Behavioural Equivalence: An introduction to Bisimulation

Ernesto Posse

February 4, 2002

1 Introduction

When modelling and reasoning about systems there is an essential question in which we are interested: "When are two given systems considered to behave 'in the same way'?". One important motivation for such a question is reusability: if two systems are proved to 'be equivalent' then we should be able to replace one by the other in any context. Another important motivation is that equivalent systems should satisfy the same properties of interest, hence reasoning about one system will tell us about all the systems equivalent to it.

The systems in which we are interested are *dynamic* in the sense that they have a *state* which evolves over time. This means that the question above can be rephrased as "when are two states considered equivalent?". If we are able to divide the total state space in equivalence classes, then reasoning about how systems behave becomes much simpler. Rather than dealing with a very large state space we would be dealing with a small state space.

We are interested in how systems *behave* rather than what is their *structure*. For instance if we were comparing programs, we might be inclined to consider two programs equivalent if they have the same text up-to variable-renamings. However this is too specific. There are many programs whose text is different, yet they behave in the same way.

The notion of *bisimulation* is introduced here as an appropriate notion of behavioural equivalence. The definitions of simulation and bisimulation are from [2] and [3].

2 Background

Here is some basic background.

Definition 1. *A* ***binary relation*** *R on a set S is a subset of $S \times S$. If two elements x and y of S are related by R, we write xRy to mean $(x, y) \in R$. Some basic types of relations are defined as follows:*

- *A binary relation $R \subseteq S \times S$ is said to be* ***reflexive*** *if for all $x \in S$, xRx.*

- *A binary relation $R \subseteq S \times S$ is said to be* ***symmetric*** *if for all $x, y \in S$, if xRy then yRx.*
- *A binary relation $R \subseteq S \times S$ is said to be* ***transitive*** *if for all $x, y, z \in S$, if xRy and yRz then xRz.*
- *A binary relation $R \subseteq S \times S$ is said to be an* ***preorder relation*** *if it is reflexive and transitive.*
- *A binary relation $R \subseteq S \times S$ is said to be an* ***equivalence relation*** *if it is a symmetric preorder.*

Given an equivalence relation and an element, we can talk about the class of all the elements that are equivalent to the given element. This is formalized by the concept of *equivalence classes.*

Definition 2. *Given an equivalence relation R over a set S, and an element $x \in S$, the* ***equivalence class*** *of x with respect to R, written $[x]_R$ is defined as:*

$$[x]_R \stackrel{def}{=} \{y \in S \mid xRy\}$$

The ***quotient*** *of S with respect to the equivalence relation R, denoted S/R is the set of equivalence classes of S, i.e.:*

$$S/R \stackrel{def}{=} \{[x]_R \mid x \in S\}$$

Note that the quotient of a set is a partition of the set, i.e. the intersection between any two equivalence classes is empty, and the union of all the equivalence classes is the original set.

3 Transition Systems

The concept of transition system is intended to formalize a kind of dynamic systems. To describe a dynamic system, we need to define what are its possible states, and how does the state of the system changes over time, i.e. for each state what can happen, and what will be the next state.

Definition 3. *A* ***labelled transition system***[1] *$\mathcal{T}$ is a tuple $(S, \mathcal{L}, \rightarrow)$ where S is a set (of possible states), $\mathcal{L}$ is a set (of labels), and $\rightarrow \subseteq S \times \mathcal{L} \times S$ is called the* ***transition relation****. If $s, s' \in S$ and $\alpha \in \mathcal{L}$ we write $s \stackrel{\alpha}{\rightarrow} s'$ for $(s, \alpha, s') \in \rightarrow$.*

The concept of transition relation is intended to describe the evolution of state in the system with respect to the set $\mathcal{L}$. There are several possible interpretations, depending on what is this set $\mathcal{L}$. In one interpretation we can think of $\mathcal{L}$ as the set of possible actions or interactions that the system can perform. Thus, $s \stackrel{\alpha}{\rightarrow} s'$ means that when the system is in state s, it can perform an action

[1]This terminology is different to the one used in the book [1], where such systems consist of a set, and a binary relation over the set, and the labels are part of the state.

