Behavioral Equivalences

Rocco De Nicola

Dipartimento di Sistemi e Informatica, Università degli Studi di Firenze
Viale Morgagni 65, 50134 Firenze, Italia
rocco.denicola@unifi.it

DEFINITION: Beahvioral equivalences serve to establish in which cases two reactive (possible concurrent) systems offer similar interaction capabilities relatively to other systems representing their operating environment. Behavioral equivalences have been mainly developed in the context of *process algebras*, mathematically rigorous languages that have been used for describing and verifying properties of concurrent communicating systems. By relying on the so called structural operational semantics (SOS), labelled transition systems, are associated to each term of a process algebra. Behavioral equivalences are used to abstract from unwanted details and identify those labelled transition systems that react "similarly" to external experiments. Due to the large number of properties which may be relevant in the analysis of concurrent systems, many different theories of equivalences have been proposed in the literature. The main contenders consider those systems equivalent that (i) perform the same sequences of actions, or (ii) perform the same sequences of actions and after each sequence are ready to accept the same sets of actions, or (iii) perform the same sequences of actions and after each sequence exhibit, recursively, the same behavior. This approach leads to many different equivalences that preserve significantly different properties of systems.

SYNONYMS: Extensional Equivalences, Behavioral Relations.

RELATED ENTRIES: Actors, Bisimulation, CCS, CSP, Pi-Calculus, Process Algebras.

1 Introduction

In many cases, it is useful to have theories which can be used to establish whether two systems are equivalent or whether one is a satisfactory "approximation" of another. It can be said that a system S_1 is equivalent to a system S_2 whenever "some" aspects of the externally observable behavior of the two systems are compatible. If the same formalism is used to model what is required of a system (its *specification*) and how it can actually be built (its *implementation*) then it is possible to use theories based on equivalences to prove that a particular concrete description is correct with respect to a given abstract one. If a step-wise development method is used, equivalences may permit substituting large specifications with equivalent concise ones. In general it is useful to be able to interchange subsystems proved behaviorally equivalent, in the sense that one subsystem may replace another as part of a larger system without affecting the behavior of the overall system.

The kind of equivalences, or approximations, involved depends very heavily on how the systems under consideration will be used. In fact, the way a system is used determines the behavioral aspects which must be taken into account and those which can be ignored. It is then important to know, for the considered equivalence, the systems properties it preserves.

In spite of the general agreement on taking an extensional approach for defining the equivalence of concurrent or nondeterministic systems, there is still disagreement on what "reasonable" observations are and how their outcomes can be used to distinguish or identify systems. Many different theories of equivalences have been proposed in the literature for models which are intended to be used to describe and reason about concurrent or nondeterministic systems. This is mainly due to the large number of properties which may be relevant in the analysis of such systems. Almost all the proposed equivalences are based on the idea that two systems are equivalent whenever no external observation can distinguish them. In fact, for any given system it is not its internal structure which is of interest but its behavior with respect to the outside world, i.e., its effect on the environment and its reactions to stimuli from the environment.

One of the most successful approaches for describing the formal, precise, behavior of concurrent systems is the so called *operational semantics*. Within this approach, concurrent programs or systems are modeled as labelled transition systems (LTSs) that consist of a set of states, a set of transition labels and a transition relation. The states of the transition systems are programs while the labels of the transitions between states represent the actions (instructions) or the interactions that are possible in a given state.

When defining behavioral equivalence of concurrent systems described as LTSs, one might think that it is possible to consider systems equivalent if they give rise to the same (isomorphic) LTSs. Unfortunately, this would lead to unwanted distinctions, e.g., it would consider the two LTSs below different

$$\begin{tikzcd} p \arrow[loop above, "a"] & & & q \arrow[r, "a"] & q_1 \arrow[loop above, "a"] \end{tikzcd}$$

in spite of the fact that their behavior is the same; they can (only) execute infinitely many a-actions, and they should thus be considered equivalent.

