Mathematical Structures in Computer Science, 20(3):319–358. © *C.U.P. 2010.*

Tree-Functors, Determinacy and Bisimulations

Rocco De Nicola[1], Daniele Gorla[2] and Anna Labella[2]

[1] *Dip. di Sistemi ed Informatica – Univ. di Firenze. Viale Morgagni 65, 50134 Firenze (IT).*
email: `rocco.denicola@unifi.it`

[2] *Dip. di Informatica – Univ. di Roma "La Sapienza". Via Salaria 113, 00198 Roma (IT).*
email: {`gorla,labella`}`@di.uniroma1.it`

We study the functorial characterization of bisimulation-based equivalences over a categorical model of labeled trees. We show that, in a setting where all labels are visible, strong bisimilarity can be characterized in terms of enriched functors by relying on *reflection of paths with their factorizations*. For an enriched functor F, this notion requires that a path (an internal morphism in our framework) π going from $F(A)$ to C corresponds to a path p going from A to K, with $F(K) = C$, such that every possible factorization of π can be lifted in an appropriate factorization of p. This last property corresponds to a *Conduché property* for enriched functors and a very rigid formulation of it has been used by Lawvere to characterize determinacy of physical systems. We also consider the setting where some labels are not visible and provide characterizations for weak and branching bisimilarity. Both equivalences are still characterized in terms of enriched functors that reflect paths with their factorizations: for branching bisimilarity, the property is the same as the one that characterized strong bisimilarity when all labels are visible; for weak bisimilarity, a weaker form of path factorization lifting is needed. This fact can be seen as an evidence that strong and branching bisimilarity are strictly related and that, differently from weak bisimilarity, they preserve process determinacy in the sense of Milner.

1. Introduction

In concurrency theory, behavioural equivalences have been used as abstraction mechanisms to equate systems that exhibit the same behaviour. Different notions of equivalence have been developed in the years; in particular, bisimulation equivalence or *bisimilarity* (Park 1981) has turned out to be a key concept in many areas of computer science. In practice, such an equivalence equates systems that offer the same interaction possibilities with the external world at every stage of their computation.

As an example, consider the two processes of Figure 1; we represent them as labeled trees, where paths represent computations and bifurcations represents nondeterministic choice between different evolutions. They are *not* equivalent, even though they exhibit the same traces, because, after the initial a, the rightmost tree can offer both b and c, while the leftmost one cannot.

In process algebra, *strong bisimilarity* (Milner 1989) is generally considered as the behavioural equivalence which provides the minimum abstraction level from the operational semantics of processes. Indeed, it is generally considered as "the" natural equivalence, when all process actions

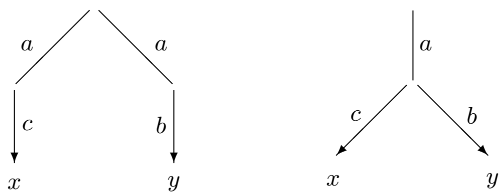


Fig. 1. Two Trees

are visible, also because it preserves what in (Milner 1989) is called *strong determinacy*, a notion deeply related to predictability of behaviours. An equivalence enjoying strong determinacy guarantees that the computations of two equated processes, at any point of their progress, have a direct 'back and forth' correspondence. Within our tree-based model, this amounts to requiring that all paths and their factorizations are preserved and reflected by the mapping associating equivalent processes via their computations (paths). By path factorization we mean a splitting of the path in two sub-paths whose concatenation yields the original path.

If one considers the two trees of Figure 1 and defines the (unique, in this case) mapping that associates paths with the same labeling, then path factorizations *are* preserved and reflected but, in doing it, the mapping does *not* associate equivalent processes. Indeed, after the initial a, the possible evolutions of the leftmost tree of Figure 1 are those of Figure 2, whereas the tree obtained from the rightmost tree of Figure 1 is the one reported in Figure 3. Of course, the three trees do not offer the same behaviour.

Fig. 2. The derivatives after a of the leftmost tree in Figure 1

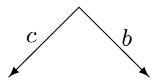


Fig. 3. The derivative after a of the rightmost tree in Figure 1

Indeed, there is no way to associate the paths of the two trees in Figure 1 in such a way that path factorizations are properly reflected.

Strong bisimilarity can be too discriminating, mainly because it keeps into account every intermediate state, even those reachable via invisible actions. For this reason, when considering systems with silent (usually called τ) actions, different variants have been proposed in the literature. The two main ones are known as *branching bisimilarity* (van Glabbeek and Weijland 1989) and *weak bisimilarity* (Milner 1989) and there are still discussions about which of them is the most appropriate. In this paper, we will study a new kind of characterization of these equivalences in terms of preservation/reflection of path factorizations and see that, in the presence of τ-actions, the reflected path factorizations are not necessarily unique.

Let us consider the pair of trees in Figure 4 and the function mapping path x into path x' and paths y_1 and y_2 into y'. We have that *all* the factorizations of x' and y' have an appropriate corresponding factorization in the corresponding source paths of the leftmost tree; in particular, the factorization of y' as a and b can be reflected both in y_1 (as $a.\tau$ and b) and in y_2 (as a and b); moreover, the corresponding intermediate states do offer the same visible behaviour.

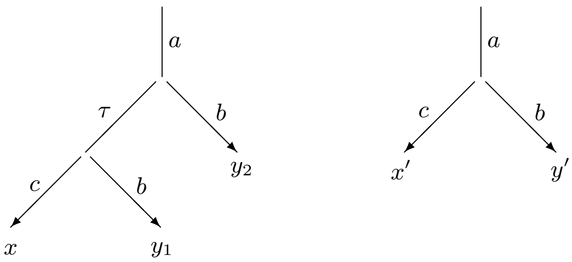


Fig. 4. Two Branching Bisimilar Trees

It can however happen that *not all* the possible factorizations of a path can be reflected in an appropriate factorization of a path mapped to it by the associating function. Let us consider the trees in Figure 5 and the function mapping y into y' and both x_1 and x_2 into x'. The factorization of x' as a and $\tau.c$ can be appropriately reflected on x_1, but not on x_2, since x_2 after a does not offer any b. Only the factorization of x' as $a.\tau$ and c can be reflected on x_2.

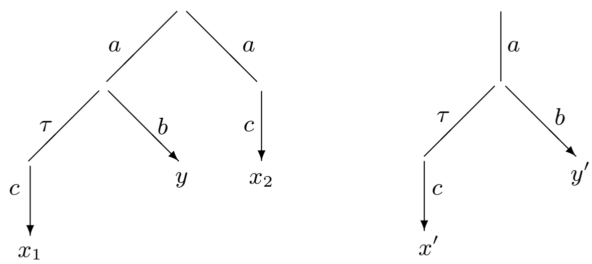


Fig. 5. Two Weakly Bisimilar Trees

We show that the mapping functions like the one used for Figure 4 characterize branching bisimulation, whereas the weaker factorized path reflection property illustrated by means of Figure 5 captures weak bisimulation. To this aim, we work in a categorical framework and characterize strong, branching and weak bisimulation equivalences through proper functorial definitions capturing the intuitions underlying the examples above. Our aim is to use category theory and its abstract constructions as a tool for understanding and assessing the relative merits of different concepts, and to better appreciate their meaning and their inter-relationships. In particular, a clear evidence of the quality of an equivalence is the *functoriality* of the map it is induced by: this is because any functor has the property of faithfully respecting the morphism structure of the category it is applied to.

We take as our starting point the tree-based categorical model of concurrency by (Kasangian and Labella 1999), which adapts the categorical modeling of finite (or infinite) state automata

(Arbib and Manes 1974; Betti and Kasangian 1982). Such models are based on a "local" approach in that an automaton is described via the languages (the sets of words on a given alphabet A) determining the transitions between any pair of states, i.e.

$$L_{s,s'} = \{w \in A^* \ : \ \delta(s,w) = s'\}.$$

Initial and terminal states can be specified when necessary. In that case, the set of words leading from the initial state to the terminal ones describes the global behaviour of the automaton†. The usual equivalence used for automata is a global one, taking into account the extensional behaviour only; computations of different automata are associated if and only if they correspond to the same sequence of actions (thus they have the same label and no path factorization is considered). The equivalence induced by this kind of associations is usually called *trace equivalence* by concurrency theorists and ignores the traversed states.

When moving from automata to nondeterministic processes, greater attention must be devoted to the notion of state, because one has to consider the possible alternative steps that are available at intermediate states. In this case,"sets of words" are no more sufficient to represent nondeterministic behaviours. For this reason, computations of Labeled Transition Systems (LTS), one of the most popular operational model of concurrency, can be better modeled via a tree-shaped labeled structure, with internal nodes representing possible choices during the evolutions; equivalences for LTS do not rely on terminal states and have a local formulation.

Our basic category is Tree, the category whose objects are trees‡ and whose morphisms can be seen as simulations between them, i.e. functions such that each "state" of the source tree is mapped into a "state" of the target tree able to simulate it. Trees are described as a set of labeled paths suitably glued together, where the gluing is sufficient to give a tree structure to a language. An important aspect of our model is that transitions between states can be rendered again as trees with paths of finite length; more technically, since Tree is a (left-closed) monoidal category, we can consider a category **Beh** that has the same objects as Tree and it is enriched over it (thus, its arrow-objects are trees as well). In this way, objects of **Beh** describe states (processes) and arrow-objects describe the evolution (operational semantics) of our models. We then provide a Tree-functorial assessment of strong bisimulation equivalence, where Tree-functoriality means that the internal morphism structure (i.e., its operational semantics) is respected.

Similarly, we can consider Tree_τ and $\mathbf{Beh}_{\tau,\tau}$, i.e. the category and the Tree_τ-category obtained like Tree and **Beh** but with a special meaning assigned to τ-actions. The category $\mathbf{Beh}_{\tau,\tau}$ has τ-labels both on its objects and on its arrow-objects but, through a change of base that eliminates τ-labels from the arrow-objects, we can consider $\mathbf{Beh}_{\tau,\tau}$ also as a Tree-category. We shall call $\mathbf{Beh}_\tau$ this category, where τ's have become invisible, and study the Tree-functors over it that characterize weak and branching equivalences.

We will provide functorial characterizations of the different bisimulation-based equivalences

† Technically, A-automata are modeled as categories enriched on the base 2-category (Walters 1981) provided by the structure of the languages, $\wp(A^*)$, and comparisons between automata are defined in terms of "change of base" (Kelly 1982), i.e. in terms of the local structure.

‡ Formally, Tree is the category of the categories enriched over a locally posetal 2-category determined by a fixed alphabet with the corresponding functors: see Remark 1.

by identifying the properties that must be enjoyed by the *Tree*-functor inducing them. In particular:

1 *Strong bisimilarity* will be characterized in **Beh** as the equivalence preserved by any *Tree*-functor F that *reflects paths*. This property requires that any path (internal morphism) from $F(\mathcal{X})$ to $\mathcal{Y}$ arises from at least one path going from $\mathcal{X}$ to $\mathcal{X}'$, with $F(\mathcal{X}') = \mathcal{Y}$. Because of the structure of **Beh**, it will also result that, if a path x' of the target tree is factorized as $x'_1; x'_2$, then *every* source path x mapped into x' can be factorized as $x_1; x_2$ in such a way that x_i is mapped into x'_i. Hence our *Tree*-functor will also *reflect path factorizations*. Existence of such a functor is provided by the definition of a standard strong representative for every tree, obtained by quotienting the set of paths.

2 *Branching bisimilarity* is the equivalence preserved in **Beh**$_\tau$ by *Tree*-functors that reflect paths with their factorizations. Notice that, while in **Beh** such a property is equivalent to a simpler one (reflection of paths), the presence of an invisible action in **Beh**$_\tau$ makes the requirement on the equivalence of all the intermediate states (reachable via invisible actions) of the paths unavoidable. Again, we show existence of such a functor by introducing a standard branching representative for every tree, in strict analogy with the strong case.

3 *Weak bisimilarity* is the equivalence preserved by *Tree*-functors that enjoy only a weaker form of factorized path reflection. This weaker property is similar to the one for strong/branching bisimulation, but only requires that to each factorization of a target path x' there corresponds a factorization of *one of* the source paths; in particular, it does not require that *each* source path does faithfully mirror all factorizations of their image. Thus, if a path x' of the target tree is factorized as $x'_1; x'_2$, it is might be that we have two source paths x and y mapped into x' with only x that can be factorized as $x_1; x_2$ in such a way that x_i is mapped into x'_i. This amounts to requiring that for each target state there is at least an equivalent source state one has gone through. The standard representative, that we use to guarantee existence of the required functor, is obtained from the one for branching equivalence by an appropriate pruning.

Reflecting factorizations has a particular meaning; it coincides with the enriched form of a necessary and sufficient condition that we call *Conduché property*, after (Conduché 1972) that a functor has to satisfy for guaranteeing that its inverse image has a right adjoint. It has been proved (Kasangian and Labella 2009) that the natural translation of Conduché property for *Tree*-functors still guarantees the existence of a right adjoint for the inverse image. If we take the standpoint of (Lawvere 1986) and (Bunge and Fiore 2000), that used a strong form of Conduché property for establishing system determinacy, we have a further evidence of the fact that weak bisimilarity does not preserve process determinacy.

The advantage of our framework is that it leads to characterizations that are parametric w.r.t. the way τ's are considered (either as visible or as invisible); moreover, it allows us to express behavioural equivalences in terms of *Tree*-functors enjoying specific properties that determine (depending on the equivalence under consideration) the kind of correspondence between the operational semantics of the equated systems. For this reason, we think that our results provide a new insight on the difference between branching and weak bisimilarity, and on the extent to which they abstract from silent actions, in terms of preservation of states/determinacy. Moreover, our approach can be easily scaled to more sophisticated models of concurrency than LTS, like, e.g., *Labeled Event Structures* that have been used to model causal (in-)dependencies of systems actions (Winskel 1988).

The rest of the paper is organized as follows. In Section 2, we present the tree model of (Kasangian and Labella 1999), define the notion of strong bisimulation between trees and prove that it can be characterized as the equivalence induced by Tree-functors that reflect path factorizations. In Section 3, we allow to ignore τ-actions, define the notion of branching and weak bisimulations, and prove that they are the equivalences induced by Tree-functors that, besides reflecting paths, reflect and weakly reflect path factorizations, respectively. In Section 4, we prove the converse, i.e. that strong, branching and weak bisimilarity imply existence of Tree-functors that, besides reflecting paths, reflect (or weakly reflect, in the weak case) path factorizations. In Section 5 we show how our treatment can be instantiated in the cases of labeled transition systems and labeled (prime) event structures. Section 6 contains a detailed comparison with related work, while in Section 7 we sum up and discuss our contribution. We assume that the reader is familiar with basic *Category Theory*, and in particular with the notions of *tensor product*, *monoidal category* and *monoidal functor*. The necessary background can be found in (Kelly 1982).

2. The Tree Model for Nondeterministic Processes

2.1. *Basic Definitions*

We now recall from (Kasangian and Labella 1999) a category of labeled trees, Tree, and some of its properties. A single tree is modeled by specifying its paths, the computations along each of them (the *extent*) and the part of the extent in which pairs of computations agree (the *agreement*). Path labels are elements of a free monoid A^*, for some (finite) alphabet A.

Example 1. Consider the trees in Figure 1 of Section 1: they both have two paths, x and y, that are labeled with ab and ac respectively; in the leftmost tree, x and y do not agree at all (i.e., their agreement is ϵ); in the rightmost tree, x and y agree by the initial a. ◇

For mathematical reasons (see Remark 1 below), it is convenient to make it explicit the order structure that is instead implicit in the free monoid $(A^*, \bullet, \epsilon)$ generated by A (where $\bullet$ denotes word concatenation and ϵ is the empty word). Indeed, $(A^*, \bullet, \epsilon)$ satisfies the following properties:

1. it is equipped with a partial order defined by $s \leq t$ iff there exists $u \in A^*$ such that $s \bullet u = t$ (this is the usual notion of prefix of a word);
2. it has meets, $\wedge$, given by the maximum common prefix, with ϵ as bottom element;
3. it has a join, $\vee$, for every family that has an upper bound;
4. it enjoys the *left-cancellation property*, i.e., for every $s, t, u \in A^*$ such that $s \bullet t = s \bullet u$, it holds that $t = u$.

The resulting structure will be called the *meet-semilattice monoid associated with* $(A^*, \bullet, \epsilon)$ and denoted by $\mathbf{A}$.

Definition 1. An $\mathbf{A}$-tree $\mathcal{X}$ is a triple (X, e_X, a_X) where X is the set of *paths*, $e_X : X \to A^*$ is the *extent map* and $a_X : X \times X \to A^*$ is the *agreement* between paths such that, for every $x, y, z \in X$, it holds that:

1. $a_X(x, x) = e_X(x)$

2 $a_X(x,y) \leq e_X(x) \wedge e_X(y)$
3 $a_X(x,y) \wedge a_X(y,z) \leq a_X(x,z)$
4 $a_X(x,y) = a_X(y,x)$

Property (1) requires that a path agrees with itself along all its length. Property (2) states that the agreement between two paths cannot be bigger than their largest common prefix (paths are forced to agree on a common initial segment and they cannot re-join once split). Property (3) states that the common agreement between x, y and z cannot be bigger than the common agreement between x and z. Finally, Property (4) states that the agreement is symmetric.

Example 2. The two trees illustrated in Figure 1 are formally specified by $\mathcal{X} = (X, e_X, a_X)$ and $\mathcal{Y} = (Y, e_Y, a_Y)$, where $X = Y = \{x, y\}$, $e_X(x) = e_Y(x) = ab$, $e_X(y) = e_Y(y) = ac$, $a_X(x,y) = \epsilon$ and $a_Y(x,y) = a$. Thus, they have the same paths and extent but a different agreement. $\diamond$

Remark 1. We can observe that $\mathbf{A}$-trees are symmetric $\mathbf{A}$-categories, or categories enriched over $\mathbf{A}$, when $\mathbf{A}$ is thought of as the locally posetal 2-category associated with $\mathbf{A}$ (Walters 1981). Therefore, the appropriate notion of comparison for trees is given in Definition 2 and coincides with the notion of $\mathbf{A}$-functor.

Definition 2. A *Tree-morphism* $f : \mathcal{X} \to \mathcal{Y}$ is a map $f : X \to Y$ such that $e_Y(f(x)) = e_X(x)$ and $a_Y(f(x), f(y)) \geq a_X(x,y)$.

