# From Enhanced Coinduction towards Enhanced Induction

DAVIDE SANGIORGI, University of Bologna, Italy and INRIA, France

There exist a rich and well-developed theory of enhancements of the *coinduction* proof method, widely used on behavioural relations such as bisimilarity. We study how to develop an analogous theory for *inductive* behaviour relations, i.e., relations defined from inductive observables. Similarly to the coinductive setting, our theory makes use of (semi)-progressions of the form $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$, where $\mathcal{R}$ is a relation on processes and $\mathcal{F}$ is a function on relations, meaning that there is an appropriate match on the transitions that the processes in $\mathcal{R}$ can perform in which the process derivatives are in $\mathcal{F}(\mathcal{R})$. For a given preorder, an enhancement corresponds to a *sound* function, i.e., one for which $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ implies that $\mathcal{R}$ is contained in the preorder; and similarly for equivalences. We introduce *weights* on the observables of an inductive relation, and a *weight-preserving* condition on functions that guarantees soundness. We show that the class of weight-preserving functions contains non-trivial functions and enjoys closure properties with respect to desirable function constructors, so to be able to derive sophisticated sound functions (and hence sophisticated proof techniques) from simpler ones. We consider both strong semantics (in which all actions are treated equally) and weak semantics (in which one abstracts from internal transitions). We test our enhancements on a few non-trivial examples.

CCS Concepts: • **Theory of computation** → **Operational semantics**; *Process calculi*; • **Software and its engineering** → *Software verification.*

Additional Key Words and Phrases: coinduction, behavioural relations, process calculi, proof techniques

**ACM Reference Format:**
Davide Sangiorgi. 2022. From Enhanced Coinduction towards Enhanced Induction. *Proc. ACM Program. Lang.* 6, POPL, Article 18 (January 2022), 29 pages. https://doi.org/10.1145/3498679

## 1 INTRODUCTION

One of the most basic elements for reasoning on programs or systems is the meaning of behavioural equality. Intuitively, two terms are equal if no difference between them can be observed by interacting with them. The two terms may be thought of as the specification and the implementation of a certain system, or as two different implementations. Sometimes one prefers a preorder to an equivalence. In this case two terms are related when all relevant observations about a term can also be made about the other one. For instance the specification of a system may leave freedom about a number of details, and a correct implementation could have a more constrained behaviour.

The meaning of equality on processes has produced a rich and profound debate, particularly in Concurrency Theory, very active in the 1970s and 1980s and not yet exhausted. A number of proposals for behavioural preorders and equalities have been made; see, e.g, van Glabbeek's spectrum [Glabbeek 1993, 2001]. Among the notions so formulated, bisimilarity has emerged as one of the most studied and used [Milner 1989; Park 1981]. While introduced in Concurrency Theory, bisimilarity has spread to other areas of Computer Science, as well as to other domains such as Mathematics and Cognitive Science.

Author's address: Davide Sangiorgi, Department of Computer Science (DISI), University of Bologna, Italy, Davide.Sangiorgi@unibo.it and INRIA, Inria Sophia Méditerranée, Sophia Antipolis, France.

2475-1421/2022/1-ART18
https://doi.org/10.1145/3498679

Bisimilarity is the union of all bisimulations. A bisimulation is a relation on the terms of a language that is invariant under the observables of the language (i.e., what can be observed of the terms). Thus the definition itself immediately leads to a well-established proof technique: to prove two terms bisimilar, find a bisimulation relation containing the two terms as a pair. Furthermore, such a proof method can be enhanced, with the goal of making it more effective (easier to use, both in paper proofs and in tools for automated or semi-automated analysis) and more broadly applicable. Examples of enhancements are 'up-to context', 'up-to transitive closure', 'up-to bisimilarity', 'up-to environment', and so on [Pous and Sangiorgi 2019]. Theories of enhancements have been proposed [Pous 2007, 2016; Sangiorgi 1998]. Most important, these theories allow one to *combine* enhancements so to obtain, for free, the soundness of more complex enhancements. Bisimilarity and the bisimulation proof method are instances of a coinductive definition and of the coinduction proof method. Analogously the bisimulation enhancements can be lifted to coinduction; see [Pous and Sangiorgi 2012] for a presentation that follows fixed-point theory.

The bisimulation proof method and its enhancements are a major reason for the success of bisimilarity. Sometimes the enhancements seem essential to be able to carry out a proof: defining a full bisimulation, with all needed pairs, can be considerably hard, let alone carrying out the whole proof. This is frequent in languages for name mobility such as the $\pi$-calculus and its many dialects, and in languages including higher-order features such as $\lambda$-calculi. In these languages bisimilarity is hardly ever applied without enhancements.

As a behavioural equivalence, however, bisimilarity has also been criticised. One of the main arguments is that it may be regarded as too fine, discriminating processes that an external observer could not tell apart. For instance, in the CSP community failure equivalence [Brookes et al. 1984] is used in place of bisimilarity. Another argument against bisimilarity is that it does not have a natural associated preorder. For instance, similarity — the 'one-way' bisimilarity — or variants of it do not yield bisimilarity as their induced equivalence [Sangiorgi 2012]. (Further, similarity as a behavioural preorder is often inadequate because it does not respect deadlocks.) Various inductive behavioural relations, both preorders and equivalences, have been put forward and studied that improve on such limitations: examples are preorders based on traces, failures, ready sets, refusals, may and must testing, ready and failure traces, e.g., [Baeten et al. 1987; Brookes et al. 1984; Brookes and Roscoe 1984; De Nicola and Hennessy 1984; Glabbeek 2001; Hennessy 1988; Olderog and Hoare 1986; Phillips 1987; Pnueli 1985; Pomello 1985]. We call them 'inductive' because resulting from inductively-defined observables, usually enriched forms of traces. The goal of this paper is to transport the theory of enhancements for bisimilarity onto such relations. Correspondingly, we sometimes call 'inductive' the resulting enhancements.

At the heart of theories of bisimulation enhancements such as [Pous 2007, 2016; Sangiorgi 1998] is the notion of *progression*. A progression from a relation $\mathcal{R}$ to a relation $\mathcal{S}$, written $\mathcal{R} \rightarrowtail \mathcal{S}$, indicates that pairs of processes in $\mathcal{R}$ can match each other's actions and their derivatives (i.e., the processes resulting from performing such actions) are in $\mathcal{S}$. The progressions that are considered are of the form $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$, where $\mathcal{F}$ is a function on relations. Conditions on functions on relations guarantee *soundness* of the progressions, meaning that if $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ then $\mathcal{R}$ only includes pairs of bisimilar processes. These are functorial-like conditions such as *respectfulness* [Sangiorgi 1998] and *compatibility* [Pous 2007], reviewed in Section 15. Such conditions can then be extended to higher-order functions, also called *constructors*, so to be able to combine sound functions, that is, to derive sophisticated sound functions — and hence sophisticated proof techniques for bisimilarity — from simpler ones. As an example, 'up-to bisimilarity' can be combined with 'up-to context' yielding 'up-to bisimilarity and context', in which one is allowed, in the bisimulation game, both to rewrite the derivative processes into bisimilar ones, and to remove, in the resulting terms, a common context. In fact, in this way even 'up-to bisimilarity' is derivable from simpler functions, namely the identity function and the constant function mapping every relation onto bisimilarity itself [Sangiorgi 1998].