The basic principles for any reasonable equivalence can be summarized as follows. It should:

- abstract from states (consider only the actions);
- abstract from internal behaviour
- identify processes whose LTSs are isomorphic;
- consider two processes equivalent only if both can execute the same actions sequences;
- allow to replace a subprocess by an equivalent counterpart without changing the overall semantics of the system.

However, these criteria are not sufficiently insightful and discriminative and the above adequacy requirements turn out to be still too loose. They have given rise to many different kinds of equivalences, even when all actions are considered visible.

The main equivalences over LTSs introduced in the literature consider as equivalent those systems that

1. perform the same sequences of actions,
2. perform the same sequences of actions and after each sequence are ready to accept the same sets of actions,
3. perform the same sequences of actions and after each sequence exhibit, recursively, the same behavior.

These three different criteria lead to three groups of equivalences that are known as *traces* equivalences, *decorated-traces* equivalences, and *bisimulation-based* equivalences. Equivalences in different classes behave differently relatively to the three labelled transition systems in Fig. 1. The three systems represent the specifications of three vending machines that accept two coins and deliver coffee or tea. The trace based equivalences equate all of them, the bisimulation based equivalences distinguish all of them, and the decorated traces distinguish the leftmost system from the other two, but equate the central and the rightmost one.

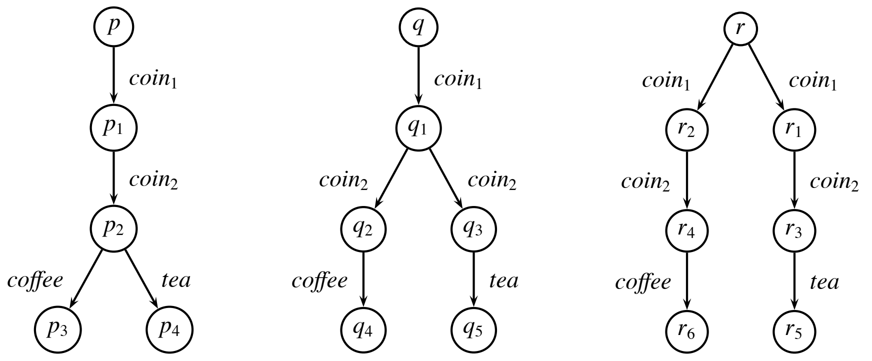


Fig. 1. Three Vending Machines

Many of these equivalences have been reviewed [11]; here, only the main ones are presented. First, equivalences that consider invisible (τ)actions just normal actions will be presented, then their variants that abstract from internal actions will be introduced.

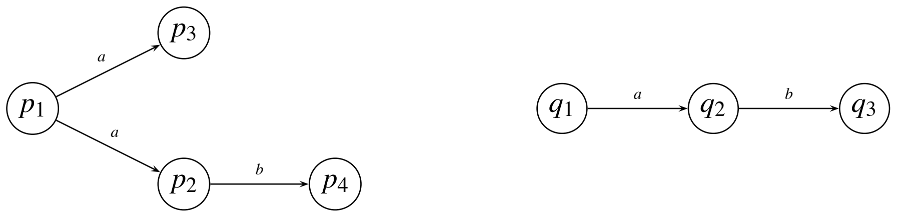
The equivalences will be formally defined on states of LTSs of the form $\langle Q, A_\tau, \overset{\mu}{\longrightarrow} \rangle$ where Q is a set of states, ranging over p, q, p', $q_1, \ldots$, A_τ is the set of labels, ranging over a, b, $c, \ldots$, that also contains the distinct silent action τ, and $\overset{\mu}{\longrightarrow}$ is the set of transitions. In the following, s will denote a generic element of A_τ^*, the set of all sequences of actions that a process might perform.