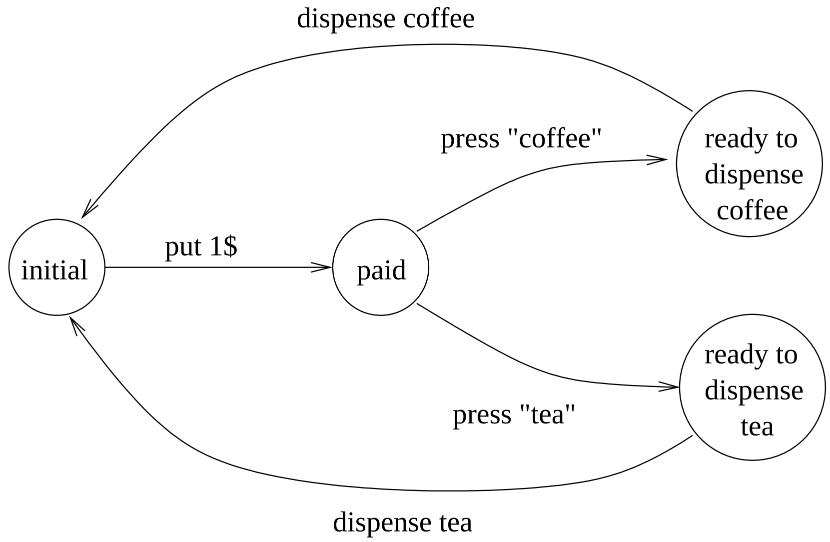

Figure 1: Simple vending machine

α and the new state will be s'. The set of labels could also represent conditions from the environment, in which case $s \xrightarrow{\alpha} s'$ means that if the system is in state s and the condition α is satisfied, then the new state is s'.

Consider the example of a vending machine. We can model such machine as a labelled transition system. A vending machine receives a coin, then lets the customer select a product, say coffee or tea, delivers the product and becomes ready again to receive more money. Graphically we can depict this as shown in figure 1.

We can define a labelled transition system describing this machine as follows. Define the following states:

- s_0 is the initial state.
- s_1 is the state "paid".
- s_2 is the state "ready to dispense coffee".
- s_3 is the state "ready to dispense tea".

Define also the following actions:

- $\$$ is the action of putting 1$ in the machine.
- c is the action of pressing the "coffee" button.
- t is the action of pressing the "tea" button.
- $\bar{c}$ is the action of dispensing coffee.

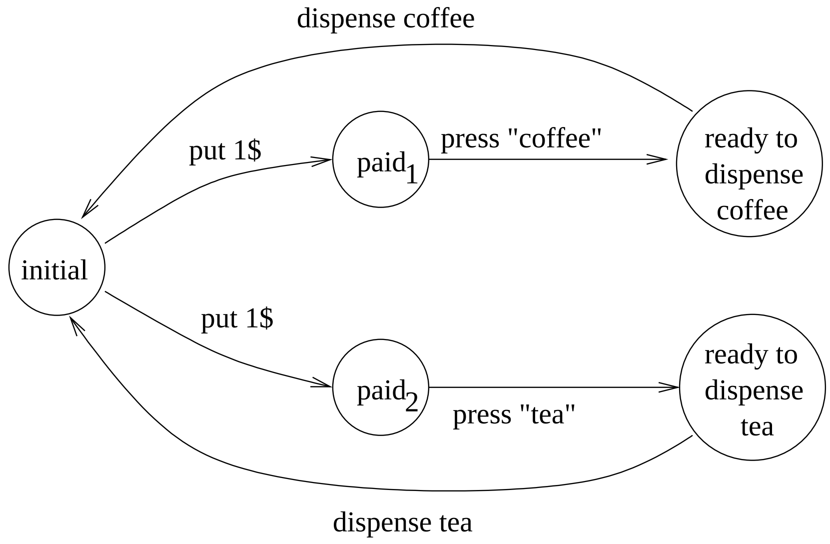

Figure 2: Mean vending machine

- $\bar{t}$ is the action of dispensing tea.

The labelled transition system is the triple $(S, \mathcal{L}, \rightarrow)$ where $S \stackrel{def}{=} \{s_0, s_1, s_2, s_3\}$, $\mathcal{L} \stackrel{def}{=} \{\$, c, t, \bar{c}, \bar{t}\}$, and $\rightarrow \stackrel{def}{=} \{s_0 \stackrel{\$}{\rightarrow} s_1, s_1 \stackrel{c}{\rightarrow} s_2, s_1 \stackrel{t}{\rightarrow} s_3, s_2 \stackrel{\bar{c}}{\rightarrow} s_0, s_3 \stackrel{\bar{t}}{\rightarrow} s_0\}$.

Note that this machine is <u>not</u> a finite state automaton: the transition relation is not a function, i.e. there are states for which some transitions are not defined, for example, in the initial state the customer does not have the option to press the "coffee" button.

4 Simulation

Consider another vending machine, designed to give the customer what the machine wants, not the customer. Figure 2 depicts such a machine.