In this paper we maintain progressions as the basic schema for studying enhancements. We actually employ one-way progressions, called semi-progressions and written $\mathcal{R} \rightarrowtail \mathcal{S}$, as we aim to capture also preorders (besides equivalences). We change however the meaning of soundness, and the conditions on functions to guarantee soundness. Moreover, everything is parametrised on a preorder, say $\precsim$, as we wish the theory to be applicable to different preorders. In our enhancements, a function $\mathcal{F}$ is *sound for* $\precsim$ if $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ implies that $\mathcal{R}$ is included in the given preorder $\precsim$; and similarly for equivalences. The crux of the theory of inductive enhancements is the condition for functional soundness that should permit composition of enhancements. For this we exploit the inductive definition of the observables. We associate a weight to each observable, intuitively related to the depth of the nesting of actions in the behaviour of a process that may have be looked at when checking that observable. We thus obtain a stratification of the preorder $\precsim$; at stage $n$ of the preorder, $\precsim_n$, only the observables of weight less than or equal to $n$ are taken into account. The condition for functional soundness, *weight-preservation*, requires that if $\mathcal{R}$ complies with $\precsim_n$, i.e., $\mathcal{R} \subseteq \precsim_n$, then also $\mathcal{F}(\mathcal{R})$ should comply, i.e., $\mathcal{F}(\mathcal{R}) \subseteq \precsim_n$. In the case of equivalences, rather than preorders, one adds analogous converse requirements on pairs of related processes.

We consider common basic functions and constructors in the literature about bisimulation enhancements. We show that they are weight-preserving, and may therefore be used also in inductive enhancements. Examples of basic functions and constructors are function composition, union, chaining (that gives us relational composition), and the context-closure function; in the paper we examine the last one in a CCS-like language. Examples of derived functions are the transitive-closure function, the closure under context and $\precsim$ (the analogous of the 'up-to context and bisimilarity' enhancement for bisimilarity).

As behavioural relations we consider the most popular inductive preorders and equivalences in the literature, following [Glabbeek 1993, 2001]. They include the trace, failure, failure trace, ready, and ready trace preorders (other preorders, like may and must testing and refusal, coincide with some of these, under mild conditions on the transitions performed by the processes), and their induced equivalences. In all cases, the soundness of the above functions and constructors is usually straightforward, the only exception being the context-closure function.

We work on ordinary Labeled Transition Systems (LTSs), and call processes the states of an LTS. We develop the theory under a strong semantics, where all actions are equally visible. We then study weak semantics, in which a special action denoting internal activity may be partly or completely ignored. It is well-known that theories of weak coinductive enhancements tend to be rather more involved than the 'strong' theories. For instance, a useful constructor, chaining, is sound only in the strong theories. Similar issues show up in the inductive enhancements. In addition, some of the weak behavioural relations make use of state predicates such as *stability*, which do not appear in the strong case. Some of the technical solutions that we adopt for the inductive setting are inspired from those used in the coinductive setting, others are specific to the weight-based conditions of the inductive setting. We consider different forms of weight (for instance distinguishing the contribution of internal and visible actions), and examine their relative advantages and disadvantages.

The usefulness of our proof techniques is examined on 3 non-trivial examples. The first example is about a key property of many languages, namely substitutivity for recursive definitions. The example also shows how the inductive enhancements may yield proofs that are *parametric* on the preorder (or equivalence) and the language adopted. The second example is about concrete systems for which bisimilarity may be argued to be too fine, or in which bisimilarity suffers from the lack of an associated preorder. The systems to be related are two servers. They differ in the order in which certain interactions with some auxiliary servers occur. We consider two instances of the systems, the second instance only suitable for preorders. In both cases the servers may be related by any of the inductive relations considered in the paper (in the second case only the preorders). The enhancements allow us to carry out the proofs using singletons relations. In none of the instances the servers are bisimilar. The third example is about the soundness of a proof system. Proof systems underpin algebraic reasoning, widely used in verification tools. The proof system we consider has been proposed by Hennessy [Hennessy 2017] for the trace preorder. The system is non-standard; for instance, in contrast with earlier proof systems by Salomaa [Salomaa 1966], Kozen [Kozen 1994], and Rabinovich [Rabinovich 1993] for trace equivalence, it does not rely on forms of unique fixed-point induction. This however makes the soundness proof delicate. We show that soundness can be explained via the inductive enhancements. Another reason for choosing the example is that it combines a number of enhancement functions. We first analyse all the examples under strong semantics; then we examine the weak versions of the examples, and variants of them, under weak semantics. All examples are described in CCS; however any language capable of expressing interaction would do.

*Structure of the paper.* In Section 2, we formalise the meaning of inductive observables, using operators from modal logics, In Section 3 we relate such observables to common inductive preorders and equivalences from the literature. In Section 4 we introduce weights and weighted stratification of the behavioural preorders and equivalences. In Section 5 we develop the theory of semi-progressions, sound functions, and weight-preserving functions. In Section 6 we derive the weight-preserving property for a number of functions; the context-closure function requires one to fix a language, and for this use a CCS-like language. In Section 7 we transport the work for preorders in earlier sections to equivalences. In Sections 8-10 we discuss the three examples of use of enhancements mentioned above. In Sections 11 to 13 we study the theory weak semantics, and in Section 14 we revisit the previous examples, and variants of them, under the weak semantics. Finally, in Sections 15 and 16, we discuss further related work and directions for future work.

For space reasons, some auxiliary definitions and results and all proofs are moved to an appendix available online in the ACM digital library as supplemental material for this paper.

*Preliminaries.* We work on ordinary *Labeled Transition Systems* (LTSs), i.e., triples $(Pr, Act, \longrightarrow)$ with domain $Pr$, set of *actions* (or *labels*) $Act$, and transition relation $\longrightarrow \subseteq Pr \times Act \times Pr$. We use $P, Q$ and $R$ to range over $Pr$ and call them *processes*; $\mu$ ranges over the actions. As usual $P \xrightarrow{\mu} Q$ stands for $(P, \mu, Q) \in \longrightarrow$, to be interpreted as "$P$ may become $Q$ by performing an action $\mu$". We write $P \xrightarrow{\mu}$ if there is $P'$ with $P \xrightarrow{\mu} P'$, and $P \not\xrightarrow{\mu}$ if there is no $P'$ such that $P \xrightarrow{\mu} P'$. Sometimes in examples we use basic CCS operators such as sum and prefix.

## 2 INDUCTIVE OBSERVABLES AND PREORDERS

We begin with ordinary LTSs and strong semantics, where all actions are equally treated.

To define our inductive enhancements, we adopt a presentation of behavioural preorders and equivalences in which the observables employed are described by means of modal formulas. The operators include the 'diamond' $\langle \mu \rangle . \theta$, to detect the possibility of performing the action $\mu$, a (possibly infinitary) 'and', to permit multiple observations, and a set of atomic observables. Without the atomic observables, this is the positive fragment of Hennessy-Milner logic [Hennessy and Milner 1985], whose satisfaction problem is straightforward — with or without atomic formulas [Stirling 2001].