2 Traces Equivalence

The first equivalence is know as *traces equivalence* and is perhaps the simplest of all; it is imported from automata theory that considers those automata equivalent that generate the same language. Intuitively, two processes are deemed traces equivalent if and only if they can perform exactly the same sequences of actions. In the formal definition, $p \overset{s}{\longrightarrow} p'$, with $s = \mu_1\mu_2 \ldots \mu_n$, denotes the sequence $p \overset{\mu_1}{\longrightarrow} p_1 \overset{\mu_2}{\longrightarrow} p_2 \ldots \overset{\mu_n}{\longrightarrow} p'$ of transitions.

Two states p and q are *traces equivalent* ($p \simeq_T q$) if :
1. $p \overset{s}{\longrightarrow} p'$ implies $q \overset{s}{\longrightarrow} q'$ for some q' and
2. $q \overset{s}{\longrightarrow} q'$ implies $p \overset{s}{\longrightarrow} p'$ for some p'.

A drawback of $\simeq_T$ is that it is not sensitive to deadlocks. For example, if we consider the two LTSs below:



we have that $P_1 \simeq_T Q_1$ but P_1, unlike Q_1, after performing action a, can reach a state in which it cannot perform any action, i.e., a deadlocked state.

Traces equivalence identifies all of the three LTSs of Fig. 1, indeed it is not difficult to see that the three vending machines can perform the same sequences of visible actions. Nevertheless, a customer with definite preferences for coffee who is offered to choose between the three machines would definitely select to interact with the leftmost one, since the others do not let him choose what to drink.

3 Bisimulation Equivalence

The classical alternative to traces equivalence is bisimilarity (also known as observational equivalence), that considers equivalent two systems that can simulate each other step after step [8]. Bisimilarity is based on the notion of bisimulation:

A relation $R \subseteq Q \times Q$ is a *bisimulation* if, for any pair of states p and q such that $\langle p, q \rangle \in R$, the following holds:

1. for all $\mu \in A_\tau$ and $p' \in Q$, if $p \xrightarrow{\mu} p'$ then $q \xrightarrow{\mu} q'$ for some $q' \in Q$ s.t. $\langle p', q' \rangle \in R$;
2. for all $\mu \in A_\tau$ and $q' \in Q$, if $q \xrightarrow{\mu} q'$ then $p \xrightarrow{\mu} p'$ for some $p' \in Q$ s.t. $\langle p', q' \rangle \in R$.

Two states p, q are *bisimilar* ($p \sim q$) if there exists a bisimulation R such that $\langle p, q \rangle \in R$.

This definition corresponds to the circular definition below, that more clearly shows that two systems are bisimilar (observationally equivalent) if they can perform the same action and reach bisimilar states. This recursive definition can be solved with the usual fixed points techniques.

Two states $p, q \in Q$ are *bisimilar*, written $p \sim q$, if and only if for each $\mu \in A_\tau$:

1. if $p \xrightarrow{\mu} p'$ then $q \xrightarrow{\mu} q'$ for some q' such that $p' \sim q'$;
2. if $q \xrightarrow{\mu} q'$ then $p \xrightarrow{\mu} p'$ for some p' such that $p' \sim q'$.

Bisimilarity distinguishes all machines of Fig. 1. This is because the basic idea behind bisimilarity is that two states are considered equivalent if by performing the same sequences of actions from these states it is possible to reach equivalent states. It is not difficult to see that bisimilarity distinguishes the first and the second machine of Fig. 1 because after receiving two coins ($coin_1$ and $coin_2$) the first machine still offers the user the possibility of choosing between having *coffee* or *tea* while the second does not. To see that also the second and the third machine are distinguished, it is sufficient to consider only the states reachable after just inserting $coin_1$, because already after this insertion the user looses his control of the third machine, indeed there is no way for this machine to reach a state bisimilar to the one that the second machine reaches after accepting $coin_1$.

