Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 347 (2019) 5–24

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

# Bisimulation Maps in Presheaf Categories

Harsh Beohar[1]

*Universität Duisburg-Essen*
*Duisburg, Germany*

Sebastian Küpper[2]

*FernUniversität in Hagen*
*Hagen, Germany*

**Abstract**

The category of presheaves on a (small) category is a suitable semantic universe to study behaviour of various dynamical systems. In particular, presheaves can be used to record the executions of a system and their morphisms correspond to simulation maps for various kinds of state-based systems. In this paper, we introduce a notion of bisimulation maps between presheaves (or executions) to capture well known behavioural equivalences in an abstract way. We demonstrate the versatility of this framework by working out the characterisations for standard bisimulation, ∀-fair bisimulation, and branching bisimulation.

*Keywords:* Presheaves, ∀-fair bisimulation, Branching bisimulation.

## 1 Introduction

The importance of formal semantics should not be underestimated, especially when aimning to design *reliable* dynamical systems in heterogeneous environments. Therefore, a variety of state based modelling frameworks at different levels of abstraction have been proposed; to quote Goguen [16]: *one person's syntax is another person's semantics*. Diversity in algorithms can be desirable; however, as argued in [1,16,34], the proliferation of semantic theories indicates our scattered understanding of concurrent systems. Thus, we seek a framework that provides semantic structure describing the behaviour of a dynamical system and its refinement independently of syntax.

This goal is shared to an extent by the theory of coalgebras [29]. In [6], we abandoned state-based modelling in favour of describing behaviour as the set of

[1] Email: harsh.beohar@uni-due.de
[2] Email: sebastian.kuepper@feu.de

https://doi.org/10.1016/j.entcs.2019.09.002

executions (inspired by [10, 28]) because the branching structure of a state in the presence of invisible actions is described by the set of executions (not states). This situation further escalates when one is interested in infinite executions (e.g. fairness properties [17, 25]) or dense executions which are omnipresent in hybrid systems (e.g. [12]). The point is not that executions are inexpressible in a coalgebra, but rather that we need a semantic framework where they are treated as first-class citizens just as states are in a coalgebraic framework. Thus, our hypothesis is that behaviour of a system is given solely by its executions.

We anticipate presheaves to be the "right" semantic structure to study executions without fixing a kind of dynamical system. Note that we are *not* the first in proposing presheaves as the mathematical universe to studying behaviour. Winskel and his colleagues [8, 15, 19, 23, 34] have already employed presheaves (among other things) by giving a denotational semantics of process-algebraic terms supported by characterisations of strong bisimulation and weak bisimulation relations using open maps in the context of transition systems. For a more modern treatment, Hirschowitz and his colleagues [13, 14, 20] advocated game semantics using (pre)sheaves.

The novelty of our work lies in refining the notion of open maps (which we christened *bisimulation maps*) in a presheaf category and using it to characterise $\forall$-fair bisimulation [18, 24] and branching bisimulation [33] relations. In addition, the prospect of having to specify notions of time and observation (which was absent in the earlier works on presheaf semantics) leads to a clearer modelling, so explicitly highlighting these two dimensions of system modelling is at the core of our contribution (cf. Section 2). This distinction was in turn essential to capture branching bisimulation in the presence of invisible actions $\tau$.

Interestingly, unlike open maps, bisimulation maps are always retracts in the category of presheaves (in turn, they are surjective at the level of executions). As a slogan, presheaf maps are refinement maps, while bisimulation maps (which are special presheaf maps) are complete refinement maps (Section 3). By moving to a finer notion, we are still able to capture functional bisimulations without fairness. However, in the context of fairness, we can show (Theorem 4.7) that the behavioural equivalence induced by a bisimulation map coincides with $\forall$-fair bisimulation relation. Note that our $\forall$-fair bisimulations are equivalence relations by definition in contrast to the existing definition [18, 24] (see the dicussion after Theorem 4.7 on Page 10). This is an improvement with respect to the previous characterisation of $\exists$-fair bisimulation [18, 24] (called *extended bisimulation* in [17]) obtained by Hildebrandt [19] using open maps, since any $\forall$-fair bisimulation relation is strictly finer than an $\exists$-fair bisimulation relation [18] and our correspondence does not impose any restrictions on the fairness predicates. These restrictions, originally from [17], asserted that fairness predicates on infinite executions are closed under the removal and the addition of finite prefixes.

Another practical aspect of the theory of presheaves is that it guides us in finding the right semantic categories once a notion of time and observation is fixed. Moreover, we can apply concepts (like, e.g., essential geometric morphism [27]) that

transform a dynamical system from one observation space $\mathcal{O}$ to another space $\mathcal{O}'$. This way we can transform (see Section 5) a presheaf of executions (induced by a given transition system) into a presheaf of minimal executions (i.e., executions in which trailing $\tau$-transitions are chopped off). This property is specific to branching bisimulation, which may be the reason why this construction was not discussed in [15] (their objective was to capture weak bisimulation).

**Organisation of the paper.**

In Section 2, we introduce our mathematical framework to model behaviour of a dynamical system with a special focus on the aspects of time and observation. Then, we introduce the notion of bisimulation maps in presheaves on an arbitrary (small) category in Section 3. Turning our attention towards the first major example, we characterise $\forall$-fair bisimulation relations in Section 4. The case of invisible actions in Section 5 is based on a change of observation space. We first outline an obvious (but ultimately failed) attempt to capture branching bisimulation, before giving the correct (yet intuitive) construction that characterises branching bisimulation.

## 2 Our universe of discourse

The objective of this section is to describe our semantic framework in which one can model behaviour of a dynamical system. By *behaviour* of a dynamical system, we understand some *phenomena that evolve over time*. Our aim is to formalise this intuition. We begin by modelling time as a small category $\mathbf{T}$, whose objects are points in time and arrows describe passing of time.

**Notation 2.1** *An object $C$ (an arrow $f$) of a category $\mathbf{C}$ will be denoted by the predicate $C \in \mathbf{C}$ ($f \in \mathbf{C}$). Moreover, the codomain and domain of an arrow $f \in \mathbf{C}$ are denoted as $cod(f)$ and $dom(f)$, respectively.*

Invariably, dynamical systems come with a notion of observation. For instance, a letter from a fixed alphabet may denote the assignment of model variables in a computer program/controller. We assume that a system under study has a display unit together with the existence of a hypothetical 'observer' $\mathcal{O}$ who is watching/measuring behaviour of the system using this display unit over time. In addition, our observer $\mathcal{O}$ can remember its observations over time, i.e., earlier observations can be deduced from the later observations. Mathematically, this amounts to saying that $\mathcal{O}$ is a contravariant functor $\mathbf{T} \longrightarrow \mathbf{Set}$.

**Proposition 2.2** *Let $\mathbf{C}$ be a small category. Then, the collection of functors of type $\mathbf{C}^{op} \longrightarrow \mathbf{Set}$ (i.e., presheaves on a category $\mathbf{C}$) and natural transformations between them form a category $\mathbf{PSh}(\mathbf{C})$.*

**Notation 2.3** *Given a presheaf $F \in \mathbf{PSh}(\mathbf{C})$, we follow [27] in writing $x \cdot f$ to denote the restriction of $x \in FC$ along $C' \xrightarrow{f} C$, i.e., $Ff(x) = x \cdot f$. In case $\mathbf{C}$ is a poset (viewed as a category) $\mathbf{C}$, we write $x \cdot C'$ to denote the restriction of $x \in FC$*

*along $C' \preceq C$. Note that we use calligraphic letters for specific presheaves, whereas arbitrary ones are denoted by capital letters as above.*

**Example 2.4** In this example, we fix the notion of time $\mathbf{T}$ and observation $\mathcal{O}$ associated with a (labelled) transition system. For time $\mathbf{T}$ we take the set of natural numbers $\mathbb{N}$ viewed as a category (arrows are the less-than-equal-to relations). For the given alphabet $A$, we now define a presheaf $\mathcal{A} \in \mathbf{PSh}(\mathbb{N})$:

$$\mathcal{A}(n) = \{\sigma \in A^\star \mid |\sigma| = n\} \qquad \text{(for every } n \in \mathbb{N}\text{)},$$

together with the action on $\mathcal{A}$ given by $\sigma \cdot n = \sigma|_n$ (for every $\sigma \in \mathcal{A}(n')$ and $n \leq n'$). In other words, $\mathcal{A}(n)$ is the set of those finite words $\sigma \in A^\star$ whose length is $n$ (denoted by $|\sigma| = n$), while the action $\_ \cdot n$ simply maps a word $\sigma$ of length $n'$ to its unique prefix of length $n$ (denoted by $\sigma|_n$). Note that $\mathcal{A}(0)$ is a singleton set containing the empty word which we denote by $\varepsilon$.