The intuition behind the notion of *Tree*-morphisms is that the target tree (viz., $\mathcal{Y}$) simulates the source one (viz., $\mathcal{X}$) in the sense of Milner: a computation x simulates a computation y if they both have the same label (i.e., perform the same actions) and x passes through intermediate states that at least offer the same behaviours as the corresponding intermediate states of y.

Example 3. Consider again the trees in Figure 1. It is well known (Milner 1989) that the rightmost tree simulates the leftmost one, but not vice versa. Indeed, the identity mapping on the set of paths induces a *Tree*-morphism from left to right, whereas there exist no *Tree*-morphisms from right to left (the identity reduces the agreement, whereas the remaining three mappings do not respect path labeling).

We denote by $\mathit{Tree}_{\mathbf{A}}$ the category of $\mathbf{A}$-trees (to make notation lighter, we shall write *Tree* instead of $\mathit{Tree}_{\mathbf{A}}$, by leaving $\mathbf{A}$ implicit when clear from the context). *Tree* has (among other features):

— an initial object: the empty tree, $\mathbf{0} = (\emptyset, \emptyset, \emptyset)$;
— a terminal object: the monoid itself considered as a tree, $(A^*, id, \wedge)$;
— finite coproducts (i.e., sums): given two trees (X, e_X, a_X) and (Y, e_Y, a_Y), their sum is obtained by joining them at the root, i.e. it is the tree $(X \uplus Y, e_X \uplus e_Y, a_X \uplus a_Y \uplus \{(x, y, \epsilon) : x \in X \text{ and } y \in Y\})$, where '$\uplus$' denotes disjoint union.

Definition 3. Given two $\mathbf{A}$-trees $\mathcal{X}$ and $\mathcal{Y}$, we can form the *sequential composition of $\mathcal{X}$ and $\mathcal{Y}$*, $\mathcal{X} \otimes \mathcal{Y} = (Z, e_Z, a_Z)$, as follows:

— $Z = X \times Y$
— $e_Z(x,y) = e_X(x) \bullet e_Y(y)$

— $a_Z((x, y), (x', y'))$ is $a_X(x, x')$, if $x \neq x'$, and $e_X(x) \bullet a_Y(y, y')$, otherwise.

Intuitively, $\mathcal{X} \otimes \mathcal{Y}$ attaches a copy of $\mathcal{Y}$ rooted at every leaf of $\mathcal{X}$. Thus, a path in $\mathcal{X} \otimes \mathcal{Y}$ is a path in $\mathcal{X}$ followed by a path in $\mathcal{Y}$, and its label is the label in $\mathcal{X}$ followed by the label in $\mathcal{Y}$. Paths that are different in $\mathcal{X}$ inherit their $\mathcal{X}$ agreement, while paths that differ only in $\mathcal{Y}$ have their $\mathcal{X}$ agreement concatenated with their $\mathcal{Y}$ agreement. In the sequel, the path (x, y) in $\mathcal{X} \otimes \mathcal{Y}$ will be denoted by $x;y$.

Proposition 1. Sequential composition defines the object part of an associative tensor product on *Tree* with unit $\mathcal{I} = (\{\star\}, [\star \mapsto \epsilon], [(\star, \star) \mapsto \epsilon])$. *Tree* with sequential composition is a monoidal category.

Since *Tree* is a monoidal category, we can define *Tree*-categories and *Tree*-functors between them (Kelly 1982).

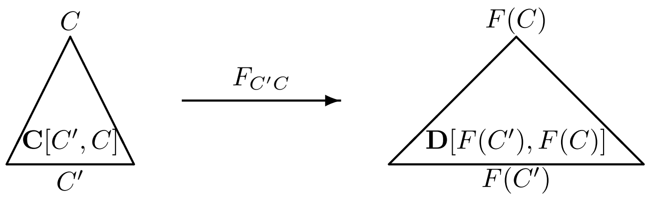
Definition 4. A *Tree*-category $\mathbf{C}$ is a set of objects such that, for every pair of objects (A, B), there is an object of *Tree*, called $\mathbf{C}[A, B]$, such that there are two *Tree*-morphisms

1 $\mathbf{C}[B, C] \otimes \mathbf{C}[A, B] \xrightarrow{m_{A\,B\,C}} \mathbf{C}[A, C]$ (*multiplication*)
2 $\mathcal{I} \xrightarrow{1_A} \mathbf{C}[A, A]$ (*identity*)

that make the diagrams in Figure 6 commute. There, $\alpha_{A\,B\,C\,D}$ is the associativity morphism of the tensor product, and $l_{\mathbf{C}[A,B]}$ and $r_{\mathbf{C}[A,B]}$ are, respectively, the left and the right composition with the tensor product identity.

Definition 5. A *Tree*-functor is a function $F : \mathbf{C} \to \mathbf{D}$ such that, for every pair (A, B) of objects in $\mathbf{C}$, there exists a *Tree*-morphism $F_{A\,B} : \mathbf{C}[A, B] \to \mathbf{D}[F(A), F(B)]$ that make the diagrams of Figure 7 commute.

Intuitively, a *Tree*-category is a class of objects $\mathbf{C}$ where we suitably put between any two of them, say C and C', a tree $\mathbf{C}[C', C]$. Moreover, a *Tree*-functor from $\mathbf{C}$ to $\mathbf{D}$ is a mapping from objects of $\mathbf{C}$ to objects of $\mathbf{D}$ that induces a *Tree*-morphism between the trees $\mathbf{C}[C', C]$ and $\mathbf{D}[F(C'), F(C)]$, for every C and C'. Pictorially:



Example 4. Consider the *Tree*-category $\mathbf{T}$ with only one object, say T. If we let $\mathbf{T}[T, T] = (A^*, id, \wedge)$, $\mathbf{T}$ will be the terminal *Tree*-category. Indeed, for every *Tree*-category $\mathbf{C}$ and for every C and C' objects of $\mathbf{C}$, there exists a unique *Tree*-morphism mapping $\mathbf{C}[C', C]$ into $\mathbf{T}[T, T]$: it suffices to associate both C and C' with T and every path of $\mathbf{C}[C', C]$ to the (unique) equally labeled path of $\mathbf{T}[T, T]$.

Tree-functors induce *Tree*-morphisms and hence they let us associate paths by taking into account their labels and their agreement. Our aim is to characterize functors by imposing further

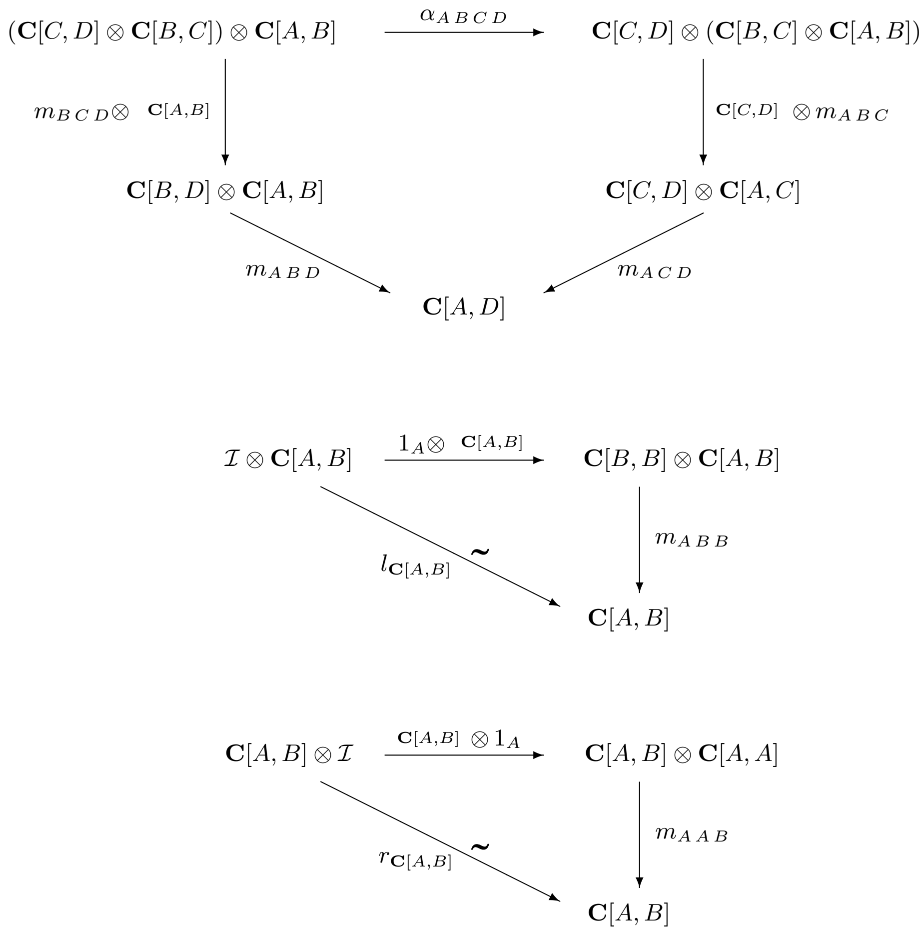


Fig. 6. The commutativity diagrams for Definition 4

conditions over their induced morphisms. For example, we want to keep into account also an 'inverse' correspondence between paths and the possibility of reflecting factorizations.

Definition 6. Let $F : \mathbf{C} \to \mathbf{D}$ be a *Tree*-functor.

1 F is *path reflecting* (PR, for short) iff for every $C \in \mathbf{C}$ and for every path $y \in \mathbf{D}[D, F(C)]$ there are $C' \in \mathbf{C}$ and $x \in \mathbf{C}[C', C]$ s.t. $F(C') = D$ and $F_{C'C}(x) = y$.

2 F is *factorized path reflecting* (FPR, for short) iff
   (a) it is PR, and
   (b) for every path $x \in \mathbf{C}[C', C]$ it holds that:
      $F_{C'C}(x) = y';y''$, with $y' \in \mathbf{D}[D', F(C)]$ and $y'' \in \mathbf{D}[D, D']$, implies that there exist $C'' \in \mathbf{C}$, $x' \in \mathbf{C}[C'', C]$ and $x'' \in \mathbf{C}[C', C'']$ such that $F_{C''C}(x') = y'$, $F_{C'C''}(x'') = y''$ and $F(C'') = D'$.

3 F is *factorized path weakly reflecting* (FPWR, for short) iff
   for every $y = y';y''$, with $y' \in \mathbf{D}[D', F(C)]$ and $y'' \in \mathbf{D}[D, D']$,
   there exist $C', C'' \in \mathbf{C}$, $x \in \mathbf{C}[C', C]$, $x' \in \mathbf{C}[C'', C]$ and $x'' \in \mathbf{C}[C', C'']$

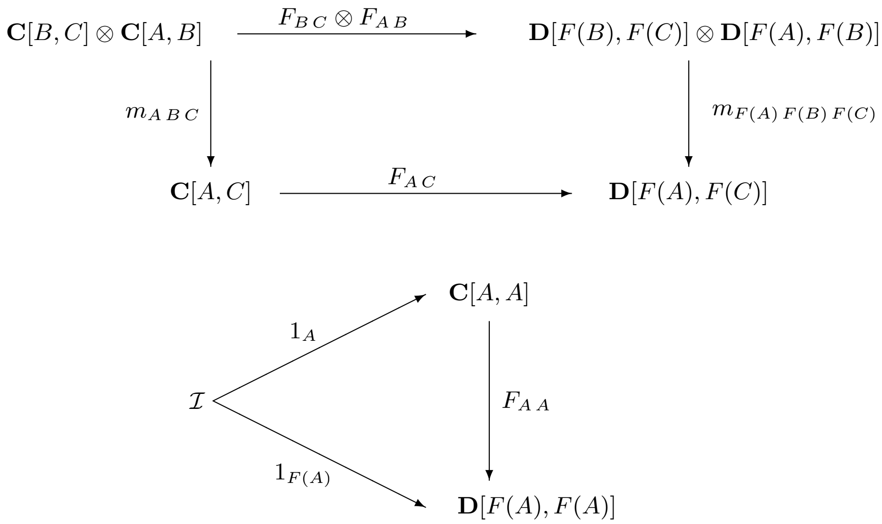


Fig. 7. The commutativity diagrams for Definition 5

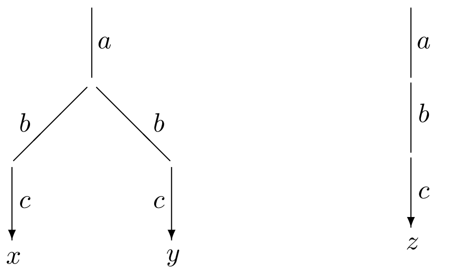
such that $F(C') = D$, $F_{C'C}(x) = y$, $F_{C''C}(x') = y'$, $F_{C'C''}(x'') = y''$ and $F(C'') = D'$.

Proposition 2. FPR implies FPWR, that implies PR.

Proof. Let $y \in \mathbf{D}[D, F(C)]$; by PR (that is holds because F is FPR) there exists a $C' \in \mathbf{C}$ and $x \in \mathbf{C}[C', C]$ such that $F(C') = D$ and $F_{C'C}(x) = y$. Then, by Definition 6(2b), every factorization of y can be reflected into x; thus, FPR implies FPWR.

Let us now fix $C \in \mathbf{C}$ and $y \in \mathbf{D}[D, F(C)]$; by FPWR, there exists a $x \in \mathbf{C}[C', C]$ such that $F_{C'C}(x) = y$ and $F(C') = D$; thus, FPWR implies PR. □

Example 5. Let $\mathbf{C}$ contain only C and C' and $\mathbf{D}$ contain only D and D'; moreover, let the two trees below



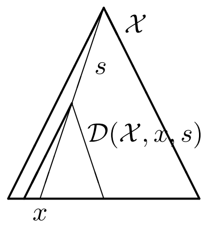
be $\mathbf{C}[C', C]$ and $\mathbf{D}[D', D]$, respectively. The *Tree*-functor that associates C with D and C' with D' is FPR, and hence it is PR and FPWR. In the next section (see, e.g., Proposition 4), we will see examples of functors that satisfy Definition 6(2b) but not PR.

Since in our model trees represent process behaviours, it is natural to define what happens to a tree after the execution of some of its actions. In analogy with language theory, we can define the derivative of $\mathcal{X}$ reached after word s along a given path x. To this aim, we denote with $t - s$ the word obtained from t by deleting the prefix s.

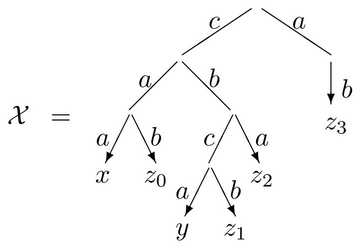
Definition 7. Let $\mathcal{X} = (X, e_X, a_X)$ be a tree. The *derivative* reached in $\mathcal{X}$ along the path x ($\in X$) after s ($\leq e_X(x)$), written $\mathcal{D}(\mathcal{X}, x, s)$, is the tree $(\{x\}, [x \mapsto \epsilon], [(x,x) \mapsto \epsilon])$ (that is isomorphic to $\mathcal{I}$ defined in Proposition 1), if $e_X(x) = s$, and otherwise it is the tree (Y, e_Y, a_Y) where

— $Y = \{x' \in X \ : \ a_X(x, x') \geq s\}$,
— $e_Y(x') = e_X(x') - s$,
— $a_Y(x', x'') = a_X(x', x'') - s$.

The following figure illustrates our terminology.

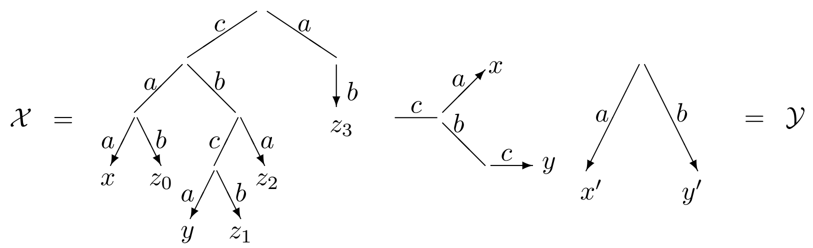


More concretely, by letting



we have that $\mathcal{D}(\mathcal{X}, x, caa) = \mathcal{I}$, $\mathcal{D}(\mathcal{X}, x, ca) = \mathcal{Y}_1$ and $\mathcal{D}(\mathcal{X}, y, cbc) = \mathcal{Y}_2$, where

Indeed, given $\mathcal{X}$ and $\mathcal{Y}$, we can consider all the paths leading from $\mathcal{X}$ to (a tree isomorphic to) $\mathcal{Y}$. The family of all such path prefixes is not just a set: it is tree shaped, as exemplified by the following:

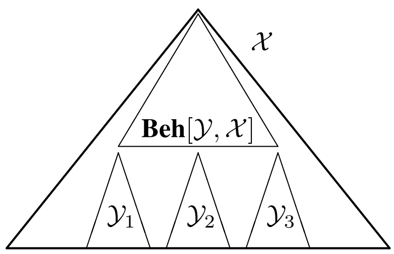


We can look at this tree as a structured set of morphisms from $\mathcal{Y}$ to $\mathcal{X}$ (the reversed notation is due to technical reasons) and consider the category **Beh** with the same objects as *Tree*, but with these new morphism-objects in place of simulations.

Definition 8. The category **Beh** is the *Tree*-category with the same objects as *Tree* and where, for any $\mathcal{X}$ and $\mathcal{Y}$, **Beh**$[\mathcal{Y}, \mathcal{X}]$ is the tree such that

— its set of paths is $\{|x|_{\mathcal{D}} \ : \ x \in X \ \text{ and } \ \mathcal{Y} \cong \mathcal{D}(\mathcal{X}, x, s)\}$, where $x \equiv_{\mathcal{D}} y$ iff $\mathcal{D}(\mathcal{X}, x, s) = \mathcal{D}(\mathcal{X}, y, s)$ and $|x|_{\mathcal{D}}$ denotes the equivalence class of x w.r.t. $\equiv_{\mathcal{D}}$;
— the extent of a path $|x|_{\mathcal{D}}$ is s, whenever $\mathcal{Y} \cong \mathcal{D}(\mathcal{X}, x, s)$;
— $a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x'|_{\mathcal{D}}) = e_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}) \wedge e_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x'|_{\mathcal{D}}) \wedge a_X(x, x')$;
— composition in **Beh** is given by the tensor product of *Tree*.

