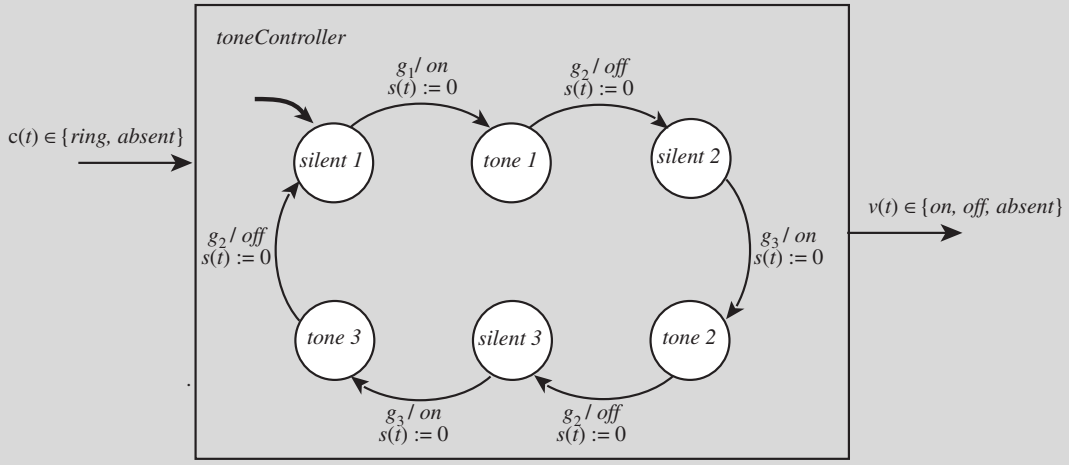
Solution: The output is present at times $t = 1, 3, 5, 7, \cdots$.

(c) Assuming that the input a can be any discrete signal at all, find a lower bound on the amount of time between events b. What input signal a (if any) achieves this lower bound?

Solution: The lower bound is 1. There is no input a that achieves this bound, but an input that comes arbitrarily close is where a is present at times $t = 1 + \varepsilon, 2 + 2\varepsilon, 3 + 3\varepsilon, \cdots$, for any $\varepsilon > 0$.

5. You have an analog source that produces a pure tone. You can switch the source on or off by the input event *on* or *off*. Construct a timed automaton that provides the *on* and *off* signals as outputs, to be connected to the inputs of the tone generator. Your system should behave as follows. Upon receiving an input event *ring*, it should produce an 80 ms-long sound consisting of three 20 ms-long bursts of the pure tone separated by two 10 ms intervals of silence. What does your system do if it receives two *ring* events that are 50 ms apart?

Solution: Assume the input alphabet is $\{ring, absent\}$ and the output alphabet is $\{on, off, absent\}$. Then the following timed automaton will control the source of the tone:



All states except *silent 1* have as a refinement the system given by

$$\dot{s}(t) = 1.$$

The guards are given by

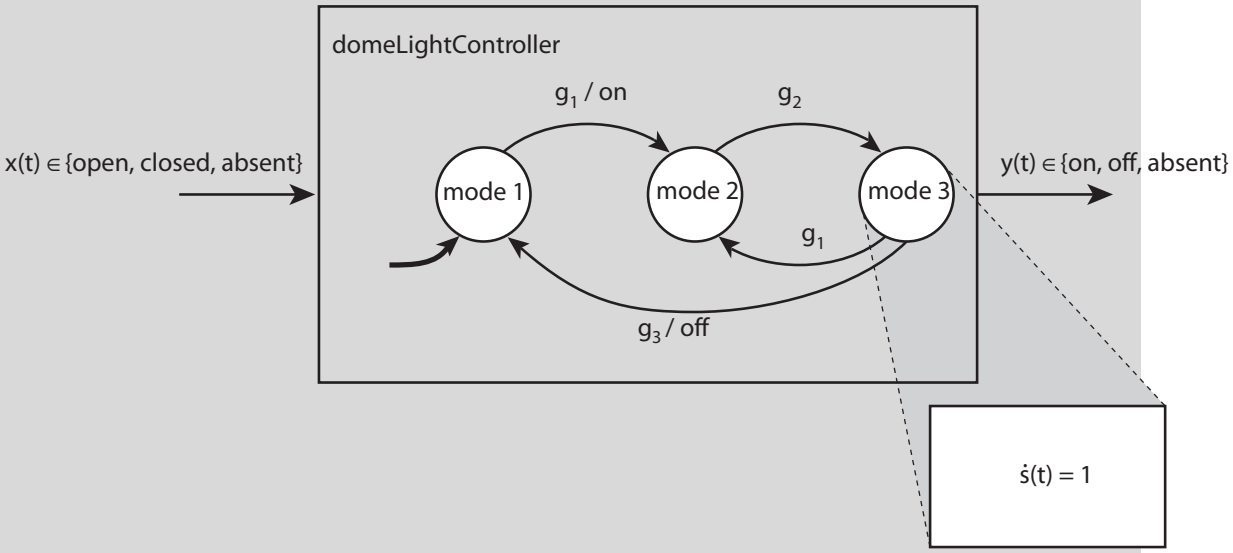
$$\begin{aligned} g_1 &= \{(c(t), s(t)) \mid c(t) = ring\} \\ g_2 &= \{(c(t), s(t)) \mid s(t) = 20\} \\ g_3 &= \{(c(t), s(t)) \mid s(t) = 10\}. \end{aligned}$$

This system ignores a *ring* input event that occurs less than 80 ms after the previous *ring* event.

6. Automobiles today have the features listed below. Implement each feature as a timed automaton.

   (a) The dome light is turned on as soon as any door is opened. It stays on for 30 seconds after all doors are shut. What sensors are needed?

Solution: Assume that the automobile provides the input *open* when the first door is opened and *closed* when the last open door is closed. The following machine provides *on* to turn on the dome light and *off* to turn it off:



The guards are given by

$$\begin{aligned} g_1 &= \{(x(t), s(t)) \mid x(t) = open\} \\ g_2 &= \{(x(t), s(t)) \mid x(t) = closed\} \\ g_3 &= \{(x(t), s(t)) \mid s(t) = 30\}. \end{aligned}$$

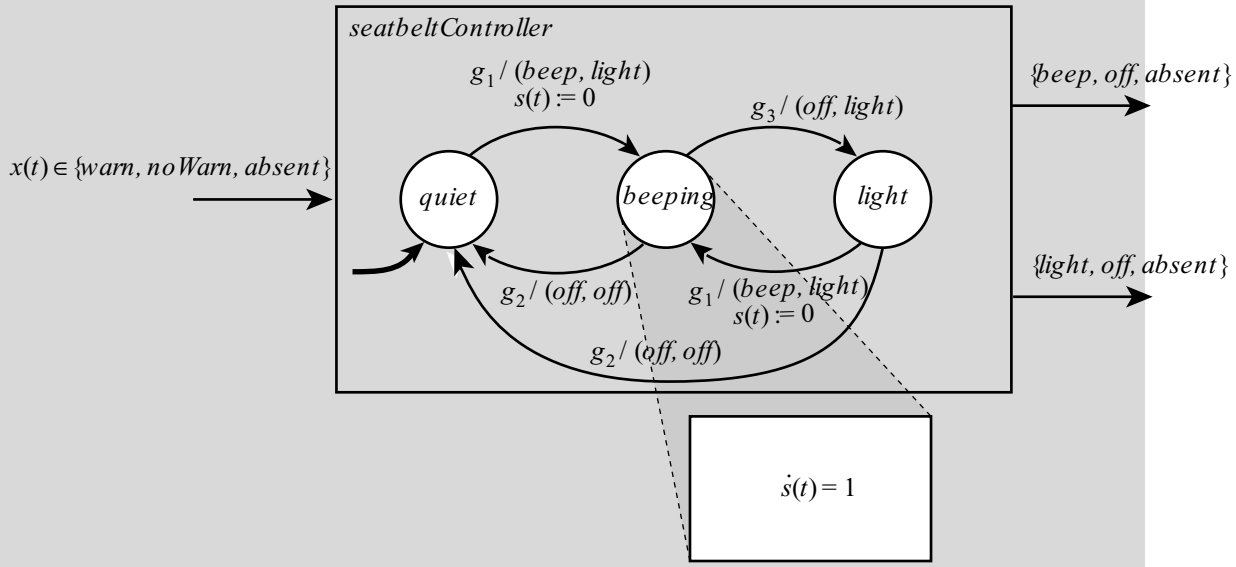
Such sensors can be easily implemented as a **daisy chain**, a series connection of switches so that if any door is open, the electrical circuit is open.