4 Testing Equivalence

The formulation of bisimilarity is mathematically very elegant and has received much attention also in other fields of computer science [10]. However, some researchers do consider it too discriminating: two processes may be deemed unrelated even though there is no practical way of ascertaining it. As an example consider the two rightmost vending machines of Fig. 1. They are not bisimilar, because after inserting the first coin in one case there is still the illusion of having the possibility of choosing what to drink. Nevertheless, a customer would not be able to appreciate their differences, since there is no possibility of deciding what to drink with both machines.

Testing equivalence has been proposed [4] (see also [5]) as an alternative to bisimilarity; it takes to the extreme the claim that when defining behavioral equivalences, one does not want to distinguish between systems that cannot be taken apart by external observers and bases the definition of the equivalences on the notions of *observers*, *observations* and *successful observations*. Equivalences are defined that consider equivalent those systems that satisfy (lead to successful observations by) the same sets of observers. An *observer* is an LTS with actions in $A_{\tau,w} \triangleq A_\tau \cup \{w\}$, with $w \notin A$. To determine whether a state q satisfies an observer with initial state o, the set $OBS(q, o)$ of all *computations* from $\langle q, o \rangle$ is considered.

Given an LTS $\langle Q, A_\tau, \xrightarrow{\mu} \rangle$ and an observer $\langle O, A_{\tau,w}, \xrightarrow{\mu} \rangle$, and a state $q \in Q$ and the initial state $o \in O$, an *observation* c from $\langle q, o \rangle$ is a maximal sequence of pairs $\langle q_i, o_i \rangle$, such that $\langle q_0, o_0 \rangle = \langle q, o \rangle$. The transition $\langle q_i, o_i \rangle \xrightarrow{\mu} \langle q_{i+1}, o_{i+1} \rangle$ can be proved using the following inference rule:

$$\frac{E \xrightarrow{\mu} E' \qquad F \xrightarrow{\mu} F'}{\langle E, F\rangle \xrightarrow{\mu} \langle E', F'\rangle}\ \mu \in A_\tau$$

An observation from $\langle q, o\rangle$ is *successful* if it contains a configuration $\langle q_n, o_n\rangle \in c$, with $n \geq 0$, such that $o_n \xrightarrow{w} o$ for some o.

When analyzing the outcome of observations, one has to take into account that, due to non-determinism, a process satisfies an observer **sometimes** or a process satisfies an observer **always**. This leads to the following definitions:

1. q MAY SATISFY o if *there exists an observation* from $\langle q, o\rangle$ that is successful;
2. q MUST SATISFY o if *all observations* from $\langle q, o\rangle$ are successful.

These notions can be used to define *may*, *must* and *testing* equivalence.

May equivalence p is *may* equivalent to q ($p \simeq_m q$) if, for all possible observers o:
p MAY SATISFY o if and only if q MAY SATISFY o;

Must equivalence p is *must* equivalent to q ($p \simeq_M q$) if, for all possible observers o:
p MUST SATISFY o if and only if q MUST SATISFY o.

Testing equivalence p is *testing* equivalent to q ($p \simeq_{test} q$) if $p \simeq_m q$ and $p \simeq_M q$.

The three vending machines of Fig. 1 are may equivalent, but only the two rightmost ones are must equivalent and testing equivalent. Indeed, in most cases must equivalence implies may equivalence and thus in most cases must and testing do coincide. The two leftmost machines are not must equivalent because one after receiving the two coins the machine cannot refuse to (must) deliver the drink chosen by the customer while the other can.