Intuitively, **Beh**$[\mathcal{Y}, \mathcal{X}]$ is the initial behaviour of $\mathcal{X}$ formed by all the computations leading to an isomorphic copy of $\mathcal{Y}$. Graphically:



where the $\mathcal{Y}_i$'s are all isomorphic to $\mathcal{Y}$. **Beh** is a *Tree*-category, exploiting monoidality of *Tree* (see Proposition 1)[§]. If $\mathcal{Y}$ is not isomorphic to $\mathcal{D}(\mathcal{X}, x, s)$ for some x and s, then **Beh**$[\mathcal{Y}, \mathcal{X}]$ is the initial tree $\mathbf{0} = (\emptyset, \emptyset, \emptyset)$. Moreover, **Beh** is well-defined, as proved in the following proposition.

Proposition 3. For every $\mathcal{X}$ and $\mathcal{Y}$ in *Tree*, it holds that **Beh**$[\mathcal{Y}, \mathcal{X}]$ is a tree.

Proof. The fact that the four properties of Definition 1 hold is trivial to check, once we have shown that the choice of the representative $|x|_D$, for every path $x \in X$, does not affect the definition of **Beh**$[\mathcal{Y}, \mathcal{X}]$. Let $x'' \in |x'|_D$; the only non-trivial thing to prove is that $a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x'|_{\mathcal{D}}) = a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x''|_{\mathcal{D}})$. Let us consider two cases:

[§] **Beh** is a *Tree*-subcategory of the canonical enrichment of the class of the objects of the category *Tree* over *Tree*. This is due to Theorem 4.1 in (Kasangian and Labella 1999) that proves that the left cancellation property enjoyed by the monoid A^* implies closedness of *Tree*. It is worth noting that we could also define *Tree*$[\mathcal{Y}, \mathcal{X}]$, but in this case it would be the initial behaviours of $\mathcal{X}$ formed by all the computations leading to a *homomorphic* (instead of isomorphic) copy of $\mathcal{Y}$.

— $a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x'|_{\mathcal{D}}) \leq e_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}) \wedge e_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x'|_{\mathcal{D}})$: by property 3 in Definition 1, $a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x'|_{\mathcal{D}}) \wedge a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x'|_{\mathcal{D}}, |x''|_{\mathcal{D}}) \leq a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x''|_{\mathcal{D}})$; since x' and x'' belong to the same $\equiv_D$-class, $a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x'|_{\mathcal{D}}, |x''|_{\mathcal{D}}) \geq e_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x'|_{\mathcal{D}})$ and, hence, $a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x'|_{\mathcal{D}}) \leq a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x''|_{\mathcal{D}})$. In a similar way, we can prove that $a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x''|_{\mathcal{D}}) \leq a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x'|_{\mathcal{D}})$ and conclude.

— $a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x'|_{\mathcal{D}}) \geq e_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}})$ (the case in which $a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x'|_{\mathcal{D}}) \geq e_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x'|_{\mathcal{D}})$ is similar): this case is simpler, since, by definition of $a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}$, it holds that $a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x'|_{\mathcal{D}}) = a_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}}, |x''|_{\mathcal{D}}) = e_{\mathbf{Beh}[\mathcal{Y},\mathcal{X}]}(|x|_{\mathcal{D}})$. □

Proposition 4. ((Kasangian and Labella 2009)) The unique *Tree*-functor from **Beh** to the terminal *Tree*-category **T** (see Example 4) is not PR, but every reflected path satisfies the property given in Definition 6 (2b) where the reflected factorization is unique.

Proposition 5. Let $F : \mathbf{Beh} \to \mathbf{Beh}$ be a PR *Tree*-functor; then, F is also FPR (and, hence, FPWR).

Proof. We only have to prove the second condition of Definition 6(2). Let $x \in \mathbf{Beh}[\mathcal{X}', \mathcal{X}]$ and $y \in \mathbf{Beh}[F(\mathcal{X}'), F(\mathcal{X})]$ the path such that $F_{\mathcal{X}'\mathcal{X}}(x) = y$. Let also consider the factorization $y = y'; y''$, with s' and s'' the extent of y' and y'' respectively, and let $\mathcal{Z} \cong \mathcal{D}(F(\mathcal{X}), y, s')$. Since F is a *Tree*-functor, the extent of x is $s's''$; thus, there exists $\mathcal{D}(\mathcal{X}, x, s')$. The claim is proved if we show that $F(\mathcal{D}(\mathcal{X}, x, s')) \cong \mathcal{Z}$. Let x' and x'' the portions of x with extent s' and s'', respectively; thus, $x = x'; x''$. By *Tree*-functoriality, $F_{\mathcal{X}'\mathcal{X}}(x'; x'') = y$ and there exist $F_{\mathcal{D}(\mathcal{X},x,s'),\mathcal{X}}(x)$ and $F_{\mathcal{X}',\mathcal{D}(\mathcal{X},x,s')}(x)$; however, by definition of **Beh**, it can only be $F_{\mathcal{D}(\mathcal{X},x,s'),\mathcal{X}}(x) = y'$ and $F_{\mathcal{X}',\mathcal{D}(\mathcal{X},x,s')}(x) = y''$; thus, $F(\mathcal{D}(\mathcal{X}, x, s')) \cong \mathcal{Z}$, since by functoriality $y' \in \mathbf{Beh}[F(\mathcal{D}(\mathcal{X}, x, s')), F(\mathcal{X})]$. □

Remark 2. It is clear from the previous proof that in **Beh** every reflected factorization is unique (up-to isomorphisms), i.e. every PR *Tree*-endofunctor also enjoys what in (Bunge and Fiore 2000) is called *unique factorization lifting* (UFL).

Though most of what we are going to prove holds for general trees, we shall restrict ourselves to regular trees, i.e. trees with finitely many non-isomorphic derivatives; we believe that this is not a limitation, since we designed our model to represent regular processes, e.g. processes definable in terms of a (finite) system of equations.

2.2. *Strong Bisimilarity*

We now work directly on our trees to provide the definition of strong bisimulation equivalence that agrees with the corresponding one usually introduced for LTS (see Section 5.1).

Definition 9. A symmetric relation between trees $\Re$ is a *strong bisimulation* if, for every $(\mathcal{X}, \mathcal{Y}) \in \Re$, it holds that $\forall x \in X\ \exists y \in Y$ such that $e_Y(y) = e_X(x)$ and $\forall s \leq e_X(x)$ it holds that that $(\mathcal{D}(\mathcal{X}, x, s), \mathcal{D}(\mathcal{Y}, y, s)) \in \Re$.

Two trees are *strongly bisimilar*, written $\mathcal{X} \simeq_S \mathcal{Y}$, if and only if there exists a strong bisimulation relating them.

We can show that strong bisimulations are the equivalences induced by factorized path reflecting *Tree*-functors. Notice that, in **Beh**, FPR coincides with PR (see Proposition 2 and Proposition 5 above); however, for the sake of uniformity with the case of branching bisimulation (see Section3.2), we shall rely on FPR.

Lemma 1. Let F : **Beh** $\rightarrow$ **Beh** be a FPR *Tree*-functor; then

1 $F(\mathcal{I}) = \mathcal{I}$;
2 $F(\mathcal{X}) = \mathcal{I}$ implies that $\mathcal{X}$ has only ϵ-labeled paths.

Proof. For the first claim, let $\pi \in$ **Beh**$[\mathcal{I}, F(\mathcal{I})]$; since F is PR, there exist $A' \in$ **Beh** and $p \in$ **Beh**$[A', \mathcal{I}]$ such that $F(A') = \mathcal{I}$. But the only such p is the path labeled with ϵ leading $\mathcal{I}$ to itself; thus, $A' = \mathcal{I}$ and, hence, $F(\mathcal{I}) = \mathcal{I}$. The second claim is trivial. □

Theorem 1. Two trees $\mathcal{X}$ and $\mathcal{Y}$ in **Beh** are strongly bisimilar if there is a FPR *Tree*-functor F : **Beh** $\rightarrow$ **Beh** such that $F(\mathcal{X}) \cong F(\mathcal{Y})$.

Proof. First of all, notice that F is PR. We prove that relation $\Re = \{(\mathcal{X}, F(\mathcal{X}))\}$ is a bisimulation, as defined in Definition 9; by transitivity of $\simeq_S$ and by the fact that isomorphic trees are also bisimilar, this will allow us to conclude, since $\mathcal{X} \simeq_S F(\mathcal{X}) \cong F(\mathcal{Y}) \simeq_S \mathcal{Y}$. Let $(\mathcal{X}, F(\mathcal{X})) \in \Re$.

1 Fix $x \in X$, that is also a path in **Beh**$[\mathcal{I}, \mathcal{X}]$; by reasoning up to $\equiv_{\mathcal{D}}$, it holds that, since F is a *Tree*-functor, $F_{\mathcal{I}\mathcal{X}}(x)$ is a path in **Beh**$[F(\mathcal{I}), F(\mathcal{X})]$ with the same extent as x. Because of Lemma 1(1), $F_{\mathcal{I}\mathcal{X}}(x)$ is a path in **Beh**$[\mathcal{I}, F(\mathcal{X})]$, i.e. $F_{\mathcal{I}\mathcal{X}}(x)$ is a path of $F(\mathcal{X})$. Now, fix a $s \leq e_X(x)$. Since F is a *Tree*-functor, it holds that $F(\mathcal{D}(\mathcal{X}, x, s)) \cong \mathcal{D}(F(\mathcal{X}), F_{\mathcal{I}\mathcal{X}}(x), s)$; this suffices to conclude that $(\mathcal{D}(\mathcal{X}, x, s), \mathcal{D}(F(\mathcal{X}), F_{\mathcal{I}\mathcal{X}}(x), s)) \in \Re$.
2 Now, take any path z in $F(\mathcal{X})$, i.e. $z \in$ **Beh**$[\mathcal{I}, F(\mathcal{X})]$; since F is PR, it holds that there exist $\mathcal{X}'$ and $x \in$ **Beh**$[\mathcal{X}', \mathcal{X}]$ such that $F(\mathcal{X}') = \mathcal{I}$ and $F_{\mathcal{X}'\mathcal{X}}(x) = z$. Because of Lemma 1(2), $\mathcal{X}'$ has only ϵ-labeled paths and so we can consider x as a path of $\mathcal{X}$. Moreover, since F is a *Tree*-functor, x and z have the same extent. Now, let s be a prefix of the extent of z and call z' the portion of z corresponding to s. By Proposition 5, F is also FPR; thus, there exist $\mathcal{X}', x', x''$ such that $x' \in$ **Beh**$[\mathcal{X}', \mathcal{X}]$, $x'' \in$ **Beh**$[\mathcal{I}, \mathcal{X}']$, $F_{\mathcal{X}'\mathcal{X}}(x') = z'$ and $F(\mathcal{X}') = \mathcal{D}(F(\mathcal{X}), z, s)$. By *Tree*-functoriality, $F_{\mathcal{X}'\mathcal{X}}(x') = z'$ implies that x' and z' have the same extent (viz., s) and so we can put $\mathcal{X}' = \mathcal{D}(\mathcal{X}, x, s)$, as required. □

3. Trees with Silent Actions

In the previous section, we gave a characterization of strong bisimilarity over trees in **Beh**. We now examine two notions of bisimulation arising in the presence of silent actions. For branching bisimilarity, factorized path reflection is still the property to use; in the weak case, a less demanding property is needed.

3.1. *Basic Definitions*

Let us specify an element of the alphabet to represent the silent action, called τ, and denote A_τ the set $A \cup \{\tau\}$, for $\tau \notin A$. Elements $t \in A_\tau^*$ have a decomposition $\tau^{i_0} a_1 \ldots \tau^{i_{n-1}} a_n \tau^{i_n}$, where $a_k \in A$; the notion of prefix in $\mathbf{A}_\tau$ is the usual one.

We also introduce the canonical function DEL : $A_\tau^* \to A^*$ which deletes τ's in words as follows:

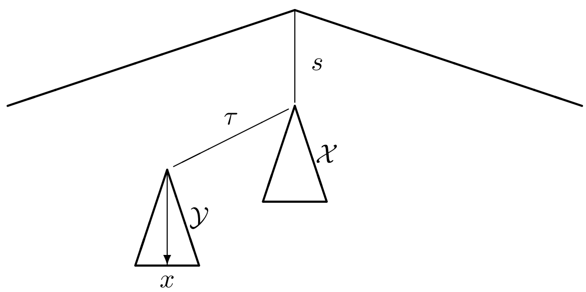
$$\text{DEL}(s) = \begin{cases} \epsilon & \text{if } s = \epsilon \\ \mu \bullet \text{DEL}(s') & \text{if } s = \mu \bullet s' \text{ and } \mu \neq \tau \\ \text{DEL}(s') & \text{if } s = \tau \bullet s' \end{cases}$$

Function DEL can be canonically extended to a monoidal functor from trees with silent actions, i.e. $\mathbf{A}_\tau$-trees (denoted Tree_τ when $\mathbf{A}$ is clear from the context), to Tree as follows.

Definition 10. DEL : $\mathit{Tree}_\tau \to \mathit{Tree}$ is a functor such that $\text{DEL}(X, e_X, a_X) = (Y, e_Y, a_Y)$, where $Y = X$, $e_Y(x) = \text{DEL}(e_X(x))$ and $a_Y(x, y) = \text{DEL}(a_X(x, y))$. Morphisms remain unchanged under DEL (indeed, functions defining morphisms from $\mathcal{X}$ to $\mathcal{Y}$ define morphisms from $\text{DEL}(\mathcal{X})$ to $\text{DEL}(\mathcal{Y})$ as well).

Since Tree_τ is a monoidal category, we can trivially define Tree_τ-categories, by rephrasing Definition 4. Then, the monoidal functor DEL just defined can be used to pass from a Tree_τ-category $\mathbf{C}$ to a Tree-category $\mathbf{C}'$: objects are the same as in $\mathbf{C}$ and $\mathbf{C}'[A, B] = \text{DEL}(\mathbf{C}[A, B])$. Such an operation is an instance of a more general operation in enriched categories called *change of base*. More precisely, in analogy with **Beh**, we can consider $\mathbf{Beh}_{\tau,\tau}$ as the Tree_τ-category whose objects and morphism-objects are objects of Tree_τ. Nonetheless, we will forget about τ's in morphism-objects of $\mathbf{Beh}_{\tau,\tau}$ by applying DEL to them. This procedure defines a Tree-category $\mathbf{Beh}_\tau$, where it is possible to define bisimulations between $\mathbf{A}_\tau$-trees without mentioning τ's.¶ In practice, $\mathbf{Beh}_\tau$ has the same objects as $\mathbf{Beh}_{\tau,\tau}$ but $\mathbf{Beh}_\tau[\mathcal{Y}, \mathcal{X}] = \text{DEL}(\mathbf{Beh}_{\tau,\tau}[\mathcal{Y}, \mathcal{X}])$.

Once we delete τ's, many derivatives can be accessed along a path after a given visible prefix s. For example, consider the tree



Here, the path x leads to both the derivative $\mathcal{X} + \tau\mathcal{Y}$ and $\mathcal{Y}$, respectively after $\text{DEL}(s)$ or $\text{DEL}(s \bullet \tau)$. It is however important to notice that there is always the largest of such trees ($\mathcal{X} + \tau\mathcal{Y}$ here) and any other tree so accessed (like $\mathcal{Y}$) is a τ-derivative of this one. The largest tree is $\mathcal{D}(\mathcal{X}, x, s)$ and its existence allows us to work with finite chains.

Notation We let $s \preccurlyeq e_X(x)$ mean that $s = \text{DEL}(t)$, for some $t \leq e_X(x)$ (where '$\leq$' is the prefix relation).

Definition 11. Let $\mathcal{X} = (X, e_X, a_X)$ be a tree in Tree_τ, $x \in X$ and $s \preccurlyeq e_X(x)$; then, $\mathcal{D}_\tau(\mathcal{X}, x, s) = \{\mathcal{D}(\mathcal{X}, x, t) \ : \ \text{DEL}(t) = s\}$.

¶ Notice that the passage from $\mathbf{Beh}_{\tau,\tau}$ to $\mathbf{Beh}_\tau$ corresponds to the passage from the arrow $\xrightarrow{\alpha}$ to the arrow $\overset{\alpha}{\Rightarrow}$ done by Milner in the context of LTS.

Intuitively, $\mathcal{D}_\tau(\mathcal{X}, x, s)$ is the family of derivatives in $\mathcal{X}$ reachable along x by s, once we forget τ's. Hence, given two of them, one is a τ-derivative of the other one.

Proposition 6. ((Kasangian and Labella 2009)) The unique *Tree*-functor from **Beh**$_\tau$ to the terminal *Tree*-category $\mathbf{T}$ (see Example 4) is not PR, but every reflected path satisfies the property given in Definition 6(2b) where the reflected factorization is unique up to silent moves.

In fact different objects of **Beh**$_\tau$ allowing the reflected factorization on a path $x = x'; x''$, with $\text{DEL}(e(x')) = s$ are elements of $\mathcal{D}_\tau(\mathcal{X}, x, s)$.

Proposition 7. Let F : **Beh**$_\tau$ $\to$ **Beh**$_\tau$ be a FPR *Tree*-functor; then the reflected factorization is unique up to silent moves.

Proof. Similar to the proof of Proposition 5. □

Remark 3. In particular, in **Beh**$_\tau$ every FPR *Tree*-endofunctor also enjoys the enriched form of Conduché property (see (Conduché 1972) and (Kasangian and Labella 2009)).

3.2. *Branching Bisimilarity*

We can now define branching bisimilarity in terms of τ-less paths; in Section 5.1 we shall prove that our definition agrees with the corresponding one usually introduced for LTS.

Definition 12. A symmetric relation on trees $\Re$ is a *branching bisimulation* if, for every $(\mathcal{X}, \mathcal{Y}) \in \Re$, it holds that $\forall x \in X\ \exists y \in Y$ such that $\text{DEL}(e_Y(y)) = \text{DEL}(e_X(x))$ and

1 $\forall s \preccurlyeq e_X(x)\ \forall \mathcal{X}' \in \mathcal{D}_\tau(\mathcal{X}, x, s)\ \exists \mathcal{Y}' \in \mathcal{D}_\tau(\mathcal{Y}, y, s)$ such that $(\mathcal{X}', \mathcal{Y}') \in \Re$;
2 $\forall s \preccurlyeq e_Y(y)\ \forall \mathcal{Y}' \in \mathcal{D}_\tau(\mathcal{Y}, y, s)\ \exists \mathcal{X}' \in \mathcal{D}_\tau(\mathcal{X}, x, s)$ such that $(\mathcal{X}', \mathcal{Y}') \in \Re$.