*Definition 2.1.* Let $Act$ and $\mathcal{A}$ be (countable) sets of *actions* and of *atomic observables*. A *family of observables (over Act and $\mathcal{A}$)* is a subset $O$ of the formulas inductively generated by the following grammar:

$$\theta \quad := \quad \langle \mu \rangle . \theta \;\Big|\; \textstyle\bigwedge_{j \in J} \theta_j \;\Big|\; \mathsf{a}$$

where $\mu \in Act$, $\mathsf{a} \in \mathcal{A}$, $J$ is a countable set, and:

- (downward-closure) if $\langle \mu \rangle . \theta \in \mathcal{O}$ then also $\theta \in \mathcal{O}$, and similarly if $\bigwedge_{j \in J} \theta_j \in \mathcal{O}$ then also $\theta_j \in \mathcal{O}$ for each $j$.

We write $\mathtt{true}$ for the formula corresponding to the empty conjunction, and $\theta_1 \wedge \theta_2$ for a binary conjunction. We let $\mathcal{O}$ range over families of observables, and $\theta, \zeta$ over observables.

The downward-closure property requires that a family of observables is syntactically saturated, in that it contains all subterms of any observable in the family. The property is needed in some of our main theorems (e.g., Theorem 5.6).

The observables of a family $\mathcal{O}$ are meant to be interpreted as predicates over the processes of an LTS over the same set of actions, and accordingly we write $P \models \theta$ when the observable $\theta$ holds on the process $P$. The interpretation is defined structurally over the observables, writing $[\![\mathsf{a}]\!]$ for the set of processes on which the atomic observable $\mathsf{a}$ holds:

$$\textsc{at}\ \frac{P \in [\![\mathsf{a}]\!]}{P \models \mathsf{a}} \qquad \textsc{act}\ \frac{P \xrightarrow{\mu} P' \qquad P' \models \theta}{P \models \langle \mu \rangle . \theta} \qquad \textsc{conj}\ \frac{P \models \theta_j \qquad \text{for all } j \in J}{P \models \bigwedge_{j \in J} \theta_j}$$

The observable $\textsc{true}$ is satisfied by all processes.

In Section 3 we discuss the characterisation of common inductive preorders and equivalences in the literature. The only kind of atomic observable that will be needed is the refusal $\mathtt{ref}_\mu$, which holds of a process $P$ and an action $\mu$ if $P \overset{\mu}{\not\longrightarrow}$. However, for generality, in the theory we allow arbitrary atomic observables. Such observables should however be *local*: they can be checked by looking only at the immediate outgoing transitions emanating from a process. That is, for each such observable, say $\mathsf{a}$, there are sets of actions $A$ and $B$ such that for all process $P$, we have $P \in [\![\mathsf{a}]\!]$ iff ( $P \xrightarrow{\mu}$ for all $\mu \in A$, and $P \overset{\mu}{\not\longrightarrow}$ for all $\mu \in B$). For instance, for the refusal $\mathtt{ref}_\mu$, we have $A = \emptyset$ and $B = \{\mu\}$.

*Example 2.2.*

- The observable $\langle a \rangle . \langle b \rangle . \mathtt{true}$ is satisfied by any process that can perform an action $a$ followed by an action $b$, i.e., the trace $a; b$.
- The observable $\mathtt{ref}_a \wedge \mathtt{ref}_b$ is satisfied by any process that cannot perform $a$ or $b$ actions.
- The observable $\langle a \rangle . \langle b \rangle . (\langle c \rangle . \mathtt{true} \wedge \mathtt{ref}_d)$ checks whether a process, upon performing the actions $a$ and $b$, can reach a state in which $c$ is possible but not $d$.

*Definition 2.3 (preorder induced by $\mathcal{O}$).* A family $\mathcal{O}$ of observables induces a preorder $\precsim^{\mathcal{O}}$ on the processes, whereby $P \precsim^{\mathcal{O}} Q$ if $P \models \theta$ implies $Q \models \theta$ for all $\theta \in \mathcal{O}$.

## 3 PREORDER INSTANCES

The operators in the grammar for observables allow us to capture all inductive preorders and equivalences in van Glabbeek's spectrum [Glabbeek 2001] (collecting the main such relations in the literature) with the exception of possible-future semantics, namely: *trace*, *failure*, *failure trace*, *ready*, *ready trace* preorders and equivalences [Baeten et al. 1987; Brookes et al. 1984; Brookes and Roscoe 1984; Olderog and Hoare 1986; Pnueli 1985; Pomello 1985]. (Assuming image-finiteness of the LTSs, meaning that for all $P$ and $\mu$, the set $\{P' \mid P \xrightarrow{\mu} P'\}$ is finite, the set of such preorders also include the *may*, *must*, and *testing* preorders [De Nicola and Hennessy 1984; Hennessy 1988], and the *refusal preorder* [Phillips 1987].) Below we only informally and briefly recall the meaning of such relations and the grammar for observables needed, referring to the Appendix for more details, and to [Glabbeek 2001] for a comprehensive treatment (including comparisons among the preorders and equivalences, modal logic characterisations and axiomatisations). The notations and terminology here and in the Appendix follow [Glabbeek 2001]. For historical reasons, we maintain however references to the papers in which the behavioural relations were first proposed, even if the papers referred to adopt a 'weak' semantics (i.e, they abstract from internal transitions of the processes, as in Section 11).

In the *trace preorder* (which coincides with the *may-testing* preorder [De Nicola and Hennessy 1984; Hennessy 1988]) two processes $P$ and $Q$ are related if all traces, i.e., finite sequences of actions, that $P$ can produce can also be produced by $Q$. The preorder is captured using the following grammar of observables:

$$\theta := \langle \mu \rangle . \theta \;\; \big| \;\; \mathsf{true}$$

Thus, there is an observable $\langle \mu_1 \rangle . \dots . \langle \mu_n \rangle . \mathsf{true}$ for each trace $\mu_1 ; \dots ; \mu_n$. No atomic observable is required — in contrast we need the refusals $\mathtt{ref}_\mu$ in all the remaining preorders.

A *failure* is a pair $(\sigma, A)$, where $\sigma$ is a trace and $A$, called *a refusal set*, is a set of actions. The failure $(\sigma, A)$ belongs to process $P$ if the process can produce the trace $\sigma$ and, by doing so, becomes a process $P'$ that may not accept any action in $A$ (i.e., $P' \stackrel{\mu}{\not\rightarrow}$ for all $\mu \in A$). In the *failure preorder* [Brookes et al. 1984; Brookes and Roscoe 1984] (which coincides with the inverse of the *must-testing* preorder [De Nicola and Hennessy 1984; Hennessy 1988]) $P$ and $Q$ are related if all failures of $P$ are also failures of $Q$. The preorder is captured by the observables of the following grammar

$$\theta := \langle \mu \rangle . \theta \;\; \big| \;\; \bigwedge_{\mu \in A} \mathtt{ref}_\mu$$

(with the convention that, if $A$ is a singleton $\{\mu\}$, then $\wedge_{\mu \in A} \mathtt{ref}_\mu$ is the same as $\mathtt{ref}_\mu$). Set $A$ may be empty, and then $\wedge_{\mu \in A} \mathtt{ref}_\mu$ is true.