**Remark 2.5** In modelling some dynamical systems, like, e.g., those arising from control theory [28], $\mathcal{O}$ may have even more structure in that global observations can be constructed by gluing the local observations (smaller neighbourhoods). In such situations, the category of sheaves $\mathbf{Sh}(\mathbf{C}, J)$ equipped with a Grothendieck topology $J$ on $\mathbf{C}$ is more suitable (cf. [30]) for semantic purposes. Moreover, sheaves equipped with discrete Grothendieck topology are exactly presheaves (cf. [27]), so our mathematical universe is actually the category of sheaves (rather than presheaves). But due to the discrete nature of dynamical systems considered in this paper, we restrict ourselves to presheaves. Nevertheless, we will state our definitions so that they are applicable on sheaves (see, e.g., Remark 3.2).

Once a notion of time $\mathbf{T}$ and an observation $\mathcal{O} \in \mathbf{PSh}(\mathbf{T})$ is fixed, then a system essentially describes the *runs* (also known as *trajectories* or *executions*) of the system and the observation associated with each run. To answer both, we envisage that a dynamical system is nothing but an object in the slice category $\mathbf{PSh}(\mathbf{C})/\mathcal{O}$. In other words, a dynamical system corresponds to a presheaf $F$ modelling the runs of the system and a natural transformation $F \xrightarrow{\alpha} \mathcal{O}$ modelling the observation associated with each run of the system. More importantly, a *system homomorphism* $\varphi$ between two systems $(F, \alpha)$ and $(G, \beta)$, denoted $(F, \alpha) \xrightarrow{\varphi} (G, \beta)$, is a natural transformation $F \xrightarrow{\varphi} G$ preserving the observations, i.e., $\beta \circ \varphi = \alpha$. Intuitively, a system homomorphism $(F, \alpha) \xrightarrow{\varphi} (G, \beta)$ says that the system $(F, \alpha)$ is a *refinement* of $(G, \beta)$ (i.e., every observable behaviour of $F$ is also part of the observable behaviour of $G$).

### *2.1 Refining our framework by unifying time and observation*

Although the slice category $\mathbf{PSh}(\mathbf{T})/\mathcal{O}$ is close to our system theoretic intuition, its presentation can be further simplified. Recall the *category of elements* of a presheaf $F \in \mathbf{PSh}(\mathbf{C})$, denoted $\mathbb{E}_{\mathbf{C}}(F)$ (we drop the subscript $\mathbf{C}$ whenever clear

from the context), has as objects the tuples $(x, C)$ with $C \in \mathbf{C}, x \in FC$ and as arrows $(x, C) \xrightarrow{f} (x', C')$ the morphism $C \xrightarrow{f} C' \in \mathbf{C}$ such that $x' \cdot f = x$.

**Theorem 2.6 ([27, Exercise III.8(a)])** *For a presheaf $F$ over a small category $\mathbf{C}$, there is an equivalence of categories $\mathbf{PSh}(\mathbf{C})/F \cong \mathbf{PSh}(\mathbb{E}(F))$.*

Note that a similar result also holds in the setting of sheaves (cf. [27, Exercise III.8(b)]).

In other words, time can be made inherent with observation and, thus, we can work in a simpler setting without worrying about the bookkeeping associated with slice categories. To see this, recall Example 2.4 and the poset of finite words $A^\star$ (a.k.a. free monoid) generated by a set $A$, which is ordered by the prefix relation $\preceq \subseteq A^\star \times A^\star$. Notice that the categories $\mathbb{E}(\mathcal{A})$ and $A^\star$ are isomorphic: since the length of a word is redundant in the objects of $\mathbb{E}(\mathcal{A})$ dropping the length results in the elements of $A^\star$. Thus, we obtain

**Corollary 2.7** *There is an equivalence of categories $\mathbf{PSh}(\mathbb{N})/\mathcal{A} \cong \mathbf{PSh}(A^\star)$.*

As a result, the category of presheaves on $A^\star$ can serve as the semantic universe to study behaviour of a transition system (cf. Example 2.8). More generally, by giving the semantics to a 'syntactic' category of a computational model $\mathbf{M}$, we mean identifying the notion of time $\mathbf{T}$ and observation $\mathcal{O} \in \mathbf{PSh}(\mathbf{T})$ together with a faithful functor $\mathbf{M} \xrightarrow{[\![\_]\!]} \mathbf{PSh}(\mathbb{E}_{\mathbf{T}}(\mathcal{O}))$, called the *semantics* functor. By interpreting an arrow $M \xrightarrow{f} M'$ in $\mathbf{M}$ as *$M$ is an implementation of $M'$ witnessed by $f$*, then faithfulness of $[\![\_]\!]$ asserts: if an implementation is witnessed by two semantically same morphisms $[\![f]\!] = [\![g]\!]$, then $f = g$ must be the same syntactically.

**Example 2.8** Consider a transition system $(X, A, \rightarrow)$ where $X$ is the set of states, $A$ is the set of actions, and $\rightarrow \subseteq X \times A \times X$ is the transition relation[3]. Then the collection of transition systems together with simulation functions form a category denoted **LTS**. Note that a *simulation function* is a function $X \xrightarrow{f} Y$ satisfying:

$$\forall_{x,x' \in X, a \in A}\ x \xrightarrow{a} x' \implies f(x) \xrightarrow{a} f(x'). \tag{1}$$

As usual, we write $x \xrightarrow{a} x'$ to denote $(x, a, x') \in \rightarrow$. Let $\downarrow \sigma = \{\sigma' \in A^\star \mid \sigma' \preceq \sigma\}$ be the prefixes of $\sigma$. Next, define a presheaf $[\![X]\!] \in \mathbf{PSh}(A^\star)$ which records all the executions whose trace is $\sigma$ at $[\![X]\!](\sigma)$:

$$[\![X]\!](\sigma) = \left\{ \downarrow \sigma \xrightarrow{p} X \mid \forall_{\sigma', a} \left( \sigma' a \preceq \sigma \implies p(\sigma') \xrightarrow{a} p(\sigma' a) \right) \right\},$$
$$p \cdot \sigma' = p_{\downarrow \sigma'} \quad \left( \text{for any } \sigma' \preceq \sigma \text{ and } p \in [\![X]\!](\sigma) \right).$$

Moreover, any function $X \xrightarrow{f} Y$ induces a family of maps $[\![f]\!]_\sigma(p) = f \circ p$ (for each $\sigma \in A^\star$).

Thus, we obtain the following result; wherein, the result that presheaf maps encode simulation maps is well known from the early work of Joyal et al. in [23].

[3] Transition systems without initial states are standard in process algebraic literature (see [3]).

**Proposition 2.9** *The map $[\![\_]\!]$ defined in Example 2.8 is a faithful functor. Moreover,*

- *if $[\![f]\!]$ is a presheaf map for any $f \in \mathbf{LTS}$, then $f$ is a simulation function.*
- *if a function $f$ between the underlying state spaces induces a presheaf map $[\![f]\!]$, then $f$ is a simulation map.*

As a result, the category of transition systems **LTS** has presheaf semantics on $A^\star$. In the subsequent sections, we will show the applicability of our semantic framework by giving presheaf semantics to different computational models; namely, transition systems with fairness predicates and invisible transitions.

## 3 Bisimulation maps: towards complete refinement

Consider a category of computational models $\mathbf{M}$ together with its semantics over a presheaf category $\mathbf{PSh}(\mathbf{C})$, i.e., $\mathbf{M} \xrightarrow{[\![\text{-}]\!]} \mathbf{PSh}(\mathbf{C})$. As mentioned earlier, an arrow between any two images of the semantics functor as a refinement map from the modelling point of view. In particular, we interpret $[\![M]\!] \xrightarrow{[\![f]\!]} [\![M']\!]$ (induced by an arrow $M \xrightarrow{f} M' \in \mathbf{M}$) as the information that $[\![M]\!]$ is an implementation of $[\![M']\!]$ witnessed by the refinement map $[\![f]\!]$. Note that this is a straightforward generalisation of Proposition 2.9 since simulation maps encode the refinement of behaviour in the case of labelled transition systems. Therefore, to study bisimulation maps at the level of presheaves (executions), we will restrict ourselves to those presheaf morphisms that represent 'complete' refinement of behaviour in the following sense. Theorem 3.3 expresses this in a more formal manner.

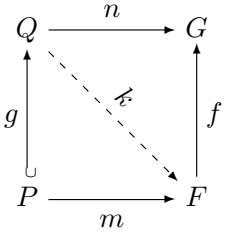
- Every observable behaviour in $[\![M']\!]$ is also observable in $[\![M]\!]$ (thus, $[\![M]\!]$ is a refinement of $[\![M']\!]$).
- Moreover, every observable behaviour in $[\![M]\!]$ can be retracted onto an observable behaviour in $[\![M']\!]$.

One possibility is to use the open maps of Joyal et al. [23], which gave a unified definition of functional bisimulations over the range of computational models. In particular, when invoked in a presheaf category, open maps correspond to natural transformations whose naturality squares are the weak-pullback squares in **Set** (see [8, Proposition 2.3]). Nevertheless, an open map falls short in capturing the complete refinement point of view since an arbitrary open map may not even be a surjective map at the level of executions; i.e., our implementation may not even implement or cover all the behaviour present in the specification.

In [5], open maps [23] were refined to embedding-open maps in the setting of concrete categories. An important difference to the classical definition of open maps is that the parametric notion of path extensions can be replaced by embeddings (see [2, Definition 8.6(2)] for a formal definition). Moreover, embedding-open maps are always retracts under some mild restrictions on concrete categories (cf. [5]). Now, if a category is concrete over itself, then an embedding corresponds to simply a monomorphism. Consequently, we propose the following definition of an embedding-

open map, which we simply call a *bisimulation* map.