Two trees $\mathcal{X}$ and $\mathcal{Y}$ are *branching bisimilar*, written $\mathcal{X} \simeq_B \mathcal{Y}$, iff there is a branching bisimulation relating them.

We are now ready to prove for branching bisimilarity the same result that we have for strong bisimilarity in the case without τ's, by simply replacing **Beh** with **Beh**$_\tau$. We start with some preliminary results.

Lemma 2. Let F : **Beh**$_\tau$ $\to$ **Beh**$_\tau$ be a PR *Tree*-functor; then

1 $F(\mathcal{I}) = \mathcal{I}$;
2 $F(\mathcal{X}) = \mathcal{I}$ implies that $\mathcal{X}$ has only τ-labeled paths.

Theorem 2. Two trees $\mathcal{X}$ and $\mathcal{Y}$ in **Beh**$_\tau$ are branching bisimilar if there is a FPR *Tree*-functor F : **Beh**$_\tau$ $\to$ **Beh**$_\tau$ such that $F(\mathcal{X}) \cong F(\mathcal{Y})$.

Proof. We prove that $\Re = \{(\mathcal{X}, F(\mathcal{X}))\}$ is a branching bisimulation; to this aim, fix any $(\mathcal{X}, F(\mathcal{X})) \in \Re$.

— Let $x \in X$, that is also a path in **Beh**$_\tau[\mathcal{I}, \mathcal{X}]$; as usual, we reason up to $\equiv_\mathcal{D}$.
  1 Since F is a *Tree*-functor, it holds that $F_{\mathcal{I}\mathcal{X}}(x)$ is a path in **Beh**$_\tau[F(\mathcal{I}), F(\mathcal{X})]$ with the same visible extent as x. By Lemma 2(1), $F_{\mathcal{I}\mathcal{X}}(x) \in$ **Beh**$_\tau[\mathcal{I}, F(\mathcal{X})]$; thus, we have found a path of $F(\mathcal{X})$ with the same visible extent as x.

2 Now fix a $s \preccurlyeq e_X(x)$ and a $\mathcal{X}' \in \mathcal{D}_\tau(\mathcal{X}, x, s)$; moreover, let $x = x_1; x_2$ for $x_1 \in$ **Beh**$_\tau[\mathcal{X}', \mathcal{X}]$ and $x_2 \in$ **Beh**$_\tau[\mathcal{I}, \mathcal{X}']$. By *Tree*-functoriality and Lemma 2(1), $F_{\mathcal{X}'\mathcal{X}}(x_1) \in$ **Beh**$_\tau[F(\mathcal{X}'), F(\mathcal{X})]$ and $F_{\mathcal{I}\mathcal{X}'}(x_2) \in$ **Beh**$_\tau[\mathcal{I}, F(\mathcal{X}')]$. By compositionality (that holds since F is a *Tree*-functor), $F_{\mathcal{X}'\mathcal{X}}(x_1); F_{\mathcal{I}\mathcal{X}'}(x_2) = F_{\mathcal{I}\mathcal{X}}(x)$ and so $F(\mathcal{X}') \in \mathcal{D}_\tau(F(\mathcal{X}), F_{\mathcal{I}\mathcal{X}}(x), s)$, up-to isomorphism.

3 Now fix a prefix s of the extent of $F_{\mathcal{I}\mathcal{X}}(x)$ and a $\mathcal{Z} \in \mathcal{D}_\tau(F(\mathcal{X}), F_{\mathcal{I}\mathcal{X}}(x), s)$; let $z'; z''$ be the factorization of z such that $\mathcal{Z}$ is the derivative after z'. Since F is FPR, we can always write x as $x'; x''$ such that $x' \in$ **Beh**$_\tau[\mathcal{X}', \mathcal{X}]$, $F_{\mathcal{X}'\mathcal{X}}(x') = z'$ and $F(\mathcal{X}') = \mathcal{Z}$. But then $\mathcal{X}' \in \mathcal{D}_\tau(\mathcal{X}, x, s)$ up-to isomorphism, as required.

— Let z be a path in $F(\mathcal{X})$. We can reason like the 2nd item in the proof of Theorem 1 (here we have to use Lemma 2 in place of Lemma 1) to prove that there exists a $x \in X$ with the same visible extent as z and that, for every derivative $\mathcal{Z}$ along z, we can find a derivative $\mathcal{X}'$ along x (after the same visible prefix) such that $F(\mathcal{X}') = \mathcal{Z}$: it suffices to consider **Beh**$_\tau$ in place of **Beh** and $\mathcal{D}_\tau$ in place of $\mathcal{D}$. To prove that, for every $s \preccurlyeq e_X(x)$ and $\mathcal{X}' \in \mathcal{D}_\tau(\mathcal{X}, x, s)$, it holds that $F(\mathcal{X}') \in \mathcal{D}_\tau(F(\mathcal{X}), F_{\mathcal{I}\mathcal{X}}(x), s)$ up-to isomorphism, we can reason like in point 2. of the previous item of this proof. □

3.3. *Weak Bisimilarity*

We now define weak bisimilarity in terms of τ-less paths; in Section 5.1 we shall prove that our definition agrees with the corresponding one usually introduced for LTS.

Definition 13. A symmetric relation on trees $\Re$ is a *weak bisimulation* if, for every $(\mathcal{X}, \mathcal{Y}) \in \Re$, it holds that $\forall x \in X\ \exists y \in Y$ such that $\text{DEL}(e_Y(y)) = \text{DEL}(e_X(x))$ and $\forall s \preccurlyeq e_X(x)\ \forall \mathcal{X}' \in \mathcal{D}_\tau(\mathcal{X}, x, s)\ \exists \mathcal{Y}' \in \mathcal{D}_\tau(\mathcal{Y}, y, s)$ such that $(\mathcal{X}', \mathcal{Y}') \in \Re$.

Two trees $\mathcal{X}$ and $\mathcal{Y}$ are *weakly bisimilar*, written $\mathcal{X} \simeq_W \mathcal{Y}$, iff there exists a weak bisimulation relating them.

It is interesting to notice that Definition 12 differs from Definition 13 only for an extra symmetry requirement. Mainly, this implies that branching bisimilarity induces a correspondence between paths and so it is an equivalence relation between paths (in the style of the ‘back-and-forth’ approach (De Nicola et al. 1990)); on the contrary, such a path correspondence cannot be defined for weak bisimulations.

Theorem 3. Two trees $\mathcal{X}$ and $\mathcal{Y}$ in **Beh**$_\tau$ are weakly bisimilar if there is a FPWR *Tree*-functor F : **Beh**$_\tau \to$ **Beh**$_\tau$ such that $F(\mathcal{X}) \cong F(\mathcal{Y})$.

Proof. We prove that $\Re = \{(\mathcal{X}, F(\mathcal{X})\}$ is a weak bisimulation. We first fix $x \in X$ and show that: (i) there exists a path z in $F(\mathcal{X})$ with the same visible extent as x; and, (ii) every $\mathcal{X}' \in \mathcal{D}_\tau(\mathcal{X}, x, s)$, where $s \preccurlyeq e_X(x)$, is such that $F(\mathcal{X}') \in \mathcal{D}_\tau(F(\mathcal{X}), z, s)$ up-to isomorphism. These facts can be proved like the 1st and 2nd point of the first item of the proof of Theorem 2 (in that proof, it was only exploited that F is FPWR). Now, let z be a path in $F(\mathcal{X})$; like in the 2nd item of the proof of Theorem 2, we have that there exists a $x \in X$ with the same visible extent as z and that, for every derivative $\mathcal{Z}$ along z, we can find a derivative $\mathcal{X}'$ along x (after the same visible prefix) such that $F(\mathcal{X}') = \mathcal{Z}$ up-to isomorphism. This suffices to conclude. □

4. Complete Characterizations and Standard Representatives

The converse of Theorems 1, 2 and 3 hold as well but, to prove them, we have to provide a *standard representative* for every tree: such a representative will be obtained by merging those paths that have the same extent and equivalent relationships with other paths in the same tree. To this aim, we will define equivalences of paths as induced by $\simeq_S$ and $\simeq_B$ (remember that a simple equivalence on paths associated to $\simeq_W$ is not possible).

4.1. *Standard Representatives for Strong Bisimilarity*

Definition 14. Let $\equiv_S$ be the equivalence relation on paths of a tree $\mathcal{X}$ defined by $x \equiv_S x'$ if and only if $e_X(x) = e_X(x')$ and, for every $s \leq e_X(x)$, $\mathcal{D}(\mathcal{X}, x, s)$ is strongly bisimilar to $\mathcal{D}(\mathcal{X}, x', s)$. Let $|x|_S$ denote the $\equiv_S$-class of x.

The *standard strong representative* of a tree $\mathcal{X} = (X, e_X, a_X)$ is the tree $\mathbf{S}\mathcal{X} = (SX, e_{SX}, a_{SX})$, where

— $SX = \{|x|_S \ : \ x \in X\}$;
— $e_{SX}(|x|_S) = e_X(x)$;
— $a_{SX}(|x|_S, |y|_S) = \bigvee_{x' \in |x|_S, y' \in |y|_S} \ a_X(x', y')$.

It is worth noting that the join above exists, thanks to the definition of meet-semilattice monoid, and it is a maximum. Moreover, we can prove a stronger property:

Lemma 3. For every $x' \in |x|_S$ there exists a $y' \in |y|_S$ such that $a_X(x', y') = a_{SX}(|x|_S, |y|_S)$.

Proof. By definition, there exist $\bar{x} \in |x|_S$ and $\bar{y} \in |y|_S$ such that $a_X(\bar{x}, \bar{y}) = a_{SX}(|x|_S, |y|_S) = s$. Clearly, $s \leq e_X(x) = e_X(\bar{x}) = e_X(x')$ and $s \leq e_X(y) = e_X(\bar{y}) = e_X(y')$. Now, by definition of $\equiv_S$, $\mathcal{D}(\mathcal{X}, x', s) \simeq_S \mathcal{D}(\mathcal{X}, \bar{x}, s)$; moreover, since $a_X(\bar{x}, \bar{y}) = s$, it holds that $\bar{y} \in \mathcal{D}(\mathcal{X}, \bar{x}, s)$. Hence, there must be a $y' \in \mathcal{D}(\mathcal{X}, x', s)$ such that $y' \equiv_S \bar{y}$ and so $a_X(x', y') \geq s$. Since s is the maximum agreement between elements in $|x|_S$ and $|y|_S$, we can conclude. □

We approach the proof that $\mathbf{S}\mathcal{X}$ is a standard representative of the $\simeq_S$-equivalence class of $\mathcal{X}$ by showing that $\mathcal{X}$ and $\mathbf{S}\mathcal{X}$ have the same transitions; this also easily implies that **S** is a *Tree*-functor.

Lemma 4. $\mathbf{S}(\mathcal{D}(\mathcal{X}, x, s)) = \mathcal{D}(\mathbf{S}\mathcal{X}, |x|_S, s)$.

Proof. If $\mathcal{D}(\mathcal{X}, x, s) \cong \mathcal{I}$, the claim is trivial. Let $y \in X$ be a path belonging to $\mathcal{D}(\mathcal{X}, x, s)$; thus, $a_X(x, y) \geq s$. By Definition 14, the agreement between $|x|_S$ and $|y|_S$ in $\mathbf{S}\mathcal{X}$ is at least $a_X(x, y)$; hence, $|y|_S$ is a path in $\mathbf{S}(\mathcal{D}(\mathcal{X}, x, s))$. Vice versa, let $|z|_S$ be a path of $\mathbf{S}\mathcal{X}$ belonging to $\mathbf{S}(\mathcal{D}(\mathcal{X}, x, s))$; by definition, the agreement between $|x|_S$ and $|z|_S$ is $s' \geq s$. By Lemma 3, there exists a $z' \in |z|_S$ ($\subseteq X$) such that $a_X(x, z') = s'$; so, z' is also a path in $\mathcal{D}(\mathcal{X}, x, s)$. A similar reasoning applies to agreements, while extents are trivially identical. □

Lemma 5. $\mathcal{X} \simeq_S \mathbf{S}\mathcal{X}$.

Proof. Define the relation $\Re = \{(\mathcal{X}, \mathbf{S}\mathcal{X})\}$, fix $(\mathcal{X}, \mathbf{S}\mathcal{X}) \in \Re$ and pick up any $x \in X$ (the case in which we pick up a path $|x|_S \in \mathbf{S}\mathcal{X}$ is similar); by construction, $|x|_S$ is a path in $\mathbf{S}\mathcal{X}$ with extent $e_X(x)$. Now, choose $s \leq e_X(x)$; by Lemma 4, $\mathbf{S}(\mathcal{D}(\mathcal{X}, x, s)) = \mathcal{D}(\mathbf{S}\mathcal{X}, |x|_S, s)$. and this suffices to conclude. □

Proposition 8. $\mathcal{X} \simeq_S \mathcal{Y}$ if and only if $\mathbf{S}\mathcal{X} \cong \mathbf{S}\mathcal{Y}$.

Proof. The "if" part trivially follows from Lemma 5. For the "only if" part, we prove that $\mathcal{X} \simeq_S \mathcal{Y}$ induces a *Tree*-isomorphism between $\mathbf{S}\mathcal{X}$ and $\mathbf{S}\mathcal{Y}$. Let $x \in X$ and $y \in Y$ be corresponding paths in the bisimulation. Hence, we can build up a bijective mapping between paths in $\mathbf{S}\mathcal{X}$ and $\mathbf{S}\mathcal{Y}$ that preserves the extent: each $x' \in |x|_S$ is associated by the bisimulation to a $y' \in Y$ such that $y' \in |y|_S$ (by transitivity, y' can also bisimulate x, that in turn bisimulates y), and vice versa. So, the mapping is surjective; it is also injective, since two different paths in the representative cannot be equivalent (by construction). This proves that $\mathbf{S}\mathcal{X}$ and $\mathbf{S}\mathcal{Y}$ have corresponding paths with the same extent; we are left with proving that their paths have also the same agreement. Choose $|x|_S$ and $|x'|_S$ in $\mathbf{S}\mathcal{X}$ and let s be their agreement; thus, there exist $x_1 \in |x|_S$ and $x_2 \in |x'|_S$ such that $a_X(x_1, x_2) = s$ and this is the maximum. Let $|y|_S$ and $|y'|_S$ be the corresponding paths in $\mathbf{S}\mathcal{Y}$; by hypothesis, $\mathcal{D}(\mathcal{X}, x_1, s) \simeq_S \mathcal{D}(\mathcal{Y}, y, s)$; since x_2 is also a path in $\mathcal{D}(\mathcal{X}, x_1, s)$, there exists a y' in $\mathcal{D}(\mathcal{Y}, y, s)$ (i.e. such that $a_Y(y, y') \geq s$) corresponding to x_2. By contradiction, let $a_Y(y, y') > s$. Then, consider $\mathcal{D}(\mathcal{Y}, y, a_Y(y, y'))$; by hypothesis of bisimilarity, it corresponds to a proper derivative of $\mathcal{D}(\mathcal{X}, x_1, s)$. But then s would not be the maximum agreement between paths in $|x|_S$ and $|x'|_S$. □

Corollary 1. $\mathbf{S}(\mathbf{S}\mathcal{X}) \cong \mathbf{S}\mathcal{X}$.

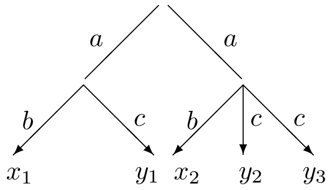
Because of Proposition 8 and Corollary 1, $\mathbf{S}\mathcal{X}$ can be considered a minimal strong representative of the $\simeq_S$-equivalence class of $\mathcal{X}$. We are now ready to prove the converse of Theorem 1.

Theorem 4. Let $\mathcal{X}$ and $\mathcal{Y}$ be strongly bisimilar trees in **Beh**; then there is a FPR *Tree*-functor F : **Beh** $\to$ **Beh** such that $F(\mathcal{X}) \cong F(\mathcal{Y})$.

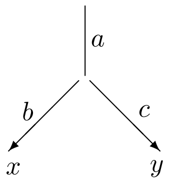
Proof. We prove that **S** can be extended to the desired *Tree*-functor. By Proposition 8, $\mathbf{S}(\mathcal{X}) \cong \mathbf{S}(\mathcal{Y})$. To prove that **S** is a *Tree*-functor, we have to prove that **S** induces a *Tree*-morphism between $\mathbf{Beh}[\mathcal{X}', \mathcal{X}]$ and $\mathbf{Beh}[\mathbf{S}\mathcal{X}', \mathbf{S}\mathcal{X}]$, for every $\mathcal{X}'$. If $\mathbf{Beh}[\mathcal{X}', \mathcal{X}]$ is the empty tree, we trivially conclude. Otherwise, take any path $x' \in \mathbf{Beh}[\mathcal{X}', \mathcal{X}]$; then, $\mathcal{X}' = \mathcal{D}(\mathcal{X}, x, s)$ up-to isomorphism, where x is a path in $\mathcal{X}$ such that x' is its initial part labeled by s. By Lemma 4, $\mathbf{S}\mathcal{X}$ evolves to $\mathbf{S}\mathcal{X}'$ via a path with the same extent as x'; moreover, by definition of standard representatives, the agreement between any two paths cannot decrease.

We have to prove that **S** is FPR; because of Proposition 5, it suffices to prove that it is PR. Fix a $\mathcal{Z}$ and a $z' \in \mathbf{Beh}[\mathcal{Z}, \mathbf{S}\mathcal{X}]$; thus, $\mathcal{Z} = \mathcal{D}(\mathbf{S}\mathcal{X}, |z|_S, s)$ up-to isomorphism, where $|z|_S$ is a path in $\mathbf{S}\mathcal{X}$ such that z' is its initial part labeled by s. By Lemma 4, $\mathcal{Z} = \mathbf{S}(\mathcal{D}(\mathcal{X}, z, s))$ up-to isomorphism; thus, we have found a $\mathcal{X}'$ (that, up-to isomorphism, is $\mathcal{D}(\mathcal{X}, z, s)$) and a $x' \in \mathbf{Beh}[\mathcal{X}', \mathcal{X}]$ (viz., the initial portion of z labeled by s) such that $\mathbf{S}\mathcal{X}' = \mathcal{Z}$ and $\mathbf{S}_{\mathcal{X}'\mathcal{X}}(x') = z'$. □

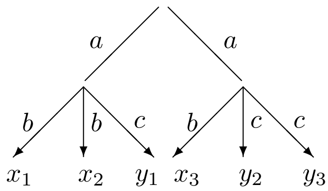
Example 6. Let us consider the tree $\mathcal{X}$:



By using the construction put forward in Definition 14, we can obtain its standard representative, i.e. the tree $\mathcal{Y}$:



If we now consider the tree $\mathcal{Z}$:



its standard representative is still $\mathcal{Y}$ and, indeed, $\mathcal{X}$ and $\mathcal{Z}$ are strong bisimilar. Indeed, by letting $f : \mathcal{X} \to \mathcal{Y}$ and $g : \mathcal{Z} \to \mathcal{Y}$ be the *Tree*-morphisms such that $f(x_1) = f(x_2) = x = g(x_1) = g(x_2) = g(x_3)$ and $f(y_1) = f(y_2) = f(y_3) = y = g(y_1) = g(y_2) = g(y_3)$, we have that

$$\mathcal{X} \xrightarrow{f} \mathcal{Y} \xleftarrow{g} \mathcal{Z}$$

i.e., there exists a span of *Tree*-morphisms between the two strongly bisimilar trees $\mathcal{X}$ and $\mathcal{Z}$. These morphisms reflect paths with their factorizations.