In the *failure trace preorder* [Glabbeek 2001] (which coincides with the *refusal preorder* [Phillips 1987]), refusal sets may be observed at any point along a trace of actions. The grammar for observables is

$$\theta := \langle \mu \rangle . \theta \;\; \big| \;\; \bigwedge_{\mu \in A} \mathtt{ref}_\mu \;\; \big| \;\; (\bigwedge_{\mu \in A} \mathtt{ref}_\mu) \wedge \theta$$

In the *ready preorder* [Olderog and Hoare 1986], one observes, after a trace, the *ready set* of the final process, that is, the set of all actions that the process can perform. A ready set $A$ is captured by the observable $\Big( \wedge_{\mu \in A} \langle \mu \rangle . \mathsf{true} \Big) \wedge \Big( \wedge_{\mu \notin A} \mathtt{ref}_\mu \Big)$. In the *ready trace preorder* [Baeten et al. 1987; Pnueli 1985; Pomello 1985], the ready sets of all the processes encountered along a trace are taken into account. We refer to the Appendix for details on the ready and ready trace preorders (definitions of the preorders, and grammars for observables).

The operators for observables in Definition 2.1 are known to be able to capture also coinductive relations such as simulation, or refinements of it such as ready simulation. Precisely, referring to van Glabbeek's spectrum [Glabbeek 2001], the operators can capture all refinements of simulation up to ready simulation [Bloom et al. 1995] (that coincides with its failure analogue, failure simulation). To go further, however, e.g., capturing Groote and Vaandrager's 2-nested simulation [Groote and Vaandrager 1992] or Milner's bisimilarity [Milner 1989], one needs extra operators (e.g., negation). In this paper we do not discuss simulation-like relations for which the natural forms of enhancement — the main goal of this paper — are the coinductive ones.

## 4 WEIGHTS

We are interested in observables that can be measured, that is, whenever $P \models \theta$ holds then a weight can be assigned to a proof of $P \models \theta$. Intuitively, a weight $n$ in $P \models^n \theta$ means that $P \models \theta$ can be proved by exploring the transition tree emanating from $P$ up to a depth not greater than $n$.

We thus refine the rules for satisfaction of observables in Section 2 by adding weights and weight constraints. To distinguish the rules with weights we add a '+' to their names:

$$\text{AT+}\ \frac{P \in [\![\mathsf{a}]\!] \qquad n \geq 1}{P \models^n \theta} \qquad\qquad \text{ACT+}\ \frac{P \xrightarrow{\mu} P' \qquad P' \models^m \theta \qquad m+1 \leq n}{P \models^n \langle \mu \rangle . \theta}$$

$$\text{CONJ+}\ \frac{P \models^{m_j} \theta_j \qquad m_j \leq n \text{ for each } j}{P \models^n \bigwedge_{j \in J} \theta_j}$$

Thus given an LTS $\mathcal{L}$ and a family $\mathcal{O}$ of observables over the same set of actions, a process $P$ of $\mathcal{L}$, and an observable $\theta \in \mathcal{O}$, an assertion $P \models^n \theta$ holds if derivable from the weighted rules above.

The condition in rule CONJ+ ensures us that the trees of transitions emanating from $P$ that are explored in the proofs of $P \models^{m_j} \theta_j$, for $j \in J$, have a bound in their depth. (In principle, the observables in Definition 2.1 need not be measurable, because they may contain infinite conjunctions. For instance, an observable requiring a process to perform $n$ consecutive $a$ actions, for each $n$, would not be measurable.) In rule AT+, the weight is at least 1 because the atomic observables are local. The only observable for which $n$ can be 0 is true, i.e., CONJ+ with $J$ empty (or conjunctions of true).

REMARK 4.1. The weights in statements $P \models^n \theta$ in this section could be set syntactically, by looking at the depth of nesting of diamond operators in $\theta$. This however will not hold in the weak semantics of Section 11.

*Example 4.2.* Suppose $P$ is a process with $P \xrightarrow{a} P' \xrightarrow{b} P''$. A proof of $P \models^2 \langle a \rangle . \langle b \rangle . \mathsf{true}$ is

$$\frac{P \xrightarrow{a} P' \qquad \dfrac{P' \xrightarrow{b} P'' \qquad \dfrac{}{P'' \models^0 \mathsf{true}} \qquad 0+1 \leq 1}{P' \models^1 \langle b \rangle . \mathsf{true}} \qquad 1+1 \leq 2}{P \models^2 \langle a \rangle . \langle b \rangle . \mathsf{true}}$$

Similarly $P \models^n \langle a \rangle . \langle b \rangle . \mathsf{true}$, for any $n \geq 2$. However, $P \models^1 \langle a \rangle . \langle b \rangle . \mathsf{true}$ does not hold.

If there is a proof $P \models^n \theta$, then there is also a proof for $P \models^{n+i} \theta$, for any $i$; it suffices to increase by $i$ all the weight of the conclusion of the last rule:

LEMMA 4.3. *If $P \models^n \theta$ and $n \leq m$ then also $P \models^m \theta$.*

*Definition 4.4.* A family of observables $\mathcal{O}$ and the corresponding preorder $\precsim^{\mathcal{O}}$ are *measurable* if, for any LTS (over the same set of actions) and process $P$ of the LTS, and for any observable $\theta \in \mathcal{O}$, whenever $P \models \theta$ then also $P \models^n \theta$ holds, for some $n \geq 0$.

The introduction of weights allows us to define approximations of the behavioural preorders.

*Definition 4.5 (stratification).* Two processes $P, Q$ are in the *$n$-th approximant of $\precsim^{\mathcal{O}}$*, written $P \precsim^{\mathcal{O}}_n Q$, if $P \models^n \theta$ implies $Q \models \theta$, for all $\theta$.

In Definition 4.5 $Q \models \theta$ could be replaced by $Q \models^n \theta$. We use $Q \models \theta$ for uniformity with the weak semantics of Section 11 (c.f., Remark 4.1).

LEMMA 4.6. *Relations $\precsim^{\mathcal{O}}_n$ are preorders.*

LEMMA 4.7. *If $\mathcal{O}$ is measurable, then $\precsim^{\mathcal{O}} = \cap_n \precsim^{\mathcal{O}}_n$.*

All preorders discussed in Section 3 are measurable.

## 5 FUNCTIONS FOR INDUCTIVE ENHANCEMENTS

We recall that, unless otherwise stated, a *relation* is meant to be a relation on processes. We let $\mathcal{R}$ and $\mathcal{S}$ range over binary relations, i.e., if $\wp$ denotes the powerset construct and $Pr$ the set of all processes, then $\mathcal{R}$ and $\mathcal{S}$ are elements of $\wp(Pr \times Pr)$. The union of relations $\mathcal{R}$ and $\mathcal{S}$ is $\mathcal{R} \cup \mathcal{S}$, and their composition is $\mathcal{R}\mathcal{S}$ (i.e., $(P, P') \in \mathcal{R}\mathcal{S}$ holds if for some $P''$, both $(P, P'') \in \mathcal{R}$ and $(P'', P') \in \mathcal{S}$ hold). We often use the infix notation for relations; hence $P \mathrel{\mathcal{R}} Q$ means $(P, Q) \in \mathcal{R}$. We use letters $I$ and $J$ for countable indexing sets in unions, conjunctions, and similar.

