Math. Struct. in Comp. Science (2010), *vol.* 20, *pp.* 319–358. © Cambridge University Press 2010
doi:10.1017/S0960129509990272 First published online 4 February 2010

Tree-functors, determinacy and bisimulations

ROCCO DE NICOLA[†], DANIELE GORLA[‡] and ANNA LABELLA[‡]

[†]*Dip. di Sistemi ed Informatica – Univ. di Firenze, Viale Morgagni 65, 50134 Firenze, Italy*
Email: rocco.denicola@unifi.it
[‡]*Dip. di Informatica – Univ. di Roma 'La Sapienza', Via Salaria 113, 00198 Roma, Italy*
Email: {gorla,labella}@di.uniroma1.it

Received 8 October 2007; revised 16 November 2008; accepted 29 June 2009

We study the functorial characterisation of bisimulation-based equivalences over a categorical model of labelled trees. We show that in a setting where all labels are visible, strong bisimilarity can be characterised in terms of enriched functors by relying on the *reflection of paths with their factorisations*. 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 factorisation of π can be lifted in an appropriate factorisation 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 characterise the determinacy of physical systems. We also consider the setting where some labels are not visible, and provide characterisations for weak and branching bisimilarity. Both equivalences are still characterised in terms of enriched functors that reflect paths with their factorisations: for branching bisimilarity, the property is the same as the one used to characterise strong bisimilarity when all labels are visible; for weak bisimilarity, a weaker form of path factorisation lifting is needed. This fact can be seen as evidence that strong and branching bisimilarity are strictly related and that, unlike weak bisimilarity, they preserve process determinacy in the sense of Milner.

1. Introduction

Behavioural equivalences have been used in concurrency theory as abstraction mechanisms to equate systems that exhibit the same behaviour. Different notions of equivalence have been developed over 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, which we represent using labelled trees, where paths represent computations and bifurcations represents non-deterministic 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 that provides the minimum abstraction level from the operational

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

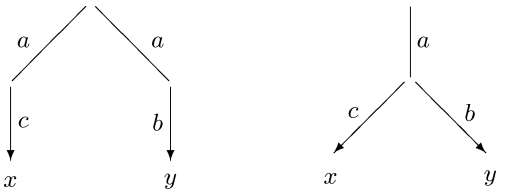


Fig. 1. Two trees.



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

semantics of processes. Indeed, it is generally considered to be 'the' natural equivalence when all process actions are visible, and also because it preserves what Milner (1989) called *strong determinacy*, which is a notion deeply related to the predictability of behaviours. An equivalence enjoying strong determinacy guarantees that the computations of two equated processes have at any point of their progress a direct 'back and forth' correspondence. Within our tree-based model, this amounts to requiring that all paths and their factorisations are preserved and reflected by the mapping associating equivalent processes through their computations (paths), where we use path factorisation to mean a splitting of the path into 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 labelling, then path factorisations *are* preserved and reflected but, in doing this, 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 shown in Figure 3. It is obvious that the three trees do not offer the same behaviour. Indeed, there is no way to associate the paths of the two trees in Figure 1 in such a way that the path factorisations are properly reflected.

Strong bisimilarity can be too discriminating, mainly because it takes 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 characterisation of these equivalences in terms of preservation/reflection of path factorisations, and show that, in the presence of τ-actions, the reflected path factorisations are not necessarily unique.

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 factorisations of x' and y' have an appropriate corresponding factorisation in the corresponding source paths of the leftmost tree: in particular, the factorisation of y' as a and b can be reflected in both y_1 (as $a.\tau$ and

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

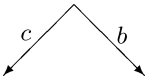


Fig. 3. The derivative after a for the rightmost tree in Figure 1.

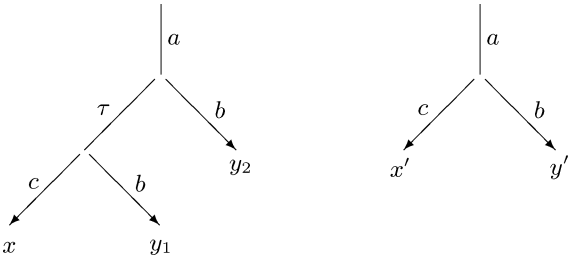


Fig. 4. Two branching bisimilar trees.

b) and y_2 (as a and b); moreover, the corresponding intermediate states offer the same visible behaviour.

However, it can happen that *not all* the possible factorisations of a path can be reflected in an appropriate factorisation of a path mapped to it by the associating function. Consider the trees in Figure 5 and the function mapping y into y' and both x_1 and x_2 into x'. The factorisation 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 factorisation of x' as $a.\tau$ and c can be reflected on x_2.

We show that mapping functions like the one used for Figure 4 characterise branching bisimulation, whereas the weaker factorised path reflection property illustrated by means of Figure 5 captures weak bisimulation. To this end, we work in a categorical framework and characterise strong, branching and weak bisimulation equivalences through proper functorial definitions capturing the intuitions underlying the above examples. 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 appreciate better their meaning and 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 in Kasangian and Labella (1999), which adapts the categorical modelling 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, that is,

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

Initial and terminal states can be specified when necessary, in which case, the set of words leading from the initial state to the terminal ones describes the global behaviour of the

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

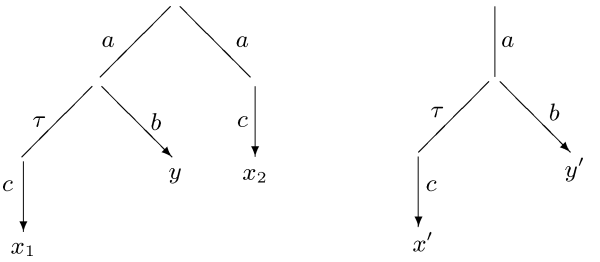


Fig. 5. Two weakly bisimilar trees.

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 factorisation is considered). The equivalence induced by this kind of association is usually called *trace equivalence* by concurrency theorists and ignores the traversed states.

When moving from automata to non-deterministic processes, greater attention must be devoted to the notion of state because one has to consider the possible alternative steps available at intermediate states. In this case, 'sets of words' are no longer sufficient for the representation of non-deterministic behaviours. For this reason, the computations of Labelled Transition Systems (LTS), which are one of the most popular operational models of concurrency, can be better modelled using a tree-shaped labelled 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, that is, 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 labelled 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 is enriched over it (thus, its arrow objects are trees also). 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 (that is, its operational semantics) is respected.

Similarly, we can consider $Tree_\tau$ and $\mathbf{Beh}_{\tau,\tau}$, that is, the category and the $Tree_\tau$-category obtained in the same way as *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

[†] Technically, A-automata are modelled 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 a 'change of base' (Kelly 1982), that is, 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.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

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 this category, where τ's have become invisible, $\mathbf{Beh}_{\tau}$, and study the *Tree*-functors over it that characterise weak and branching equivalences.

We will provide functorial characterisations of the different bisimulation-based equivalences by identifying the properties that must be enjoyed by the *Tree*-functor inducing them. In particular:

(1) *Strong bisimilarity* will be characterised in **Beh** as the equivalence preserved by any *Tree*-functor F that *reflects paths*. This property requires that any path (internal morphism) from $F(\mathscr{X})$ to $\mathscr{Y}$ arises from at least one path going from $\mathscr{X}$ to $\mathscr{X}'$, with $F(\mathscr{X}') = \mathscr{Y}$. Because of the structure of **Beh**, we get that if a path x' of the target tree is factorised as $x'_1; x'_2$, then *every* source path x mapped into x' can be factorised 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 factorisations*. The existence of such a functor is given by the definition of a standard strong representative for every tree, which is obtained by quotienting the set of paths.

(2) *Branching bisimilarity* is the equivalence preserved in $\mathbf{Beh}_{\tau}$ by *Tree*-functors that reflect paths with their factorisations. Notice that, while in **Beh** such a property is equivalent to a simpler one (reflection of paths), the presence of an invisible action in $\mathbf{Beh}_{\tau}$ means that the requirement on the equivalence of all the intermediate states (reachable via invisible actions) of the paths is unavoidable. Again, we show the 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 only enjoy a weaker form of factorised path reflection. This weaker property is similar to the one for strong/branching bisimulation, but only requires that for each factorisation of a target path x', there is a corresponding factorisation of *one of* the source paths: in particular, it does not require that *each* source path faithfully mirrors all factorisations of its image. Thus, if a path x' of the target tree is factorised as $x'_1; x'_2$, it might be that we have two source paths x and y mapped into x' with only x that can be factorised 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 that one has gone through. The standard representative, which we use to guarantee the existence of the required functor, is obtained from the one for branching equivalence by an appropriate pruning.

The reflection of factorisations has a particular meaning: it coincides with the enriched form of a necessary and sufficient condition that a functor has to satisfy in order to guarantee that its inverse image has a right adjoint – we call this the *Conduché property*, after Conduché (1972). It has been proved (Kasangian and Labella 2010) that the natural translation of the 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), which used a strong form of the Conduché property to establish

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

system determinacy, we have further evidence of the fact that weak bisimilarity does not preserve process determinacy.

The advantage of our framework is that it leads to characterisations that are parametric with respect to 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 into the difference between branching and weak bisimilarity, and the extent to which they abstract from silent actions, in terms of the preservation of states/determinacy. Moreover, it is easy to scale our approach to more sophisticated models of concurrency than LTS, such as, *Labelled Event Structures*, which have been used to model causal (in-)dependencies of systems actions (Winskel 1988).

Structure of the paper

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 characterised as the equivalence induced by *Tree*-functors that reflect path factorisations. In Section 3, we allow τ-actions to be ignored, 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 factorisations, respectively. In Section 4, we prove the converse, that is, that strong, branching and weak bisimilarity imply the existence of *Tree*-functors that, besides reflecting paths, reflect (or weakly reflect, in the weak case) path factorisations. In Section 5, we show how our treatment can be applied in the cases of labelled transition systems and labelled (prime) event structures. Section 6 contains a detailed comparison with related work, while in Section 7 we sum up and discuss our contribution.

Throughout this paper we assume a familiarity with basic *Category Theory*, and, in particular, with the notions of *tensor product*, *monoidal category* and *monoidal functor*: if required, the necessary background can be found in Kelly (1982).

2. The tree model for non-deterministic processes

2.1. *Basic definitions*

In this section we recall from Kasangian and Labella (1999) a category of labelled trees, *Tree*, and some of its properties. A single tree is modelled 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, with the labels ab and ac, respectively. In the leftmost tree, x and y do not agree at all (that is, their agreement is ϵ), but in the rightmost tree, x and y agree through the initial a.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

For mathematical reasons (see Remark 1 below), it is convenient to make explicit the order structure that is otherwise 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 $s \leqslant t$ if and only if there exists $u \in A^*$ such that $s \bullet u = t$ (this is the usual notion of the 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*, that is, for every $s, t, u \in A^*$ such that $s \bullet t = s \bullet u$, we have $t = u$.

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 $\mathscr{X}$ 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 $x, y, z \in X$, we have:

(1) $a_X(x, x) = e_X(x)$.
(2) $a_X(x, y) \leqslant e_X(x) \wedge e_X(y)$.
(3) $a_X(x, y) \wedge a_X(y, z) \leqslant a_X(x, z)$.
(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 shown in Figure 1 are formally specified by $\mathscr{X} = (X, e_X, a_X)$ and $\mathscr{Y} = (Y, e_Y, a_Y)$, 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.

Remark 1. Note 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 an $\mathbf{A}$-functor.

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

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

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

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 labelling).