To conclude, notice that **S** is characterizable also through a maximality condition among those enjoying factorized path reflection, i.e., every PR *Tree*-functor is annihilated by **S**.

Proposition 9. Every FPR *Tree*-functor F : **Beh** $\to$ **Beh** is such that **S**$F \cong$ **S**.

Proof. Since F reflects paths, $F_{\mathcal{I},\mathcal{X}}$ is an epimorphism from **Beh**$[\mathcal{I}, \mathcal{X}]$ to **Beh**$[\mathcal{I}, F(\mathcal{X})]$ (see Lemma 1(1)); therefore, there is an epimorphism from $\mathcal{X}$ to $F(\mathcal{X})$. This means that $F(\mathcal{X})$ is a quotient of $\mathcal{X}$ w.r.t. a strong bisimulation; but **S** induces the coarsest strong bisimulation, hence the assert. *Tree*-naturality of the isomorphism holds since both images are induced via quotienting on paths. □

4.2. *Standard Representatives for Branching Bisimilarity*

We now rephrase the results of the previous section to deal with branching bisimilarity. However, due to the presence of τ's, we need some extra efforts to build the standard branching representative. The same technicalities will be useful for building the standard weak representative in the next section.

Definition 15. Let $\mathcal{X} = (X, e_X, a_X)$ be a tree in $Tree_\tau$, $x \in X$ and $s \preccurlyeq e_X(x)$. Then

— $\mathcal{D}_d(\mathcal{X}, x, s) = \{\text{DEL}(\mathcal{Z}) \ : \ \mathcal{Z} \in \mathcal{D}_\tau(\mathcal{X}, x, s)\}$;
— $\mathcal{D}_S(\mathcal{X}, x, s) = \{\mathbf{S}(\mathcal{Z}) \ : \ \mathcal{Z} \in \mathcal{D}_d(\mathcal{X}, x, s)\}$.

Intuitively, $\mathcal{D}_d(\mathcal{X}, x, s)$ and $\mathcal{D}_S(\mathcal{X}, x, s)$ are the families of τ-less derivatives reachable along x by s, pruned by DEL and also quotiented by **S**, respectively.

Notice that the sets $\mathcal{D}_\tau(\mathcal{X}, x, s)$, $\mathcal{D}_d(\mathcal{X}, x, s)$ and $\mathcal{D}_S(\mathcal{X}, x, s)$ are chains w.r.t. the property of being τ-summand, that are finite thanks to regularity of our trees; DEL induces also a monotonic function from the first to the second one preserving the extremes; similarly, **S** induces a monotonic function from the second to the third one preserving the extremes. This last correspondence is actually made out of a family of epimorphisms in $Tree$.

Definition 16. Let $\equiv_B$ be the equivalence relation on paths of a given tree $\mathcal{X}$ defined by $x \equiv_B x'$ if and only if $\text{DEL}(e_X(x)) = \text{DEL}(e_X(x'))$ and, for every $s \preccurlyeq e_X(x)$, $\mathcal{D}_S(\mathcal{X}, x, s)$ is in a bijective correspondence with $\mathcal{D}_S(\mathcal{X}, x', s)$, such that corresponding trees are isomorphic; let $|x|_B$ denote the $\equiv_B$-equivalence class of x.

The *standard branching representative* of a tree $\mathcal{X} = (X, e_X, a_X)$ is the tree $\mathbf{B}\mathcal{X} = (BX, e_{BX}, a_{BX})$, where

— $BX = \{|x|_B \ : \ x \in X\}$;
— $e_{BX}(|x|_B) = \tau^{i_1} s_1 \tau^{i_2} s_2 \dots \tau^{i_n} s_n \tau^{i_{n+1}}$, with $i_k = |\mathcal{D}_S(\mathcal{X}, x, s_1 \dots s_{k-1})| - 1$, for $1 \leq k \leq n+1$ and $e_X(x) = \tau^{k_1} s_1 \tau^{k_2} s_2 \dots \tau^{k_n} s_n \tau^{k_{n+1}}$;
— $a_{BX}(|x|_B, |y|_B) = \tau^{i_1} s_1 \tau^{i_2} s_2 \dots s_m \tau^{i_{m+1}}$, with
  - $i_k = |\mathcal{D}_S(\mathcal{X}, x, s_1 \dots s_{k-1})| - 1$, for $1 \leq k \leq m$, and
  - $i_{m+1} = h - 1$, where h is the length of the longest common prefix of the chains $\mathcal{D}_S(\mathcal{X}, x, s_1 s_2 \dots s_m)$ and $\mathcal{D}_S(\mathcal{X}, y, s_1 s_2 \dots s_m)$ (i.e., $\mathcal{D}_S(\mathcal{X}, x, s_1 s_2 \dots s_m) = \{\mathcal{X}_1, \dots, \mathcal{X}_k, \mathcal{X}'_{h+1} \dots, \mathcal{X}'_{h+p}\}$ and $\mathcal{D}_S(\mathcal{X}, y, s_1 s_2 \dots s_m) = \{\mathcal{X}_1, \dots, \mathcal{X}_h, \mathcal{X}''_{h+1} \dots, \mathcal{X}''_{h+q}\}$, for either $p = q = 0$, or $p = 0$ and $q > 0$, or $p > 0$ and $q = 0$, or $\mathcal{X}'_{h+1} \neq \mathcal{X}''_{h+1}$)

  whenever $\bigvee_{x' \in |x|_B, y' \in |y|_B} a_X(x', y') = \tau^{k_1} s_1 \tau^{k_2} s_2 \dots \tau^{k_m} s_m \tau^{k_{m+1}}$.

Lemma 6.

1 If $\mathcal{Y} \in \mathcal{D}_\tau(\mathcal{X}, x, s)$, then $\mathbf{B}\mathcal{Y} \in \mathcal{D}_\tau(\mathbf{B}\mathcal{X}, |x|_B, s)$;
2 if $\mathcal{Z} \in \mathcal{D}_\tau(\mathbf{B}\mathcal{X}, |x|_B, s)$, then $\mathcal{Z} = \mathbf{B}\mathcal{Y}$, for some $\mathcal{Y} \in \mathcal{D}_\tau(\mathcal{X}, x, s)$.

Proof. Fix a $x \in X$ and a $s = a_1 \dots a_h \preccurlyeq e_X(x)$; by construction of $\mathbf{B}\mathcal{X}$, it holds that $|\mathcal{D}_\tau(\mathbf{B}\mathcal{X}, |x|_B, s)| = |\mathcal{D}_S(\mathcal{X}, x, s)|$ because $e_{BX}(|x|_B) = \tau^{i_1} a_1 \tau^{i_2} \dots a_h \tau^{i_{h+1}} \dots a_m \tau^{i_{m+1}}$, where $i_{h+1} = |\mathcal{D}_S(\mathcal{X}, x, s)| - 1$. We let ϕ be the monotonic bijective correspondence between $\mathcal{D}_S(\mathcal{X}, x, s)$ and $\mathcal{D}_\tau(\mathbf{B}\mathcal{X}, |x|_B, s)$; thus, $\phi \circ \mathbf{S} \circ \text{DEL} : \mathcal{D}_\tau(\mathcal{X}, x, s) \to \mathcal{D}_\tau(\mathbf{B}\mathcal{X}, |x|_B, s)$ is a surjective function (it is a composition of surjective functions). We can prove both claims of this Lemma if we show that $\phi \circ \mathbf{S} \circ \textsc{del}$ coincides with $\mathbf{B}$. To this aim, assume that $\mathcal{Z}$ is the n-th element of the chain $\mathcal{D}_\tau(\mathbf{B}\mathcal{X}, |x|_B, s)$ and take any $\mathcal{Y} \in \mathcal{D}_\tau(\mathcal{X}, x, s)$ such that $\mathbf{S}(\textsc{del}(\mathcal{Y}))$ is the n-th element of the chain $\mathcal{D}_S(\mathcal{X}, x, s)$; we shall prove that $\mathbf{B}\mathcal{Y} = \mathcal{Z}$. This fact will trivially prove the first claim, whereas the second claim is implied by surjectivity of $\phi \circ \mathbf{S} \circ \textsc{del}$.

By construction, $\mathcal{Z} = (Z, e_Z, a_Z)$, where $Z = \{|y|_B : a_{BX}(|x|_B, |y|_B) \geq \tau^{i_1} a_1 \tau^{i_2} \dots a_h \tau^{n-1}\}$, $e_Z(|y|_B) = \tau^{i_{h+1}-(n-1)} a_{h+1} \tau^{i_{h+2}} \dots a_m \tau^{i_{m+1}}$ and $a_Z(|y|_B, |y'|_B) = a_{BX}(|y|_B, |y'|_B) - \tau^{i_1} a_1 \tau^{i_2} \dots a_h \tau^{n-1}$. To prove that $\mathbf{B}\mathcal{Y} = \mathcal{Z}$, we have to prove three facts about $\mathbf{B}\mathcal{Y}$:

its set of paths is Z: Let $y \in Y$; thus, $a_X(x, y) \geq \tau^{k_1} a_1 \tau^{k_2} \dots a_h \tau^k$, where $\mathcal{Y} = \mathcal{D}(\mathcal{X}, x, \tau^{k_1} a_1 \tau^{k_2} \dots a_h \tau^k)$. Hence, y corresponds to a path in the n-th element of the chain $\mathcal{D}_S(\mathcal{X}, x, a_1 \dots a_h)$; so, $\mathcal{D}_S(\mathcal{X}, x, a_1 \dots a_h)$ and $\mathcal{D}_S(\mathcal{X}, y, a_1 \dots a_h)$ share at least their first n elements. This implies that $a_{BX}(|x|_B, |y|_B) \geq \tau^{i_1} a_1 \tau^{i_2} \dots a_h \tau^{n-1}$ and hence $|y|_B \in Z$. Conversely, let $|y|_B \in Z$; then, $a_{BX}(|x|_B, |y|_B) = \tau^{i_1} a_1 \tau^{i_2} \dots a_h \tau^{n-1}$. Then, we can find a $\mathcal{Y}'$ and $y \in Y' \cap |y|_B$ such that $\mathbf{S}(\textsc{del}(\mathcal{Y}'))$ is the n-th element in $\mathcal{D}_S(\mathcal{X}, x, a_1 \dots a_h)$; but then $\mathcal{D}_S(\mathcal{X}, x, a_1 \dots a_h)$ and $\mathcal{D}_S(\mathcal{X}, y', a_1 \dots a_h)$ have at least the first n derivatives in common. Since also $\mathbf{S}(\textsc{del}(\mathcal{Y}))$ is the n-th element in $\mathcal{D}_S(\mathcal{X}, x, a_1 \dots a_h)$, we can find a $y'' \in Y$ such that $\mathcal{D}_S(\mathcal{X}, y', a_1 \dots a_h)$ and $\mathcal{D}_S(\mathcal{X}, y'', a_1 \dots a_h)$ are in a bijective correspondence whose corresponding derivatives are isomorphic. But, hence, $y'' \equiv_B y' \equiv_B y$ and we can conclude.

its extent function is e_Z**:** first notice that $|\mathcal{D}_S(\mathcal{Y}, x, \epsilon)| = |\mathcal{D}_S(\mathcal{X}, x, s)| - (n-1)$. Thus, every $|y|_B \in Z$ has extent that starts with $|\mathcal{D}_S(\mathcal{X}, x, s)| - (n-1) - 1$ τ's and then is $e_{BX}(|x|_B) - \tau^{i_1} a_1 \tau^{i_2} \dots a_h \tau^{i_{h+1}}$. Recall that $|\mathcal{D}_S(\mathcal{X}, x, s)| - 1 = i_{h+1}$; thus, $|\mathcal{D}_S(\mathcal{X}, x, s)| - (n-1) - 1 = i_{h+1} - (n-1)$. So, $e_{BY}(|y|_B) = e_Z(|y|_B)$, as required.

its agreement function is a_Z**:** similar to the extent. □

Lemma 7. $\mathcal{X} \simeq_B \mathbf{B}\mathcal{X}$.

Proof. Define the relation $\Re = \{(\mathcal{X}, \mathbf{B}\mathcal{X})\}$, any $(\mathcal{X}, \mathbf{B}\mathcal{X}) \in \Re$ and any $x \in X$; then, by construction, $|x|_B$ is a path in $\mathbf{B}\mathcal{X}$ with extent $e_X(x)$ such that $\textsc{del}(e(|x|_B)) = \textsc{del}(e_X(x))$. Now, choose $s \preccurlyeq e_X(x)$ and let $\mathcal{Y} \in \mathcal{D}_\tau(\mathcal{X}, x, s)$. By Lemma 6(1), $\mathbf{B}\mathcal{Y} \in \mathcal{D}_\tau(\mathbf{B}\mathcal{X}, x, s)$ and, by construction, $(\mathcal{Y}, \mathbf{B}\mathcal{Y}) \in \Re$. The converse is proved by using Lemma 6(2). □

Proposition 10. $\mathcal{X} \simeq_B \mathcal{Y}$ if and only if $\mathbf{B}\mathcal{X} \cong \mathbf{B}\mathcal{Y}$.

Proof. If $\mathbf{B}\mathcal{X} \cong \mathbf{B}\mathcal{Y}$, then $\mathcal{X} \simeq_B \mathcal{Y}$ holds because of Lemma 7. Suppose $\mathcal{X} \simeq_B \mathcal{Y}$. Then, given $x \in X$, there exists $y \in Y$ with the properties in Definition 12; we have to prove that the function associating $|x|_B$ to $|y|_B$ can be extended to an isomorphism of trees. Let $s \preccurlyeq e_X(x)$; then $\mathcal{D}_\tau(\mathcal{X}, x, s)$ is a chain and $\mathcal{D}_\tau(\mathcal{Y}, y, s)$ is the corresponding chain in $\mathcal{Y}$. By the symmetry in the first item of Definition 12, there is a correspondence between the two chains which is surjective on both sides, and, being monotonic, preserves the extremes. This correspondence induces a correspondence between $\mathcal{D}_d(\mathcal{X}, x, s)$ and $\mathcal{D}_d(\mathcal{Y}, y, s)$, via application of $\textsc{del}$, which preserves order and extremes and, again, between $\mathcal{D}_S(\mathcal{X}, x, s)$ and $\mathcal{D}_S(\mathcal{Y}, y, s)$, via application of the function $\mathbf{S}$. In this last stage, the correspondence becomes injective on both sides because two adjacent members of the chain cannot be equivalent. Hence we have a bijection as required, because, due to the fact that we are dealing with standard representatives, corresponding trees must be isomorphic. As a consequence, $|x|_B$ and $|y|_B$ have the same extent, and also agreement with any other path is strictly preserved. On the other hand the symmetric case can be dealt with analogously, hence we have an isomorphism of trees. □

To prove the converse of Theorem 2, we need to prove that the correspondence which associates $\mathbf{B}\mathcal{X}$ with $\mathcal{X}$ is a FPR *Tree*-functor. Moreover, similarly to the strong case, **B** can be also characterized through a maximality condition among those that reflect factorized paths.

Theorem 5. If two trees $\mathcal{X}$ and $\mathcal{Y}$ in $\mathbf{Beh}_\tau$ are branching bisimilar, then there is a FPR *Tree*-functor $F : \mathbf{Beh}_\tau \to \mathbf{Beh}_\tau$ such that $F(\mathcal{X}) \cong F(\mathcal{Y})$.

Proof. Consider the correspondence induced by **B**. Such a correspondence is a *Tree*-functor (by Lemma 6(1)) that reflects paths (by Lemma 6(2)); this fact can be proved like in Theorem 4, by using Lemma 6 in place of Lemma 4 and $\mathbf{Beh}_\tau$ in place of **Beh**.

We are left to prove that **B** is FPR. To this aim, fix a $\mathcal{Z}$ and a $z \in \mathbf{Beh}_\tau[\mathcal{Z}, \mathbf{B}\mathcal{X}]$; by paths reflection, we know that there exist $\mathcal{X}'$ and $x \in \mathbf{Beh}_\tau[\mathcal{X}', \mathcal{X}]$ such that $\mathbf{B}(\mathcal{X}') = \mathcal{Z}$ and $\mathbf{B}_{\mathcal{X}'\mathcal{X}}(x) = z$. We must prove that, for every such x and for every factorization $z = z_1; z_2$ (where $z_2 \in \mathbf{Beh}_\tau[\mathcal{Z}, \mathcal{Z}']$ and $z_1 \in \mathbf{Beh}_\tau[\mathcal{Z}', \mathbf{B}\mathcal{X}]$) we can find a $\mathcal{X}''$ and a factorization $x = x_1; x_2$ such that $x_2 \in \mathbf{Beh}_\tau[\mathcal{X}', \mathcal{X}'']$, $x_1 \in \mathbf{Beh}_\tau[\mathcal{X}'', \mathcal{X}]$, $\mathbf{B}(\mathcal{X}'') = \mathcal{Z}'$ and $\mathbf{B}_{\mathcal{X}''\mathcal{X}}(x_1) = z_1$. By Lemma 6(2), $\mathcal{Z}'$ is the branching representative of a $\mathcal{X}'' \in \mathcal{D}_\tau(\mathcal{X}, x, s)$ up-to isomorphism, for some $\mathcal{X}''$ and s that is the visible extent of the paths x_1 and z_1. □

Example 7. Consider the two trees in Figure 4: the rightmost (call it $\mathcal{X}$) is the standard branching representative of the leftmost (call it $\mathcal{Y}$). It suffices to apply the construction put forward in Definition 16 to $\mathcal{Y}$, and let $x' = |x|_B$ and $y' = |y_1|_B = |y_2|_B$. Thus, similarly to the case of strong bisimulation, we can find a span of *Tree*-morphisms

$$\mathcal{Y} \xrightarrow{f} \mathcal{X} \xleftarrow{g} \mathcal{X}$$

where f has been intuitively described in the introduction (just before Figure 4) and g is the identity. These morphisms reflect paths with their factorizations, as prescribed by Definition 6(2).