The definition below of semi-progression is essentially the "one-way" version of the progressions in the theory of enhanced coinduction [Sangiorgi 1998]. The converse clause is missing because we deal with preorders, rather than equivalences as in enhanced coinduction.

*Definition 5.1 (semi-progression).* Given two relations $\mathcal{R}$ and $\mathcal{S}$, we say that $\mathcal{R}$ *semi-progresses to* $\mathcal{S}$, written $\mathcal{R} \rightarrowtail \mathcal{S}$, if $P \mathrel{\mathcal{R}} Q$ implies:

- whenever $P \xrightarrow{\mu} P'$, there exists $Q'$ such that $Q \xrightarrow{\mu} Q'$ and $P' \mathrel{\mathcal{S}} Q'$.

We consider semi-progressions of the form $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ where $\mathcal{F}$ is a function on relations, i.e. a function from $\wp(Pr \times Pr)$ to $\wp(Pr \times Pr)$. We call these *first-order functions*, briefly *functions*. Below, $\mathcal{F}$ and $\mathcal{G}$ range over such functions.

In the inductive setting, the definitions of certain behavioural relations make use of atomic observables such as refusals, which do not appear in the definition of bisimulation. To handle them in the theory of inductive enhancements we use the following compliance requirement.

*Definition 5.2 (compliance).* A relation $\mathcal{R}$ is *compliant with* $\mathcal{O}$ (or simply *compliant*, if $\mathcal{O}$ is clear) if, for all atomic observables $\mathsf{a}$ in $\mathcal{O}$ and for all $P \mathrel{\mathcal{R}} Q$, $P \models \mathsf{a}$ implies $Q \models \mathsf{a}$.

Remark 5.3. Separating compliance and (semi-)progression allows us to maintain the standard notion of progression in the literature, which only refers to action transitions. Moreover, in this way progressions and semi-progressions are defined without reference to any family of observables.

*Definition 5.4 (soundness).* A function $\mathcal{F}$ is *sound for* $\precsim^{\mathcal{O}}$ if $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ implies $\mathcal{R} \subseteq \precsim^{\mathcal{O}}$, for any compliant $\mathcal{R}$.

Not all functions are sound. An example is the function that maps every relation onto the universal relation (the set of all pairs of processes). We wish to isolate a class of sound functions for which membership is easy to check, which includes interesting functions, and which satisfies interesting properties. For this we propose the notion of weight-preserving function. The usability of this notion will be examined in Section 6.

*Definition 5.5 (weight-preserving function).* A function $\mathcal{F}$ *preserves weights on* $\precsim^{\mathcal{O}}$ when, for all $n$ and for all $\mathcal{R}$, if $\mathcal{R} \subseteq \precsim^{\mathcal{O}}_n$, then also $\mathcal{F}(\mathcal{R}) \subseteq \precsim^{\mathcal{O}}_n$. Function $\mathcal{F}$ *preserves weights* if it preserves weights on $\precsim^{\mathcal{O}}$, for any $\precsim^{\mathcal{O}}$.

We first show that indeed any weight-preserving function is sound.

Theorem 5.6 (soundness of weight-preserving functions for preorders). *If $\mathcal{O}$ is measurable and $\mathcal{F}$ preserves weights on $\precsim^{\mathcal{O}}$, then $\mathcal{F}$ is sound for $\precsim^{\mathcal{O}}$.*

Proof. Suppose $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ with $\mathcal{R}$ compliant. We have to show $\mathcal{R} \subseteq \precsim^{\mathcal{O}}$. Appealing to Lemma 4.7, we show that $\mathcal{R} \subseteq \precsim^{\mathcal{O}}_n$, for all $n$, by induction on $n$. Details may be found in the Appendix.

Remark 5.7. In the proof of Theorem 5.6 it may be puzzling that compliance for $\mathcal{R}$ is ensured by hypothesis, but no analogous hypothesis is made on $\mathcal{F}(\mathcal{R})$. However $\mathcal{F}$ preserves weights, which, given the hypothesis of the theorem, entails compliance for $\mathcal{F}(\mathcal{R})$.

Remark 5.8. If in Definition 5.5 we had taken 'preserving weights for $\precsim^{O}$' to mean

$$\text{`whenever } \mathcal{R} \subseteq \precsim^{O} \text{ then also } \mathcal{F}(\mathcal{R}) \subseteq \precsim^{O}\text{'}, \qquad (*)$$

the theorem above would not hold. As a counterexample, consider processes that are purely sequential and deterministic, i.e., mere sequences of actions such as $a_1. \dots a_n$, and a function $\mathcal{F}$ that removes the first action of the processes; for instance, yielding $\mathcal{F}(\{(a_1. \dots a_n, b_1. \dots b_m)\})= \{(a_2. \dots a_n, b_2. \dots b_m)\}$. If $\precsim^{O}$ is trace preorder, then such a function would preserve weights according to $(*)$. Yet for $\mathcal{R} \overset{\text{def}}{=} \{(a.b, a.c)\}$ we have $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$, although the processes in $\mathcal{R}$ are not in the trace preorder.

Technically, in the proof of Theorem 5.6 we do not know whether $\mathcal{R} \subseteq \precsim^{O}$ holds, hence we cannot use property $(*)$ in the proof (the current proof goes by induction, hence at step $n+1$ we can exploit the weight property for $\precsim^{O}_{n}$).

Remark 5.9 (completeness). By completeness for a preorder $\precsim^{O}$, we refer to the possibility of handling any relation $\mathcal{R} \subseteq \precsim^{O}$ in the framework. We can distinguish completeness for sound functions, meaning that for any $\mathcal{R} \subseteq \precsim^{O}$ there is a sound function $\mathcal{F}$ with $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$; and the analogous completeness for weight-preserving functions.

The inductive enhancements are trivially complete for sound functions: the function that maps a relation $\mathcal{R}$ onto the universal relation if $\mathcal{R} \subseteq \precsim^{O}$, and to the empty relation otherwise, is sound.

In the form stated above, completeness for weight-preserving functions fails. For instance, taking $\precsim^{\text{TR}}$ to be the trace preorder, there are processes $P, Q$ such that $P \precsim^{\text{TR}} Q$ but there is no compliant relation $\mathcal{R}$ containing the pair $(P, Q)$ that semi-progresses to $\mathcal{F}(\mathcal{R})$ for some $\mathcal{F}$ that preserves weights. To see this, consider $P \overset{\text{def}}{=} a.(b+c)$ and $Q \overset{\text{def}}{=} a.b + a.c$, where $P \precsim^{\text{TR}} Q$ holds. We have $P \xrightarrow{a} b+c$, which can only be matched by $Q \xrightarrow{a} b$ or $Q \xrightarrow{a} c$. However, we cannot have $(b+c, b)$ or $(b+c, c)$ in $\mathcal{F}(\mathcal{R})$, for some $\mathcal{R}$ with $(P, Q) \in \mathcal{R} \subseteq \precsim^{\text{TR}}$ and $\mathcal{F}$ that preserves weights: when a function preserves weights, it must map any relation included in $\precsim^{\text{TR}}$ onto a relation with the same property, but $b+c \not\precsim^{\text{TR}} b$ and $b+c \not\precsim^{\text{TR}} c$.