We will use $Tree_{\mathbf{A}}$ to denote the category of **A**-trees (to simplify the notation, we shall write *Tree* instead of $Tree_{\mathbf{A}}$, and leave **A** implicit when it is clear from the context). *Tree* has (among other features):

— an initial object – the empty tree, $\mathbf{0} = (\varnothing, \varnothing, \varnothing)$;
— a terminal object – the monoid itself considered as a tree, $(A^*, id, \wedge)$;
— finite coproducts (that is, 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, that is, 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 **A**-trees $\mathscr{X}$ and $\mathscr{Y}$, we can form the *sequential composition of* $\mathscr{X}$ *and* $\mathscr{Y}$, $\mathscr{X} \otimes \mathscr{Y} = (Z, e_Z, a_Z)$, 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, $\mathscr{X} \otimes \mathscr{Y}$ attaches a copy of $\mathscr{Y}$ rooted at every leaf of $\mathscr{X}$. Thus, a path in $\mathscr{X} \otimes \mathscr{Y}$ is a path in $\mathscr{X}$ followed by a path in $\mathscr{Y}$, and its label is the label in $\mathscr{X}$ followed by the label in $\mathscr{Y}$. Paths that are different in $\mathscr{X}$ inherit their $\mathscr{X}$ agreement, while paths that differ only in $\mathscr{Y}$ have their $\mathscr{X}$ agreement concatenated with their $\mathscr{Y}$ agreement. In the following, the path (x, y) in $\mathscr{X} \otimes \mathscr{Y}$ will be denoted by $x;y$.

Proposition 1. Sequential composition defines the object part of an associative tensor product on *Tree* with unit $\mathscr{I} = (\{\star\}, [\star \mapsto \epsilon], [(\star, \star) \mapsto \epsilon])$. *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 **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) $\mathscr{I} \xrightarrow{1_A} \mathbf{C}[A, A]$ (*identity*)

that make the diagrams in Figure 6 commute. In Figure 6, $\alpha_{A\,B\,C\,D}$ 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 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 **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.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

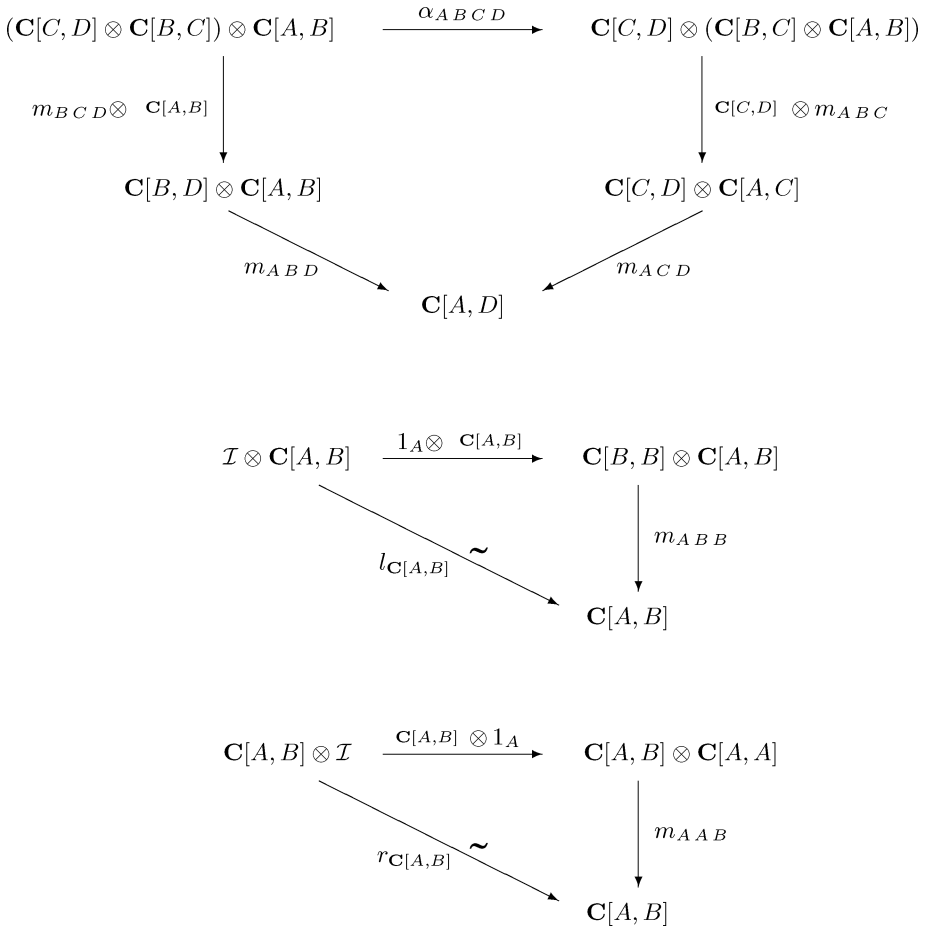
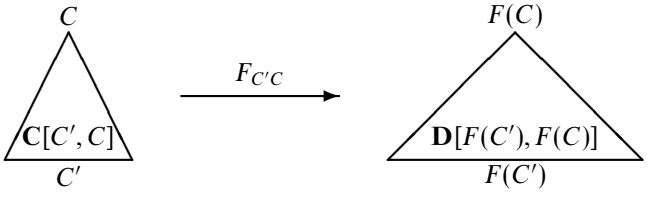


Fig. 6. The commutativity diagrams for Definition 4.

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'. Graphically:



https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

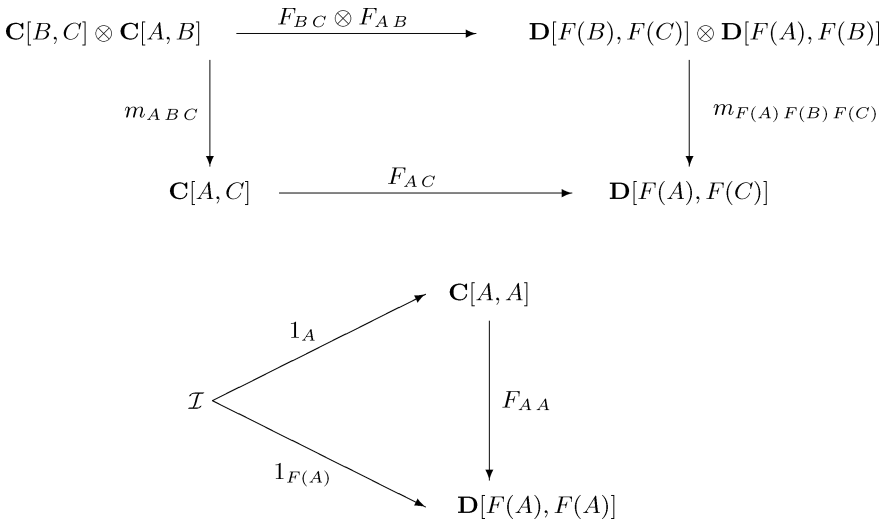


Fig. 7. The commutativity diagrams for Definition 5.

Example 4. Consider the *Tree*-category $\mathbf{T}$ with only one object, say T. If we let $\mathbf{T}[T,T] = (A^*, id, \wedge)$, then $\mathbf{T}$ will be the terminal *Tree*-category. Indeed, for every *Tree*-category $\mathbf{C}$ and 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 labelled path of $\mathbf{T}[T,T]$.

Tree-functors induce *Tree*-morphisms, so they let us associate paths by taking into account their labels and their agreement. Our aim is to characterise functors by imposing further conditions over their induced morphisms. For example, we also want to take into account an 'inverse' correspondence between paths and the possibility of reflecting factorisations.

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

(1) F is *path reflecting* (PR, for short) if and only if for every $C \in \mathbf{C}$ and every path $y \in \mathbf{D}[D, F(C)]$ there are $C' \in \mathbf{C}$ and $x \in \mathbf{C}[C',C]$ such that $F(C') = D$ and $F_{C'C}(x) = y$.

(2) F is *factorised path reflecting* (FPR, for short) if and only if:

  (a) it is PR; and

  (b) for every path $x \in \mathbf{C}[C',C]$, we have

  $F_{C'C}(x) = y';y''$, 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 *factorised path weakly reflecting* (FPWR, for short) if and only if:

  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 $x'' \in \mathbf{C}[C',C'']$ such that $F(C') = D$, $F_{C'C}(x) = y$, $F_{C''C}(x') = y'$, $F_{C'C''}(x'') = y''$ and $F(C'') = D'$.

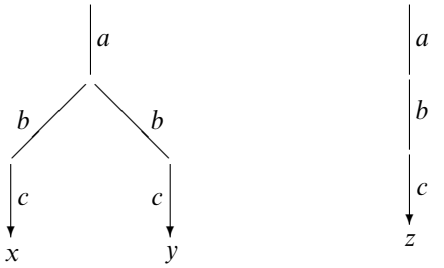
https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

Proposition 2. FPR implies FPWR, which implies PR.

Proof. Let $y \in \mathbf{D}[D, F(C)]$. By PR (which 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 factorisation of y can be reflected into x, so FPR implies FPWR.

We 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 $F(C') = D$, so 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



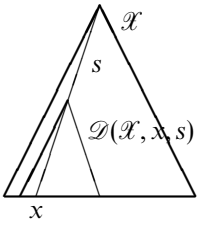
be $\mathbf{C}[C', C]$ and $\mathbf{D}[D', D]$, respectively. The *Tree*-functor that associates C with D and C' with D' is FPR, and thus PR and FPWR. In the next section (see, for example, Proposition 4), we will see examples of functors that satisfy Definition 6(2b) but not PR.

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 $\mathscr{X}$ reached after word s along a given path x. To this end, we use $t - s$ to denote the word obtained from t by deleting the prefix s.

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

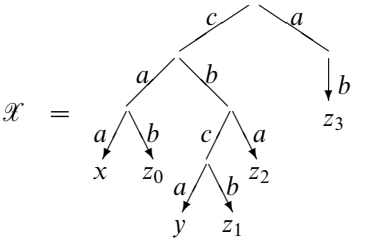
— $Y = \{x' \in X \ : \ a_X(x, x') \geqslant s\}$
— $e_Y(x') = e_X(x') - s$
— $a_Y(x', x'') = a_X(x', x'') - s$.

The following figure illustrates our terminology:



https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

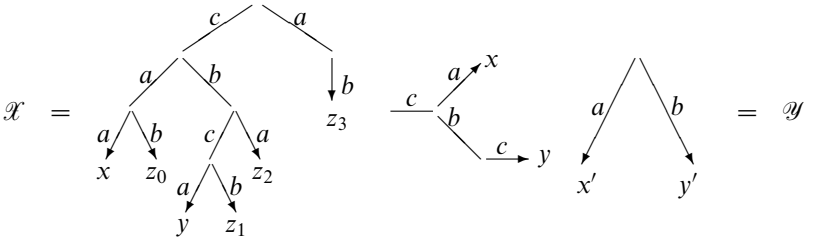
More concretely, by letting



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



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



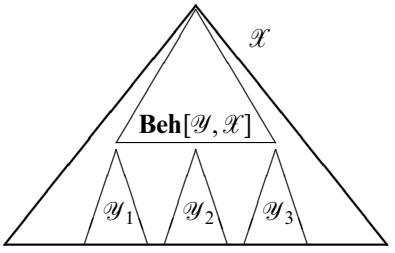
We can look at this tree as a structured set of morphisms from $\mathscr{Y}$ to $\mathscr{X}$ (the reversed notation is due to technical reasons) and consider the category **Beh** with the same objects as *Tree*, but with these new morphism objects instead of simulations.

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

— Its set of paths is $\{|x|_{\mathscr{D}} \ : \ x \in X \ \text{ and } \ \mathscr{Y} \cong \mathscr{D}(\mathscr{X}, x, s)\}$, where $x \equiv_{\mathscr{D}} y$ if and only if $\mathscr{D}(\mathscr{X}, x, s) = \mathscr{D}(\mathscr{X}, y, s)$ and $|x|_{\mathscr{D}}$ denotes the equivalence class of x with respect to $\equiv_{\mathscr{D}}$.
— The extent of a path $|x|_{\mathscr{D}}$ is s whenever $\mathscr{Y} \cong \mathscr{D}(\mathscr{X}, x, s)$.
— $a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x'|_{\mathscr{D}}) = e_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}) \wedge e_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x'|_{\mathscr{D}}) \wedge a_X(x, x')$.
— Composition in **Beh** is given by the tensor product of *Tree*.

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

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press



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

Proposition 3. For every $\mathscr{X}$ and $\mathscr{Y}$ in *Tree*, we have $\mathbf{Beh}[\mathscr{Y},\mathscr{X}]$ is a tree.