Proposition 11. Every FPR *Tree*-functor $F : \mathbf{Beh}_\tau \to \mathbf{Beh}_\tau$ is such that $\mathbf{B}F \cong \mathbf{B}$.

Proof. Since F reflects paths, $F_{\mathcal{I},\mathcal{X}}$ induces an epimorphism from $\mathbf{Beh}_\tau[\mathcal{I}, \mathcal{X}]$ to $\mathbf{Beh}_\tau[\mathcal{I}, F(\mathcal{X})]$ (see Lemma 2); therefore, there is an epimorphism from $\mathcal{X}$ to $F(\mathcal{X})$. Since F reflects factorized paths, $F(\mathcal{X})$ is a quotient of $\mathcal{X}$ w.r.t. a branching bisimulation; but **B** induces the coarsest branching bisimulation, hence the assert. *Tree*-naturality of the isomorphism holds because both images are induced via quotienting on paths. □

4.3. *Standard Representatives for Weak Bisimilarity*

To build up the standard weak representative, we shall work on the branching representative and prune it of some paths, namely the non-maximal (w.r.t. the induced chains of derivatives) ones. We now formally define such an order relation over trees and use it to build the representative.

Definition 17. Let $\leq_B$ be the preorder relation on paths of a given tree $\mathcal{X}$ defined by $x \leq_B x'$ if and only if $\text{DEL}(e_X(x)) = \text{DEL}(e_X(x'))$ and, for every $s \preccurlyeq e_X(x)$, there is a monotonic injective function from $\mathcal{D}_S(\mathcal{X}, x, s)$ to $\mathcal{D}_S(\mathcal{X}, x', s)$ such that corresponding trees are isomorphic.

When considered on $\equiv_B$-classes of paths, $\leq_B$ becomes a partial order with finite chains (recall that finiteness is guaranteed by the fact that we restricted ourselves to regular trees) and, therefore, with maximal elements; representatives of maximal classes will be called *maximal paths* in the original tree. In the sequel, $|x|_W$ will denote the equivalence class $|x'|_B$ of a maximal path x' such that $|x|_B \leq_B |x'|_B$.

Definition 18. The *standard weak representative* of a tree $\mathcal{X} = (X, e_X, a_X)$ is the tree $\mathbf{W}\mathcal{X} = (WX, e_{WX}, a_{WX})$, where

— $WX = \{|x|_W \,:\, x \in X \text{ and } x \text{ is maximal}\}$;
— $e_{WX}(|x|_W) = e_{BX}(|x|_B)$;
— $a_{WX}(|x|_W, |y|_W) = a_{BX}(|x|_B, |y|_B)$.

We start the completeness proof with a technical proposition that will be useful in the sequel; we then follow a path very similar to the ones followed for the strong and branching cases.

Proposition 12. Let $\mathcal{X}$ be a tree; then, for all $x \in X$ there is a $x' \in |x|_W$ such that for all $y \in X$ there is a $y' \in |y|_W$ such that $\text{DEL}(a_X(x', y')) = \text{DEL}(a_{WX}(|x|_W, |y|_W))$.

Proof. First, notice that for all $x, y \in X$ and for all $x' \in |x|_B$ there exists a $y' \in |y|_B$ such that $\text{DEL}(a_X(x', y')) = \text{DEL}(a_{BX}(|x|_B, |y|_B))$. Indeed, by definition, $\forall x \in X\ \forall y \in X\ \exists x' \in |x|_B\ \exists y' \in |y|_B$ such that $\text{DEL}(a_X(x', y')) = \text{DEL}(a_{BX}(|x|_B, |y|_B))$; moreover, since $x' \in |x|_B$, we have that $x' \equiv_B x$.

To prove this Proposition, we just notice that $|x|_W = |x'|_B$, for a maximal x', and $|y|_W = |y'|_B$, for a maximal y'. What we have just proved for $\equiv_B$-classes entails that we can take such x' and y' to obtain the claim. □

Lemma 8.

1 if $\mathcal{Y} \in \mathcal{D}_\tau(\mathcal{X}, x, s)$, then $\mathbf{W}\mathcal{Y} \in \mathcal{D}_\tau(\mathbf{W}\mathcal{X}, |x|_W, s)$;
2 if $\mathcal{Z} \in \mathcal{D}_\tau(\mathbf{W}\mathcal{X}, |x|_W, s)$, then $\mathcal{Z} = \mathbf{W}\mathcal{Y}$, for some $\mathcal{Y} \in \mathcal{D}_\tau(\mathcal{X}, x', s)$ and $x' \in |x|_W$.

Proof. The same argument as in the proof of Lemma 6, by only considering maximal paths. To prove the second claim, we exploit Proposition 12: x' is the path fixed by the first existential of that result. □

Lemma 9. $\mathcal{X} \simeq_W \mathbf{W}\mathcal{X}$.

Proof. Similar to the proof of Lemma 7, but using Lemma 8. □

Proposition 13. $\mathcal{X} \simeq_W \mathcal{Y}$ if and only if $\mathbf{W}\mathcal{X} \cong \mathbf{W}\mathcal{Y}$.

Proof. If $\mathbf{W}\mathcal{X} \cong \mathbf{W}\mathcal{Y}$, then $\mathcal{X} \simeq_W \mathcal{Y}$ holds because of Lemma 7. Suppose $\mathcal{X} \simeq_W \mathcal{Y}$. Then, given $x \in X$, there exists $y \in Y$ with the properties as in Definition 13; we have to prove that the function associating $|x'|_B$ to $|y'|_B$ for maximal paths can be extended to an isomorphism of trees. Let us take a maximal x; then, for a corresponding y, we will have $|y|_B \leq_B |y'|_B$, for y' maximal. Let $s \preccurlyeq e_X(x)$, with $\mathcal{D}_\tau(\mathcal{X}, x, s)$ and $\mathcal{D}_\tau(\mathcal{Y}, y, s)$ its corresponding chains.

By Definition 13, there is a correspondence between the two chains which is surjective on the side of $\mathcal{D}_\tau(\mathcal{X}, x, s)$. This correspondence induces a correspondence between $\mathcal{D}_d(\mathcal{X}, x, s)$ and $\mathcal{D}_d(\mathcal{Y}, y, s)$, via application of DEL, which preserves order and, again, between $\mathcal{D}_S(\mathcal{X}, x, s)$ and $\mathcal{D}_S(\mathcal{Y}, y, s)$, via application of the function **S**. In this last stage, the correspondence becomes injective because two adjacent members of the chain cannot be equivalent. On the other hand, there is an injection from $\mathcal{D}_S(\mathcal{Y}, y, s)$ to $\mathcal{D}_S(\mathcal{Y}, y', s)$, hence we have an injection from $\mathcal{D}_S(\mathcal{X}, x, s)$ to $\mathcal{D}_S(\mathcal{Y}, y', s)$. Corresponding trees must be isomorphic as required, because we are dealing with **S**-standard representatives.

At this point, we can say that $e(|x|_B)$ and $e(|y'|_B)$ are mapped to the same word by function DEL, with $e(|y'|_B)$ possibly longer (it can have more τ's). Using the symmetry in Definition 13, $|y'|_B$ will determinate a maximal $|x'|_B$ such that $e(|y'|_B)$ and $e(|x'|_B)$ will be mapped to the same word by DEL, with $e(|x'|_B)$ possibly longer. Let us now compare $|x|_B$ with $|x'|_B$: they are maximal elements of the same $\leq_B$-chain, hence they are equal. This implies that $e(|x'|_B) = e(|y'|_B)$ and the correspondence between paths in $\mathbf{W}\mathcal{X}$ and $\mathbf{W}\mathcal{Y}$ is a bijection preserving extent. A similar reasoning proves preservation of agreement. □

To prove the converse of Theorem 3, we need to prove that the correspondence which associates $\mathbf{W}\mathcal{X}$ with $\mathcal{X}$ is a FPWR *Tree*-functor.

Theorem 6. If two trees $\mathcal{X}$ and $\mathcal{Y}$ in $\mathbf{Beh}_\tau$ are weak bisimilar, then there is a FPWR *Tree*-functor $F : \mathbf{Beh}_\tau \to \mathbf{Beh}_\tau$ such that $F(\mathcal{X}) \cong F(\mathcal{Y})$.

Proof. The desired functor is defined as **W** on objects and Lemma 8(1) allows to define it on arrow-objects; FPWR derives from Lemma 8(2). □

Example 8. Consider the two trees in Figure 5: the rightmost (call it $\mathcal{X}$) is the standard weak representative of the leftmost (call it $\mathcal{Y}$), once we let $y' = |y|_W$ and $x' = |x_1|_W = |x_2|_W$. Indeed, the standard branching representative of $\mathcal{Y}$ is itself; moreover, the pruning defined in Definition 18 reduces it to $\mathcal{X}$, since $|x_2|_B$ is smaller (with respect to the ordering defined in Definition 17) than $|x_1|_B$. Also in this case, we can find a span of *Tree*-morphisms

$$\mathcal{Y} \xrightarrow{f} \mathcal{X} \xleftarrow{g} \mathcal{X}$$

where f has been intuitively described in the introduction (just before Figure 5) and g is the identity. However, f reflects paths with their factorizations only in the weaker way prescribed by Definition 6(3).

Proposition 14. Every FPWR *Tree*-functor $F : \mathbf{Beh}_\tau \to \mathbf{Beh}_\tau$ is such that $\mathbf{W}F \cong \mathbf{W}$.

Proof. Similar to the proof of Proposition 11. □

5. Applications

The theory developed so far does not need any assumption on the alphabet A. We now show that, by properly instantiating it, we can easily model LTS and labeled (prime) event structures. Moreover, we show that the notions of equivalences for the resulting trees do indeed correspond to standard equivalences already developed for the two models.

5.1. *A Tree-based Model for LTS*

Definition 19. A *labeled transition system* (or LTS) is a quadruple $(S, E, \rightarrow, s_0)$, where S is a set of *states*, ranged over by $s, u, \cdots$, E is a set of *actions*, for $E \subseteq A_\tau$, $\rightarrow \subseteq S \times E \times S$ is the *transition relation* and $s_0 \in S$ is the *starting state*.

As usual, we write $s \xrightarrow{\mu} s'$ rather than $(s, \mu, s') \in \rightarrow$. Moreover, if s, u, s', u' are states of $\mathcal{S}$, we write $\Longrightarrow$ for the reflexive and transitive closure of $\xrightarrow{\tau}$, $s \stackrel{\mu}{\Longrightarrow} u$ if there exist s', u' such that $s \Longrightarrow s' \xrightarrow{\mu} u' \Longrightarrow u$, and $s \stackrel{\hat{\mu}}{\Longrightarrow} u$ for $s \stackrel{\mu}{\Longrightarrow} u$, if $\mu \neq \tau$, and for $s \Longrightarrow u$, otherwise. We now introduce three well-known bisimulation-based relations.

Definition 20. Let $(S, E, \rightarrow, s_0)$ be an LTS. A symmetric relation $\Re \subseteq S \times S$ is a

1 *strong bisimulation* if $(s, u) \in \Re$ and $s \xrightarrow{\mu} s'$ imply that $u \xrightarrow{\mu} u'$ and $(s', u') \in \Re$, for some u';
2 *branching bisimulation* if $(s, u) \in \Re$ and $s \xrightarrow{\mu} s'$ imply that either $\mu = \tau$ and $(s', u) \in \Re$, or $u \Longrightarrow u_1 \xrightarrow{\mu} u_2 \Longrightarrow u_3$ for some u_1, u_2, u_3 such that $(s, u_1) \in \Re$, $(s', u_2) \in \Re$ and $(s', u_3) \in \Re$.
3 *weak bisimulation* if $(s, u) \in \Re$ and $s \xrightarrow{\mu} s'$ imply that $u \stackrel{\hat{\mu}}{\Longrightarrow} u'$ and $(s', u') \in \Re$, for some u'.

Two states are said to be *strongly/branching/weakly bisimilar* if there exists an eponymous bisimulation relating them. Two LTS are strongly/branching/weakly bisimilar if, when considering their disjoint union, their starting states are strongly/branching/weakly bisimilar. We write $\sim_S$, $\approx_B$ and $\approx_W$ for *strong*, *branching* and *weak* bisimilarity on LTS, respectively.

We now show that the equivalences that we have defined for general trees in Section 2 and Section 3 agree with the standard bisimulation-based equivalences defined in literature for labeled transition systems. To do this, we introduce the notion of unfolding of an LTS; then, we prove that the classical equivalences on LTS correspond to the ones we have defined, once we apply them to the trees obtained as unfoldings of LTS.

We define the *unfolding* of $\mathcal{S}$ to be $\text{UNF}(\mathcal{S}) = (\text{RUNS}(\mathcal{S}), e_{\mathcal{S}}, a_{\mathcal{S}})$, where:

— $\text{RUNS}(\mathcal{S}) = \{s_0\mu_1 s_1 \mu_2 \ldots \mu_n s_n \ : \ s_0 \xrightarrow{\mu_1} s_1 \xrightarrow{\mu_2} \cdots \xrightarrow{\mu_n} s_n\}$;
— $e_{\mathcal{S}}(s_0\mu_1 s_1 \ldots \mu_n s_n) = \mu_1 \ldots \mu_n$;
— $a_{\mathcal{S}}(s_0\mu_1 s_1 \ldots \mu_n s_n, s_0\nu_1 u_1 \ldots \nu_m u_m) = \mu_1 \ldots \mu_l$ where, for all $k \leq l$, it holds that $\nu_k = \mu_k$, $s_k = u_k$ and either $\mu_{l+1} \neq \nu_{l+1}$ or $s_{l+1} \neq u_{l+1}$ or $l = n$ or $l = m$.

By definition, $\text{UNF}(\mathcal{S})$ is a tree; it is obtained by taking all the *finite* approximations of all the possible sequences of actions that can be generated in the LTS. By construction, this implies that all the (possibly infinite) paths of $\text{UNF}(\mathcal{S})$ have finite length.

We now prove that this simple modeling of LTS via trees enjoys full abstraction w.r.t. strong, branching and weak bisimilarities. In the following proofs, we shall use notation $\mathcal{S}\downarrow_s$ to denote the LTS obtained from $\mathcal{S}$ by letting its initial state be s; moreover, we shall write $s \not\longrightarrow$ to mean that there exist no μ nor s' such that $s \xrightarrow{\mu} s'$.

Lemma 10. Let $x = s_0\mu_1 s_1 \ldots \mu_n s_n \in \text{RUNS}(\mathcal{S})$, then we have that $\mathcal{D}(\text{UNF}(\mathcal{S}), x, \mu_1 \ldots \mu_k) \cong \text{UNF}(\mathcal{S}\downarrow_{s_k})$ whenever either $k < n$, or $k = n$ and $s_n \not\longrightarrow$.

Proof. Let $k < n$, or $k = n$ and $s_n \not\longrightarrow$; we now prove that

$$D = \{s_0\mu_1 s_1 \ldots \mu_k s_k \bar{\mu}_1 \bar{s}_1 \ldots \bar{\mu}_m \bar{s}_m \ : \ s_k \bar{\mu}_1 \bar{s}_1 \ldots \bar{\mu}_m \bar{s}_m \in \text{RUNS}(\mathcal{S}\downarrow_{s_k})\}$$

is the set of paths in $\mathcal{D}(\text{UNF}(\mathcal{S}), x, \mu_1 \ldots \mu_k)$. This will easily allow us to conclude: we can easily define a bijection between D and the paths of $\text{UNF}(\mathcal{S}\downarrow_{s_k})$ such that, trivially, corresponding paths have the same extent and the same agreement with other paths in the same tree.

Let $y = s_0\hat{\mu}_1\hat{s}_1 \ldots \hat{\mu}_h\hat{s}_h$ be a path of $\mathcal{D}(\text{UNF}(\mathcal{S}), x, \mu_1 \ldots \mu_k)$; by definition, $a_\mathcal{S}(x, y) \geq \mu_1 \ldots \mu_k$ and, by construction of the unfolding, we can say that $h \geq k$ and, for every $i = 1 \ldots k$, we have that $\hat{\mu}_i = \mu_i$ and $\hat{s}_i = s_i$. Hence, $s_k\hat{\mu}_{k+1}\hat{s}_{k+1} \ldots \hat{\mu}_h\hat{s}_h \in \text{RUNS}(\mathcal{S}\downarrow_{s_k})$, that implies $y \in D$, as desired.

Vice versa, let $y = s_0\mu_1 s_1 \ldots \mu_k s_k \bar{\mu}_1 \bar{s}_1 \ldots \bar{\mu}_m \bar{s}_m \in D$; by definition, $s_k \bar{\mu}_1 \bar{s}_1 \ldots \bar{\mu}_m \bar{s}_m \in \text{RUNS}(\mathcal{S}\downarrow_{s_k})$ and so $y \in \text{RUNS}(\mathcal{S})$. Now, we can trivially conclude that y is a path of $\mathcal{D}(\text{UNF}(\mathcal{S}), x, \mu_1 \ldots \mu_k)$, since $a_\mathcal{S}(x, y) \geq \mu_1 \ldots \mu_k$. □

Proposition 15. Let $\mathcal{S}$ and $\mathcal{S}'$ be two transition systems. Then, $\mathcal{S} \sim_S \mathcal{S}'$ if and only if $\text{UNF}(\mathcal{S}) \simeq_S \text{UNF}(\mathcal{S}')$.