However a weaker form of completeness holds: if $P \precsim Q$ then there is a weight-preserving function $\mathcal{F}$ and a compliant relation $\mathcal{R}$ with $\mu.P \;\mathcal{R}\; \mu.Q$, for some $\mu$, and with $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ (we can use for this the constant function that maps any relation onto the trace preorder). We can then derive $P \precsim Q$ from $\mu.P \precsim \mu.Q$ using the cancellation law for prefixes

$$\text{`}\mu.P \precsim \mu.Q \text{ implies } P \precsim Q\text{'}$$

The law is valid in all preorders of Section 3; in fact, it is valid in any preorder whose grammar for observables includes the diamond operator (see on this also the Appendix).

## 6 INSTANCES OF ENHANCEMENT

We test the usability of the weight-based condition for enhancements (Definition 5.5). We show that a number of useful functions preserve weights. For this we also introduce forms of functional composition that preserve weights, with the idea of producing complex sound functions from simpler ones. The first-order and higher-order functions considered in this section are inspired by the most common ones used in coinductive enhancements. Below we omit the family $O$ when clear, e.g., abbreviating $\precsim^{O}$ as $\precsim$.

*Basic functions and constructors.* A function that takes first-order functions as arguments and yields back another first-order function as a result is a *second-order function* or, briefly, a *constructor*. A constructor *preserves weights* if whenever its first-order function arguments preserve weights on $\precsim$, then also the first-order function result preserves weights on $\precsim$, for any $\precsim$.

We write $\mathcal{I}$ for the identity function, and $\mathcal{X}$ for the reflexive function, mapping any relation onto the purely reflexive relation whose elements are all pairs of the form $(P, P)$. For a given preorder $\precsim$, we write $\mathcal{U}_{\precsim}$ for the constant-to-$\precsim$ function, mapping every relation onto the preorder $\precsim$ itself.

Lemma 6.1 (identity and constant-to-$\precsim$ functions). *The identity function $\mathcal{I}$, the reflexive function $\mathcal{X}$, and the constant-to-$\precsim$ function $\mathcal{U}_{\precsim}$ preserve weights on $\precsim$, for any $\precsim$.*

The basic constructors we consider are *composition* ($\circ$), *union* ($\cup$) and chaining ($\frown$), so defined:

$$(\mathcal{G}\circ\mathcal{F})(\mathcal{R}) \overset{\text{def}}{=} \mathcal{G}(\mathcal{F}(\mathcal{R})) \qquad (\bigcup_{i\in I}\mathcal{F}_i)(\mathcal{R}) \overset{\text{def}}{=} \bigcup_{i\in I}(\mathcal{F}_i(\mathcal{R}))$$
$$(\mathcal{G}^\frown\mathcal{F})(\mathcal{R}) \overset{\text{def}}{=} \mathcal{G}(\mathcal{R})\,\mathcal{F}(\mathcal{R}) = \{(P,P') \mid \exists P''.\,(P,P'') \in \mathcal{G}(\mathcal{R}),\,(P'',P') \in \mathcal{F}(\mathcal{R})\}$$

(Formally, for arity reasons, there is a different union operator for all $n \in \{0, 1, \ldots, \omega\}$.)

Lemma 6.2 (composition, union, chaining). *The composition, union and chaining constructors preserve weights.*

*Derived functions.* We derive now more complex functions, exploiting the earlier basic functions and constructors. Then we examine the context-closure function.

Corollary 6.3. *The following functions preserve weights on $\precsim$, for any $\precsim$:*

$$\mathcal{D}_n \overset{def}{=} \mathcal{I}^\frown\ldots{}^\frown\mathcal{I}, \quad n \text{ times } (n > 0) \qquad \mathcal{P}_{\precsim} \overset{def}{=} \mathcal{U}_{\precsim}{}^\frown\mathcal{I}^\frown\mathcal{U}_{\precsim}$$
$$\mathcal{T} \overset{def}{=} \bigcup_{n>0}\mathcal{D}_n \qquad \mathcal{B} \overset{def}{=} \mathcal{X}\cup\mathcal{T}$$

On a relation $\mathcal{R}$: function $\mathcal{D}_n$ makes the composition of $\mathcal{R}$ with itself $n$ times; function $\mathcal{P}_{\precsim}$ is the 'preorder counterpart' of the 'bisimulation up to bisimilarity' enhancement [Milner 1989], as $(P, Q) \in \mathcal{P}_{\precsim}(\mathcal{R})$ if there are $P'$ and $Q'$ with $P \precsim P'\ \mathcal{R}\ Q' \precsim Q$; function $\mathcal{T}$ returns the transitive closure of $\mathcal{R}$; function $\mathcal{B}$ returns the preorder induced by $\mathcal{R}$ (the reflexive and transitive closure of $\mathcal{R}$).

The composition and union constructors are particularly useful. Composition allows us to freely combine a set of weight-preserving functions, and union allows us to use different combinations of the functions in different parts of a proof. We formalise this via the closure construction below, which we shall use later in examples.

Let $\{\mathcal{F}_1, \ldots, \mathcal{F}_m\}$ be a set of functions. It is convenient to assume that the identity $\mathcal{I}$ is in the set, otherwise we can add it. Abbreviating $\{\mathcal{F}_1, \ldots, \mathcal{F}_m\}$ as $\widetilde{\mathcal{F}}$, the *n-th closure of a relation $\mathcal{R}$ with respect to $\widetilde{\mathcal{F}}$*, written $\mathcal{L}_n^{\widetilde{\mathcal{F}}}(\mathcal{R})$, is inductively defined as follows:

$$\mathcal{L}_1^{\widetilde{\mathcal{F}}}(\mathcal{R}) \overset{\text{def}}{=} \mathcal{I}(\mathcal{R}) = \mathcal{R} \qquad \mathcal{L}_{n+1}^{\widetilde{\mathcal{F}}}(\mathcal{R}) \overset{\text{def}}{=} \bigcup_i \mathcal{F}_i(\mathcal{L}_n^{\widetilde{\mathcal{F}}}(\mathcal{R}))$$

As $\mathcal{I} \in \widetilde{\mathcal{F}}$, we have $\mathcal{L}_n^{\widetilde{\mathcal{F}}}(\mathcal{R}) \subseteq \mathcal{L}_{n+1}^{\widetilde{\mathcal{F}}}(\mathcal{R})$, for any $n$. Then the *closure* $\mathcal{L}^{\widetilde{\mathcal{F}}}(\mathcal{R})$ *with respect to* $\widetilde{\mathcal{F}}$ is $\mathcal{L}^{\widetilde{\mathcal{F}}}(\mathcal{R}) \overset{\text{def}}{=} \bigcup_n \mathcal{L}_n^{\widetilde{\mathcal{F}}}(\mathcal{R})$. Thus a pair of processes is in the closure $\mathcal{L}^{\widetilde{\mathcal{F}}}(\mathcal{R})$ if the pair can be obtained starting from $\mathcal{R}$ and then applying the functions in $\widetilde{\mathcal{F}}$ an arbitrary number of times, using composition and union.