**Definition 3.1** A map $F \xrightarrow{f} G \in \mathbf{PSh}(\mathbf{C})$ is a *bisimulation* if, and only if, for every commutative square depicted in (2) with a mono $P \xhookrightarrow{g} Q$ and maps $m, n$ in $\mathbf{PSh}(\mathbf{C})$, there is a map $Q \xrightarrow{k} F \in \mathbf{PSh}(\mathbf{C})$ such that the two triangles commute, i.e., $k \circ g = m$ and $f \circ k = n$.




(2)

Intuitively, (2) states that every extension of behaviour observable in $G$ can be reflected as an extension observable in $F$ through the witnesser $f$.

**Remark 3.2** It is interesting to note that a similar definition for sheaves can be derived from the definition of embedding-open maps as given in [5]. First, note that $\mathbf{Sh}(\mathbf{C}, J)$ is concrete over $\mathbf{PSh}(\mathbf{C})$ due to the forgetful functor $\mathfrak{i}$, which is fully faithful. Moreover, embeddings in this concrete category are actually monomorphisms. This is because any mono is a regular mono in $\mathbf{Sh}(\mathbf{C}, J)$ and the faithful functor $\mathfrak{i}$ preserves regular monos (since $\mathfrak{i}$ is right adjoint to the associated sheaf functor $\mathfrak{a}$ [27]). Lastly, every regular mono is an embedding whenever the faithful functor preserves regular monos [2, Proposition 8.7.3]. Thus, we have the exact same definition of bisimulation maps in $\mathbf{Sh}(\mathbf{C}, J)$.

**Theorem 3.3** *Every bisimulation map in a (pre)sheaf category is a retract.*

We end this section by capturing functional bisimulations in terms of bisimulation maps whose proof can be extracted from the proof of Theorem 4.3. Note that a similar theorem was proven earlier in the seminal paper [23] for functional bisimulation; however, the difference is that we use bisimulation maps (not open maps) in our characterisation.

**Theorem 3.4** *Given a simulation function $X \xrightarrow{f} Y$, then $[\![f]\!]$ is a bisimulation map in $\mathbf{PSh}(A^\star)$ if, and only if, the function $f$ is a surjection satisfying:*

$$\forall_{x \in X, y \in Y} \ \big(f(x) \xrightarrow{a} y \implies \exists_{x' \in X} \ (x \xrightarrow{a} x' \wedge f(x') = y)\big). \tag{3}$$

## 4 The case of fairness

When considering infinite, rather than finite, behaviour of systems, it is common to take fairness into account. There are various notions of fairness [18] but the general idea for fair (bi)simulation is to demand that fair executions are matched by fair executions, in addition to classical (bi)simulation properties. In this section, we give

a presheaf semantics to fair transition systems and outline how $\forall$-fair bisimulation can be captured via bisimulation maps.

Let $A^\omega = \{\sigma \mid \mathbb{N} \xrightarrow{\sigma} A\}$ be the set of infinite words generated from $A$ and fix $A^\infty = A^\star \cup A^\omega$, which is ordered by the prefix relation $\preceq$. The object of study are *fair transition systems* $(X, A, \to, \mathrm{Fair}_X)$, where $(X, A, \to)$ is a transition system and $\mathrm{Fair}_X$ is a fairness predicate on infinite executions. An *infinite execution* is a function $\downarrow\sigma \xrightarrow{p} X \cup \{\Omega\}$ whose domain is the history of an infinite word $\sigma \in A^\omega$ such that $p(\sigma) = \Omega$ and $\forall_{\sigma', a}\ (\sigma' a \prec \sigma \implies p(\sigma') \xrightarrow{a} p(\sigma' a))$. Here, $\Omega$ is the *chaos* state which the system enters once it has executed an infinite execution. Let $\mathrm{FExec}(X) = \mathrm{Fair}_X \cup \mathrm{Exec}(X)$ be the set of all fair executions ordered by the prefix relation $\preceq$, i.e., $p \preceq p' \iff p'|_{\mathrm{dom}(p)} = p$ (for $p, p' \in \mathrm{FExec}(X)$). The set of (finite) executions $\mathrm{Exec}(X)$ is defined as earlier, i.e., $\mathrm{Exec}(X) = \{\downarrow\sigma \xrightarrow{p} X \mid \sigma \in A^\star \wedge \forall_{\sigma' a \in \downarrow\sigma}\ p(\sigma') \xrightarrow{a} p(\sigma' a)\}$.

**Definition 4.1** A chaos preserving extension $X \cup \{\Omega\} \xrightarrow{f_\Omega} Y \cup \{\Omega\}$ (i.e., $f_\Omega(x) = f(x)$ for $x \in X$ and $f_\Omega(\Omega) = \Omega$) of a function $X \xrightarrow{f} Y$ is a *fair simulation* between $(X, A, \to, \mathrm{Fair}_X), (Y, A, \to, \mathrm{Fair}_Y)$ if, and only if, $f$ satisfies (1) and $\forall_{p \in \mathrm{Fair}_X}\ f_\Omega \circ p \in \mathrm{Fair}_Y$. Henceforth, we do not distinguish between $f_\Omega$ and $f$.

Note that an infinite, but fair, execution can be seen as the limit of a monotonically increasing sequence of finite executions in $\mathrm{FExec}(X)$. Since this limiting sequence is part of behaviour, we should reflect it. Thus, we say a *fair bisimulation* $X \cup \{\Omega\} \xrightarrow{f} Y \cup \{\Omega\}$ is a surjective fair simulation $f$ satisfying (3) and the following condition for any increasing sequence of finite executions $(p_i)_{i \in \mathbb{N}}$ in $X$:

$$\bigsqcup_{i \in \mathbb{N}} f \circ p_i \approx f \circ \bigsqcup_{i \in \mathbb{N}} p_i. \tag{4}$$

Here, $\approx$ is the Kleene equality used to equate the partially defined terms above.

To the best of our knowledge, the above notion of fair bisimulation is novel; however, below we will establish its connection with the literature after characterising it in terms of presheaf morphisms. So let **FTS** be the category of fair transition systems and fair simulation between them. In addition,

- Time: Since infinite executions are allowed in a fair transition system, we take $\mathbf{T} = \mathbb{N} \cup \{\infty\}$ to be the category of natural numbers extended by a number representing infinity (i.e., $\forall_{n \in \mathbb{N}}\ n \leq \infty$).
- Observation: Using the definition of $\mathcal{A}$ in Example 2.4, we define: $\mathcal{O}(\infty) = A^\omega$ and $\mathcal{O}(n) = \mathcal{A}(n)$ (for $n \in \mathbb{N}$).

Clearly, the categories $\mathbb{E}(\mathcal{O})$ and $A^\infty$ are isomorphic and by applying Theorem 2.6 we obtain $\mathbf{PSh}(\mathbb{E}(\mathcal{O})) \cong \mathbf{PSh}(A^\infty)$. So, we take the category $\mathbf{PSh}(A^\infty)$ as the semantic universe to study fair transition systems. Just as in Example 2.8, for a given fair transition system $(X, A, \to, \mathrm{Fair}_X)$, we define a presheaf

$$[\![X]\!](\sigma) = \{p \in \mathrm{FExec}(X) \mid \max \mathrm{dom}(p) = \sigma\} \quad (\text{for each } \sigma \in A^\infty),$$

and the action is given by restricting the domain of an execution. Moreover, for any fair simulation function $f$, we let $[\![f]\!]_\sigma = f \circ \_$ (for each $\sigma \in A^\infty$); thus, resulting in a presheaf semantics to fair transition systems.

**Proposition 4.2** *The above map* $\mathbf{FTS} \xrightarrow{[\![\text{-}]\!]} \mathbf{PSh}(A^\infty)$ *is a faithful functor.*

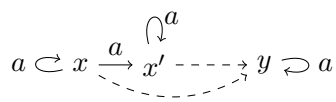
Invoking the bisimulation map definition between any two presheaves generated by fair transition systems results in the characterisation of fair bisimulation.

**Theorem 4.3** *A fair simulation function $f$ is a fair bisimulation if, and only if, the underlying map $[\![f]\!]$ is a bisimulation map in* $\mathbf{PSh}(A^\infty)$.

*Remark 4.4.* It should be noted that fair bisimulation maps are stronger than the open maps studied by Hildebrandt [19] in $\mathbf{PSh}(A^\infty)$ to characterise the extended bisimulation of Hennessy and Stirling [17] for pointed systems, i.e. systems with explicitly initial states. To demonstrate this, recall that an $\mathrm{Inf}_\perp$-open map of Hildebrandt [19, Proposition 25] is a fair simulation function (not necessarily surjective) $X \cup \{\Omega\} \xrightarrow{f} Y \cup \{\Omega\}$ satisfying (3) and the property:

$$\forall_{x \in X, q \in \mathrm{FExec}(Y)} \big(q(\varepsilon) = f(x) \implies \exists_{p \in \mathrm{FExec}(X)} \, (p(\varepsilon) = x \wedge f \circ p = q)\big). \tag{5}$$

Consider the two systems drawn on the right with a function $f$ between the states depicted by dashed lines.