Proof. It is trivial to check that the four properties of Definition 1 hold once we have shown that the choice of the representative $|x|_D$, for every path $x \in X$, does not affect the definition of $\mathbf{Beh}[\mathscr{Y},\mathscr{X}]$. Let $x'' \in |x'|_D$. The only non-trivial thing to prove is that

$$a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x'|_{\mathscr{D}}) = a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x''|_{\mathscr{D}}).$$

We consider two cases:

(1) $a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x'|_{\mathscr{D}}) \leqslant e_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}) \wedge e_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x'|_{\mathscr{D}})$:

By property 3 in Definition 1,

$$a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x'|_{\mathscr{D}}) \wedge a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x'|_{\mathscr{D}}, |x''|_{\mathscr{D}}) \leqslant a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x''|_{\mathscr{D}}).$$

Since x' and x'' belong to the same $\equiv_D$-class, $a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x'|_{\mathscr{D}}, |x''|_{\mathscr{D}}) \geqslant e_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x'|_{\mathscr{D}})$, so

$$a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x'|_{\mathscr{D}}) \leqslant a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x''|_{\mathscr{D}}).$$

In a similar way, we can prove that

$$a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x''|_{\mathscr{D}}) \leqslant a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x'|_{\mathscr{D}}),$$

and conclude.

(2) $a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x'|_{\mathscr{D}}) \geqslant e_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}})$:

(The case in which $a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x'|_{\mathscr{D}}) \geqslant e_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x'|_{\mathscr{D}})$ is similar.)
This case is simpler, since, by the definition of $a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}$,

$$a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x'|_{\mathscr{D}}) = a_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}, |x''|_{\mathscr{D}}) = e_{\mathbf{Beh}[\mathscr{Y},\mathscr{X}]}(|x|_{\mathscr{D}}).$$ □

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

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

Proposition 4 (Kasangian and Labella 2010). 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 factorisation 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 $x \in \mathbf{Beh}[\mathscr{X}', \mathscr{X}]$ and $y \in \mathbf{Beh}[F(\mathscr{X}'), F(\mathscr{X})]$ be the path such that $F_{\mathscr{X}'\mathscr{X}}(x) = y$. Consider the factorisation $y = y'; y''$, with s' and s'' the extent of y' and y'', respectively, and let $\mathscr{Z} \cong \mathscr{D}(F(\mathscr{X}), y, s')$. Since F is a *Tree*-functor, the extent of x is $s's''$, so there exists $\mathscr{D}(\mathscr{X}, x, s')$. The claim is proved if we can show that $F(\mathscr{D}(\mathscr{X}, x, s')) \cong \mathscr{Z}$. Let x' and x'' be the portions of x with extent s' and s'', respectively. So $x = x'; x''$. By *Tree*-functoriality, $F_{\mathscr{X}'\mathscr{X}}(x'; x'') = y$, and there exist $F_{\mathscr{D}(\mathscr{X},x,s'),\mathscr{X}}(x)$ and $F_{\mathscr{X}',\mathscr{D}(\mathscr{X},x,s')}(x)$. However, by the definition of **Beh**, we must have $F_{\mathscr{D}(\mathscr{X},x,s'),\mathscr{X}}(x) = y'$ and $F_{\mathscr{X}',\mathscr{D}(\mathscr{X},x,s')}(x) = y''$, so $F(\mathscr{D}(\mathscr{X}, x, s')) \cong \mathscr{Z}$, since, by functoriality, $y' \in \mathbf{Beh}[F(\mathscr{D}(\mathscr{X}, x, s')), F(\mathscr{X})]$. □

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

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

2.2. *Strong bisimilarity*

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

Definition 9. A symmetric relation between trees $\mathfrak{R}$ is a *strong bisimulation* if, for every $(\mathscr{X}, \mathscr{Y}) \in \mathfrak{R}$, we have $\forall x \in X\ \exists y \in Y$ such that $e_Y(y) = e_X(x)$ and $\forall s \leqslant e_X(x)$ we have $(\mathscr{D}(\mathscr{X}, x, s), \mathscr{D}(\mathscr{Y}, y, s)) \in \mathfrak{R}$.

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

We can show that strong bisimulations are the equivalences induced by factorised path reflecting *Tree*-functors. Notice that in **Beh**, FPR coincides with PR (see Propositions 2 and 5). However, in order to give a uniform treatment with the case of branching bisimulation (see Section 3.2), we shall rely on FPR.

Lemma 1. Let $F : \mathbf{Beh} \to \mathbf{Beh}$ be a FPR *Tree*-functor. Then:

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

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

Proof. For the first claim, let $\pi \in \mathbf{Beh}[\mathscr{I}, F(\mathscr{I})]$. Since F is Pr, there exist $A' \in \mathbf{Beh}$ and $p \in \mathbf{Beh}[A', \mathscr{I}]$ such that $F(A') = \mathscr{I}$. But the only such p is the path labelled with ϵ leading $\mathscr{I}$ to itself, so $A' = \mathscr{I}$ and thus $F(\mathscr{I}) = \mathscr{I}$.

The second claim is trivial. □

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

Proof. We begin by noting that F is Pr. We will prove that the relation $\mathfrak{R} = \{(\mathscr{X}, F(\mathscr{X}))\}$ is a bisimulation, as defined in Definition 9. Then, by the transitivity of $\simeq_S$ and the fact that isomorphic trees are also bisimilar, we will be able to conclude since $\mathscr{X} \simeq_S F(\mathscr{X}) \cong F(\mathscr{Y}) \simeq_S \mathscr{Y}$. So we let $(\mathscr{X}, F(\mathscr{X})) \in \mathfrak{R}$.

(1) Fix $x \in X$ that is also a path in $\mathbf{Beh}[\mathscr{I}, \mathscr{X}]$. Reasoning up to $\equiv_{\mathscr{D}}$, we have, since F is a *Tree*-functor, that $F_{\mathscr{I}\mathscr{X}}(x)$ is a path in $\mathbf{Beh}[F(\mathscr{I}), F(\mathscr{X})]$ with the same extent as x. From Lemma 1(1), $F_{\mathscr{I}\mathscr{X}}(x)$ is a path in $\mathbf{Beh}[\mathscr{I}, F(\mathscr{X})]$, that is, $F_{\mathscr{I}\mathscr{X}}(x)$ is a path of $F(\mathscr{X})$. Now fix an $s \leqslant e_X(x)$. Since F is a *Tree*-functor, we have $F(\mathscr{D}(\mathscr{X}, x, s)) \cong \mathscr{D}(F(\mathscr{X}), F_{\mathscr{I}\mathscr{X}}(x), s)$, which suffices to conclude that $(\mathscr{D}(\mathscr{X}, x, s), \mathscr{D}(F(\mathscr{X}), F_{\mathscr{I}\mathscr{X}}(x), s)) \in \mathfrak{R}$.

(2) Now take any path z in $F(\mathscr{X})$, that is, $z \in \mathbf{Beh}[\mathscr{I}, F(\mathscr{X})]$. Since F is Pr, there exist $\mathscr{X}'$ and $x \in \mathbf{Beh}[\mathscr{X}', \mathscr{X}]$ such that $F(\mathscr{X}') = \mathscr{I}$ and $F_{\mathscr{X}'\mathscr{X}}(x) = z$. From Lemma 1(2), $\mathscr{X}'$ has only ϵ-labelled paths, so we can consider x as a path of $\mathscr{X}$. 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 let z' be the portion of z corresponding to s. By Proposition 5, F is also Fpr, so there exist $\mathscr{X}', x', x''$ such that $x' \in \mathbf{Beh}[\mathscr{X}', \mathscr{X}]$, $x'' \in \mathbf{Beh}[\mathscr{I}, \mathscr{X}']$, $F_{\mathscr{X}'\mathscr{X}}(x') = z'$ and $F(\mathscr{X}') = \mathscr{D}(F(\mathscr{X}), z, s)$. By *Tree*-functoriality, $F_{\mathscr{X}'\mathscr{X}}(x') = z'$ implies that x' and z' have the same extent (*viz.* s), so we can put $\mathscr{X}' = \mathscr{D}(\mathscr{X}, x, s)$, as required. □

3. Trees with silent actions

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

3.1. *Basic definitions*

We specify an element of the alphabet to represent the silent action, called τ, and use A_τ to denote 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 $a_k \in A$, and we use the usual notion for prefix in $\mathbf{A}_\tau$.

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

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

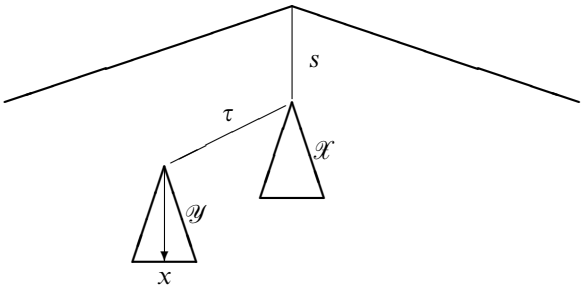
https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

The function DEL can be canonically extended to a monoidal functor from trees with silent actions, that is, $\mathbf{A}_\tau$-trees (which are denoted by $Tree_\tau$ when $\mathbf{A}$ is clear from the context), to *Tree* as follows.

Definition 10. DEL : $Tree_\tau \to Tree$ 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 $\mathscr{X}$ to $\mathscr{Y}$ also define morphisms from $\text{DEL}(\mathscr{X})$ to $\text{DEL}(\mathscr{Y})$).

Since $Tree_\tau$ is a monoidal category, we can trivially define $Tree_\tau$-categories by rephrasing Definition 4. So the monoidal functor DEL just defined can be used to pass from a $Tree_\tau$-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 a *change of base*. More precisely, in analogy with **Beh**, we can consider $\mathbf{Beh}_{\tau,\tau}$ as the $Tree_\tau$-category whose objects and morphism objects are objects of $Tree_\tau$. 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 $\mathbf{Beh}_\tau$ in which 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 $\mathbf{Beh}_{\tau,\tau}$, except that $\mathbf{Beh}_\tau[\mathscr{Y}, \mathscr{X}] = \text{DEL}(\mathbf{Beh}_{\tau,\tau}[\mathscr{Y}, \mathscr{X}])$.

Once we have deleted 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 $\mathscr{X} + \tau\mathscr{Y}$ and $\mathscr{Y}$, after $\text{DEL}(s)$ or $\text{DEL}(s \bullet \tau)$, respectively. However, it is important to note that one of these trees is always the largest ($\mathscr{X} + \tau\mathscr{Y}$ here), and any other tree so accessed (like $\mathscr{Y}$) is a τ-derivative of this one. The largest tree is $\mathscr{D}(\mathscr{X}, x, s)$ and its existence allows us to work with finite chains.

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

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

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

[†] Notice that the passage from $\mathbf{Beh}_{\tau,\tau}$ to $\mathbf{Beh}_\tau$ corresponds to the way Milner passes from the arrow $\xrightarrow{\alpha}$ to the arrow $\stackrel{\alpha}{\Rightarrow}$ in the context of LTS.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

Proposition 6 (Kasangian and Labella 2010). The unique *Tree*-functor from $\mathbf{Beh}_\tau$ 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 factorisation is unique up to silent moves.

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

Proposition 7. Let $F : \mathbf{Beh}_\tau \to \mathbf{Beh}_\tau$ be a Fpr *Tree*-functor. Then the reflected factorisation is unique up to silent moves.

Proof. The proof is similar to the proof of Proposition 5. □

Remark 3. In particular, in $\mathbf{Beh}_\tau$, every Fpr *Tree*-endofunctor also enjoys the enriched form of the Conduché property (Conduché 1972; Kasangian and Labella 2010).

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 definition usually introduced for LTS.

Definition 12. A symmetric relation on trees $\mathfrak{R}$ is a *branching bisimulation* if, for every $(\mathscr{X}, \mathscr{Y}) \in \mathfrak{R}$, we have $\forall x \in X\ \exists y \in Y$ such that $\mathrm{DEL}(e_Y(y)) = \mathrm{DEL}(e_X(x))$ and:

(1) $\forall s \leqslant e_X(x)\ \forall \mathscr{X}' \in \mathscr{D}_\tau(\mathscr{X}, x, s)\ \exists \mathscr{Y}' \in \mathscr{D}_\tau(\mathscr{Y}, y, s)$ such that $(\mathscr{X}', \mathscr{Y}') \in \mathfrak{R}$.
(2) $\forall s \leqslant e_Y(y)\ \forall \mathscr{Y}' \in \mathscr{D}_\tau(\mathscr{Y}, y, s)\ \exists \mathscr{X}' \in \mathscr{D}_\tau(\mathscr{X}, x, s)$ such that $(\mathscr{X}', \mathscr{Y}') \in \mathfrak{R}$.