Proof. For the "if" part, we have to prove that the relation $\Re = \{(\mathcal{S}, \mathcal{S}') \ : \ \text{UNF}(\mathcal{S}) \simeq_S \text{UNF}(\mathcal{S}')\}$ is a strong bisimulation on LTS. Consider a step $s_0 \xrightarrow{\mu} s_1$ in $\mathcal{S}$. If $s_1 \not\longrightarrow$, we let x be $s_0\mu s_1$; otherwise, we let x be $s_0\mu s_1\bar{\mu}\bar{s}$, for any $\bar{\mu}$ and $\bar{s}$ such that $s_1 \xrightarrow{\bar{\mu}} \bar{s}$ (at least one exists, since it does not hold that $s_1 \not\longrightarrow$). By strong bisimulation on trees, there exists a path $y \in \text{RUNS}(\mathcal{S}')$ whose extent starts with μ such that $\mathcal{D}(\text{UNF}(\mathcal{S}), x, \mu) \simeq_S \mathcal{D}(\text{UNF}(\mathcal{S}'), y, \mu)$. Let, $y = s'_0\mu s'_1 \cdots$; since x and y are related by a bisimulation on trees, it is easy to prove that $s'_1 \not\longrightarrow$ if and only if $s_1 \not\longrightarrow$, so both x and y satisfy the premises of Lemma 10. This easily allows us to conclude: by Lemma 10, $\text{UNF}(\mathcal{S}\downarrow_{s_1}) \cong \mathcal{D}(\text{UNF}(\mathcal{S}), x, \mu) \simeq_S \mathcal{D}(\text{UNF}(\mathcal{S}), x, \mu) \cong \text{UNF}(\mathcal{S}'\downarrow_{s'_1})$ and, by definition, $(\mathcal{S}\downarrow_{s_1}, \ \mathcal{S}'\downarrow_{s'_1}) \in \Re$.

For the converse, we have to prove that the relation $\Re = \{(\text{UNF}(\mathcal{S}), \text{UNF}(\mathcal{S}')) \ : \ \mathcal{S} \sim_S \mathcal{S}'\}$ is a strong bisimulation on trees. Pick up a $x \in \text{RUNS}(\mathcal{S})$, for $x = s_0\mu_1 s_1 \cdots \mu_n s_n$; there exist $s'_1, \ldots, s'_n$ such that $s'_0 \xrightarrow{\mu_1} s'_1 \cdots \xrightarrow{\mu_n} s'_n$ and $s_i \sim_S s'_i$, for every i. Thus, $y = s'_0\mu_1 s'_1 \cdots \mu_n s'_n \in \text{RUNS}(\mathcal{S}')$; moreover, y has the same extent as x and $s_n \not\longrightarrow$ if and only if $s'_n \not\longrightarrow$. Thus, by Lemma 10, the derivatives along x are all in relation $\Re$ with the corresponding derivatives along y; this suffices to conclude. □

Lemma 11.

1 Let $\mathcal{X}_0, \mathcal{X}_1, \cdots, \mathcal{X}_m$ and $\mathcal{Y}_0, \mathcal{Y}_1, \cdots, \mathcal{Y}_n$ be two chains of τ-derivatives; if $\mathcal{X}_i \simeq_B \mathcal{Y}_j$, $\mathcal{X}_{i+1} \simeq_B \mathcal{Y}_k$ and $\mathcal{Y}_j \not\simeq_B \mathcal{Y}_k$ then $k > j$.

2 If $\mathcal{X} \simeq_B \mathcal{Y}$ then, for every $x \in X$ and $s \preccurlyeq e_X(x)$, the maximum element of $\mathcal{D}_\tau(\mathcal{X}, x, s)$ is branching bisimilar to the maximum element of $\mathcal{D}_\tau(\mathcal{Y}, y, s)$, where y is the path of $\mathcal{Y}$ corresponding to x in the given branching bisimulation between $\mathcal{X}$ and $\mathcal{Y}$.

Proof. For the first claim, reason by contradiction and assume $k < j$. This means that $\mathcal{Y}_j$ is a τ-derivative of $\mathcal{Y}_k$; since $\mathcal{Y}_j \not\simeq_B \mathcal{Y}_k$, there exists a $\bar{y}$ in $\mathcal{Y}_k$ that cannot be simulable in $\mathcal{Y}_j$. However, $\mathcal{X}_{i+1} \simeq_B \mathcal{Y}_k$ and, hence, there exists a $\bar{x}$ in $\mathcal{X}_{i+1}$ that simulates $\bar{y}$; but $\bar{x}$ is also a path of $\mathcal{X}_i$. Since $\mathcal{X}_i \simeq_B \mathcal{Y}_j$, there exists a $\bar{y}'$ in $\mathcal{Y}_j$ that simulates $\bar{x}$; by transitivity, $\bar{y}'$ also simulates $\bar{y}$, contradiction.

For the second claim, we know that, since $\mathcal{X} \simeq_B \mathcal{Y}$, every element of $\mathcal{D}_\tau(\mathcal{X}, x, s)$ (that is a chain of τ-derivatives $\mathcal{X}_0, \mathcal{X}_1, \cdots, \mathcal{X}_m$) is branching bisimilar an element of $\mathcal{D}_\tau(\mathcal{Y}, y, s)$ (that is a chain of τ-derivatives $\mathcal{Y}_0, \mathcal{Y}_1, \cdots, \mathcal{Y}_n$). We have to prove that $\mathcal{X}_0 \simeq_B \mathcal{Y}_0$. Let $\mathcal{X}_0 \simeq_B \mathcal{Y}_h$; if $h = 0$, we have done. Otherwise, by the first claim of this Lemma, $\mathcal{Y}_0, \ldots, \mathcal{Y}_{h-1}$ must all be branching bisimilar to an element of $\mathcal{D}_\tau(\mathcal{X}, x, s)$ branching bisimilar to $\mathcal{X}_0$. By transitivity, we easily conclude. □

Proposition 16. Let $\mathcal{S}$ and $\mathcal{S}'$ be two transition systems. Then

1 $\mathcal{S} \approx_B \mathcal{S}'$ if and only if $\textsc{unf}(\mathcal{S}) \simeq_B \textsc{unf}(\mathcal{S}')$;
2 $\mathcal{S} \approx_W \mathcal{S}'$ if and only if $\textsc{unf}(\mathcal{S}) \simeq_W \textsc{unf}(\mathcal{S}')$.

Proof. We only prove the first claim, that is more complex. For the "if" part, we have to prove that the relation $\Re = \{(\mathcal{S}, \mathcal{S}') : \textsc{unf}(\mathcal{S}) \simeq_B \textsc{unf}(\mathcal{S}')\}$ is a branching bisimulation on LTS. Consider a step $s_0 \xrightarrow{\mu} s_1$ in $\mathcal{S}$; let x be $s_0 \mu s_1$, if $s_1 \not\longrightarrow$, and be $s_0 \mu s_1 \bar{\mu} \bar{s}$ (for any $\bar{\mu}$ and $\bar{s}$ such that $s_1 \xrightarrow{\bar{\mu}} \bar{s}$), otherwise; let also $\mathcal{X}$ be $\mathcal{D}(\textsc{unf}(\mathcal{S}), x, \mu)$ that, by Lemma 10, is isomorphic to $\textsc{unf}(\mathcal{S}\downarrow_{s_1})$.

— If $\mu \neq \tau$, we have that $\mathcal{X} \in \mathcal{D}_\tau(\textsc{unf}(\mathcal{S}), x, \mu)$; by hypothesis, there exists a y in $\textsc{unf}(\mathcal{S}')$ such that $e_{\mathcal{S}'}(y) = \tau^m \mu \tau^n w$ and $\mathcal{X} \simeq_B \mathcal{Y}$, for some $\mathcal{Y} \in \mathcal{D}_\tau(\textsc{unf}(\mathcal{S}'), y, \mu)$. Since x and y are put in correspondence by a branching bisimulation, it is easy to prove that $s_1 \not\longrightarrow$ if and only if $u_3 \not\longrightarrow$, where $s'_0(\xrightarrow{\tau})^m \xrightarrow{\mu} (\xrightarrow{\tau})^n\ u_3$ and s'_0 is the starting state of $\mathcal{S}'$; this allows us to freely apply Lemma 10 in what follows. Let $s'_0(\xrightarrow{\tau})^m\ u_1 \xrightarrow{\mu} u_2(\xrightarrow{\tau})^n\ u_3$; by Lemma 10, $\mathcal{Y}_1 = \mathcal{D}(\textsc{unf}(\mathcal{S}'), y, \tau^m) \cong \textsc{unf}(\mathcal{S}' \downarrow_{u_1})$, $\mathcal{Y}_2 = \mathcal{D}(\textsc{unf}(\mathcal{S}'), y, \tau^m \mu) \cong \textsc{unf}(\mathcal{S}' \downarrow_{u_2})$ and $\mathcal{Y} = \mathcal{D}(\textsc{unf}(\mathcal{S}'), y, \tau^m \mu \tau^n) \cong \textsc{unf}(\mathcal{S}' \downarrow_{u_3})$. Since $\mathcal{X} \simeq_B \mathcal{Y}$, we have that $(\mathcal{S}\downarrow_{s_1}, \mathcal{S}' \downarrow_{u_3}) \in \Re$. By Definition 12.1(b), $\textsc{unf}(\mathcal{S}) \simeq_B \mathcal{Y}_1$ (indeed, the only tree in $\mathcal{D}_\tau(\textsc{unf}(\mathcal{S}), x, \epsilon)$ is $\textsc{unf}(\mathcal{S})$, since $\mu \neq \tau$); this implies that $(\mathcal{S}, \mathcal{S}' \downarrow_{u_1}) \in \Re$. To conclude, observe that $\mathcal{X}$ and $\mathcal{Y}_2$ are the maximum elements in $\mathcal{D}_\tau(\textsc{unf}(\mathcal{S}), x, \mu)$ and $\mathcal{D}_\tau(\textsc{unf}(\mathcal{S}'), y, \mu)$, respectively; thus, by Lemma 11(2), $(\mathcal{S}\downarrow_{s_1}, \mathcal{S}' \downarrow_{u_2}) \in \Re$. This suffices to conclude.
— If $\mu = \tau$, we have that $\mathcal{X} \in \mathcal{D}_\tau(\textsc{unf}(\mathcal{S}), x, \epsilon)$; by hypothesis, there exists a $y \in \textsc{runs}(\mathcal{S}')$ such that $\mathcal{X} \simeq_B \mathcal{Y}$, for some $\mathcal{Y} \in \mathcal{D}_\tau(\textsc{unf}(\mathcal{S}'), y, \epsilon)$. If $\mathcal{Y} = \textsc{unf}(\mathcal{S}')$, we then conclude that $(\mathcal{S}\downarrow_{s_1}, \mathcal{S}') \in \Re$. Otherwise, $\mathcal{Y}$ is a τ-derivative of $\textsc{unf}(\mathcal{S}')$ and, so, we can find $\mathcal{Y}_1$ and $\mathcal{Y}_2$ in $\mathcal{D}_\tau(\textsc{unf}(\mathcal{S}'), y, \epsilon)$ such that $\textsc{unf}(\mathcal{S}')$, $\mathcal{Y}_1$, $\mathcal{Y}_2$ and $\mathcal{Y}$ appear in this order in the chain associated to $\mathcal{D}_\tau(\textsc{unf}(\mathcal{S}'), y, \epsilon)$. By Lemma 10, we have that $\mathcal{Y}_1 \cong \textsc{unf}(\mathcal{S}' \downarrow_{u_1})$, $\mathcal{Y}_2 \cong \textsc{unf}(\mathcal{S}' \downarrow_{u_2})$ and $\mathcal{Y} \cong \textsc{unf}(\mathcal{S}' \downarrow_{u_3})$, for $s'_0 \Longrightarrow u_1 \xrightarrow{\mu} u_2 \Longrightarrow u_3$. Since $\mathcal{X} \simeq_B \mathcal{Y}$, we have that $(\mathcal{S}\downarrow_{s_1}, \mathcal{S}' \downarrow_{u_3}) \in \Re$. Since $\textsc{unf}(\mathcal{S}') \simeq_B \textsc{unf}(\mathcal{S})$ and $\mathcal{Y} \simeq_B \mathcal{X}$, by virtue of Lemma 11(1), we have that the elements in $\mathcal{D}_\tau(\textsc{unf}(\mathcal{S}), x, \epsilon)$ branching bisimilar to $\mathcal{Y}_1$ and $\mathcal{Y}_2$ can only be (branching bisimilar to) $\textsc{unf}(\mathcal{S})$ and $\mathcal{X}$, respectively; thus, $(\mathcal{S}, \mathcal{S}' \downarrow_{u_1})$ and $(\mathcal{S}\downarrow_{s_1}, \mathcal{S}' \downarrow_{u_2})$ are both in $\Re$, and this suffices to conclude.

For the "only if" part, we have to prove that the relation $\Re = \{(\textsc{unf}(\mathcal{S}), \textsc{unf}(\mathcal{S}')) : \mathcal{S} \approx_B \mathcal{S}'\}$ is a branching bisimulation on trees. Choose $x = s_0 \mu_1 s_1 \cdots \mu_n s_n \in \textsc{runs}(\mathcal{S})$. From the hypothesis, we know that we can find a path $y = u_0^1 \mu_1^1 u_1^1 \cdots \mu_1^{k_1} u_1^{k_1} \mu_2^1 u_2^1 \cdots \mu_2^{k_2} u_2^{k_2} \cdots \mu_n^1 u_n^1 \cdots \mu_n^{k_n} u_n^{k_n} \in \textsc{runs}(\mathcal{S}')$ such that $u_0^1 = s'_0$ and

— either $\mu_i = \tau$, $k_i = 0$ and $s_i \approx_B u_{i-1}^{k_{i-1}}$,

— or $k_i = k'_i + 1 + k''_i$, $\mu_i^j = \tau$, for every $j \in \{1, \ldots, k'_i, k'_i + 2, \ldots, k_i\}$, and $\mu_i^{k'_i+1} = \mu_i$; moreover, $s_{i-1} \approx_B u_i^{k'_i}$, $s_i \approx_B u_i^{k'_i+1}$ and $s_i \approx_B u_i^{k_i}$.

By construction, $\text{DEL}(e_{\mathcal{S}}(x)) = \text{DEL}(e_{\mathcal{S}'}(y))$; moreover, it is not difficult to prove that, for every $w \preccurlyeq \mu_1 \cdots \mu_n$ and for every $\mathcal{X} \in \mathcal{D}_\tau(\text{UNF}(\mathcal{S}), x, w)$, there exists a $\mathcal{Y} \in \mathcal{D}_\tau(\text{UNF}(\mathcal{S}'), y, w)$ such that $(\mathcal{X}, \mathcal{Y}) \in \Re$, and vice versa. This suffices to conclude. □

5.2. *Labeled event structures*

We will show now that our machinery can provide a model for labeled event structures (LES), equipped with a pomset-bisimulation relation. It has to be said, that other bisimulations can be dealt with similarly.

Definition 21.

1 A *labeled (prime) event structure* is a tuple $\mathcal{E} = (E, \prec, \#, L, \ell)$ where
   (a) E is a set of so-called *events* taken from a universe Ev,
   (b) $\prec$ and $\#$ are two binary relations on E, that are called *causality* and *conflict*, and are such that $\prec \cap \# = \emptyset$,
   (c) L is an *alphabet* and $\ell : E \to L$ is a *labelling function*,
   (d) $\prec$ is a partial order such that for, every $e \in E$, the set $\{e' | e' \prec e\}$ is finite,
   (e) $\#$ is symmetric, irreflexive and enjoys conflict heredity: if $e \# e'$ and $e' \prec e''$ then $e \# e''$.

2 An event structure is said to be *conflict free* if $\# = \emptyset$.

3 A *configuration* in $\mathcal{E}$ is a finite, conflict free and downwards closed subset of E.

4 Two configurations $\chi = (X, \prec, L, \ell)$ and $\chi' = (X', \prec', L, \ell')$ are *isomorphic* if there is an order-preserving, labeling-preserving bijection between X and X'.

5 A *pomset* is an isomorphism class of finite configurations. $Pom_{Ev,L}$ denotes the set of pomsets on the set of events Ev, labeled by L.

A prime event structure represents a concurrent system in the following way: action names $a \in L$ represent the actions the system might have performed, an event $e \in E$ labeled with a represents an occurrence of a during a possible run of the system, $d \prec e$ means that d is a prerequisite for e, and $d \# e$ means that d and e cannot have occurred in the same run.

The behaviour of a prime event structure is described by means of configurations, that explain which subsets of events constitute possible (partial) runs of the represented system. Thus, configurations must be conflict-free; furthermore, they must be closed with respect to causal predecessors: all prerequisites for any event occurring in the run must also have occurred.

It is well known (see, e.g., (Winskel and Nielsen 1995; Roggenbach and Majster-Cederbaum 2000)) that we can consider a (prime) labeled event structure as a transition system, labeled by pomsets, where states are configurations and the label on the transition relation denotes the partially ordered multiset of moves induced by the events that occurred in going from the first configuration to the second one. Notice that this intuitive interpretation is somehow dual w.r.t. the one given for processes: there, when going from the first to the second one by performing an action, this action (and possibly many others) are erased from the possible behavior of the second one. Here, in some sense, we suppose to find in the second one the first one and the record of what has been performed.

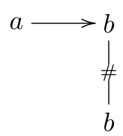
Definition 22. A (prime) labeled event structure $\mathcal{E} = (E, \prec, \#, l)$, generates the pomset-labeled transition system $T_{Pom}(\mathcal{E}) = (Conf(\mathcal{E}), Pom_{Ev,L}, \rightarrow, \emptyset)$, where, for every $\chi = (X, \prec, L, \ell)$ and $\chi' = (X', \prec', L, \ell')$ in $Conf(\mathcal{E})$, we let $\chi \xrightarrow{p} \chi'$ iff $X \subset X'$ and $p \in Pom_{Ev,L}$ is the isomorphism class of $X' \setminus X$ as poset.

Definition 23 ((van Glabbeeck and Goltz 1990)). Two (prime) labeled event structures $\mathcal{E} = (E, \prec, \#, l)$ and $\mathcal{F} = (F, \prec, \#, l)$, are pomset bisimilar if there is a relation $R \subset E \times F$ s.t.: for every $(\chi, \zeta) \in R$ and $p \in Pom_{Ev,L}$ we have:

— (i) if $\chi \in Conf(\mathcal{E})$, $\chi \xrightarrow{p} \chi'$, then there is $\zeta' \in Conf(\mathcal{F})$ such that $\zeta \xrightarrow{p} \zeta'$, $(\chi', \zeta') \in R$, and
— (ii) if $\zeta \in Conf(\mathcal{F})$, $\zeta \xrightarrow{p} \zeta'$, then there is $\chi' \in Conf(\mathcal{E})$ such that $\chi \xrightarrow{p} \chi'$, $(\chi', \zeta') \in R$.