We formalize it similarly as follows: let t_0 be the initial state, t_1 be the state "paid$_1$", t_2 be the state "paid$_2$", t_3 be the state "ready to dispense coffee", and t_4 be the state "ready to dispense tea". We use the same set of labels $\mathcal{L}$ as before. The labelled transition system is then $(S', \mathcal{L}, \rightarrow')$ where $S' \stackrel{def}{=} \{t_0, t_1, t_2, t_3, t_4\}$, $\mathcal{L} \stackrel{def}{=} \{\$, c, t, \bar{c}, \bar{t}\}$, and $\rightarrow' \stackrel{def}{=} \{t_0 \stackrel{\$}{\rightarrow} t_1, t_0 \stackrel{\$}{\rightarrow} t_2, t_1 \stackrel{c}{\rightarrow} t_3, t_2 \stackrel{t}{\rightarrow} t_4, t_3 \stackrel{\bar{c}}{\rightarrow} t_0, t_4 \stackrel{\bar{t}}{\rightarrow} t_0\}$.

Note that although this is a non-deterministic system, since the machine has a choice in the initial state, it is <u>not</u> an NFA. Standard Automata Theory ([4]) doesn't differentiate between "internal choice" and "external choice". In this example, the choice is internal (made by the machine), whereas in the previous it was external (made by the customer). NFA's do not deal either with the concept of rejection (absence of a transition from a state), or the concept of "observable difference".

Any reasonable notion of behavioural equivalence must take into account observable differences. This is, if the machines interact differently with an external observer, then they should be regarded as different.

The two machines behave in a very different way from the point of view of an external observer that interacts with them (the customer), even though, considered as NFA's, they both generate the same language. Therefore, "generating the same language" is not a good answer for the question "do the machines behave in the same way?".

A possible alternative is to say that the machines are equivalent if they can "simulate each other", or "match each other's moves". This is made precise be the notion of *simulation*.

Definition 4. *Given a labelled transition system $\mathcal{T} = (S, \mathcal{L}, \rightarrow)$, and two states $s, s' \in S$ we say that s'* ***simulates*** *s, written $s \preceq s'$, if whenever $s \xrightarrow{\alpha} t$ for some $\alpha \in \mathcal{L}$ and some $t \in S$, then there is a $t' \in S$ such that $s' \xrightarrow{\alpha} t'$ and t' simulates t.*

An informal reading of this definition is as follows: the state s' simulates s if whenever s makes a move α to some state t, then s' can match the same move α and result in a state t' that simulates t.

Note that the fact that we defined two separate labelled transition systems for each machine is no obstacle to talk about one simulating the other, since we can create a new transition system that is the disjoint union of the two given systems.[2]

A basic fact about the similarity relation is the following:

Proposition 1. *The similarity relation $\preceq$ is a preorder.*

Continuing with the example, suppose that we have a labelled transition system that combines both machines so that we can ask the question of whether the states of one system simulate the states of the other. We can see that $s_0 \not\preceq t_0$, this is, that t_0 does not simulate s_0. The reason for this is that the machine that starts in t_0 cannot match all the moves of the machine starting in s_0. To see this, notice there is no state t_i that simulates s_1. The state s_1 has two options: $s_1 \xrightarrow{c} s_2$ and $s_1 \xrightarrow{t} s_3$. However there is no state in the second machine that has both options c and t: $t_1 \xrightarrow{c} t_3$ but $t_1 \not\xrightarrow{t} t_4$ and $t_2 \xrightarrow{t} t_4$ but $t_2 \not\xrightarrow{c} t_3$. Hence we can be sure that the two machines behave differently.

On the other hand, it is the case that the machine in figure 1 simulates the machine in figure 2, that is, $t_0 \preceq s_0$. This is so because $t_1 \preceq s_1$, $t_2 \preceq s_1$, $t_3 \preceq s_2$, and $t_4 \preceq s_3$. Each of these can be easily checked. For instance, $t_2 \preceq s_1$ because the only possible action for t_2 is $t_2 \xrightarrow{t} t_4$, which can be matched by s_1, since $s_1 \xrightarrow{t} s_3$, and $t_4 \preceq s_3$.

[2]Technically speaking, if $T = (S, \mathcal{L}, \rightarrow)$ and $T' = (S', \mathcal{L}', \rightarrow')$ are two labelled transition systems, we can define a new labelled transition system $T + T' \stackrel{def}{=} (S \uplus S', \mathcal{L} \uplus \mathcal{L}, \rightarrow \uplus \rightarrow')$.