In the left system, the infinite executions visiting $x'$ infinitely often are considered fair, whereas the only infinite execution in the right system is fair. Clearly, $f$ is a fair simulation satisfying (5). But $f$ is *not* a fair bisimulation because the sequence of finite executions $(p_i)_{i \in \mathbb{N}}$ formed by unfolding the self-loop on $x$ has a fair execution $\bigsqcup_{i \in \mathbb{N}} f \circ p_i$ (looping on $y$) as the limit in the right system. Yet, $(p_i)_{i \in \mathbb{N}}$ has no limit, thus, violating (4).

**Remark 4.5** In [19], (separated) presheaves with sup topology are used because an increasing sequence of finite executions induced by a fair transition system has at most one limit point. Unlike [19], we are not interested in one-to-one semantic representation of our syntactical models, so we work with arbitrary presheaves. This is because the choice whether the semantic universe should be (separated) presheaves or sheaves depends on observations, i.e., how $\mathcal{O}$ is modelled. This is also why we do *not* require our semantic functor $[\![\text{-}]\!]$ to be full.

Next, we relate fair bisimulation maps with $\forall$-fair bisimulation relations.

**Definition 4.6** A $\forall$-*fair bisimulation* on $(X, A, \to, \mathrm{Fair}_X)$ is an equivalence relation $\mathcal{R} \subseteq X \times X$ satisfying the following transfer properties:

(i) $\forall_{x,y,x',a} \big((x \xrightarrow{a} x' \wedge x \mathcal{R} y) \implies \exists_{y'} \, (y \xrightarrow{a} y' \wedge x' \mathcal{R} y')\big)$, and

(ii) $\forall_{p,q}\ \big((p =_{\mathcal{R}} q \wedge p \in \mathrm{Fair}_X) \implies q \in \mathrm{Fair}_X\big)$.

Here, $p =_{\mathcal{R}} q$ is an abbreviation for $\mathrm{dom}(q) = \mathrm{dom}(p) \wedge \forall_{\sigma \in \mathrm{dom}(p) \cap A^\star}\ p(\sigma)\mathcal{R}q(\sigma)$.

**Theorem 4.7** *Two states $x$ and $x'$ are related by a $\forall$-fair bisimulation relation if, and only if, there is a fair bisimulation function $f$ such that $f(x) = f(x')$.*

The requirement of equivalence relations in the above definition may look superfluous at first glance. This is because, traditionally (i.e., when fairness predicates are empty sets), a strong bisimulation relation by definition is not necessarily an equivalence relation on the set of states. Moreover, *bisimilarity* which is defined as the union of all strong bisimulation relations turns out to be both an equivalence relation and a strong bisimulation relation. However, such closure results do not hold in general for $\forall$-fair bisimulation relations. $\forall$-fair bisimulation relations are not closed under union and the relational composition (even if we relax Definition 4.6 by replacing 'an equivalence relation' for 'a symmetric relation')[4]. Thus, $\forall$-fair bisimilarity may, in general, neither be an equivalence relation nor a $\forall$-fair bisimulation; in other words, $\forall$-fair bisimilarity is not a coinductive definition (like how strong bisimilarity is). Nevertheless, we deemed all $\forall$-fair bisimulation relations to be equivalences because: first, the main use of a $\forall$-fair bisimulation relation is to equate two systems that have the same behaviour sensitive to fair executions; second, the mathematics tells us that the kernel of a (fair bisimulation) function is an equivalence relation.

# 5 The case of invisible actions

The behaviour of a specification and its implementation is often spread over different levels of abstraction. The standard process algebraic way to relate the behaviour of an implementation with its specification is by delineating the effect of actions in lower levels of abstraction as invisible. For instance, removing a message from a buffer is considered unobservable in a rendezvous between communicating processes. This is made formal by reinterpreting the notion of (bi)simulation functions in the presence of the invisible action $\tau$.

## *5.1 Branching bisimulation*

**Notation 5.1** *Henceforth, $\tau \notin A$ will denote the invisible action and $A_\tau = A \cup \{\tau\}$. Furthermore, Exec$(X, \sigma)$ is the set of executions $p$ having trace $\sigma$, i.e.,* $\max dom(p) = \sigma$ *and $\twoheadrightarrow\ \subseteq X \times A^\star \times X$ is the* weak reachability *relation on $(X, A_\tau, \to)$ given as the smallest relation satisfying the following conditions:*

$$\frac{}{x \overset{\varepsilon}{\twoheadrightarrow} x} \qquad \frac{x \overset{\tau}{\to} x'}{x \overset{\varepsilon}{\twoheadrightarrow} x'} \qquad \frac{x \overset{\varepsilon}{\twoheadrightarrow} x' \wedge x' \overset{\varepsilon}{\twoheadrightarrow} x''}{x \overset{\varepsilon}{\twoheadrightarrow} x''} \qquad \frac{x \overset{\sigma}{\twoheadrightarrow} x' \wedge x' \overset{a}{\to} x''}{x \overset{\sigma a}{\longrightarrow} x''}.$$

[4] This is how $\forall$-fair bisimulation relations are defined in [24] on the states of Kripke structures. In regards to the above closure properties, we are only aware of [21] who showed that $\forall$-fair simulation relations are not closed under union.

**Definition 5.2** A *branching simulation* function $f$ between systems $(X, A_\tau, \to)$ and $(Y, A_\tau, \to)$ is a function $X \xrightarrow{f} Y$ satisfying the following properties:

(i) Simulation of observable transitions, i.e., $\forall_{x,x' \in X, a \in A}\ x \xrightarrow{a} x' \implies f(x) \xrightarrow{a} f(x')$, and

(ii) (Possible) simulation of invisible transitions, i.e., $\forall_{x,x' \in X}\ x \xrightarrow{\tau} x' \implies \big(f(x) = f(x') \vee f(x) \xrightarrow{\tau} f(x')\big)$.

(iii) Stuttering of $\tau$-transitions, i.e., for any $x_1, x_2, x_3$ we have

$$\big(x_1 \xrightarrow{\varepsilon} x_2 \xrightarrow{\varepsilon} x_3 \wedge f(x_1) = f(x_3)\big) \implies f(x_1) = f(x_2). \tag{6}$$

A *branching bisimulation* $f$ is a branching simulation surjection $f$ satisfying the 'weak' reflection of transitions, i.e., for any $x \in X, y \in Y$, and $a \in A_\tau$ we have

$$f(x) \xrightarrow{a} y \implies \exists_{x',x'' \in X}\ (x \xrightarrow{\varepsilon} x' \xrightarrow{a} x'' \wedge f(x') = f(x) \wedge f(x'') = y). \tag{7}$$

**Definition 5.3** Given a labelled transition system $(X, A_\tau, \to)$, then a symmetric relation $\mathcal{R} \subseteq X \times X$ is a *branching bisimulation* [33] if, and only if, the following transfer property is satisfied

$$\forall_{x_1,x_2,y_1,a \in A_\tau}\Big((x_1 \xrightarrow{a} x_2 \wedge x_1 \mathcal{R} y_1) \implies (a = \tau \wedge x_2 \mathcal{R} y) \vee \exists_{y,y_2}\ \big(y_1 \xrightarrow{\varepsilon} y \xrightarrow{a} y_2 \wedge x_1 \mathcal{R} y \wedge x_2 \mathcal{R} y_2\big)\Big).$$

Two states $x, x' \in X$ are branching bisimilar if there exists a branching bisimulation $\mathcal{R}$ such that $x \mathcal{R} x'$.

We work with branching bisimulation functions (not relations) because of the following result (Theorem 5.4), which is similar in spirit to Lemma 2.7 proved by Caucal in [9]. The difference is that Caucal's branching bisimulation functions (which he calls *reduction* in [9]) do not respect the stuttering of invisible steps (6). Nevertheless, we are still able to obtain the following correspondence since the largest branching bisimulation relation satisfies the so-called *stuttering lemma* of [33]. In particular, any reduction $X \xrightarrow{f} Y$ in the sense of Caucal can be extended to a branching bisimulation function by composing it with the quotient map $Y \xrightarrow{q} Y/\mathcal{R}$, where $\mathcal{R} \subseteq Y \times Y$ is the largest branching bisimulation relation on $Y$.