Two trees $\mathscr{X}$ and $\mathscr{Y}$ are *branching bisimilar*, written $\mathscr{X} \simeq_B \mathscr{Y}$, if and only if 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 $\mathbf{Beh}_\tau$. We begin with some preliminary results.

Lemma 2. Let $F : \mathbf{Beh}_\tau \to \mathbf{Beh}_\tau$ be a Pr *Tree*-functor. Then:

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

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

Proof. We prove that $\mathfrak{R} = \{(\mathscr{X}, F(\mathscr{X}))\}$ is a branching bisimulation. To this end, fix any $(\mathscr{X}, F(\mathscr{X})) \in \mathfrak{R}$.

— Let $x \in X$ that is also a path in $\mathbf{Beh}_\tau[\mathscr{I}, \mathscr{X}]$. As usual, we reason up to $\equiv_{\mathscr{D}}$.

(1) Since F is a *Tree*-functor, $F_{\mathscr{I}\mathscr{X}}(x)$ is a path in $\mathbf{Beh}_\tau[F(\mathscr{I}), F(\mathscr{X})]$ with the same visible extent as x. By Lemma 2(1), $F_{\mathscr{I}\mathscr{X}}(x) \in \mathbf{Beh}_\tau[\mathscr{I}, F(\mathscr{X})]$, so we have found a path of $F(\mathscr{X})$ with the same visible extent as x.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

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

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

— Let z be a path in $F(\mathscr{X})$. We can reason as in the second item in the proof of Theorem 1, but using Lemma 2 instead of Lemma 1, to prove that there exists a $x \in X$ with the same visible extent as z and that, for every derivative $\mathscr{Z}$ along z, we can find a derivative $\mathscr{X}'$ along x (after the same visible prefix) such that $F(\mathscr{X}') = \mathscr{Z}$: it suffices to consider $\mathbf{Beh}_\tau$ in place of $\mathbf{Beh}$ and $\mathscr{D}_\tau$ in place of $\mathscr{D}$. To prove that, for every $s \preccurlyeq e_X(x)$ and $\mathscr{X}' \in \mathscr{D}_\tau(\mathscr{X}, x, s)$, we have $F(\mathscr{X}') \in \mathscr{D}_\tau(F(\mathscr{X}), F_{\mathscr{I}\mathscr{X}}(x), s)$ up-to isomorphism, we can reason as in item (2) of the first part 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 definition usually introduced for LTS.

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

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

Note that the only difference between Definition 12 and Definition 13 is an extra symmetry requirement. The main implication of this is that branching bisimilarity induces a correspondence between paths, so it is an equivalence relation between paths (in the style of the 'back-and-forth' approach (De Nicola *et al.* 1990)); on the other hand, it is not possible to define such a path correspondence for weak bisimulations.

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

Proof. We prove that $\mathfrak{R} = \{(\mathscr{X}, F(\mathscr{X})\}$ is a weak bisimulation. We first fix $x \in X$ and show that:

(i) There exists a path z in $F(\mathscr{X})$ with the same visible extent as x.
(ii) Every $\mathscr{X}' \in \mathscr{D}_\tau(\mathscr{X}, x, s)$, where $s \preccurlyeq e_X(x)$, is such that $F(\mathscr{X}') \in \mathscr{D}_\tau(F(\mathscr{X}), z, s)$ up-to isomorphism.

These facts can be proved in the same way as (1) and (2) in the first item of the proof of Theorem 2 (in that proof, we only used the fact that F is FPWR). Now, let z be a path in $F(\mathscr{X})$. As in the second item of the proof of Theorem 2, there exists a $x \in X$ with the

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

same visible extent as z and, for every derivative $\mathscr{Z}$ along z, we can find a derivative $\mathscr{X}'$ along x (after the same visible prefix) such that $F(\mathscr{X}') = \mathscr{Z}$ up-to isomorphism. This is then enough to give the result. □

4. Complete characterisations and standard representatives

The converses of Theorems 1, 2 and 3 also hold, 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 end, we will define equivalences of paths as induced by $\simeq_S$ and $\simeq_B$ (remember that a simple equivalence on paths associated with $\simeq_W$ is impossible).

4.1. *Standard representatives for strong bisimilarity*

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

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

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

Note that the above join exists, because of the definition of a meet-semilattice monoid, and that it is a maximum. Moreover, we can prove the following 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,

$$s \leqslant e_X(x) = e_X(\bar{x}) = e_X(x')$$

and

$$s \leqslant e_X(y) = e_X(\bar{y}) = e_X(y').$$

Now, by the definition of $\equiv_S$, we have $\mathscr{D}(\mathscr{X}, x', s) \simeq_S \mathscr{D}(\mathscr{X}, \bar{x}, s)$, and since $a_X(\bar{x}, \bar{y}) = s$, we have $\bar{y} \in \mathscr{D}(\mathscr{X}, \bar{x}, s)$. Hence, there must be a $y' \in \mathscr{D}(\mathscr{X}, x', s)$ such that $y' \equiv_S \bar{y}$, so $a_X(x', y') \geqslant s$. Since s is the maximum agreement between elements in $|x|_S$ and $|y|_S$, we are done. □

We prepare for the proof that $\mathbf{S}\mathscr{X}$ is a standard representative of the $\simeq_S$-equivalence class of $\mathscr{X}$ by showing that $\mathscr{X}$ and $\mathbf{S}\mathscr{X}$ have the same transitions, which also easily implies that $\mathbf{S}$ is a *Tree*-functor.

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

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

Proof. If $\mathscr{D}(\mathscr{X}, x, s) \cong \mathscr{I}$, the claim is trivial. Let $y \in X$ be a path belonging to $\mathscr{D}(\mathscr{X}, x, s)$, so $a_X(x, y) \geqslant s$. By Definition 14, the agreement between $|x|_S$ and $|y|_S$ in $\mathbf{S}\mathscr{X}$ is at least $a_X(x, y)$, so $|y|_S$ is a path in $\mathbf{S}(\mathscr{D}(\mathscr{X}, x, s))$. Conversely, let $|z|_S$ be a path of $\mathbf{S}\mathscr{X}$ belonging to $\mathbf{S}(\mathscr{D}(\mathscr{X}, x, s))$. By definition, the agreement between $|x|_S$ and $|z|_S$ is $s' \geqslant s$. 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 $\mathscr{D}(\mathscr{X}, x, s)$. A similar reasoning applies to agreements, while the extents are trivially identical. □

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

Proof. We define the relation $\mathfrak{R} = \{(\mathscr{X}, \mathbf{S}\mathscr{X})\}$, fix $(\mathscr{X}, \mathbf{S}\mathscr{X}) \in \mathfrak{R}$ and pick up any $x \in X$ (the case in which we pick up a path $|x|_S \in \mathbf{S}\mathscr{X}$ is similar). By construction, $|x|_S$ is a path in $\mathbf{S}\mathscr{X}$ with extent $e_X(x)$. Now choose $s \leqslant e_X(x)$. By Lemma 4, $\mathbf{S}(\mathscr{D}(\mathscr{X}, x, s)) = \mathscr{D}(\mathbf{S}\mathscr{X}, |x|_S, s)$, which is enough to give the result. □

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

Proof. The 'if' part follows trivially from Lemma 5.

For the 'only if' part, we will prove that $\mathscr{X} \simeq_S \mathscr{Y}$ induces a *Tree*-isomorphism between $\mathbf{S}\mathscr{X}$ and $\mathbf{S}\mathscr{Y}$. 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 $\mathbf{S}\mathscr{X}$ and $\mathbf{S}\mathscr{Y}$ 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, which 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 $\mathbf{S}\mathscr{X}$ and $\mathbf{S}\mathscr{Y}$ have corresponding paths with the same extent. We are left to prove that their paths also have the same agreement. Choose $|x|_S$ and $|x'|_S$ in $\mathbf{S}\mathscr{X}$ and let s be their agreement. So 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 $\mathbf{S}\mathscr{Y}$. By hypothesis, $\mathscr{D}(\mathscr{X}, x_1, s) \simeq_S \mathscr{D}(\mathscr{Y}, y, s)$. Since x_2 is also a path in $\mathscr{D}(\mathscr{X}, x_1, s)$, there exists a y' in $\mathscr{D}(\mathscr{Y}, y, s)$ (that is, such that $a_Y(y, y') \geqslant s$) corresponding to x_2. To show a contradiction, let $a_Y(y, y') > s$ and consider $\mathscr{D}(\mathscr{Y}, y, a_Y(y, y'))$. By the hypothesis of bisimilarity, this corresponds to a proper derivative of $\mathscr{D}(\mathscr{X}, x_1, s)$. But then s would not be the maximum agreement between paths in $|x|_S$ and $|x'|_S$, which gives a contradiction. □

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

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

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

Proof. We will prove that **S** can be extended to the desired *Tree*-functor. By Proposition 8, $\mathbf{S}(\mathscr{X}) \cong \mathbf{S}(\mathscr{Y})$. To prove that **S** is a *Tree*-functor, we have to prove that **S** induces a *Tree*-morphism between $\mathbf{Beh}[\mathscr{X}', \mathscr{X}]$ and $\mathbf{Beh}[\mathbf{S}\mathscr{X}', \mathbf{S}\mathscr{X}]$ for every $\mathscr{X}'$.

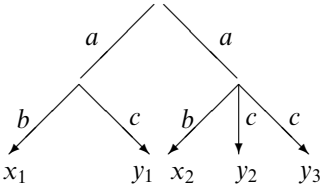
The case when $\mathbf{Beh}[\mathscr{X}', \mathscr{X}]$ is the empty tree is trivial.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

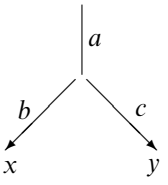
For the other cases, take any path $x' \in \mathbf{Beh}[\mathscr{X}', \mathscr{X}]$. Then $\mathscr{X}' = \mathscr{D}(\mathscr{X}, x, s)$ up-to isomorphism, where x is a path in $\mathscr{X}$ such that x' is its initial part labelled by s. By Lemma 4, $\mathbf{S}\mathscr{X}$ evolves to $\mathbf{S}\mathscr{X}'$ via a path with the same extent as x'. Moreover, by the definition of standard representatives, the agreement between any two paths cannot decrease.

We now have to prove that $\mathbf{S}$ is Fpr. However, because of Proposition 5, it is enough to prove that it is Pr. Fix a $\mathscr{Z}$ and a $z' \in \mathbf{Beh}[\mathscr{Z}, \mathbf{S}\mathscr{X}]$. So $\mathscr{Z} = \mathscr{D}(\mathbf{S}\mathscr{X}, |z|_S, s)$ up-to isomorphism, where $|z|_S$ is a path in $\mathbf{S}\mathscr{X}$ such that z' is its initial part labelled by s. By Lemma 4, $\mathscr{Z} = \mathbf{S}(\mathscr{D}(\mathscr{X}, z, s))$ up-to isomorphism, so we have found an $\mathscr{X}'$ (which, up-to isomorphism, is $\mathscr{D}(\mathscr{X}, z, s)$) and an $x' \in \mathbf{Beh}[\mathscr{X}', \mathscr{X}]$ (*viz.* the initial portion of z labelled by s) such that we have $\mathbf{S}\mathscr{X}' = \mathscr{Z}$ and $\mathbf{S}_{\mathscr{X}'\mathscr{X}}(x') = z'$. □

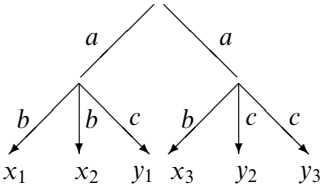
Example 6. Consider the following tree $\mathscr{X}$:



Using the construction of Definition 14, we can obtain its standard representative, that is, the following tree $\mathscr{Y}$:



Now consider the following tree $\mathscr{Z}$:



Its standard representative is still $\mathscr{Y}$ and, indeed, $\mathscr{X}$ and $\mathscr{Z}$ are strongly bisimilar. Indeed, letting $f : \mathscr{X} \to \mathscr{Y}$ and $g : \mathscr{Z} \to \mathscr{Y}$ be the *Tree*-morphisms such that

$$f(x_1) = f(x_2) = x = g(x_1) = g(x_2) = g(x_3)$$

and

$$f(y_1) = f(y_2) = f(y_3) = y = g(y_1) = g(y_2) = g(y_3),$$

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

we have

$$\mathscr{X} \xrightarrow{f} \mathscr{Y} \xleftarrow{g} \mathscr{Z}.$$

That is, there exists a cospan of *Tree*-morphisms between the two strongly bisimilar trees $\mathscr{X}$ and $\mathscr{Z}$. These morphisms reflect paths with their factorisations.