May and must equivalences have nice alternative characterizations. It has been shown that may equivalence coincides with traces equivalence and that must equivalence coincides with *failures equivalence*, another well-studied relation that is inspired by traces equivalence but takes into account the possible interactions (*failures*) after each trace and is thus more discriminative than trace equivalence [7]. Failures equivalence relies on pairs of the form $\langle s, F\rangle$, where s is a trace and F is a set of labels. Intuitively, $\langle s, F\rangle$ is a failure for a process if it can perform the sequence of actions s to evolve into a state from which no action in F is possible. This equivalence can be formulated on LTS as follows:

Failures Equivalence Two states p and q are *failures-equivalent,* written $p \simeq_F q$, if and only if they possess the same failures, i.e., if for any $s \in A_\tau^*$ and for any $F \subseteq A_\tau$:

1. $p \xrightarrow{s} p'$ and $Init(p') \cap F = \emptyset$ implies $q \xrightarrow{s} q'$ for some q' and $Init(q') \cap F = \emptyset$;
2. $q \xrightarrow{s} q'$ and $Init(q') \cap F = \emptyset$ implies $p \xrightarrow{s} p'$ for some p' and $Init(p') \cap F = \emptyset$.

where $Init(q)$ represents the immediate actions of state q.

5 Hierarchy of Equivalences

The equivalences considered above can be precisely related (see [3] for a first study). Their relationships over the class of finite transition systems with only visible actions are summarized by the figure below, where the upward arrow indicates containment of the induced relations over states and $\equiv$ indicates coincidence.

$$\begin{tikzcd}
\sim \\
\simeq_F \equiv \simeq_M \arrow[u] \\
\simeq_T \equiv \simeq_m \arrow[u]
\end{tikzcd}$$

Overall, the figure states that may testing gives rise to a relation ($\simeq_m$) that coincides with traces equivalence, while must testing gives rise to a relation $\simeq_M$ that coincides with failures equivalence. For the considered class of systems it also holds that must and testing equivalence $\simeq_{test}$ do coincide. Thus, bisimilarity implies testing equivalence that in turn implies traces equivalence.

6 Weak Variants of the Equivalences

When considering abstract versions of systems making use of invisible actions it turns out that all equivalences considered above are too discriminating. Indeed, traces, testing/failures and observation equivalence would distinguish the two machines of Fig. 2 that, nevertheless, exhibit similar observable behaviors: get a coin and deliver a coffee. The second one can be obtained, e.g., from the term

$$coin.grinding.\overline{coffee}.nil$$

by hiding the *grinding* action that is irrelevant for the customer.

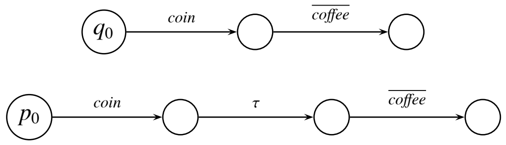


Fig. 2. Weakly equivalent vending machines

Because of this overdiscrimination, *weak* variants of the equivalences have been defined that permit ignoring (to different extents) internal actions when considering the behavior of systems. The key step of their definition is the introduction of a new transition relation that ignores silent actions. Thus, $q \overset{a}{\Rightarrow} q'$ denotes that q reduces to q' by performing the visible action a possibly preceded and followed by any number (also 0) of invisible actions (τ). The transition $q \overset{s}{\Rightarrow} q'$, instead, denotes that q reduces to q' by performing the sequence s of visible actions, each of which can be preceded and followed by τ-actions, while $\overset{\epsilon}{\Rightarrow}$ indicates that only τ-actions, possibly none, are performed.

Weak Traces Equivalence The weak variant of traces equivalence is obtained by simply replacing the transitions $p \overset{s}{\rightarrow} p'$ above with the observable transitions $p \overset{s}{\Rightarrow} p'$.

Two states p and q are *weak traces equivalent* ($p \approxeq_T q$) if for any $s \in A^*$:

1. $p \overset{s}{\Rightarrow} p'$ implies $q \overset{s}{\Rightarrow} q'$ for some q' and
2. $q \overset{s}{\Rightarrow} q'$ implies $p \overset{s}{\Rightarrow} p'$ for some p'.