**Theorem 5.4** *Two states $x, x' \in X$ of a transition system $(X, A_\tau, \to)$ are branching bisimilar if, and only if, there are a transition system $(Y, A_\tau, \to)$ and a branching bisimulation function $X \xrightarrow{f} Y$ such that $fx = fx'$.*

### *5.2 The setup*

Our first step towards the characterisation of branching bisimulation is the presheaf representation of executions induced by a transition system with invisible actions. Since a transition system evolves in a step-based manner, it is sufficient to model the

time by the set of natural numbers $\mathbb{N}$. The only difference, when compared to the strong case (cf. Example 2.4), is in the notion of observation. Instead of recording words from $A^\star$, our hypothetical 'observer' now records words from $A_\tau^\star$. So, consider a presheaf $\mathbb{N}^{\mathrm{op}} \xrightarrow{\mathcal{A}_\tau} \mathbf{Set}$ in the spirit of $\mathcal{A}$ as defined in Example 2.4. Recall that the slice category $\mathbf{PSh}(\mathbb{N})/\mathcal{A}_\tau$ is equivalent to the category of presheaves on the category of elements $\mathbb{E}(\mathcal{A}_\tau)$, which is isomorphic to the category $A_\tau^\star$. Thus, for a given transition system $(X, A_\tau, \to)$, we define a presheaf $F_X \in \mathbf{PSh}(A_\tau^\star)$ as follows:

$$F_X(\sigma) = \mathrm{Exec}(X, \sigma), \quad (\text{for every } \sigma \in A_\tau^\star). \tag{8}$$

The action on $F_X$ is given by the restriction of the domain of an execution.

Incidentally, a branching simulation function $X \xrightarrow{f} Y$ between $(X, A_\tau, \to)$ and $(Y, A_\tau, \to)$ does *not* induce a system homomorphism between the underlying dynamical systems (presheaves) $F_X$ and $F_Y$ because a branching simulation function does not necessarily preserve the length of the executions. For example, a sequence of transitions $\bullet \xrightarrow{\tau} \bullet \xrightarrow{a} \bullet$ may get mapped to a transition $\bullet \xrightarrow{a} \bullet$.

Thus, we need a procedure that transforms a given presheaf on $A_\tau^\star$ to a presheaf on $A^\star$. It turns out that there already is a general result in category theory for this purpose, which we explain next. In particular, recall the cocompletion of a category $\mathbf{C}$ through the Yoneda embedding $\mathbf{C} \xrightarrow{\mathbb{Y}_\mathbf{C}} \mathbf{PSh}(\mathbf{C})$.

**Theorem 5.5 (see [27])** *For any functor $\mathbf{C} \xrightarrow{h} \mathbf{D}$, when $\mathbf{C}$ is small and $\mathbf{D}$ is cocomplete, there is a colimit preserving functor $\mathbf{PSh}(\mathbf{C}) \xrightarrow{L_h} \mathbf{D}$ satisfying $L_h \circ \mathbb{Y}_\mathbf{C} \cong h$. Moreover, $L_h$ has a right adjoint $R_h$ given by: $R_h D(C) = \mathbf{D}(hC, D)$, for each $C \in \mathbf{C}, D \in \mathbf{D}$.*

Using the language of Kan extensions, $L_h$ is the left Kan extension of $h$ along the Yoneda embedding $\mathbb{Y}_\mathbf{C}$. Moreover, using the notion of a coend [26], we have:

$$L_h F \cong \int^{C \in \mathbf{C}} FC \odot hC,$$

where $\mathbf{D} \xrightarrow{S\odot -} \mathbf{D}$ is the copower functor given by $S \odot D = \coprod_{s \in S} D$ (taking $S$ disjoint copies of $D$). If we replace $\mathbf{C} \xrightarrow{h} \mathbf{D}$ in the above diagram by a map $\mathbf{C} \xrightarrow{h} \mathbf{D} \xrightarrow{\mathbb{Y}_\mathbf{D}} \mathbf{PSh}(\mathbf{D})$, then we obtain the so-called *essential geometric morphism* [27] between $\mathbf{PSh}(\mathbf{C}) \longrightarrow \mathbf{PSh}(\mathbf{D})$. In full, this means that the composition functor $h^*$ not only has a right adjoint $\Pi_h$, but also a left adjoint $\Sigma_h$ (see, for instance, [27] for a formal definition of a geometric morphism). Below, the composition functor $h^*$, its left adjoint $\Sigma_h$, and its right adjoint $\Pi_h$ are given by (up to isomorphism) $R_{\mathbb{Y}_\mathbf{D} h}, L_{\mathbb{Y}_\mathbf{D} h}$, and $R_{h^* \mathbb{Y}_\mathbf{D}}$, respectively.

**Corollary 5.6 (see [27])** *Given a functor $\mathbf{C} \xrightarrow{h} \mathbf{D}$ between small categories, then the inverse image functor $\mathbf{PSh}(\mathbf{D}) \xrightarrow{h^*} \mathbf{PSh}(\mathbf{C})$ given by $h^* G = G \circ h^{op}$ (for*

*each $G \in \mathbf{PSh}(\mathbf{D})$) has both left and right adjoints:*

$$\Sigma_h \dashv h^* \dashv \Pi_h.$$

So, in principle, there are two ways to land in $\mathbf{PSh}(A^\star)$ from $\mathbf{PSh}(A_\tau^\star)$ whenever there is a functor between $A_\tau^\star \longrightarrow A^\star$. Nevertheless, since our aim is to characterise branching bisimulation functions, we will choose the left adjoint (over the right adjoint) of the composition functor for this task. This is because to reflect (recall (7)) an observable transition – say, of the form $\bullet \xrightarrow{a} \bullet$ –, we only need *minimal* executions which are of the form $\bullet \twoheadrightarrow \bullet \xrightarrow{a} \bullet$ (cf. Definition 5.9). In particular, we will show (cf. Theorem 5.11) that the left adjoint $\Sigma_h$ (induced by a suitable $h$) transforms a presheaf of executions into a presheaf of minimal executions (or those executions in which trailing $\tau$-transitions are chopped off).

### *5.3 An obvious, but failed attempt*

There is an evident functor $A_\tau^\star \xrightarrow{h} A^\star$ which treats the letter $\tau$ as an empty word. It is then natural to investigate whether the bisimulation maps in $\mathbf{PSh}(A^\star)$ between any two induced presheaves $\Sigma_h F_X, \Sigma_h F_Y$ characterise the branching bisimulation map. Unfortunately, the answer is no! We explain this extensively, as this lays the formal foundation for the desired characterisation given in the next subsection.

So, consider the hiding function $A_\tau \xrightarrow{h} A^\star$ which treats $\tau$ as the unit of $A^\star$ (i.e., $h(\tau) = \varepsilon$) and treats an action $a \in A$ as observable (i.e., $h(a) = a$ when $a \in A$). This lifts to an order-preserving function $A_\tau^\star \xrightarrow{h} A^\star$ (denoted again as $h$ by abuse of notation). Thus, we have a functor $A_\tau^\star \xrightarrow{h} A^\star$.

**Proposition 5.7** *The categories $A_\tau^\star, A^\star$ have binary products.*

**Remark 5.8** It is worthwhile noting that $h$ does not generally preserve finite limits (or infimum $\sqcap$ in this case). Consider the words $a\tau b, ab \in A_\tau^\star$. Then, $h(a\tau b \sqcap ab) = h(a) = a$; however, $h(a\tau b) \sqcap h(ab) = ab \sqcap ab = ab$.

Furthermore, $S \odot X \cong S \times X$ (for any two sets $S, X$) and the (co)limit in any presheaf category is computed point-wise. Thus, using these observations, we calculate $\Sigma_h F$ (for $F \in \mathbf{PSh}(A_\tau^\star)$) at $\varrho \in A^\star$ as follows:

$$\Sigma_h F(\varrho) \cong \int^{\sigma \in A_\tau^\star} F(\sigma) \times \mathbb{Y}_{A^\star}(h(\sigma))(\varrho) \;\cong \varinjlim_{\sigma \in A_\tau^{\star\,\mathrm{op}},\, \varrho \preceq h(\sigma)} F(\sigma). \tag{9}$$

Note that the reason for the last isomorphism is that every colimit can be encoded as a coend (see [26, p. 224–225] for a dual statement). In addition, (9) is the colimit of $A_\tau^{\star\,\mathrm{op}} \xrightarrow{F} \mathbf{Set}$ when $\varrho = \varepsilon$. Notice that $A_\tau^{\star\,\mathrm{op}}$ is directed; thus, the colimit of the filtered diagram is $\coprod_{\varepsilon \preceq h(\sigma)} F\sigma/\!\sim$, where $\sim \subseteq \coprod_{\varepsilon \preceq h(\sigma)} F(\sigma) \times \coprod_{\varepsilon \preceq h(\sigma)} F(\sigma)$ is the equivalence relation defined as follows:

$$(\sigma, p) \sim (\sigma', p') \iff \exists_{\sigma''} \left(\sigma'' \preceq \sigma \wedge \sigma'' \preceq \sigma' \wedge p \cdot \sigma'' = p' \cdot \sigma''\right). \tag{10}$$

It turns out that from a system theoretic viewpoint, the set $\varinjlim_{\varepsilon \preceq h(\sigma)} F_X(\sigma)$ is nothing but the set of minimal executions whose observable trace is an empty observation. Theorem 5.11 states this result in full generality.