To conclude, notice that **S** is also characterisable through a maximality condition among those enjoying factorised path reflection, that is, every Pr *Tree*-functor is annihilated by **S**.

Proposition 9. Every Fpr *Tree*-functor $F : \mathbf{Beh} \to \mathbf{Beh}$ is such that $\mathbf{S}F \cong \mathbf{S}$.

Proof. Since F reflects paths, $F_{\mathscr{I},\mathscr{X}}$ is an epimorphism from $\mathbf{Beh}[\mathscr{I}, \mathscr{X}]$ to $\mathbf{Beh}[\mathscr{I}, F(\mathscr{X})]$ (see Lemma 1(1)). Therefore, there is an epimorphism from $\mathscr{X}$ to $F(\mathscr{X})$. This means that $F(\mathscr{X})$ is a quotient of $\mathscr{X}$ with respect to a strong bisimulation. But **S** induces the coarsest strong bisimulation, so we get the assertion. *Tree*-naturality of the isomorphism holds since both images are induced through quotienting on paths. □

4.2. *Standard representatives for branching bisimilarity*

We will now rephrase the results of the previous section to deal with branching bisimilarity. However, the presence of τ's means that building the standard branching representative requires some extra effort – the same techniques will be useful for building the standard weak representative in the next section.

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

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

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

Note that the sets $\mathscr{D}_\tau(\mathscr{X}, x, s)$, $\mathscr{D}_d(\mathscr{X}, x, s)$ and $\mathscr{D}_S(\mathscr{X}, x, s)$ are chains with respect to the property of being a τ-summand, which are finite thanks to the regularity of our trees. DEL also induces 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 $\mathscr{X}$ defined by $x \equiv_B x'$ if and only if $\text{DEL}(e_X(x)) = \text{DEL}(e_X(x'))$ and, for every $s \leqslant e_X(x)$, we have $\mathscr{D}_S(\mathscr{X}, x, s)$ is in a bijective correspondence with $\mathscr{D}_S(\mathscr{X}, x', s)$ such that corresponding trees are isomorphic. We will use $|x|_B$ to denote the $\equiv_B$-equivalence class of x.

The *standard branching representative* of a tree $\mathscr{X} = (X, e_X, a_X)$ is the tree $\mathbf{B}\mathscr{X} = (BX, e_{BX}, a_{BX})$, 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 $i_k = |\mathscr{D}_S(\mathscr{X}, x, s_1 \dots s_{k-1})| - 1$, for $1 \leqslant k \leqslant n+1$ and $e_X(x) = \tau^{k_1} s_1 \tau^{k_2} s_2 \dots \tau^{k_n} s_n \tau^{k_{n+1}}$.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

— $a_{BX}(|x|_B, |y|_B) = \tau^{i_1} s_1 \tau^{i_2} s_2 \dots s_m \tau^{i_{m+1}}$, with:
  - $i_k = |\mathscr{D}_S(\mathscr{X}, x, s_1 \dots s_{k-1})| - 1$, for $1 \leqslant k \leqslant m$, and
  - $i_{m+1} = h - 1$, where h is the length of the longest common prefix of the chains $\mathscr{D}_S(\mathscr{X}, x, s_1 s_2 \dots s_m)$ and $\mathscr{D}_S(\mathscr{X}, y, s_1 s_2 \dots s_m)$ (that is,

$$\mathscr{D}_S(\mathscr{X}, x, s_1 s_2 \dots s_m) = \{\mathscr{X}_1, \dots, \mathscr{X}_k, \mathscr{X}'_{h+1} \dots, \mathscr{X}'_{h+p}\}$$

    and

$$\mathscr{D}_S(\mathscr{X}, y, s_1 s_2 \dots s_m) = \{\mathscr{X}_1, \dots, \mathscr{X}_h, \mathscr{X}''_{h+1} \dots, \mathscr{X}''_{h+q}\}$$

    for either $p = q = 0$, or $p = 0$ and $q > 0$, or $p > 0$ and $q = 0$, or $\mathscr{X}'_{h+1} \neq \mathscr{X}''_{h+1}$)

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

Proof. Fix an $x \in X$ and an $s = a_1 \dots a_h \leqslant e_X(x)$. By the construction of $\mathbf{B}\mathscr{X}$, we have

$$|\mathscr{D}_\tau(\mathbf{B}\mathscr{X}, |x|_B, s)| = |\mathscr{D}_S(\mathscr{X}, x, s)|$$

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 $i_{h+1} = |\mathscr{D}_S(\mathscr{X}, x, s)| - 1$. Let ϕ be the monotonic bijective correspondence between $\mathscr{D}_S(\mathscr{X}, x, s)$ and $\mathscr{D}_\tau(\mathbf{B}\mathscr{X}, |x|_B, s)$. So

$$\phi \circ \mathbf{S} \circ \textsc{del} : \mathscr{D}_\tau(\mathscr{X}, x, s) \to \mathscr{D}_\tau(\mathbf{B}\mathscr{X}, |x|_B, s)$$

is a surjective function (it is a composition of surjective functions). We can prove both claims of this Lemma by showing that $\phi \circ \mathbf{S} \circ \textsc{del}$ coincides with $\mathbf{B}$. To this end, we assume that $\mathscr{Z}$ is the nth element of the chain $\mathscr{D}_\tau(\mathbf{B}\mathscr{X}, |x|_B, s)$ and take any $\mathscr{Y} \in \mathscr{D}_\tau(\mathscr{X}, x, s)$ such that $\mathbf{S}(\textsc{del}(\mathscr{Y}))$ is the nth element of the chain $\mathscr{D}_S(\mathscr{X}, x, s)$. We shall prove that $\mathbf{B}\mathscr{Y} = \mathscr{Z}$, which trivially proves the first claim, and the second claim is implied by the surjectivity of $\phi \circ \mathbf{S} \circ \textsc{del}$.

By construction, $\mathscr{Z} = (Z, e_Z, a_Z)$, where

$$\begin{aligned} Z &= \{|y|_B \ : \ a_{BX}(|x|_B, |y|_B) \geqslant \tau^{i_1} a_1 \tau^{i_2} \dots a_h \tau^{n-1}\} \\ e_Z(|y|_B) &= \tau^{i_{h+1} - (n-1)} a_{h+1} \tau^{i_{h+2}} \dots a_m \tau^{i_{m+1}} \\ a_Z(|y|_B, |y'|_B) &= a_{BX}(|y|_B, |y'|_B) - \tau^{i_1} a_1 \tau^{i_2} \dots a_h \tau^{n-1}. \end{aligned}$$

To prove that $\mathbf{B}\mathscr{Y} = \mathscr{Z}$, we have to prove the following three facts about $\mathbf{B}\mathscr{Y}$:

(1) Its set of paths is Z:
  Let $y \in Y$. So $a_X(x, y) \geqslant \tau^{k_1} a_1 \tau^{k_2} \dots a_h \tau^k$, where $\mathscr{Y} = \mathscr{D}(\mathscr{X}, x, \tau^{k_1} a_1 \tau^{k_2} \dots a_h \tau^k)$. Hence, y corresponds to a path in the nth element of the chain $\mathscr{D}_S(\mathscr{X}, x, a_1 \dots a_h)$. So $\mathscr{D}_S(\mathscr{X}, x, a_1 \dots a_h)$ and $\mathscr{D}_S(\mathscr{X}, y, a_1 \dots a_h)$ share at least their first n elements. This implies that $a_{BX}(|x|_B, |y|_B) \geqslant \tau^{i_1} a_1 \tau^{i_2} \dots a_h \tau^{n-1}$ and hence $|y|_B \in Z$.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

Conversely, letting $|y|_B \in Z$, we have $a_{BX}(|x|_B, |y|_B) = \tau^{i_1} a_1 \tau^{i_2} \dots a_h \tau^{n-1}$. So we can find a $\mathscr{Y}'$ and $y \in Y' \cap |y|_B$ such that $\mathbf{S}(\text{DEL}(\mathscr{Y}'))$ is the nth element in $\mathscr{D}_S(\mathscr{X}, x, a_1 \dots a_h)$. But then $\mathscr{D}_S(\mathscr{X}, x, a_1 \dots a_h)$ and $\mathscr{D}_S(\mathscr{X}, y', a_1 \dots a_h)$ have at least the first n derivatives in common. Since $\mathbf{S}(\text{DEL}(\mathscr{Y}))$ is also the nth element in $\mathscr{D}_S(\mathscr{X}, x, a_1 \dots a_h)$, we can find a $y'' \in Y$ such that $\mathscr{D}_S(\mathscr{X}, y', a_1 \dots a_h)$ and $\mathscr{D}_S(\mathscr{X}, y'', a_1 \dots a_h)$ are in a bijective correspondence whose corresponding derivatives are isomorphic. But this means we have $y'' \equiv_B y' \equiv_B y$, and we are done.

(2) Its extent function is e_Z:
First note that $|\mathscr{D}_S(\mathscr{Y}, x, \epsilon)| = |\mathscr{D}_S(\mathscr{X}, x, s)| - (n-1)$. So every $|y|_B \in Z$ has extent that starts with $|\mathscr{D}_S(\mathscr{X}, x, s)| - (n-1) - 1$ τ's and is thus $e_{BX}(|x|_B) - \tau^{i_1} a_1 \tau^{i_2} \dots a_h \tau^{i_{h+1}}$. Recall that $|\mathscr{D}_S(\mathscr{X}, x, s)| - 1 = i_{h+1}$, so

$$|\mathscr{D}_S(\mathscr{X}, x, s)| - (n-1) - 1 = i_{h+1} - (n-1).$$

Hence, $e_{BY}(|y|_B) = e_Z(|y|_B)$, as required.

(3) Its agreement function is a_Z:
This is shown in a similar way to the previous point. □

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

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

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

Proof. If $\mathbf{B}\mathscr{X} \cong \mathbf{B}\mathscr{Y}$, then $\mathscr{X} \simeq_B \mathscr{Y}$ because of Lemma 7. Suppose $\mathscr{X} \simeq_B \mathscr{Y}$. 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 $\mathscr{D}_\tau(\mathscr{X}, x, s)$ is a chain and $\mathscr{D}_\tau(\mathscr{Y}, y, s)$ is the corresponding chain in $\mathscr{Y}$. By the symmetry in the first item of Definition 12, there is a correspondence between the two chains that is surjective on both sides, and, being monotonic, preserves the extremes. This correspondence induces a correspondence between $\mathscr{D}_d(\mathscr{X}, x, s)$ and $\mathscr{D}_d(\mathscr{Y}, y, s)$, via an application of DEL, that preserves order and extremes and, again, between $\mathscr{D}_S(\mathscr{X}, x, s)$ and $\mathscr{D}_S(\mathscr{Y}, y, s)$, via an application of the function $\mathbf{S}$. In this final 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. As the symmetric case can be dealt with analogously, we have an isomorphism of trees. □

To prove the converse of Theorem 2, we need to prove that the correspondence that associates $\mathbf{B}\mathscr{X}$ with $\mathscr{X}$ is an FPR *Tree*-functor. Moreover, as in the strong case, $\mathbf{B}$ can also

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

be characterised through a maximality condition amongst those that reflect factorised paths.