   (b) Once the engine is started, a beeper is sounded and a red light warning is indicated if there are passengers that have not buckled their seat belt. The beeper stops sounding after 30 seconds, or as soon the seat belts are buckled, whichever is sooner. The warning light is on all the time the seat belt is unbuckled. **Hint:** Assume the sensors provide a *warn* event when the ignition is turned on and there is a seat with passenger not buckled in, or if the ignition is already on and a passenger sits in a seat without buckling the seatbelt. Assume further that the sensors provide a *noWarn* event when a passenger departs from a seat, or when the buckle is buckled, or when the ignition is turned off.

Solution: Assume that the vehicle sensors provide the following input alphabet,

$$Inputs = \{warn, noWarn, absent\},$$

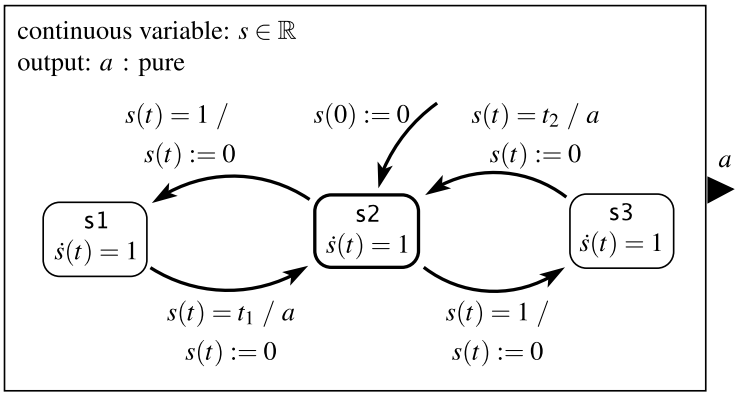
as suggested in the hint. The following model provides the requisite control:



The guards are given by

$$
\begin{aligned}
g_1 &= \{(x(t), s(t)) \mid x(t) = warn\} \\
g_2 &= \{(x(t), s(t)) \mid x(t) = noWarn\} \\
g_3 &= \{(x(t), s(t)) \mid s(t) = 30 \wedge x(t) = absent\}.
\end{aligned}
$$

8. Consider the following timed automaton:



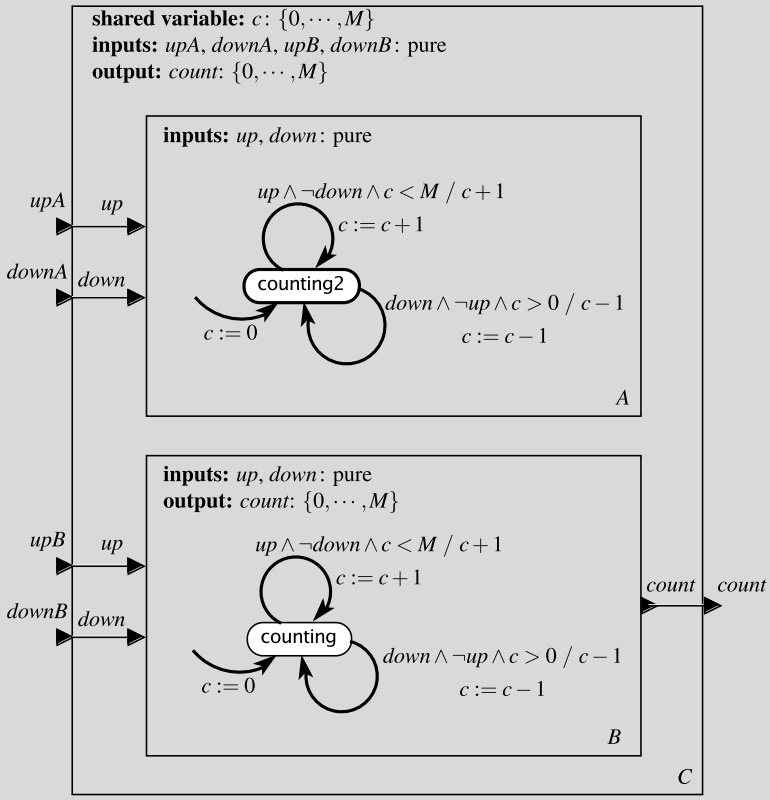
Assume t_1 and t_2 are positive real numbers. What is the minimum amount of time between events a? That is, what is the smallest possible time between two times when the signal a is present?