Corollary 6.4. *If functions $\widetilde{\mathcal{F}}$ preserve weights on $\precsim$, then so does the closure function $\mathcal{L}^{\widetilde{\mathcal{F}}}$.*

*Context closure.* We now introduce a function that gives us the context-closure of a relation. For this, we have to assume — as usual in process algebras — that the class of processes is defined as the term algebra generated by some signature. A *context* is a an element of the term algebra with at most one occurrence of the hole $[\cdot]$ in it, taking $[\cdot]$ as a further operator of the signature with arity 0.

*Definition 6.5.* If $\mathscr{C}$ is a set of contexts, then the *$\mathscr{C}$-closure function* is the function $C_{\mathscr{C}}$ that takes a relation and returns the closure of that relation with respect to the contexts in $\mathscr{C}$:

$$C_{\mathscr{C}}(\mathcal{R}) \overset{\text{def}}{=} \{(C[P], C[Q]) \mid P \mathrel{\mathcal{R}} Q \text{ and } C \text{ is a context in } \mathscr{C}\}$$

We simply write $C$ when $\mathscr{C}$ is the set of all contexts, and call it the *context-closure function.*

It is sufficient to use contexts with at most one occurrence of a unique hole. More sophisticated closures, involving *n-ary contexts*, i.e., contexts that may contain different holes, to be filled with possibly different processes, can be recovered with a combination of the function $C_{\mathscr{C}}$ and the transitive-closure function as $\mathcal{T} \circ C_{\mathscr{C}}$.

We present an abstract property on contexts that can be used to show that the corresponding function $C_{\mathscr{C}}$ preserves weights.

*Definition 6.6.* A context $C$ is *measurably faithful for (a family of measurable observables) $O$* if for all $n, P$ and $\theta \in O$ there is $\zeta$ such that $C[P] \models^n \theta$ implies $P \models^n \zeta$, and moreover for all $Q$ with $Q \models \zeta$ also $C[Q] \models \theta$ holds.

Theorem 6.7. *If all contexts in $\mathscr{C}$ are measurably faithful for $O$, then function $C_{\mathscr{C}}$ preserves weights on $\precsim^{O}$.*

The proof is straightforward, following the definitions. Below we discuss a CCS language, showing that all its contexts are measurably faithful for the observables required by all the preorders discussed in Section 3.

*Context closure in CCS$^!$.* We have so far developed a theory of inductive enhancements based on functions that are weight-preserving and, earlier in this section, we have presented a number of functions that are captured by such a theory. Among them we have discussed the context-closure function, introducing an abstract condition on contexts, measurable faithfulness, that guarantees soundness. Below we show an example of how such a property can be proved. For this, however, we have to pick a specific language and family of observables. Thus the language is essentially CCS and the observables are any family used for the preorders examined in Section 3.

We assume a (countably) infinite set $\{a, b, \dots, \}$ of *names*, which does not contain the special symbol $\tau$. The class of the CCS$^!$ processes is built from the operators of input prefix, output prefix, silent prefix, parallel composition, countable sum, restriction, and replication:

$$\begin{array}{lcl} P & := & \mu.P \;\big|\; P_1 \mid P_2 \;\big|\; \sum_{i \in I} P_i \;\big|\; \boldsymbol{\nu} a\, P \;\big|\; !P \\ \\ \mu & := & a \;\big|\; \overline{a} \;\big|\; \tau\,. \end{array}$$

We sometimes use the binary sum $P_1 + P_2$; and write $\mathbf{0}$ for the sum in which the indexing set is empty. We have omitted the relabeling operator (it would not however bring complications into the theory), and included replication, which will be used in some examples —- hence the name CCS$^!$. (Other operators could be added to the process language, see Remark 6.10.) The operational semantics of the calculus is reported in the Appendix.

Lemma 6.8. *In CCS$^!$, all contexts are measurably faithful for $O$, for all the families of observables $O$ described in Section 3 (i.e., from those yielding trace preorder to those yielding ready trace preorder).*

Corollary 6.9. *In CCS$^!$, the context-closure function $C$ (Definition 6.5) preserves weights on $\precsim^O$, for all the families of observables $O$ described in Section 3.*

We can thus combine the 'up-to context' function $C$ with other weight-preserving functions, using the constructors introduced earlier in this section. For instance we thus obtain the soundness of the 'up-to $n$-ary context' function $C^{\mathcal{T}} \overset{\text{def}}{=} \mathcal{T} \circ C$ or, for any preorder $\precsim$, and of the 'up-to $\precsim$ and context' function $C^{\precsim} \overset{\text{def}}{=} \mathcal{P}_{\precsim} \circ C = \mathcal{U}_{\precsim} \frown C \frown \mathcal{U}_{\precsim}$. That is,

$$C^{\precsim}(\mathcal{R}) \quad \overset{\text{def}}{=} \quad \{(P,Q) \ \mid \ \text{there are } P', Q', C \text{ with } P \precsim C[P'], C[Q'] \precsim Q \text{ and } P' \mathcal{R} Q'\}$$

(The function can be made more powerful using $\mathcal{T} \circ C_{\mathscr{C}}$. in place of $C$.) Function $C^{\precsim}$ is the counterpart of the common 'up-to context and bisimilarity' enhancement of the coinductive setting, with a preorder in place of an equivalence.

Remark 6.10. Corollary 6.9 can be strengthened to show that the set of contexts produced by the operators of CCS$^!$ preserves weights in any extension of the language. That is, consider a process language whose contexts include the set $\mathscr{C}$ described by the following grammar:

$$C \quad := \quad \mu.C \;\Big|\; C \mid P \;\Big|\; P \mid C \;\Big|\; C + P \;\Big|\; P + C \;\Big|\; \nu a\, C \;\Big|\; !C$$

where $P$ is any process of the language and where, for simplicity, we use only binary sums. Then function $C_{\mathscr{C}}$ preserves weights. In some of the examples we shall indeed employ extensions of CCS$^!$, for instance including a recursion operator.

## 7 EQUIVALENCES

The equivalence induced by a preorder $\precsim$ is $\precsim \cap \succsim$. We write $\sim^O$ for the equivalence induced by the preorder $\precsim^O$; we simply write $\sim$ when non ambiguous. All definitions and results in previous sections can be straightforwardly extended to equivalences, simply adding the 'converse direction'. Below are the main results needed for obtaining the enhancement techniques. In particular, we move from *semi-progressions* to *progressions*. We write $\mathcal{R}^{-1}$ for the inverse of a relation $\mathcal{R}$.

*Definition 7.1 (progression).* Given two relations $\mathcal{R}$ and $\mathcal{S}$, we say that $\mathcal{R}$ *progresses to* $\mathcal{S}$, written $\mathcal{R} \rightarrowtail \mathcal{S}$, if both $\mathcal{R} \rightarrowtail \mathcal{S}$ and $\mathcal{R}^{-1} \rightarrowtail \mathcal{S}^{-1}$.

*Definition 7.2 (soundness, for equivalences).* A function $\mathcal{F}$ is *sound for* $\sim^O$ if $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ implies $\mathcal{R} \subseteq \sim^O$.

In the definition of soundness, now compliance is not mentioned as implied by the use of progressions in place of semi-progressions.