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

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

We are now left to prove that $\mathbf{B}$ is FPR. To this end, we fix a $\mathscr{Z}$ and a $z \in \mathbf{Beh}_\tau[\mathscr{Z}, \mathbf{B}\mathscr{X}]$. By paths reflection, we know that there exist $\mathscr{X}'$ and $x \in \mathbf{Beh}_\tau[\mathscr{X}', \mathscr{X}]$ such that $\mathbf{B}(\mathscr{X}') = \mathscr{Z}$ and $\mathbf{B}_{\mathscr{X}'\mathscr{X}}(x) = z$. We must prove that, for every such x and for every factorisation $z = z_1; z_2$ (where $z_2 \in \mathbf{Beh}_\tau[\mathscr{Z}, \mathscr{Z}']$ and $z_1 \in \mathbf{Beh}_\tau[\mathscr{Z}', \mathbf{B}\mathscr{X}]$), we can find a $\mathscr{X}''$ and a factorisation $x = x_1; x_2$ such that $x_2 \in \mathbf{Beh}_\tau[\mathscr{X}', \mathscr{X}'']$, $x_1 \in \mathbf{Beh}_\tau[\mathscr{X}'', \mathscr{X}]$, $\mathbf{B}(\mathscr{X}'') = \mathscr{Z}'$ and $\mathbf{B}_{\mathscr{X}''\mathscr{X}}(x_1) = z_1$. By Lemma 6(2), $\mathscr{Z}'$ is the branching representative of an $\mathscr{X}'' \in \mathscr{D}_\tau(\mathscr{X}, x, s)$ up-to isomorphism, for some $\mathscr{X}''$ 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 $\mathscr{X}$) is the standard branching representative of the leftmost (call it $\mathscr{Y}$). It suffices to apply the construction of Definition 16 to $\mathscr{Y}$, and let $x' = |x|_B$ and $y' = |y_1|_B = |y_2|_B$. Thus, as in the case of strong bisimulation, we can find a cospan of *Tree*-morphisms

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

where f was described intuitively in the Introduction (just before Figure 4) and g is the identity. These morphisms reflect paths with their factorisations, 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, $F_{\mathscr{I},\mathscr{X}}$ induces an epimorphism from $\mathbf{Beh}_\tau[\mathscr{I}, \mathscr{X}]$ to $\mathbf{Beh}_\tau[\mathscr{I}, F(\mathscr{X})]$ (see Lemma 2), so there is an epimorphism from $\mathscr{X}$ to $F(\mathscr{X})$. Since F reflects factorised paths, $F(\mathscr{X})$ is a quotient of $\mathscr{X}$ with respect to a branching bisimulation. But $\mathbf{B}$ induces the coarsest branching bisimulation, so we get the assertion. *Tree*-naturality of the isomorphism holds because both images are induced via quotienting on paths. □

4.3. *Standard representatives for weak bisimilarity*

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

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

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

When considered on $\equiv_B$-classes of paths, $\leqslant_B$ 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 following, we will use $|x|_W$ to denote the equivalence class $|x'|_B$ of a maximal path x' such that $|x|_B \leqslant_B |x'|_B$.

Definition 18. The *standard weak representative* of a tree $\mathscr{X} = (X, e_X, a_X)$ is the tree $\mathbf{W}\mathscr{X} = (WX, e_{WX}, a_{WX})$, 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 begin the completeness proof with a technical proposition that will be useful later. We will then follow a path very similar to those followed for the strong and branching cases.

Proposition 12. Let $\mathscr{X}$ be a tree. Then, for all $x \in X$ there is an $x' \in |x|_W$ 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, note that for all $x, y \in X$ and 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 $x' \equiv_B x$.

To prove this Proposition, we just observe 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 an x' and y' to obtain the claim. □

Lemma 8.

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

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

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

Proof. The proof is similar to the proof of Lemma 7, but using Lemma 8. □

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

Proof. If $\mathbf{W}\mathscr{X} \cong \mathbf{W}\mathscr{Y}$, then $\mathscr{X} \simeq_W \mathscr{Y}$ because of Lemma 7. Suppose $\mathscr{X} \simeq_W \mathscr{Y}$. 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. We take a maximal x. Then, for a corresponding y, we will have $|y|_B \leqslant_B |y'|_B$ for y' maximal. Let $s \leqslant e_X(x)$, with $\mathscr{D}_\tau(\mathscr{X}, x, s)$ and $\mathscr{D}_\tau(\mathscr{Y}, y, s)$ its corresponding chains. By Definition 13, there is a correspondence between the two chains that is surjective on the side of $\mathscr{D}_\tau(\mathscr{X}, x, s)$. This correspondence induces a correspondence

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

between $\mathscr{D}_d(\mathscr{X}, x, s)$ and $\mathscr{D}_d(\mathscr{Y}, y, s)$, via an application of DEL, that preserves order and, again, between $\mathscr{D}_S(\mathscr{X}, x, s)$ and $\mathscr{D}_S(\mathscr{Y}, y, s)$, via an 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 $\mathscr{D}_S(\mathscr{Y}, y, s)$ to $\mathscr{D}_S(\mathscr{Y}, y', s)$, so we have an injection from $\mathscr{D}_S(\mathscr{X}, x, s)$ to $\mathscr{D}_S(\mathscr{Y}, y', s)$. 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. Comparing $|x|_B$ with $|x'|_B$, they are maximal elements of the same $\leqslant_B$-chain, so they are equal. This implies that $e(|x'|_B) = e(|y'|_B)$, and the correspondence between paths in $\mathbf{W}\mathscr{X}$ and $\mathbf{W}\mathscr{Y}$ is a bijection preserving extent. Similar reasoning proves the preservation of agreement. □

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

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

Proof. The desired functor is defined as **W** on objects, and Lemma 8(1) allows us to define it on arrow objects. The fact that it is FPWR follows from Lemma 8(2). □

Example 8. Consider the two trees in Figure 5. The rightmost (call it $\mathscr{X}$) is the standard weak representative of the leftmost (call it $\mathscr{Y}$), once we let $y' = |y|_W$ and $x' = |x_1|_W = |x_2|_W$. Indeed, the standard branching representative of $\mathscr{Y}$ is itself. Moreover, the pruning defined in Definition 18 reduces it to $\mathscr{X}$, 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 cospan of *Tree*-morphisms

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

where f was described intuitively in the Introduction (just before Figure 5) and g is the identity. However, f only reflects paths with their factorisations 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. The proof is similar to the proof of Proposition 11. □

5. Applications

The theory developed so far does not need us to make any assumptions about the alphabet A. We now show that, by properly instantiating it, we can easily model LTS and labelled (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.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

5.1. *A tree-based model for LTS*

Definition 19. A *labelled 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.*
— $s_0 \in S$ 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 $\mathscr{S}$, we write:

— $\Longrightarrow$ for the reflexive and transitive closure of $\xrightarrow{\tau}$;
— $s \stackrel{\mu}{\Longrightarrow} u$ if there exist s', u' such that $s \Longrightarrow s' \xrightarrow{\mu} u' \Longrightarrow u$;
— $s \stackrel{\hat{\mu}}{\Longrightarrow} u$ 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 $\mathfrak{R} \subseteq S \times S$ is a:

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

We will 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 labelled transition systems. To do this, we will introduce the notion of the unfolding of an LTS, and then prove that the classical equivalences on LTS correspond to the ones we have defined when we apply them to the trees obtained as unfoldings of LTS.

We define the *unfolding* of $\mathscr{S}$ to be $\textsc{unf}(\mathscr{S}) = (\textsc{runs}(\mathscr{S}), e_{\mathscr{S}}, a_{\mathscr{S}})$, where:

— $\textsc{runs}(\mathscr{S}) = \{s_0 \mu_1 s_1 \mu_2 \ldots \mu_n s_n \ : \ s_0 \xrightarrow{\mu_1} s_1 \xrightarrow{\mu_2} \cdots \xrightarrow{\mu_n} s_n\}$.
— $e_{\mathscr{S}}(s_0 \mu_1 s_1 \ldots \mu_n s_n) = \mu_1 \ldots \mu_n$.
— $a_{\mathscr{S}}(s_0 \mu_1 s_1 \ldots \mu_n s_n, s_0 v_1 u_1 \ldots v_m u_m) = \mu_1 \ldots \mu_l$ where, for all $k \leqslant l$, we have $v_k = \mu_k$, $s_k = u_k$ and either $\mu_{l+1} \neq v_{l+1}$ or $s_{l+1} \neq u_{l+1}$ or $l = n$ or $l = m$.

By definition, $\textsc{unf}(\mathscr{S})$ is a tree, which 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 $\textsc{unf}(\mathscr{S})$ have finite length.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

We now prove that this simple modelling of LTS via trees enjoys full abstraction with respect to strong, branching and weak bisimilarities. In the following proofs, we shall use $\mathscr{S}\downarrow_s$ to denote the LTS obtained from $\mathscr{S}$ by letting its initial state be s. Moreover, we shall write $s \not\longrightarrow$ to mean that there exists no μ or s' such that $s \xrightarrow{\mu} s'$.

Lemma 10. Let $x = s_0\mu_1 s_1 \dots \mu_n s_n \in \text{RUNS}(\mathscr{S})$. Then:

$$\mathscr{D}(\text{UNF}(\mathscr{S}), x, \mu_1 \dots \mu_k) \cong \text{UNF}(\mathscr{S}\downarrow_{s_k})$$

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 will now prove that

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

is the set of paths in $\mathscr{D}(\text{UNF}(\mathscr{S}), x, \mu_1 \dots \mu_k)$. The result is then straightforward as it is easy to define a bijection between D and the paths of $\text{UNF}(\mathscr{S}\downarrow_{s_k})$ such that, trivially, corresponding paths have the same extent and the same agreement with other paths in the same tree.

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

Conversely, let

$$y = s_0\mu_1 s_1 \dots \mu_k s_k \bar{\mu}_1 \bar{s}_1 \dots \bar{\mu}_m \bar{s}_m \in D.$$

By definition,

$$s_k \bar{\mu}_1 \bar{s}_1 \dots \bar{\mu}_m \bar{s}_m \in \text{RUNS}(\mathscr{S}\downarrow_{s_k}),$$

so $y \in \text{RUNS}(\mathscr{S})$. Now we can trivially conclude that y is a path of $\mathscr{D}(\text{UNF}(\mathscr{S}), x, \mu_1 \dots \mu_k)$, since $a_{\mathscr{S}}(x, y) \geqslant \mu_1 \dots \mu_k$. □

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

Proof.

'If': We have to prove that the relation $\mathfrak{R} = \{(\mathscr{S}, \mathscr{S}') \ : \ \text{UNF}(\mathscr{S}) \simeq_S \text{UNF}(\mathscr{S}')\}$ is a strong bisimulation on LTS. Consider a step $s_0 \xrightarrow{\mu} s_1$ in $\mathscr{S}$. 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 we do not have $s_1 \not\longrightarrow$). By strong bisimulation on trees, there exists a path $y \in \text{RUNS}(\mathscr{S}')$ whose extent starts with μ such that $\mathscr{D}(\text{UNF}(\mathscr{S}), x, \mu) \simeq_S \mathscr{D}(\text{UNF}(\mathscr{S}'), y, \mu)$. 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. It is then easy to conclude since, by Lemma 10,

$$\text{UNF}(\mathscr{S}\downarrow_{s_1}) \cong \mathscr{D}(\text{UNF}(\mathscr{S}), x, \mu) \simeq_S \mathscr{D}(\text{UNF}(\mathscr{S}), x, \mu) \cong \text{UNF}(\mathscr{S}'\downarrow_{s'_1})$$

and, by definition, $(\mathscr{S}\downarrow_{s_1}\ ,\ \mathscr{S}'\downarrow_{s'_1}) \in \mathfrak{R}$.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

'Only if': We have to prove that the relation

$$\mathfrak{R} = \{(\text{UNF}(\mathscr{S}), \text{UNF}(\mathscr{S}')) \ : \ \mathscr{S} \sim_S \mathscr{S}'\}$$

is a strong bisimulation on trees. So we pick up an $x \in \text{RUNS}(\mathscr{S})$ 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. So $y = s'_0\mu_1 s'_1 \cdots \mu_n s'_n \in \text{RUNS}(\mathscr{S}')$. Moreover, y has the same extent as x and $s_n \not\rightarrow$ if and only if $s'_n \not\rightarrow$. Thus, by Lemma 10, the derivatives along x are all in relation $\mathfrak{R}$ with the corresponding derivatives along y, which is enough to complete the proof. □

Lemma 11.