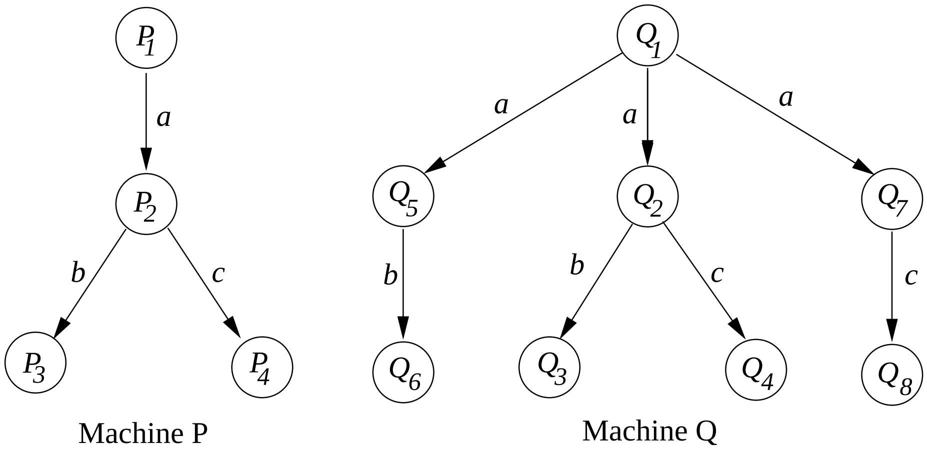

Figure 3: Are these machines equivalent?

Definition 5. *Given a labelled transition system $(S, \mathcal{L}, \rightarrow)$, and two states $s, t \in S$, we say that s and t are* ***two-way similar****, written $s \asymp t$, if $s \preceq t$ and $t \preceq s$, that is, if one simulates the other and viceversa.*

Proposition 2. *Two-way similarity is an equivalence relation.*

Summarizing for the two machines we have that $t_0 \preceq s_0$ but $s_0 \not\preceq t_0$. Hence $s_0 \not\asymp t_0$, i.e. they are not two-way similar.

5 Bisimulation

There is a problem with two-way similarity as a behavioural equivalence. Consider the machines shown in figure 3. These machines are two-way similar, i.e. $P_1 \asymp Q_1$. However there is something strange about them. For instance there is no state in P that corresponds the state Q_5 of Q, in the sense that Q_5 does not simulate any state of P. This means that the machine does not always interact in the same way with its environment. This is so, because P_2 simulates Q_5, but Q_5 does not have a c transition, while P_2 does, so Q_5 does not simulate P_2.

This example shows that although two-way simulation seems a reasonable candidate for behavioural equivalence, it does not completely capture the concept we want. To deal with this, a new notion, closely related to simulation is introduced. Even though the following definition looks very similar to simulation, it is in fact quite different.

Definition 6. *Given a labelled transition system $\mathcal{T} = (S, \mathcal{L}, \rightarrow)$, and two states $s, s' \in S$ we say that s and s' are* ***bisimilar****, written $s \sim s'$, if:*

- *Whenever $s \stackrel{\alpha}{\rightarrow} t$ for some $\alpha \in \mathcal{L}$ and some $t \in S$, then there is a $t' \in S$ such that $s' \stackrel{\alpha}{\rightarrow} t'$ and $t \sim t'$, and*

- *Whenever $s' \stackrel{\alpha}{\to} t'$ for some $\alpha \in \mathcal{L}$ and some $t' \in S$, then there is a $t \in S$ such that $s \stackrel{\alpha}{\to} t$ and $t \sim t'$.*

This definition now captures the concept of "matching each other's actions" with respect to an external observer, in a much better fashion.

The following are some basic facts about bisimilarity.

Proposition 3 (Basic bisimilarity properties).

(i) Bisimilarity is an equivalence relation.

(ii) Bisimilarity is stronger than similarity: $\sim \subseteq \preceq$, this is, if two states are bisimilar, then they are similar.

(iii) Bisimilarity is stronger than two-way similarity: $\sim \subseteq \asymp$, this is, if two states are bisimilar, then they are two-way similar.

In the example of figure 3, we have that $P_0 \asymp Q_0$ but $P_0 \not\sim Q_0$.

6 Conclusions

We have defined the notion of labelled transition systems as a way to model dynamic systems. Based on this we have defined two notions of behavioural equivalence: two-way similarity, and bisimilarity. Bisimilarity is stronger, which means that given a state space S, $S/\sim$ is larger than $S/\asymp$, or in other words, that less systems are considered "equivalent" by bisimilarity, than by two-way similarity. This is because bisimilarity is more sensitive to "observable differences". This makes bisimilarity an appropriate notion of behavioural equivalence.

References

[1] Michael R. A. Huth and Mark D. Ryan. *Logic in Computer Science: Modelling and reasoning about systems.* Cambridge University Press, 2000.

[2] Robin Milner. *Communication and Concurrency.* Prentice Hall, 1989.

[3] Robin Milner. *Communicating and mobile systems: the π-calculus.* Cambridge University Press, 1999.

[4] Miachel Sipser. *Introduction to the Theory of Computation.* PWS, Boston, MA, 1996.