**Definition 5.9** Given a transition system $(X, A_\tau, \to)$ and a word $\varrho \in A^\star$, we say an execution $p$ has an *observable trace* $\varrho$ if $h(\max \mathrm{dom}(p)) = \varrho$. Furthermore, $\mathrm{MExec}(X, \varrho)$ is the set of all *minimal executions* w.r.t. the prefix order $\preceq$ on executions whose observable trace is $\varrho$. Formally, $p \preceq p' \iff p'|_{\mathrm{dom}(p)} = p$ and

$$\mathrm{MExec}(X, \varrho) = \Big\{ p \mid {\downarrow} p \cap \{ q \in \mathrm{Exec}(X) \mid h(\max \mathrm{dom}(q)) = \varrho \} = \{p\} \Big\}.$$

Before we prove Theorem 5.11, we need a category theoretic result (which is probably folklore; see [26, Exercise IV.2.7] for a dual statement), namely, that the colimit of a diagram can be decomposed into the coproducts of the colimits of diagrams with smaller shapes under certain restrictions.

**Lemma 5.10** *Given a set $J$ and a small category $\mathbf{C} = \coprod_{j \in J} \mathbf{C}_j$ with injections $\iota_j$, then for any functor $F$ from $\mathbf{C}$ to a cocomplete category $\mathbf{D}$, we have $\varinjlim_{\mathbf{C}} F \cong \coprod_{j \in J} \varinjlim_{\mathbf{C}_j} F \circ \iota_j$.*

**Theorem 5.11** *For a given transition system $(X, A_\tau, \to)$ and $\varrho \in A^\star$, we have $\Sigma_h F_X(\varrho) \cong \mathit{MExec}(X, \varrho)$.*

Next, we explore the action of the presheaf $\Sigma_h F_X$ from a system theoretic viewpoint. Firstly, it is defined by the universal property of the colimit. Let $\varrho' \preceq \varrho$ and let $\sigma$ be a minimal word such that $h(\sigma) = \varrho$. Then, there is a unique minimal word $\sigma_{\varrho'} = \bigsqcap \{ \sigma' \preceq \sigma \mid h(\sigma') = \varrho' \}$. Note that this infimum exists since the history of a word in $A_\tau^\star$ is a finite totally ordered set of words. Therefore, the family of arrows depicted by the dotted arrows in (11) forms a cone, where $A^\star_{\tau,\varrho} = \{ \sigma \in A_\tau^\star \mid \varrho \preceq h(\sigma) \}$ is a sub-forest of $A_\tau^\star$. Thus, the universal property of the colimit gives a map $\Sigma_h F_X \varrho \longrightarrow \Sigma_h F_Y \varrho'$, which we denote by $\Sigma_h(\varrho, \varrho')$.

$$\begin{array}{ccc} F_X\sigma & \dashrightarrow & F_X\sigma_{\varrho'} \\ \downarrow & (11) & \vdots \\ \varinjlim_{\sigma \in A^\star_{\tau,\varrho}} F_X \iota_\varrho \sigma & \dashrightarrow & \varinjlim_{\sigma \in A^\star_{\tau,\varrho'}} F_X \iota_{\varrho'} \sigma \end{array} \qquad \begin{array}{ccc} \Sigma_h F_X \varrho & \xrightarrow{\cong} & \mathrm{MExec}(X, \varrho) \\ \downarrow \Sigma_h(\varrho, \varrho') & (12) & \downarrow \mathrm{mpast}(\varrho, \varrho') \\ \Sigma_h F_X \varrho' & \xrightarrow{\cong} & \mathrm{MExec}(X, \varrho') \end{array}$$

Secondly, for any $\varrho' \preceq \varrho$ with $\varrho, \varrho' \in A^\star$, we define a map:

$$\mathrm{MExec}(X, \varrho) \xrightarrow{\mathrm{mpast}(\varrho, \varrho')} \mathrm{MExec}(X, \varrho') \quad \text{given by} \quad p \mapsto p|_{{\downarrow}\sigma_{\varrho'}} \ ,$$

where $\sigma = \max \mathrm{dom}(p)$. Now we can establish that the isomorphism in Theorem 5.11 is natural in $\varrho \in A^\star$.

**Theorem 5.12** *For any $\varrho, \varrho' \in A^\star$ with $\varrho' \preceq \varrho$, the square in (12) commutes.*

Now that we know what $\Sigma_h$ does to a presheaf, we use the category $\mathbf{PSh}(A^\star)$ as our semantic universe to handle invisible actions. Thus, for a given $(X, A_\tau, \rightarrow)$, we let $[\![X]\!] = \Sigma_h F_X$. In addition, for a branching simulation function $X \xrightarrow{f} Y$, we use the isomorphic view of minimal executions and let $[\![f]\!]_\varrho(p) = p_f$ (for each $\varrho \in A^\star$), where $p_f$ is an execution defined inductively using the following rules.

- If $\mathrm{dom}(p) = \varepsilon$ then $\mathrm{dom}(p_f) = \varepsilon$ and $p_f(\varepsilon) = f(p(\varepsilon))$.
- If $p \xrightarrow{a} p'$, $p_f \xrightarrow{a} q$, $\mathrm{last}(q) = f(\mathrm{last}(p'))$, and $a \in A$ then $p'_f = q$.
- If $p \xrightarrow{\tau} p'$ and $f(\mathrm{last}(p)) = f(\mathrm{last}(p'))$ then $p'_f = p_f$.
- If $p \xrightarrow{\tau} p'$, $f(\mathrm{last}(p)) \neq f(\mathrm{last}(p'))$, $p_f \xrightarrow{\tau} q, \mathrm{last}(q) = f(\mathrm{last}(p'))$ then $p'_f = q$.

Here, $p \xrightarrow{a} p' \iff \mathrm{dom}(p') = \mathrm{dom}(p)a \ \wedge \ p \prec p'$; the function $\mathrm{last}(p)$ returns the last visited state by the execution $p$, i.e., $\mathrm{last}(p) = p(\max \mathrm{dom}(p))$.

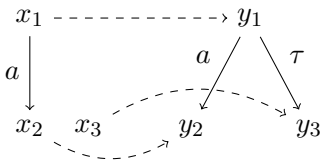
**Lemma 5.13** *For a given branching simulation function $X \xrightarrow{f} Y$ and a minimal execution $p \in MExec(X, \varrho)$, we have $p_f \in MExec(Y, \varrho)$.*

Next let $\mathbf{LTS}_\tau$ be the category of transition systems with possible invisible steps, which comprises of transition systems as objects and morphisms are the collection of identity functions and branching simulation functions. Thus, we can now give a presheaf semantics for the category $\mathbf{LTS}_\tau$.

**Theorem 5.14** *The mapping $\mathbf{LTS}_\tau \xrightarrow{[\![-]\!]} \mathbf{PSh}(A^\star)$ is a faithful functor.*

Finally, we can invoke the bisimulation maps in $\mathbf{PSh}(A^\star)$ but the characterisation of branching bisimulation functions fails to hold. The mismatch is in the reflection (7) of invisible steps (not with the reflection of observable transitions), which we explain next in the following example.

**Example 5.15** Consider the two transition systems and the branching simulation $f$ depicted below with dashed lines. Note that $f$ is *not* a branching bisimulation function (even though it is surjective) because it fails to reflect the transition $y_1 \xrightarrow{\tau} y_3$.




Yet $[\![f]\!]$ is a bisimulation map in $\mathbf{PSh}(A^\star)$. For this consider a given commutative square (2) in $\mathbf{PSh}(A^\star)$. The crucial case in defining the functions $k_\varrho$ is when $\varrho \in \{\varepsilon, a\}$.

- Let $q \in Q(\varepsilon)$. If $n_\varepsilon(q) = \varepsilon_{y_i}$ then we let $k_\varepsilon(q) = \varepsilon_{x_i}$ (for $i \in \{1, 2, 3\}$).
- Let $q \in Q(a)$. Then, $n_a(q)$ is the minimal execution witnessing $y_1 \xrightarrow{a} y_2$ since there are no other minimal executions in $\mathrm{MExec}(Y, a)$. Thus, we let $k_a(q)$ to be the minimal execution witnessing the transition $x_1 \xrightarrow{a} x_2$.

### 5.4 Characterisation of branching bisimulation

Unfortunately, all the information related to silent transitions is lost by transforming a presheaf using the functor $\Sigma_h$ because all the executions with empty observations get fused into their corresponding empty (minimal) executions. In retrospect, the problem lies in our specification of observation since our hypothetical 'observer' $\mathcal{O}$ is unable to differentiate between *empty observation due to no system move* and *empty observation due to zero or more system moves*. Therefore, to model this latter observation, we introduce a constant $\bar{\tau}$ which can be observed anytime after 0. So for this section, we define $\mathcal{O}$ in the next page as follows:

$$\mathcal{O}(0) = \mathcal{A}_\tau(0) \quad \text{and} \quad \mathcal{O}(n) = \mathcal{A}_\tau(n) \cup \{\bar{\tau}\} \quad (\text{for every } n > 0);$$

$$\sigma \cdot m = \begin{cases} \sigma|_m, & \text{if } \sigma \neq \bar{\tau} \\ \varepsilon, & \text{if } m = 0 \wedge \sigma = \bar{\tau} \\ \bar{\tau}, & \text{if } m > 0 \wedge \sigma = \bar{\tau}. \end{cases} \qquad \Big(\text{for any } m \leq n \text{ and } \sigma \in \mathcal{O}(n)\Big).$$

Put differently, $\bar{\tau}$ is 'really' a constant observation over time except at 0. The introduction of $\bar{\tau}$ is new and inspired from the constant $\eta$ [4] which can be viewed as an *empty observation due to at least one system move*.