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

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

Proof.

(1) To show a contradiction, we assume $k < j$. This means that $\mathscr{Y}_j$ is a τ-derivative of $\mathscr{Y}_k$. Since $\mathscr{Y}_j \not\simeq_B \mathscr{Y}_k$, there exists a $\bar{y}$ in $\mathscr{Y}_k$ that cannot be simulable in $\mathscr{Y}_j$. However, $\mathscr{X}_{i+1} \simeq_B \mathscr{Y}_k$, so there exists a $\bar{x}$ in $\mathscr{X}_{i+1}$ that simulates $\bar{y}$. But $\bar{x}$ is also a path of $\mathscr{X}_i$. Since $\mathscr{X}_i \simeq_B \mathscr{Y}_j$, there exists a $\bar{y}'$ in $\mathscr{Y}_j$ that simulates $\bar{x}$, but, by transitivity, $\bar{y}'$ also simulates $\bar{y}$, which gives a contradiction.

(2) We know that since $\mathscr{X} \simeq_B \mathscr{Y}$, every element of $\mathscr{D}_\tau(\mathscr{X}, x, s)$ (which is a chain of τ-derivatives $\mathscr{X}_0, \mathscr{X}_1, \cdots, \mathscr{X}_m$) is branching bisimilar to an element of $\mathscr{D}_\tau(\mathscr{Y}, y, s)$ (which is a chain of τ-derivatives $\mathscr{Y}_0, \mathscr{Y}_1, \cdots, \mathscr{Y}_n$). We have to prove that $\mathscr{X}_0 \simeq_B \mathscr{Y}_0$. Let $\mathscr{X}_0 \simeq_B \mathscr{Y}_h$. If $h = 0$, we are done. Otherwise, by part (1) of this Lemma, $\mathscr{Y}_0, \ldots, \mathscr{Y}_{h-1}$ must all be branching bisimilar to an element of $\mathscr{D}_\tau(\mathscr{X}, x, s)$ that is branching bisimilar to $\mathscr{X}_0$. The result is then easy by transitivity. □

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

(1) $\mathscr{S} \approx_B \mathscr{S}'$ if and only if $\text{UNF}(\mathscr{S}) \simeq_B \text{UNF}(\mathscr{S}')$.

(2) $\mathscr{S} \approx_W \mathscr{S}'$ if and only if $\text{UNF}(\mathscr{S}) \simeq_W \text{UNF}(\mathscr{S}')$.

Proof. We will only prove the first claim, which is more complex.

'If': We have to prove that the relation

$$\mathfrak{R} = \{(\mathscr{S}, \mathscr{S}') \ : \ \text{UNF}(\mathscr{S}) \simeq_B \text{UNF}(\mathscr{S}')\}$$

is a branching bisimulation on LTS. Consider a step $s_0 \xrightarrow{\mu} s_1$ in $\mathscr{S}$. Let x be $s_0\mu s_1$ if $s_1 \not\rightarrow$, and $s_0\mu s_1\bar{\mu}\bar{s}$ (for any $\bar{\mu}$ and $\bar{s}$ such that $s_1 \xrightarrow{\bar{\mu}} \bar{s}$) otherwise. Let $\mathscr{X}$ be $\mathscr{D}(\text{UNF}(\mathscr{S}), x, \mu)$, which, by Lemma 10, is isomorphic to $\text{UNF}(\mathscr{S}\downarrow_{s_1})$. We consider two cases:

$\mu \neq \tau$:

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

We have $\mathscr{X} \in \mathscr{D}_\tau(\text{UNF}(\mathscr{S}), x, \mu)$. By hypothesis, there exists a y in $\text{UNF}(\mathscr{S}')$ such that $e_{\mathscr{S}'}(y) = \tau^m \mu \tau^n w$ and $\mathscr{X} \simeq_B \mathscr{Y}$, for some $\mathscr{Y} \in \mathscr{D}_\tau(\text{UNF}(\mathscr{S}'), y, \mu)$. 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 $s'_0 (\overset{\tau}{\longrightarrow})^m \overset{\mu}{\longrightarrow} (\overset{\tau}{\longrightarrow})^n u_3$ and s'_0 is the starting state of $\mathscr{S}'$, which allows us to apply Lemma 10 freely in the following. Let $s'_0 (\overset{\tau}{\longrightarrow})^m u_1 \overset{\mu}{\longrightarrow} u_2 (\overset{\tau}{\longrightarrow})^n u_3$. By Lemma 10,

$$\begin{aligned} \mathscr{Y}_1 &= \mathscr{D}(\text{UNF}(\mathscr{S}'), y, \tau^m) \cong \text{UNF}(\mathscr{S}' \downarrow_{u_1}) \\ \mathscr{Y}_2 &= \mathscr{D}(\text{UNF}(\mathscr{S}'), y, \tau^m \mu) \cong \text{UNF}(\mathscr{S}' \downarrow_{u_2}) \\ \mathscr{Y} &= \mathscr{D}(\text{UNF}(\mathscr{S}'), y, \tau^m \mu \tau^n) \cong \text{UNF}(\mathscr{S}' \downarrow_{u_3}). \end{aligned}$$

Since $\mathscr{X} \simeq_B \mathscr{Y}$, we have $(\mathscr{S} \downarrow_{s_1}, \mathscr{S}' \downarrow_{u_3}) \in \mathfrak{R}$. By Definition 12.1(b), $\text{UNF}(\mathscr{S}) \simeq_B \mathscr{Y}_1$ (indeed, the only tree in $\mathscr{D}_\tau(\text{UNF}(\mathscr{S}), x, \epsilon)$ is $\text{UNF}(\mathscr{S})$, since $\mu \neq \tau$), which implies that $(\mathscr{S}, \mathscr{S}' \downarrow_{u_1}) \in \mathfrak{R}$. To conclude, we observe that $\mathscr{X}$ and $\mathscr{Y}_2$ are the maximum elements in $\mathscr{D}_\tau(\text{UNF}(\mathscr{S}), x, \mu)$ and $\mathscr{D}_\tau(\text{UNF}(\mathscr{S}'), y, \mu)$, respectively, so, by Lemma 11(2), we have $(\mathscr{S} \downarrow_{s_1}, \mathscr{S}' \downarrow_{u_2}) \in \mathfrak{R}$, which is enough to complete the proof for this part.

$\mu = \tau$:

We have $\mathscr{X} \in \mathscr{D}_\tau(\text{UNF}(\mathscr{S}), x, \epsilon)$. By hypothesis, there exists a $y \in \text{RUNS}(\mathscr{S}')$ such that $\mathscr{X} \simeq_B \mathscr{Y}$ for some $\mathscr{Y} \in \mathscr{D}_\tau(\text{UNF}(\mathscr{S}'), y, \epsilon)$. If $\mathscr{Y} = \text{UNF}(\mathscr{S}')$, we then conclude that $(\mathscr{S} \downarrow_{s_1}, \mathscr{S}') \in \mathfrak{R}$. Otherwise, $\mathscr{Y}$ is a τ-derivative of $\text{UNF}(\mathscr{S}')$, so we can find $\mathscr{Y}_1$ and $\mathscr{Y}_2$ in $\mathscr{D}_\tau(\text{UNF}(\mathscr{S}'), y, \epsilon)$ such that $\text{UNF}(\mathscr{S}')$, $\mathscr{Y}_1$, $\mathscr{Y}_2$ and $\mathscr{Y}$ appear in this order in the chain associated with $\mathscr{D}_\tau(\text{UNF}(\mathscr{S}'), y, \epsilon)$. By Lemma 10, we have $\mathscr{Y}_1 \cong \text{UNF}(\mathscr{S}' \downarrow_{u_1})$, $\mathscr{Y}_2 \cong \text{UNF}(\mathscr{S}' \downarrow_{u_2})$ and $\mathscr{Y} \cong \text{UNF}(\mathscr{S}' \downarrow_{u_3})$, for $s'_0 \Longrightarrow u_1 \overset{\mu}{\longrightarrow} u_2 \Longrightarrow u_3$. Since $\mathscr{X} \simeq_B \mathscr{Y}$, we have $(\mathscr{S} \downarrow_{s_1}, \mathscr{S}' \downarrow_{u_3}) \in \mathfrak{R}$. Since $\text{UNF}(\mathscr{S}') \simeq_B \text{UNF}(\mathscr{S})$ and $\mathscr{Y} \simeq_B \mathscr{X}$, by virtue of Lemma 11(1), we have that the elements in $\mathscr{D}_\tau(\text{UNF}(\mathscr{S}), x, \epsilon)$ that are branching bisimilar to $\mathscr{Y}_1$ and $\mathscr{Y}_2$ can only be (branching bisimilar to) $\text{UNF}(\mathscr{S})$ and $\mathscr{X}$, respectively, so $(\mathscr{S}, \mathscr{S}' \downarrow_{u_1})$ and $(\mathscr{S} \downarrow_{s_1}, \mathscr{S}' \downarrow_{u_2})$ are both in $\mathfrak{R}$, which is enough to complete the proof for this part.

'Only if': We have to prove that the relation $\mathfrak{R} = \{(\text{UNF}(\mathscr{S}), \text{UNF}(\mathscr{S}')) \ : \ \mathscr{S} \approx_B \mathscr{S}'\}$ is a branching bisimulation on trees. Choose $x = s_0 \mu_1 s_1 \cdots \mu_n s_n \in \text{RUNS}(\mathscr{S})$. From the hypothesis, we know that we can find a path

$$y = u_0^1 \mu_1^1 u_1^1 \cdots \mu_1^{k_1} u_1^{k_1} \mu_2^1 u_2^1 \cdots \mu_2^{k_2} u_2^{k_2} \cdots \mu_n^1 u_n^1 \cdots \mu_n^{k_n} u_n^{k_n} \in \text{RUNS}(\mathscr{S}')$$

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 $\mu_i^{k'_i+1} = \mu_i$, and, 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, $\text{DEL}(e_{\mathscr{S}}(x)) = \text{DEL}(e_{\mathscr{S}'}(y))$. Moreover, it is not difficult to prove that for every $w \preccurlyeq \mu_1 \cdots \mu_n$ and every $\mathscr{X} \in \mathscr{D}_\tau(\text{UNF}(\mathscr{S}), x, w)$, there exists a $\mathscr{Y} \in$

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

$\mathscr{D}_\tau(\text{UNF}(\mathscr{S}'), y, w)$ such that $(\mathscr{X}, \mathscr{Y}) \in \mathfrak{R}$, and *vice versa*, which is enough to complete the proof.

5.2. *Labelled event structures*

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

Definition 21.

(1) A *labelled (prime) event structure* is a tuple $\mathscr{E} = (E, \prec, \#, L, \ell)$ where:
   (a) E is a set of so-called *events* taken from a universe Ev.
   (b) $\prec$ and $\#$ are two binary relations on E, which are called *causality* and *conflict* and for which $\prec \cap \# = \varnothing$.
   (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 and 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 $\# = \varnothing$.
(3) A *configuration* in $\mathscr{E}$ 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, labelling-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, labelled 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 $e \in E$ labelled 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.
— $d \# e$ 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, which explain which subsets of events constitute possible (partial) runs of the represented system. Thus, configurations must be conflict-free; they must be closed with respect to causal predecessors; and all prerequisites for any event occurring in the run must also have occurred.

It is well known (see, for example, Winskel and Nielsen (1995) and Roggenbach and Majster-Cederbaum (2000)) that we can consider a (prime) labelled event structure as a transition system, labelled 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 to the second configuration. Notice that

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

this intuitive interpretation is in some sense dual to the one given for processes: there, when going from the first to the second process by performing an action, this action (and possibly many others) are erased from the possible behaviour of the second process. Here, in some sense, we are supposed to find the first process in the second one, together with a record of what has been performed.