Weak Testing Equivalence To define the weak variants of may, must and testing equivalences (denoted by $\approxeq_m, \approxeq_M, \approxeq_{test}$ respectively) it suffices to change experiments so that processes and observers can freely perform silent actions. To this purpose, one only needs to change the inference rule of the observation step: $\langle q_i, o_i \rangle \overset{\mu}{\rightarrow} \langle q_{i+1}, o_{i+1} \rangle$ that can now be proved using:

$$\frac{E \xrightarrow{\tau} E'}{\langle E,F\rangle \xrightarrow{\tau} \langle E',F\rangle} \qquad \frac{F \xrightarrow{\tau} F'}{\langle E,F\rangle \xrightarrow{\tau} \langle E,F'\rangle} \qquad \frac{E \xrightarrow{a} E' \quad F \xrightarrow{a} F'}{\langle E,F\rangle \xrightarrow{a} \langle E',F'\rangle}\ a \in A$$

To define, instead, *weak failures equivalence*, it suffices to replace $p \xrightarrow{s} p'$ with $p \overset{s}{\Rightarrow} p'$ in the definition of its strong variant. It holds that weak traces equivalence coincides with weak may equivalence, and that weak failures equivalence $\approxeq_F$ coincides with weak must equivalence.

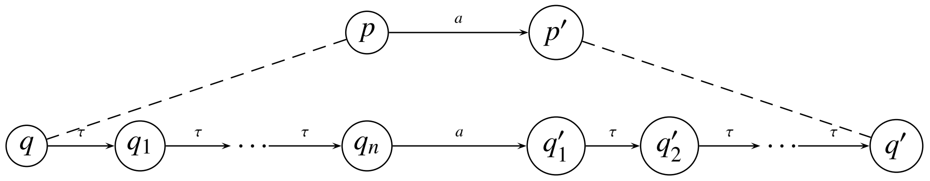
Weak Bisimulation Equivalence For defining weak observational equivalence a new notion of (weak) bisimulation is defined that again assigns a special role to τ's. To avoid having four items, the definition below requires that the candidate bisimulation relations be symmetric:

A **symmetric** relation $R \subseteq Q \times Q$ is a *weak bisimulation* if, for any pair of states p and q such that $\langle p,q\rangle \in R$, the following holds:

- for all $a \in A$ and $p' \in Q$, if $p \xrightarrow{a} p'$ then $q \overset{a}{\Rightarrow} q'$ for some $q' \in Q$ s. t. $\langle p',q'\rangle \in R$;
- for all $p' \in Q$, if $p \xrightarrow{\tau} p'$ then $q \overset{\epsilon}{\Rightarrow} q'$ for some $q' \in Q$ s.t. $\langle p',q'\rangle \in R$.

Two states p, q are *weakly bisimilar* ($p \approx q$) if there exists a weak bisimulation R such that $\langle p,q\rangle \in R$.

The figure below describes the intuition behind weak bisimilarity. In order to consider two states, say p and q, equivalent, it is necessary that for each visible action performed by one of them the other has to have the possibility of performing the same visible action possibly preceded and followed by any number of invisible actions.



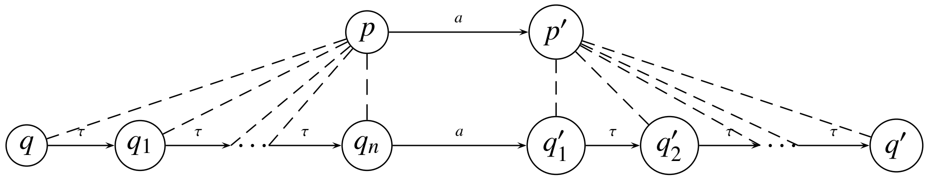
Branching Bisimulation Equivalence An alternative to weak bisimulation has also been proposed that considers those τ-actions important that appear in branching points of systems descriptions: only silent actions that do not eliminate possible interaction with external observers are ignored.