**Proposition 5.16** *The above mapping* $\mathbb{N}^{op} \xrightarrow{\mathcal{O}}$ **Set** *is a presheaf.*

Moreover, the categories $\mathbf{PSh}(\mathbb{N})/\mathcal{O}$ and $\mathbf{PSh}(\mathbb{E}(\mathcal{O}))$ are equivalent (Theorem 2.6). So, consider the category of elements $\mathbb{E}(\mathcal{O})$ whose objects are tuples $(n, \sigma)$ (for $n \in \mathbb{N}, \sigma \in \mathcal{O}(n)$) and whose arrows are given by the rule:

$$(n, \sigma) \to (n', \sigma') \iff n \leq n' \wedge \sigma' \cdot n = \sigma.$$

Just as the category $\mathbb{E}(\mathcal{A}_\tau)$ has a simpler description in terms of $A^\star_\tau$, we simplify $\mathbb{E}(\mathcal{O})$ by defining a set $A^\star_{\tau\bar{\tau}} = A^\star_\tau \cup \{(n, \bar{\tau}) \mid n > 0\}$ ordered by the smallest relation $\preceq \subseteq A^\star_{\tau\bar{\tau}} \times A^\star_{\tau\bar{\tau}}$ satisfying the following rules:

$$\frac{(|\sigma|, \sigma) \to (|\sigma'|, \sigma') \in \mathbb{E}(\mathcal{A}_\tau)}{\sigma \preceq \sigma'} \qquad \frac{(m, \bar{\tau}) \to (n, \bar{\tau}) \in \mathbb{E}(\mathcal{O})}{(m, \bar{\tau}) \preceq (n, \bar{\tau})} \qquad \frac{n > 0}{\varepsilon \preceq (n, \bar{\tau})}.$$

Note that the leftmost rule is concerned with the elements of $A^\star_\tau$. Thus, the slice category $\mathbf{PSh}(\mathbb{N})/\mathcal{O}$ is equivalent to the category $\mathbf{PSh}(A^\star_{\tau\bar{\tau}})$ since $\mathbb{E}(\mathcal{O}) \cong A^\star_{\tau\bar{\tau}}$. Using the presheaf $F_X$ (cf. (8)) induced by a given transition system $(X, A_\tau, \to)$, define a presheaf $\bar{F}_X \in \mathbf{PSh}(A^\star_{\tau\bar{\tau}})$ as follows:

$$\bar{F}_X(\sigma) = \begin{cases} F_X(\sigma), & \text{if } \sigma \in A^\star_\tau \\ \coprod_{n\in\mathbb{N}} F_X(\tau^n), & \text{otherwise} \end{cases}, \text{ where } \begin{array}{l} \tau^0 = \{\varepsilon\} \\ \tau^{n+1} = \tau^n \cup \prod_{n+1}\{\tau\} \end{array}.$$

The executions based on invisible steps are seen *stretchable in time* by $\mathcal{O}$, i.e., a system may perform invisible executions of length independent of time instants.

This is encoded in the second clause of $\bar{F}_X$. To complete the definition of $\bar{F}_X$ as a presheaf, we define, for any $p \in \bar{F}_X(\sigma)$ and $\sigma' \preceq \sigma$ in $A^\star_{\tau\bar{\tau}}$, the action of $\bar{F}_X$ as:

$$p \cdot \sigma' = \begin{cases} p|_{\downarrow\sigma'}, & \text{if } \sigma \in A^\star_\tau \wedge \sigma' \in A^\star_\tau \\ p|_{\{\varepsilon\}}, & \text{if } \sigma \notin A^\star_\tau \wedge \sigma' = \varepsilon \\ p, & \text{if } \sigma \notin A^\star_\tau \wedge \sigma' \notin A^\star_\tau. \end{cases}$$

Notice how the above formulation closely follows the three defining clauses of $\mathcal{O}$.

**Proposition 5.17** *The mapping $\bar{F}_X$ defined above is a contravariant functor.*

To characterise branching bisimulation, define a structure $A^\star_{\bar{\tau}}$ similar to $A^\star$ and a structure preserving map $A^\star_{\tau\bar{\tau}} \longrightarrow A^\star_{\bar{\tau}}$ similar to $h$ (Section 5.3). To this end, our semantic category for branching bisimulation will be $A^\star_{\bar{\tau}} = A^\star \cup \{\bar{\tau}\}$ ordered by the relation $\preceq \subseteq A^\star_{\bar{\tau}} \times A^\star_{\bar{\tau}}$ consisting of prefix relation and the pairs $(\varepsilon, \bar{\tau}), (\bar{\tau}, \bar{\tau})$. We consider $A^\star_{\tau\bar{\tau}} \xrightarrow{\bar{h}} A^\star_{\bar{\tau}}$ given by: $\bar{h}(\sigma) = h(\sigma)$, if $\sigma \in A^\star_\tau$; and $\bar{h}(n, \bar{\tau}) = \bar{\tau}$ (for all $n \in \mathbb{N}$).

Next, we aim to compute the presheaf $\Sigma_{\bar{h}}\bar{F}_X$ at $\varrho \in A^\star_{\bar{\tau}}$. At this stage, we can take the advantage of similarity between the structures $A^\star_\tau, A^\star_{\tau\bar{\tau}}$ and $A^\star, A^\star_{\bar{\tau}}$ to compute the colimits, similar to (9) where ${A^\star_\tau}^{\mathrm{op}}$ is replaced by ${A^\star_{\tau\bar{\tau}}}^{\mathrm{op}}$ as the indexing category. In lieu of Theorems 5.11 and 5.12, we obtain

**Theorem 5.18** *For a given transition system $(X, A_\tau, \rightarrow)$, we find that $\Sigma_{\bar{h}}\bar{F}_X(\bar{\tau}) \cong \coprod_{n\in\mathbb{N}} Exec(X, \tau^n)$ and $\Sigma_{\bar{h}}\bar{F}_X(\varrho) \cong MExec(X, \varrho)$ (for any $\varrho \neq \bar{\tau}$). Moreover, the above isomorphisms are natural in $\varrho \in A^\star_{\bar{\tau}}$. I.e., for any $\varrho' \preceq \varrho$ with $\varrho, \varrho' \in A^\star$, the square in (12) and the following square commute.*

$$\begin{array}{ccc} \Sigma_{\bar{h}}F_X(\bar{\tau}) & \xrightarrow{\cong} & \coprod_{n\in\mathbb{N}} Exec(X, \tau^n) \\ \downarrow{\scriptstyle \Sigma_{\bar{h}}(\bar{\tau}, \varepsilon)} & & \downarrow{\scriptstyle \_|_\varepsilon} \\ \Sigma_h F_X(\varepsilon) & \xrightarrow{\cong} & MExec(X, \varepsilon) \end{array}$$

Note that the definition of the maps $\Sigma_{\bar{h}}(\varrho', \varrho)$ (for $\varrho' \preceq \varrho$) is similar to the maps $\Sigma_h(\varrho', \varrho)$ defined by the universal property of colimits (see (11)). Just like in the previous subsection, we utilise the isomorphic view of minimal executions to define our semantic map $\mathbf{LTS}_\tau \xrightarrow{[\![\_]\!]} \mathbf{PSh}(A^\star_{\bar{\tau}})$:

- For a given transition system, we let $[\![X]\!] = \Sigma_{\bar{h}}\bar{F}_X$.
- For a given branching simulation function $X \xrightarrow{f} Y$, we let $[\![f]\!]_\varrho(p) = p_f$.

**Lemma 5.19** *The above mapping $[\![\_]\!]$ is a faithful functor.*

Finally, we have obtained the desired result of this section.

**Theorem 5.20** *A branching simulation function $f$ is a branching bisimulation function iff $[\![f]\!]$ is a bisimulation map in $\mathbf{PSh}(A^\star_{\bar{\tau}})$.*

# 6 Related work and Conclusion

The core idea of **Goguen's sheaf semantics** [16] is: systems are diagrams of sheaves, behaviour (interconnection) of systems is their limit (colimit). In retrospect, Goguen gave a sheaf semantics of nondeterministic automata by constructing presheaves on the Alexandroff topology induced by the downward closed subsets of $A^\star$ (in contrast to presheaves over $A^\star$). However, it is well known that the category of sheaves on Alexandroff spaces induced by a poset $(X, \preceq)$ is equivalent to the category of presheaves on $X$ (a consequence of the so-called comparison lemma in topos theory; see [27, Corollary 3 on Page 590]). In short, we use simpler structures to represent executions and focused on defining bisimulations abstractly which were absent in [16].

**Relational presheaves** [31] generalise transition systems that are labelled by words from the free monoid $A^\star$. The idea was to accommodate the earlier presheaf approaches [23,34] with algebraic structure on labels. The most insightful observation of [31] was the well-known 'saturation' construction on transition systems can be captured using a 2-adjunction induced by a homomorphism between $A^\star_\tau, A^\star$. Unlike [31], our left adjoint $\Sigma_h$ records the minimal executions induced by a transition system with silent steps. We expect this to be relevant for probabilistic systems, where minimal executions are used to define a probability measure (cf. [6]).