Solution: $1 + \min(t_1, t_2)$.

5 Composition of State Machines — Exercises

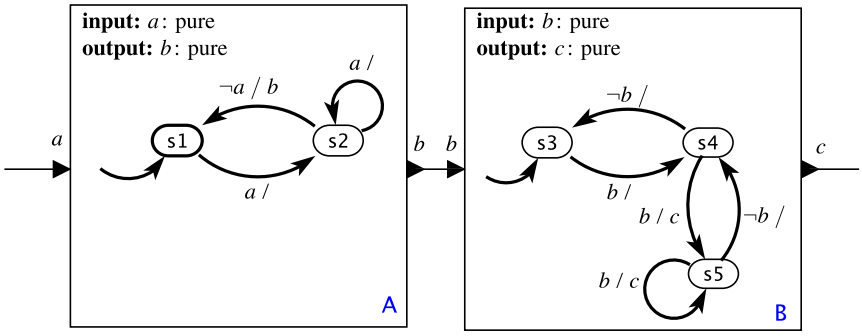
1. Consider the extended state machine model of Figure 3.8, the garage counter. Suppose that the garage has two distinct entrance and exit points. Construct a side-by-side concurrent composition of two counters that share a variable c that keeps track of the number of cars in the garage. Specify whether you are using synchronous or asynchronous composition, and define exactly the semantics of your composition by giving a single machine modeling the composition. If you choose synchronous semantics, explain what happens if the two machines simultaneously modify the shared variable. If you choose asynchronous composition, explain precisely which variant of asynchronous semantics you have chosen and why. Is your composition machine deterministic?

Solution: A solution is shown here:



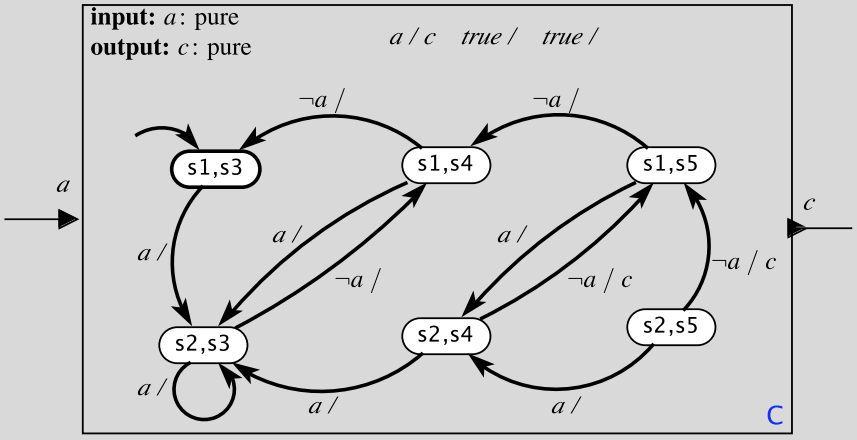
It would not be acceptable in this case to miss events, so asynchronous semantics 1 will not be a good choice. We can choose, for example, a synchronous interleaving semantics where machine *A* always reacts before machine *B*. Note that if we allow the order of reactions to be nondeterministic, then the *count* output will not necessarily reflect the final number of cars in the garage.

3. Consider the following synchronous composition of two state machines A and B:



Construct a single state machine C representing the composition. Which states of the composition are unreachable?

Solution:



The following states are unreachable: (s1,s5), (s2,s4), and (s2,s5).