A **symmetric** relation $R \subseteq Q \times Q$ is a *branching bisimulation* if, for any pair of states p and q such that $\langle p,q\rangle \in R$, if $p \xrightarrow{\mu} p'$, with $\mu \in A_\tau$ and $p' \in Q$, at least one of the following conditions holds:

- $\mu = \tau$ and $\langle p',q\rangle \in R$
- $q \overset{\epsilon}{\Rightarrow} q'' \xrightarrow{\mu} q'$ for some $q', q'' \in Q$ such that $\langle p,q''\rangle \in R$ and $\langle p',q'\rangle \in R$.

Two states p, q are *branching bisimilar* ($p \approx_b q$) if there exists a branching bisimulation R such that $\langle p,q\rangle \in R$.

The figure below describes the intuition behind branching bisimilarity; it corresponds to the definition above although it might appear, at first glance, more demanding. In order to consider two states, say p and q, equivalent, it is necessary, like for weak bisimilarity, that for each visible action performed by one of them the other has to have the possibility of performing the same visible action possibly preceded and followed by any number of invisible actions. Branching bisimilarity, however, imposes the additional requirement that all performed internal actions are not used to change equivalent class. Thus, all states reached via τ's before performing the visible action are required to be equivalent to p, while all states reached via τ's after performing the visible action are required to be equivalent to q.



7 Hierarchy of Weak Equivalences

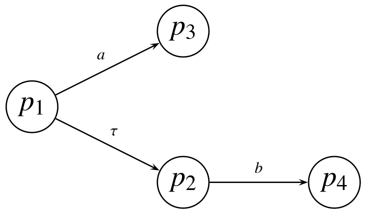
Like for the strong case, also weak equivalences can be clearly related. Their relationships over the class of finite transition systems with invisible actions but without τ-loops (so called *non-divergent* or *strongly convergent* LTSs) are summarized by the figure below, where the upward arrow indicates containment of the induced relations over states.

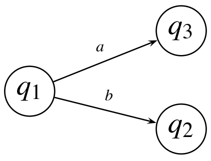
$$\begin{tikzcd}
\approx_b \\
\approx \arrow[u] \\
\approxeq_F \equiv \approxeq_M \arrow[u] \\
\approxeq_T \equiv \approxeq_m \arrow[u]
\end{tikzcd}$$

Thus, over *strongly convergent* LTSs with silent actions, branching bisimilarity imlies weak bisimilarity and this implies testing and failures equivalences; and these implies traces equivalence.

A number of counterexamples can be provided to show that the implications of the figure above are proper and thus that the converse does not hold.

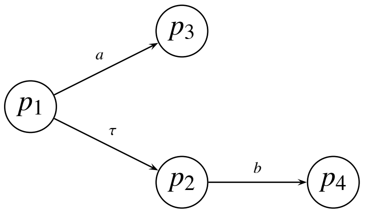
The two LTSs reported below are weak traces equivalent and weakly may equivalent but are distinguished by all the other equivalences.

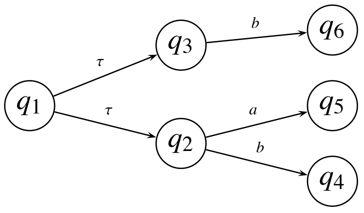




Indeed, they can perform exactly the same weak traces but while the former can silently reach a state in which an a-action can be refused the second cannot.

The next two LTSs are equated by weak trace and weak must equivalences but are distinguished by weak bisimulation and branching bisimulation.





Both of them, after zero or more silent actions, can be either in a state where both actions a and b are possible or in a state in which only a b transition is possible. However, via a τ-action, the rightmost system can reach a state that has no equivalent one in the leftmost one, thus they are not weakly bisimilar.