**Open maps** between presheaves as defined in [23] are instances of the open maps in a topos as introduced in [22]. This is because open maps between presheaves (as in [23]) are natural transformations whose naturality square is a weak pullback in **Set** (cf. [22, Example 1.1]). We discarded the open maps between presheaves because they are incapable of establishing complete refinement between an implementation and its specficiation; though it is still interesting to assert whether the bisimulation maps (Def. 3.1) satisfy the axioms given in [22].

**Prefix orders** are generalisations of trees proposed in [10] to study executions of dynamical systems in an order theoretic manner. In [5], the authors defined functional bisimulation between prefix orders by reinterpreting the definition of open maps in concrete categories. Our bisimulation maps are an instance of this general definition (Section 3). It is unclear, though, how to enrich prefix orders with observations so that we can model labelled executions in a uniformly. This question lead us to model observations as presheaves.

To sum up, bisimulation maps between presheaves are versatile enough to capture different notions of behavioural equivalence. We demonstrated this by characterising $\forall$-fair bisimulation and branching bisimulation, two notions that are notoriously difficult to capture with a coalgebraic approach. The clear distinction between time and observation proved fruitful in dealing with silent actions, but we also expect our framework to lend itself well to modelling hybrid systems. For instance, sheaves over the translation-invariant interval domain $\mathbb{IR}_{/\rhd}$ were introduced in [30] to model hybrid systems. It will be interesting to explore whether bisimulation maps between such sheaves (Remark 3.2) coincides with stateless bisimulation [7,11,32].

## Acknowledgement

We thank the anonymous reviewers of FOSSACS'19 for their feedback on an earlier draft that greatly improved this manuscript. In particular, Reviewer 1 not only identified a mistake in our earlier characterisation of branching bisimulation, but also provided a detailed feedback including the proofs for which we are grateful. We thank Barbara König and Christina Mika for valuable comments on earlier drafts of this paper. We also thank Pieter Cuijpers for asking us about the importance of presheaf categories in system modelling, which formed the basis of Section 2. Moreover, we are grateful to Alex Simpson for discussing early ideas and in particular, for guiding our attention from sheaves on Alexandroff spaces induced by the poset $A^\star$ to just presheaves on $A^\star$. Finally, we acknowledge Paul Taylor's diagram package for commutative diagrams.

## References

[1] Abramsky, S., *What are the fundamental structures of concurrency?: We still don't know!*, Electronic Notes in Theoretical Computer Science **162** (2006), pp. 37 – 41, essays on Algebraic Process Calculi.

[2] Adámek, J., H. Herrlich and G. E. Strecker, "Abstract and Concrete Categories," Wiley-Interscience, New York, NY, USA, 1990.

[3] Baeten, J. C. M., T. Basten and M. A. Reniers, "Process Algebra: Equational Theories of Communicating Processes," Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, 2009.

[4] Baeten, J. C. M. and R. J. van Glabbeek, *Another look at abstraction in process algebra*, in: T. Ottmann, editor, *Automata, Languages and Programming* (1987), pp. 84–94.

[5] Beohar, H. and P. J. L. Cuijpers, *Open maps in concrete categories and branching bisimulation for prefix orders*, Electronic Notes in Theoretical Computer Science **319** (2015), pp. 51 – 66, the 31st Conference on the Mathematical Foundations of Programming Semantics (MFPS XXXI).

[6] Beohar, H. and S. Küpper, *On path-based coalgebras and weak notions of bisimulation*, in: F. Bonchi and B. König, editors, *7th Conference on Algebra and Coalgebra in Computer Science (CALCO 2017)*, Leibniz International Proceedings in Informatics (LIPIcs) **72** (2017), pp. 6:1–6:17.

[7] Beohar, H., D. E. Nadales Agut, D. A. van Beek and P. J. L. Cuijpers, *Hierarchical states in the compositional interchange format*, in: *Proceedings Seventh Workshop on Structural Operational Semantics, SOS*, 2010, pp. 42–56.

[8] Cattani, G. L. and G. Winskel, *Profunctors, open maps and bisimulation*, Mathematical Structures in Computer Science **15** (2005), p. 553–614.

[9] Caucal, D., *Branching bisimulation for context-free processes*, in: R. K. Shyamasundar, editor, *FSTTCS* (1992), pp. 316–327.

[10] Cuijpers, P. J. L., *Prefix orders as a general model of dynamics*, in: I. Mackie, M. Ayala-Rincón and E. Bonelli, editors, *Proc. of Developments in Computation Models*, DCM'13, Buenos Aires, Argentina, 2013, pp. 25–29.

[11] Cuijpers, P. J. L. and M. A. Reniers, *Hybrid process algebra*, The Journal of Logic and Algebraic Programming **62** (2005), pp. 191 – 245.

[12] Cuijpers, P. J. L. and M. A. Reniers, *Lost in translation: Hybrid-time flows vs. real-time transitions*, in: M. Egerstedt and B. Mishra, editors, *Hybrid Systems: Computation and Control* (2008), pp. 116–129.

[13] Eberhart, C. and T. Hirschowitz, *What's in a game?: A theory of game models*, in: *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18 (2018), pp. 374–383. URL http://doi.acm.org/10.1145/3209108.3209114

[14] Eberhart, C., T. Hirschowitz and T. Seiller, *An intensionally fully-abstract sheaf model for π (expanded version)*, Logical Methods in Computer Science **Volume 13, Issue 4** (2017).
URL https://lmcs.episciences.org/4069

[15] Fiore, M., G. L. Cattani and G. Winskel, *Weak bisimulation and open maps*, in: *Proc. 14th Symposium on Logic in Computer Science*, 1999, pp. 67–76.

[16] Goguen, J. A., *Sheaf semantics for concurrent interacting objects*, in: *Mathematical Structures in Computer Science*, 1992, pp. 159–191.

[17] Hennessy, M. C. B. and C. P. Stirling, *The power of the future perfect in program logics*, Information and Control **67** (1985), pp. 23 – 52.

[18] Henzinger, T. A., O. Kupferman and S. K. Rajamani, *Fair simulation*, Inf. Comput. **173** (2002), pp. 64–81.

[19] Hildebrandt, T. T., *Towards categorical models for fairness: fully abstract presheaf semantics of SCCS with finite delay*, Theoretical Computer Science **294** (2003), pp. 151 – 181.

[20] Hirschowitz, T. and D. Pous, *Innocent strategies as presheaves and interactive equivalences for ccs*, Scientific Annals of Computer Science **22** (2013), pp. 147–199.

[21] Hojati, R., "A BDD-Based Environment for Formal Verification of Hardware Systems," Ph.D. thesis, EECS Department, University of California, Berkeley (1996).
URL http://www2.eecs.berkeley.edu/Pubs/TechRpts/1996/3052.html

[22] Joyal, A. and I. Moerdijk, *A completeness theorem for open maps*, Annals of Pure and Applied Logic **70** (1994), pp. 51 – 86.

[23] Joyal, A., M. Nielsen and G. Winskel, *Bisimulation from open maps*, Information and Computation **127** (1996), pp. 164 – 185.

[24] Kupferman, O., N. Piterman and M. Y. Vardi, "Fair Equivalence Relations," Springer, Berlin, Heidelberg, 2003 pp. 702–732.

[25] Kwiatkowska, M., *Survey of fairness notions*, Information and Software Technology **31** (1989), pp. 371 – 386.

[26] Mac Lane, S., "Categories for the Working Mathematician," Number 5 in Graduate texts in Mathematics, Springer, NY, 1998, second edition.

[27] Mac Lane, S. and I. Moerdijk, "Sheaves in geometry and logic: a first introduction to topos theory," Universitext, Springer, 1992.

[28] Polderman, J. W. and J. C. Willems, "Introduction to Mathematical Systems Theory: A Behavioral Approach," Texts in Applied Mathematics **26**, Springer-Verlag, New York, 1998, 1 edition.

[29] Rutten, J. J. M. M., *Universal coalgebra: a theory of systems*, Theoretical Computer Science **249** (2000), pp. 3 – 80.

[30] Schultz, P. and D. I. Spivak, *Temporal Type Theory: A topos-theoretic approach to systems and behavior* (2017).
URL https://arxiv.org/abs/1710.10258

[31] Sobociński, P., *Relational presheaves, change of base and weak simulation*, Journal of Computer and System Sciences **81** (2015), pp. 901 – 910, 11th International Workshop on Coalgebraic Methods in Computer Science, CMCS 2012 (Selected Papers).

[32] van Beek, D. A., K. L. Man, R. M. A., J. E. Rooda and R. R. H. Schiffelers, *Syntax and consistent equation semantics of hybrid Chi*, The Journal of Logic and Algebraic Programming **68** (2006), pp. 129 – 210.

[33] van Glabbeek, R. J. and W. P. Weijland, *Branching time and abstraction in bisimulation semantics*, J. ACM **43** (1996), pp. 555–600.

[34] Winskel, G. and M. Nielsen, *Presheaves as transition systems*, in: *Proceedings of the DIMACS Workshop on Partial Order Methods in Verification*, POMIV '96 (1997), pp. 129–140.