Once we have $T_{Pom}(\mathcal{E})$, we can unfold it like in Section 5.1 and obtain a tree whose paths are labeled with sequences of pomsets. Thus, in our framework, we can consider the free monoid $(Pom_{Ev,L})^*$ and carry our construction as usual, by describing the behaviour of an event structure as a pomset-labeled tree. Pomset-bisimulation will coincide with strong bisimulation on our model (this result can be proved like Proposition 15) and a minimal representative can be obtained. We illustrate the procedure by an example.

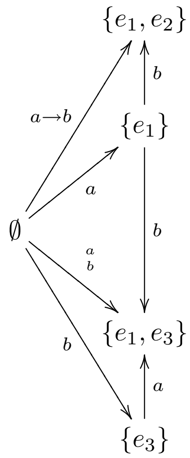
Example 9. Consider the event structure $\mathcal{E}$:



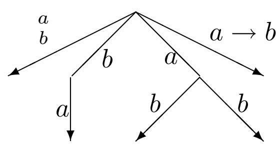
where '$\longrightarrow$' denotes causality and '—#—' denotes conflict. Formally, $\mathcal{E}$ is defined as the 5-tuple $(E, \prec, \#, L, \ell)$, where:

— $E = \{e_1, e_2, e_3\}$;
— $e_1 \prec e_2$;
— $e_2 \# e_3$ and $e_3 \# e_2$;
— $L = \{a, b\}$;
— $\ell(e_1) = a$ and $\ell(e_2) = \ell(e_3) = b$.

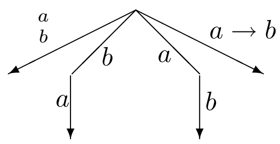
The configurations of $\mathcal{E}$ are $\emptyset$, $\{e_1\}$, $\{e_3\}$, $\{e_1, e_2\}$ and $\{e_1, e_3\}$; they correspond to the pomsets $\emptyset$, a, b, $a \rightarrow b$ and $\overset{a}{b}$. Thus, we have that $T_{Pom}(\mathcal{E})$ is the LTS:



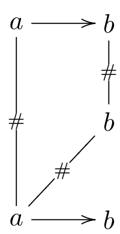
The unfolding of this LTS is the pomset-labeled tree:



whose standard strong representative is the tree $\mathcal{X}$:



Now, consider the event structure $\mathcal{F}$:

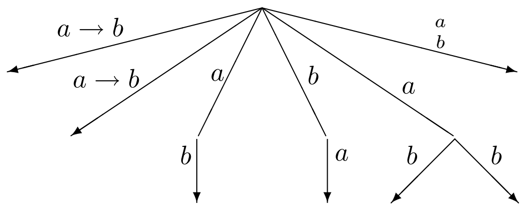


Formally, $\mathcal{F}$ is defined as the 5-tuple $(F, \prec', \#', L, \ell')$, where:

— $F = \{f_1, f_2, f_3, f_4, f_5\}$;
— $f_1 \prec' f_2$ and $f_4 \prec' f_5$;

— $f_1 \#' f_4$, $f_1 \#' f_5$, $f_2 \#' f_3$, $f_2 \#' f_4$, $f_2 \#' f_5$, $f_3 \#' f_4$ and $f_3 \#' f_5$ (plus their symmetric versions);
— $L = \{a, b\}$;
— $\ell'(f_1) = \ell'(f_3) = a$ and $\ell'(f_2) = \ell'(f_4) = \ell'(f_5) = b$.

The configurations of $\mathcal{F}$ are $\emptyset$, $\{f_1\}$, $\{f_3\}$, $\{f_4\}$, $\{f_1, f_2\}$, $\{f_1, f_3\}$ and $\{f_4, f_5\}$; they correspond to the pomsets $\emptyset$, a, b, $a \to b$ and $\begin{smallmatrix} a \\ b \end{smallmatrix}$. We leave to the reader the construction of $T_{Pom}(\mathcal{E})$ and just provide its unfolding, i.e., the tree



Again, its standard strong representative is the tree $\mathcal{X}$.

6. Related works

Let us now compare our approach with other mathematical models for bisimulation. First of all, we have to say (borrowing the terminology from category theory) that we take a "local" automata-theoretic approach, by modeling and comparing processes in terms of their the local behavior: to each ordered pair of states of a given process, we associate the structured set of transitions leading from the first state to the second one. To the best of our knowledge, all the other approaches modeling bisimulations consider comparisons between "global" process behaviors: to each state, it is associated the structured set of possible evolutions into all the other states. Another distinguishing feature of our approach is that we aim at classifying bisimulations; hence, we use different definitions for them on the same model. In all the other approaches, the definition of bisimulation is the same and the basic model is modified to capture the different kinds of bisimulation.

For more detailed comparisons, we start by considering the categorical characterization of bisimulation-based equivalences as *open maps* (Joyal et al. 1993). In that approach, *behaviours* are considered as objects of a given category **M**, *maps* are (partial) simulations, and *paths within behaviours* are morphisms from objects in a given subcategory **P**, representing the shape of their paths. The relation among objects in **M** is expressed in terms of the existence of a *span* of maps between them, open with respect to **P**: this means that, if a path can be extended in the image behavior, then it can be extended in the domain in a similar way.

In our approach, the base locally posetal 2-category associated with the free monoid of labels **A** (see Remark 1) plays the role of **P** and local behaviours are canonically defined over it as symmetric **A**-categories. Therefore, we cannot define morphisms arbitrarily; they must be **A**-functors. As a consequence, our morphisms, or simulations, map a path into one of identical length. Our notion of path reflection required for *Tree*-functors led us to prove Lemmata 4, 6 and 8 that amount to requiring a kind of "openness property"; a possible formulation of the latter property (see (Joyal et al. 1993)) requires that, for a given open morphism $\sigma : T \to T'$ between LTSs, it holds that:

$$\text{if } \sigma(s) \xrightarrow{a} s' \text{ in } T' \text{ then } s \xrightarrow{a} u \text{ in } T \text{ and } \sigma(u) = s' \text{ for some } u \text{ of } T.$$

In particular, the second part of our lemmata states that, if the image (through **S**, **B**, **W**) of a tree $\mathcal{X}$ can perform an action along a path to go into another tree $\mathcal{Y}$, than the original object can perform the same action and go into a tree $\mathcal{Z}$ in the inverse image of $\mathcal{Y}$.

By changing the model, the openness condition, that indeed captures the intuition underlying strong bisimulation, is then used to model other equivalences (Cheng and Nielsen 1995). On the contrary, we model the three kinds of bisimulations by keeping the same model and imposing further requirements to the path factorization property to be satisfied by the Tree-functors.

Another possible modelization of bisimulations has been proposed by relying on coalgebras (see, e.g., (Rutten 2000; Roggenbach and Majster-Cederbaum 2000)). Now labeled transition systems are modeled as coalgebras for a suitable endofunctor. A bisimulation between two given coalgebras is a subcoalgebra of the product of the two. Hence, using the projections, one can get a span of coalgebra homomorphisms from the bisimulation into the two given coalgebras. The notion of homomorphism for coalgebra is strictly related to the notion of open map, when the used simulations are total functions. The relationships between the open maps and the coalgebraic approach are carefully studied in (Lasota 2002) where it is shown that open maps correspond to coalgebra morphisms for a suitably chosen endofunctor in a category of many-sorted sets.

In (Pavlović 1995; Pavlović 1996) the basic intuitions underlying open maps and coalgebras are exploited to obtain standard representatives for strong, weak and branching bisimulations. (Pavlović 1995) starts from automata (in a global approach), and studies how to forget their "geometry" to ignore redundant states. A category, isomorphic to a subcategory of transition systems, is defined, where morphisms are simulations (spans of sober and saturated arrows) between processes. Two transition systems are (strongly) bisimilar if there exists a span of bisimulation morphisms between them, where these morphisms are a kind of open maps. Equivalently, the two systems are bisimilar if there exists a cospan of bisimulation morphisms between them, i.e., a common quotient. It is also shown that every automaton has an irredundant quotient and a couniversal quotient is provided for each process by exploiting coalgebraic methods. Branching and weak bisimilarity are considered in (Pavlović 1996). Like for strong bisimilarity, the functor inducing the quotient is the initial one and enjoys the property of reducing to equality the relations between objects induced by spans. Standard representatives are obtained for each class by a coalgebraic definition, either in the original category or in a subcategory obtained by suitably "repleting" the given transition systems. The former holds for strong bisimulation and we have that the representatives are the minimal ones; the latter holds for branching and weak bisimulation, and representatives are not always minimal.

The coalgebraic and the open-maps approaches are based on the fact that, in a cartesian category, a relation $R \rightarrowtail X \times Y$ between two objects of a category $\mathcal{C}$ can be represented by a pair of maps $X \leftarrow R \rightarrow Y$, i.e. a span, obtained by composing the inclusion morphism with the projections. However, if the considered category has appropriate properties, one can introduce an equivalence relation $\mathcal{R} \rightarrowtail \mathcal{C} \times \mathcal{C}$ and consider the quotient $\mathcal{C}/\mathcal{R}$. This was done in (Pavlović 1995; Pavlović 1996). But one can do even better: if a kind of "homomorphism theorem" holds, it can be proved that the quotient $\mathcal{C}/\mathcal{R}$ is induced by an endofunctor identifying equivalent objects.

This last intuition is the one underlying our approach. Our maps (Tree-functors F) induce a quotient over the class of all the objects and produce a cospan of tree morphisms between "related" trees $\mathcal{X} \rightarrow F(\mathcal{X}) \cong F(\mathcal{Y}) \leftarrow \mathcal{Y}$. By putting a suitable lifting factorisation condition on F, we guarantee that this cospan, that locally inherits the suitable lifting factorization property, relates equivalent trees. By exhibiting a standard representative for strong, branching and weak equivalence classes, we can define the corresponding functor and guarantee that a terminal relation, that quotients as much as possible, does exist for every class. The representatives corresponding to the terminal relation in their class (i.e., the Tree-functors $\mathbf{S}$, $\mathbf{B}$ and $\mathbf{W}$) are terminal among those enjoying the required properties and the standard representatives are the "smallest" in the three cases. This is possible because we do not need to replete representatives with extra paths, but have only to select existing ones. Our standard representatives, at least in the case of strong and branching bisimulation, also enjoy a very nice property: due to the path factorization lifting property of $\mathbf{S}$ and $\mathbf{B}$, they can replace their originals in every universal and couniversal construction (Kasangian and Labella 2009).

Another feature of our approach is that our trees can be seen as presentations of sheaves; hence, we are closely related to the (pre)sheaf approach of (Cattani and Winskel 1997). Sheaf models for processes have also been presented in (Bunge and Fiore 2000), where processes are seen as categories of states equipped with a control functor on a category of paths, over which the *Unique Factorization Lifting* (ULF) property is imposed; they then characterize bisimulations as open maps. Our trees are categories enriched over a 2-category associated with a semilattice (representing elementary paths); for this reason, they can be thought of as a presentation of sheaves (Walters 1981) and thus are more abstract, but simpler, than sheaves (because they lack restrictions). On the other hand, our Tree-categories $\mathbf{Beh}$ and $\mathbf{Beh}_\tau$ enjoy a sort of UFL property, in the sense that their unique Tree-functor to the terminal Tree-category $\mathbf{T}$ enjoys path factorization lifting (see Proposition 4). Actually, UFL holds in the case of $\mathbf{Beh}$, and a more relaxed variants holds for $\mathbf{Beh}_\tau$.

In our view, the use of the same concept of lifting factorization both for the construction of models (as in the case of Bunge and Fiore) and for defining equivalences (specific of our case) opens interesting directions for future work.

7. Conclusions

We have presented a characterization of strong, branching and weak bisimulation on a tree-based model via enriched functors enjoying a factorized path reflection property. Our machinery is able to classify bisimulations according to the properties of the functor defining them, for different kinds of concurrency. This is mainly due to the change of base machinery, a very natural tool to go from one model to another one, once the local point of view is assumed.

The current work can be seen as the continuation of the line of research started in (De Nicola and Labella 1998); however, in this paper we have:

— considered infinite (regular) structures;

— provided a functorial characterization also for weak bisimilarity;

— simplified and clarified the characterization of branching bisimilarity.

Our results are summarized in the following Table, that for each of the three considered equivalences, lists the properties that have to enjoyed by the functor needed to capture them.

	PR	FPWR	FPR	UFL
$\simeq_S$	×	×	×	×
$\simeq_B$	×	×	×	
$\simeq_W$	×	×		

The abbreviations used in the table correspond to the following:

— PR is the path reflection property, requiring that every target path comes from some source path, and it corresponds to the openness property by (Joyal et al. 1993);

— FPWR is the factorized path weakly reflection property, requiring that, to each factorization of a target path, there corresponds a factorization of *one* of the associated source paths;

— FPR is the factorized path reflection property, requiring PR and that all the intermediate states of each factorization have a corresponding state path in *all* the associated source paths;

— UFL is the *unique factorization lifting* property, that corresponds to FPR plus the requirement that the reflected factorization is unique up-to isomorphism. It is worth noticing that FPR collapses on UFL when all actions are visible.

The requirement that every possible factorization of a target path is reflected in an appropriate factorization of its source path corresponds to a Conduché property (Conduché 1972) for enriched functors. A very rigid formulation of it has been used in (Lawvere 1986) to characterize determinacy of physical systems and to explain determinacy on states: in his view, a physical transformation is deterministic if it can be "controlled" by a Conduché functor[||], in the sense that it is always possible to recover the sequence of states traversed during system evolution.

In 'ordinary' (i.e., not enriched) categories, Conduché property characterizes those functors that have a right adjoint to the inverse image functor associated with them (Conduché 1972). In our setting, existence of a right adjoint (proved in (Kasangian and Labella 2009)) ensures that the inverse image functor associated with an equivalence preserves colimits, as well as limits. Because of this, we could operate on standard representatives and pull the constructions back on ordinary behaviours while preserving universality.

If we take Lawvere's standpoint, we can evaluate our functors according to the notion of "state determinacy preservation"; in this way, we can set up another criterion to assess the three bisimulation equivalences we have considered. Thus, we have that weak bisimilarity partially destroys the notion of state, because it forgets relevant information about the different states a system has gone through, while the other two equivalences keep track of them. In fact, we have that weak bisimilarity only enjoys a weaker form of lifting factorization property, that strong bisimilarity enjoys uniqueness of lifted factorizations (it faithfully keeps track of every state traversed along a path), and that branching bisimilarity enjoys a lifting factorization property that permits forgetting 'useless' states. In some sense, we could say that we have a new mathematical justification for Milner's claim (Milner 1989) that weak bisimilarity does not preserve strong determinacy of

[||] Actually, Lawvere deals with the UFL property, that he names *Moebius property*.

processes, whereas strong bisimilarity does. Moreover, we have also formally detected a close correspondence between strong and branching bisimilarity in terms of preservation of systems determinacy.

References

M.A. Arbib and E.G. Manes. Machines in a category: an expository introduction. *SIAM Review*, 16(2):163–192, 1974.

R. Betti, and S. Kasangian. A quasi-universal realization of automata. *Rend. Ist. Mat. Univ. Trieste*, 14:41-48, 1982.

M. Bunge and M.P. Fiore. Unique factorization lifting functors and categories of linearly-controlled processes. *Mathem. Structures in Comp. Sci.*, 10(2):137–163, 2000.

G.L. Cattani and G. Winskel. Preshaf models for concurrency. *Proc. of CSL '96*, volume 1258 of *LNCS*, pages 106–126. Springer, 1997.

A. Cheng and M. Nielsen. Open Maps (at) Work. *BRICS Report Series*, n. 23, 1995.

F. Conduché. Au sujet de l'existence d'adjoints à droîte aux foncteurs "image reciproque" dans la catégorie des catégories. *C.R.Acad. Sci. Paris*, 275 (1972), A891–894.

R. De Nicola, U. Montanari and F. Vaandrager. Back and forth bisimulations. In *Proc. of CONCUR'90*, volume 458 of *LNCS*, pages 152–165. Springer, 1990.

R. De Nicola and A. Labella. Tree Morphisms and Bisimulations. In *Proc. of MFCS'98 Workshop on Concurrency*, volume 18 of *ENTCS*. Elsevier, 1998.

R. van Glabbeeck and U. Goltz, Equivalences and refinement. In *Semantics of Systems of Concurrent Processes*, volume 469 in *LNCS*, pages 309-333. Springer, 1990.

R. van Glabbeek and W. Weijland. Branching Time and Abstraction in Bisimulation Semantics. *Journal of the ACM*, 43(3):555–600, 1996.

A. Joyal, M. Nielsen and G. Winskel. Bisimulations and Open Maps. In *Proc. of LICS*, pages 418–427. IEEE, 1993.

S. Kasangian and A. Labella. Observational trees as models for concurrency. *Math. Struct. in Comp. Sci.*, 9: 687–718, 1999.

S. Kasangian and A. Labella. Conduché property and Tree-based categories. To appear in the *Journal of Pure and Applied Algebra*.

G. Kelly. *Basic Concepts of Enriched Category Theory*. Cambridge University Press, 1982.

S. Lasota. Coalgebra morphisms subsume open maps. *Theoretical Computer Science* 280(2002) 123-135.

F.W. Lawvere. State categories and response functors. Unpublished manuscript, 1986.

R. Milner. *Communication and concurrency*. Prentice Hall International, 1989.

D. Park. Concurrency and automata on infinite sequences. In *Proc. of Theoretical Computer Science*, volume 104 of *LNCS*, pages 167–183. Springer, 1981.

D. Pavlović. Convenient Category of Processes and Simulations 1: Modulo Strong Bisimilarity. In Proc. of Category Theory and Computer Science, publisher = Springer, volume 953 in *LNCS*, pages 3–23. Springer, 1995.

D. Pavlović. Convenient categories of asynchronous processes and simulations II, in Theory and Formal Methods of Computing 96, A. Edalat ed., World Scientific (1996) 156-167.

M. Roggenbach, M. Majster-Cederbaum. Towards a unified view of bisimulation: a comparative study. *Theoretical Computer Science* 238 (2000) 81-130.

J. Rutten. Universal coalgebra – a theory of systems. *Theoretical Computer Science*, 249(1):3–80, 2000.

R. Walters. Sheaves and Cauchy-complete categories. *Cahiers de Topologie et Geometrie Diff.*, 22:283–286, 1981.

G. Winskel An introduction to event structures, In *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, volume 354 of *LNCS*, pages 364-397. Springer, 1988.

G. Winskel, M. Nielsen. Models for Concurrency. In *Handbook of Logic in Computer Science*, S. Abramsky, D. Gabbay, T. S. E. Maibaum editors. Oxford University Press, 1995.