The next two LTSs are instead equated by weak bisimilarity, and thus by weak trace and and weak must equivalences, but are not branching bisimilar.

$$\begin{tikzcd}
 & & p_2 & \\
p_0 \arrow[r, "a"] & p_1 \arrow[ur, "c"] \arrow[dr, "\tau"] & & \\
 & & p_3 \arrow[r, "b"] & p_4
\end{tikzcd}
\qquad
\begin{tikzcd}
 & & q_6 & \\
 & q_5 \arrow[ur, "b"] & & \\
q_0 \arrow[ur, "a"] \arrow[dr, "a"] & & q_2 & \\
 & q_1 \arrow[ur, "c"] \arrow[dr, "\tau"] & & \\
 & & q_3 \arrow[r, "b"] & q_4
\end{tikzcd}$$

It is easy to see that from the states p_0 and q_0 the same visible action is possible and bisimilar states can be reached. The two states p_0 and q_0 are instead not branching bisimilar because p_0, in order to match the a action of q_0 to q_5 and reach a state equivalent to q_5, needs to reach p_3 through p_1 but these two states, connected by a τ-action, are not branching bisimilar.

It is worth concluding that the two LTSs of Fig. 2 are equated by all the considered weak equivalences.

8 Future Directions

The study on behavioral equivalences of transition systems is still continuing. LTSs are increasingly used as the basis for specifying and proving properties of reactive systems. For example they are used in *model checking* as the model against which logical properties are checked. It is then important to be able to use minimal systems that are, nevertheless, equivalent to the original larger ones so that preservation of the checked properties is guaranteed. Thus, further research is expected on devising efficient algorithms for equivalence checking and on understanding more precisely the properties of systems that are preserved by the different equivalences.

9 Bibliographic Notes and Further Reading

The theories of equivalences can be found in a number of books targeted to describing the different process algebras. The theory of bisimulation is introduced in [8] while failure and trace semantics are considered in [7] and [9]. The testing approach is presented in [5].

Moreover, interesting papers relating the different approaches are [3], the first paper to establish precise relationships between the many equivalences proposed in the literature, and the two papers by R. van Glabbeek: [11] , considering systems with only visible actions, and [12], considering also systems with invisible actions. In his two companion papers R. van Glabbeek provides a uniform, model-independent account of many of the equivalences proposed in the literature and proposes several motivating testing scenarios, phrased in terms of “button pushing experiments” on reactive machines to capture them.

Bisimulation and its relationships with modal logics is deeply studied in [6], while a deep study of its origins and its use in other areas of computer science is provided in [10]. Branching bisimulation was first introduced in [1], while the testing based equivalences were introduced in [4]. Failure semantic was first introduced in [2].

References

1. J.C.M. Baeten and W.P. Weijland. *Process Algebra.* Cambridge University Press, 1990.
2. S. D. Brookes, C. A. R. Hoare, and A. W. Roscoe. A theory of communicating sequential processes. *J. ACM*, 31(3):560–599, 1984.

3. R. De Nicola. Extensional equivalences for transition systems. *Acta Informatica*, 24(2):211–237, 1987.
4. R. De Nicola and Matthew Hennessy. Testing equivalences for processes. *Theor. Comput. Sci.*, 34:83–133, 1984.
5. M. Hennessy. *Algebraic theory of Processes*. The MIT Press, 1988.
6. M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *J. ACM*, 32(1):137–161, 1985.
7. C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, Inc., 1985.
8. R. Milner. *Communication and concurrency.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989.
9. A.W. Roscoe. *The Theory and Practice of Concurrency*. Prentice-Hall, Inc., 1998.
10. D. Sangiorgi. On the origins of bisimulation and coinduction. *ACM Trans. Program. Lang. Syst.*, 31(4), 2009.
11. R.J. van Glabbeek. The linear time - branching time spectrum I: the semantics of concrete, sequential processes. In *Handbook of Process Algebra (J.A. Bergstra, A. Ponse and S.A. Smolka, eds.)*, pages 3–99. Elsevier, 2001.
12. R.J. van Glabbeek. The linear time - branching time spectrum II. In *CONCUR '93, 4th International Conference on Concurrency Theory, (E. Best ed.)*, volume 715, pages 66–81. Springer–Verlag, 1993.