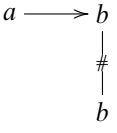
Definition 22. A (prime) labelled event structure $\mathscr{E} = (E, \prec, \#, l)$ generates the pomset-labelled transition system $T_{Pom}(\mathscr{E}) = (Conf(\mathscr{E}), Pom_{Ev,L}, \rightarrow, \varnothing)$, where, for every $\chi = (X, \prec, L, \ell)$ and $\chi' = (X', \prec', L, \ell')$ in $Conf(\mathscr{E})$, we let $\chi \xrightarrow{p} \chi'$ if and only if $X \subset X'$ 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) labelled event structures $\mathscr{E} = (E, \prec, \#, l)$ and $\mathscr{F} = (F, \prec, \#, l)$, are pomset bisimilar if there is a relation $R \subset E \times F$ such that for every $(\chi, \zeta) \in R$ and $p \in Pom_{Ev,L}$, we have:

(i) If $\chi \in Conf(\mathscr{E})$, $\chi \xrightarrow{p} \chi'$, there is $\zeta' \in Conf(\mathscr{F})$ such that $\zeta \xrightarrow{p} \zeta'$, $(\chi', \zeta') \in R$.
(ii) If $\zeta \in Conf(\mathscr{F})$, $\zeta \xrightarrow{p} \zeta'$, there is $\chi' \in Conf(\mathscr{E})$ such that $\chi \xrightarrow{p} \chi'$, $(\chi', \zeta') \in R$.

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

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

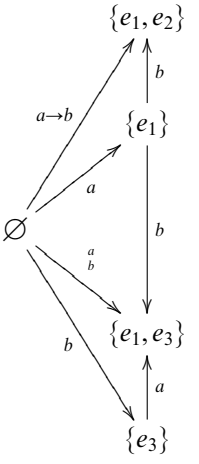


where '$\longrightarrow$' denotes causality and '$—\#—$' denotes conflict. Formally, $\mathscr{E}$ is defined as the 5-tuple $(E, \prec, \#, L, \ell)$, where:

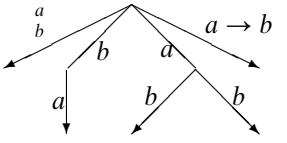
— $E = \{e_1, e_2, e_3\}$.
— $e_1 \prec e_2$.
— $e_2 \# e_3$ and $e_3 \# e_2$.
— $L = \{a, b\}$.
— $\ell(e_1) = a$ and $\ell(e_2) = \ell(e_3) = b$.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

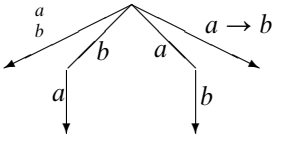
The configurations of $\mathscr{E}$ are $\varnothing$, $\{e_1\}$, $\{e_3\}$, $\{e_1, e_2\}$ and $\{e_1, e_3\}$, which correspond to the pomsets $\varnothing$, a, b, $a \to b$ and $\overset{a}{b}$. Thus, we have that $T_{Pom}(\mathscr{E})$ is the LTS



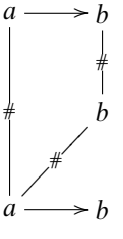
The unfolding of this LTS is the pomset-labelled tree



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



Now consider the following event structure $\mathscr{F}$:

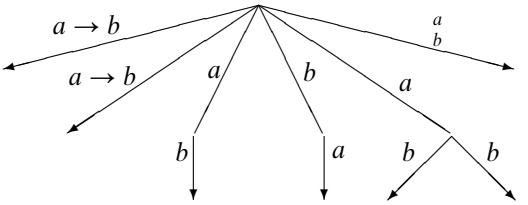


https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

Formally, $\mathscr{F}$ 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 $\mathscr{F}$ are $\varnothing$, $\{f_1\}$, $\{f_3\}$, $\{f_4\}$, $\{f_1, f_2\}$, $\{f_1, f_3\}$ and $\{f_4, f_5\}$, which correspond to the pomsets $\varnothing$, a, b, $a \to b$ and $\overset{a}{{}_{b}}$. We leave the reader to construct $T_{Pom}(\mathscr{E})$ and just provide its unfolding, that is, the tree



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

6. Related work

In this section we will compare our approach with other mathematical models for bisimulation. First, we should say (borrowing the terminology from category theory) that we take a 'local' automata-theoretic approach by modelling and comparing processes in terms of their local behaviour: 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 other approaches to modelling bisimulations have considered comparisons between 'global' process behaviours: to each state, they associate the structured set of possible evolutions into all the other states. Another distinguishing feature of our approach is that we aim to classify bisimulations, so 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 a more detailed comparison, we will begin by considering the categorical characterisation of bisimulation-based equivalences as *open maps* (Joyal *et al.* 1993). In this 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**, which represent 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, which is open with respect to **P**: this means that if a path can be extended in the image behaviour, 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

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

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. The notion of path reflection we require for *Tree*-functors led us to prove Lemmas 4, 6 and 8, which amount to requiring a kind of 'openness property'. A possible formulation of this property (Joyal *et al.* 1993) requires that, for a given open morphism $\sigma : T \to T'$ between LTSs,

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

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

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

Another possible modelisation of bisimulations has been proposed by relying on coalgebras (see, for example, Rutten (2000) and Roggenbach and Majster-Cederbaum (2000)). In this approach, labelled transition systems are modelled 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 a homomorphism for a coalgebra is strictly related to the notion of an open map when the simulations used are total functions. The relationships between the open maps and the coalgebraic approach were carefully studied in Lasota (2002), where it was shown that open maps correspond to coalgebra morphisms for a suitably chosen endofunctor in a category of many-sorted sets.

Pavlović (1995; 1996) exploited the basic intuitions underlying open maps and coalgebras 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 map. Equivalently, the two systems are bisimilar if there exists a cospan of bisimulation morphisms between them, that is, a common quotient. It was 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 were considered in Pavlović (1996). As in the case of strong bisimilarity, the functor inducing the quotient is the initial one and enjoys the property of reducing the relations between objects induced by spans to equality. 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

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

for strong bisimulation, and the representatives are the minimal ones; the latter holds for branching and weak bisimulation, and the representatives are not always minimal.

The coalgebraic and 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 $\mathscr{C}$ can be represented by a pair of maps $X \leftarrow R \rightarrow Y$, that is, a span, obtained by composing the inclusion morphism with the projections. However, if the category under consideration has appropriate properties, one can introduce an equivalence relation $\mathscr{R} \rightarrowtail \mathscr{C} \times \mathscr{C}$ and consider the quotient $\mathscr{C}/\mathscr{R}$ – this was done in Pavlović (1995; 1996). But one can do even better than this since if a kind of 'homomorphism theorem' holds, one can prove that the quotient $\mathscr{C}/\mathscr{R}$ is induced by an endofunctor identifying equivalent objects.

It is this last intuition that underlies 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 $\mathscr{X} \rightarrow F(\mathscr{X}) \cong F(\mathscr{Y}) \leftarrow \mathscr{Y}$. By putting a suitable lifting factorisation condition on F, we guarantee that this cospan, which locally inherits the suitable lifting factorisation 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, which quotients as much as possible, does exist for every class. The representatives corresponding to the terminal relation in their class (that is, the *Tree*-functors **S**, **B** and **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 fill out representatives with extra paths, but only have 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 factorisation lifting property of **S** and **B**, they can replace their originals in every universal and couniversal construction (Kasangian and Labella 2010).

Another feature of our approach is that our trees can be seen as presentations of sheaves; hence, it is 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 Factorisation Lifting* (ULF) property is imposed – they then characterise 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 are thus more abstract, but simpler, than sheaves (because they lack restrictions). On the other hand, our *Tree*-categories **Beh** and $\mathbf{Beh}_\tau$ enjoy a sort of UFL property in the sense that their unique *Tree*-functor to the terminal *Tree*-category **T** enjoys path factorisation lifting (see Proposition 4). Actually, UFL holds in the case of **Beh**, and a more relaxed variant holds for $\mathbf{Beh}_\tau$.

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

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

7. Conclusions

We have presented a characterisation of strong, branching and weak bisimulation on a tree-based model via enriched functors enjoying a factorised 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, which is a very natural tool for going from one model to another once the local point of view is assumed.

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

— considered infinite (regular) structures;

— also provided a functorial characterisation for weak bisimilarity;

— simplified and clarified the characterisation of branching bisimilarity.

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

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

where:

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

— FPWR is the factorised path weakly reflection property, which requires that, to each factorisation of a target path, there corresponds a factorisation of *one* of the associated source paths.

— FPR is the factorised path reflection property, which requires PR and that all the intermediate states of each factorisation have a corresponding state path in *all* the associated source paths.

— UFL is the *unique factorisation lifting* property, which corresponds to FPR plus the requirement that the reflected factorisation is unique up-to isomorphism – it is worth noting that FPR collapses on UFL when all actions are visible.

The requirement that every possible factorisation of a target path is reflected in an appropriate factorisation of its source path corresponds to a Conduché property (Conduché 1972) for enriched functors. A very rigid formulation of it was used in Lawvere (1986) to characterise determinacy of physical systems and to explain determinacy on states: in his view, a physical transformation is deterministic if it can be 'controlled'

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

by a Conduché functor[†] in the sense that it is always possible to recover the sequence of states traversed during system evolution.

In 'ordinary' (that is, non-enriched) categories, the Conduché property characterises those functors that have a right adjoint to the inverse image functor associated with them (Conduché 1972). In our setting, the existence of a right adjoint (which was proved in Kasangian and Labella (2010)) 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', and 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:

— Weak bisimilarity only enjoys a weaker form of lifting factorisation property.
— Strong bisimilarity enjoys uniqueness of lifted factorisations (it faithfully keeps track of every state traversed along a path).
— Branching bisimilarity enjoys a lifting factorisation 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 the strong determinacy of processes, while strong bisimilarity does. Moreover, we have also formally detected a close correspondence between strong and branching bisimilarity in terms of the preservation of system determinacy.

References

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

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

Bunge, M. and Fiore, M. P. (2000) Unique factorisation lifting functors and categories of linearly-controlled processes. *Mathematical Structures in Computer Science* **10** (2) 137–163.

Cattani, G. L. and Winskel, G. (1997) Preshaf models for concurrency. Proc. of CSL '96. *Springer-Verlag Lecture Notes in Computer Science* **1258** 106–126.

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

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

De Nicola, R., Montanari, U. and Vaandrager, F. (1990) Back and forth bisimulations. In: Proc. of CONCUR'90. *Springer-Verlag Lecture Notes in Computer Science* **458** 152–165.

[†] Actually, Lawvere deals with the UFL property, which he called the *Moebius property*.

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press

De Nicola, R. and Labella, A. (1998) Tree Morphisms and Bisimulations. In: Proc. of MFCS'98 Workshop on Concurrency. *Electronic Notes in Theoretical Computer Science* **18**.

van Glabbeeck, R. and Goltz, U. (1990) Equivalences and refinement. In: Semantics of Systems of Concurrent Processes. *Springer-Verlag Lecture Notes in Computer Science* **469** 309–333.

van Glabbeek, R. and Weijland, W. (1989) Branching Time and Abstraction in Bisimulation Semantics. *Journal of the ACM* **43** (3) 555–600.

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

Kasangian, S. and Labella, A. (1999) Observational trees as models for concurrency. *Mathematical Structures in Computer Science* **9** 687–718.

Kasangian, S. and Labella, A. (2010) Conduché property and Tree-based categories. *Journal of Pure and Applied Algebra* **214** 221–235.

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

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

Lawvere, F. W. (1986) State categories and response functors (unpublished manuscript).

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

Park, D. (1981) Concurrency and automata on infinite sequences. In: Proc. of Theoretical Computer Science. *Springer-Verlag Lecture Notes in Computer Science* **104** 167–183.

Pavlović, D. (1995) Convenient Category of Processes and Simulations 1: Modulo Strong Bisimilarity. In: Proc. of Category Theory and Computer Science. *Springer-Verlag Lecture Notes in Computer Science* **953** 3–23.

Pavlović, D. (1996) Convenient categories of asynchronous processes and simulations II. In: Edalat, A. (ed.) *Theory and Formal Methods of Computing '96*, World Scientific 156–167.

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

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

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

Winskel, G. (1988) An introduction to event structures. In: Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency. *Springer-Verlag Lecture Notes in Computer Science* **354** 364–397.

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

https://doi.org/10.1017/S0960129509990272 Published online by Cambridge University Press