Theorem 7.3 (soundness of weight-preserving functions for equivalences). *Suppose $\mathcal{F}$ preserves weights on $\precsim^O$. Then $\mathcal{F}$ is sound for $\sim^O$.*

In Theorem 7.3, we need not have the same $\mathcal{R}$ or the same function to prove the two directions of the progression:

Theorem 7.4. *Suppose $\mathcal{F}$ and $\mathcal{G}$ preserve weights on $\precsim^O$. If $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ and $\mathcal{S} \rightarrowtail \mathcal{G}(\mathcal{S})$ then $\mathcal{R} \cap \mathcal{S}^{-1} \subseteq \sim^O$.*

The function and constructors examined in Section 6 for preorders and semi-progressions may thus be used to reason about equivalences.

## 8 EXAMPLE 1: CONGRUENCE FOR RECURSION

A congruence proof that is notoriously delicate in concurrency concerns the recursion operator. In this section we show that the enhancement techniques can relieve such a proof. Precisely we exhibit a congruence proof that is *parametric* on the process language and the preorder $\precsim$. Besides some mild assumptions, we only require $\precsim$ to be measurable (Definition 4.4). Below we sketch the elements for the proof, referring to the Appendix for more details.

The language includes a recursion operator $\mathtt{rec}\ X.\, P$, which, as usual, represents a binding occurrence for $X$ in the body $P$. The main assumptions on language and preorder are: the presence of an action prefix operator $\mu.\, P$, akin to CCS prefix, so to be able to say that an expression $E$ is *guarded in* $G$ if all occurrences of $E$ in $G$ are underneath a prefix; the variable $X$ of a recursion $\mathtt{rec}\ X.\, E$ is guarded in the body $E$; a cancellation property for prefixes, that is, $\mu.\, P \precsim \mu.\, Q$ implies $P \precsim Q$. (These are basic and standard assumptions, see also the discussion in the Appendix.)

We use $X, Y$ as recursion variables, and write $\mathsf{fv}(G)$ for the free recursion variables of an expression $G$. We let $E, F, G$ range over open expressions (i.e., expressions that may contain free occurrences of the recursion variables), and $P, Q$ over the closed ones. A preorder $\precsim$ is extended to open expressions $E, F$, with $\mathsf{fv}(E, F) \subseteq \{\widetilde{X}\}$, by stipulating that $E \precsim F$ if $E\{\widetilde{P}/\widetilde{X}\} \precsim F\{\widetilde{P}/\widetilde{X}\}$ for all closed processes $\widetilde{P}$.

Theorem 8.1 (congruence for recursion). *Assume $X$ guarded in $E$ and $F$. If $E \precsim F$, then $\mathtt{rec}\ X.\, E \precsim \mathtt{rec}\ X.\, F$.*

Proof. [Sketch] Following the definition of $\precsim$ on open expressions, it is sufficient to carry out the proof in the case when $X$ is the only possible free variable of $E, F$. Let $P \overset{\text{def}}{=} \mathtt{rec}\ X.\, E$ and $Q \overset{\text{def}}{=} \mathtt{rec}\ X.\, F$. Take

$$\mathcal{R} \overset{\text{def}}{=} \{(G\{P/Y\}, G\{Q/Y\}) \mid Y \text{ guarded in } G\}$$

(processes in $\mathcal{R}$ are closed, thus $G$ has at most $Y$ as a free recursion variable). If we show $\mathcal{R} \subseteq \precsim$, then we are done: we have $\mu.\, P\ \mathcal{R}\ \mu.\, Q$, for any $\mu$, and then the result in the theorem follows from the cancellation property on prefixes. Relation $\mathcal{R}$ is compliant; we prove the semi-progression $\mathcal{R} \rightarrowtail \mathcal{P}_{\precsim}(\mathcal{R})$ (that is, $\mathcal{R} \rightarrowtail \precsim\mathcal{R}\precsim$). For this, we consider a transition $G\{P/Y\} \xrightarrow{\mu} P'$ and show that we can find $G^{\#}$ guarded such that

$$P' \precsim G^{\#}\{P/Y\} \qquad \text{and} \qquad G\{Q/Y\} \xrightarrow{\mu} \succsim G^{\#}\{Q/Y\}$$

We first write $P'$ as $G^{*}\{P/Y\}$, for some $G^{*}$, and on the unguarded occurrences of $Y$ in $G^{*}$ we unfold the substitution $\{P/Y\}$ as $\{(E\{P/X\})/Y\}$. We proceed similarly on $G\{Q/Y\}$ and, in addition, we replace all unguarded occurrences of $F$ so produced with $E$, using the hypothesis of the theorem. We extract $G^{\#}$ from the final expression.

The result can be transported to the equivalence induced by the preorder, exploiting the result for the preorder.

## 9 EXAMPLE 2: TWO SERVERS

We use the enhancement techniques to related two servers, and some variants of them. The results are valid for all preorders of Sections 3 and their induced equivalences; the last two systems may however be related only by the preorders. None of the results are valid for coinductive behavioral relations such as bisimilarity (not even for similarity). Thus in this section $\precsim$ is any of the preorders of Sections 3, and $\sim$ its induced equivalence.

As a language, we use the value-passing version of the CCS language in Section 6. While the value-passing language could be translated into pure CCS [Milner 1989], having explicit values improves readability (adapting the results in this paper to value-passing CCS anyhow is simple). In a value-passing calculus, $a(x).\,P$ is an input at $a$ in which $x$ is the placeholder for the value received, whereas $\overline{a}\langle n\rangle.\,P$ is an output at $a$ of the value $n$.

We wish to implement a server that may be interrogated by clients at a channel $c$, so to start a certain interaction protocol with the client; for this however the server must also consult two auxiliary servers $A$ and $B$, respectively at $a$ and $b$. The auxiliary servers $A$ and $B$ are nondeterministic, and always return, respectively, an arbitrary integer and arbitrary boolean (we write $\mathtt{t}$ and $\mathtt{f}$ for the two boolean values):

$$A \overset{\text{def}}{=} !\,\Sigma_{n\in N}\ \overline{a}\langle n\rangle \qquad\qquad B \overset{\text{def}}{=} !\,(\overline{b}\langle \mathtt{t}\rangle + \overline{b}\langle \mathtt{f}\rangle)$$

We consider two implementations, $L$ and $M$, of the server. The difference between them is that the order in which the two auxiliary servers are interrogated is swapped. Both servers are represented using a replication operator; a channel $e$ is used to activate a new copy of the body of the replication.

$$\begin{array}{rcl} L & \overset{\text{def}}{=} & !e.\,b(y).\,c(z).\,a(x).\,(\overline{e}\mid R(c,x,y,z)) \\ M & \overset{\text{def}}{=} & !e.\,a(x).\,c(z).\,b(y).\,(\overline{e}\mid R(c,x,y,z)) \end{array}$$

Here $R(c,x,y,z)$ represents the interaction protocol that is started with a client, and can be any process. It may use, and depend on, the values $z, x$ and $y$ (obtained from the client and the auxiliary servers). Process $R(c,x,y,z)$ may also use channel $c$, and therefore trigger further interactions with the server; in contrast, $R(c,x,y,z)$ may not use $a$ and $b$ to interrogate the auxiliary servers. We wish to relate the composition of the two servers with the auxiliary servers $A$ and $B$ :

$$LS \overset{\text{def}}{=} (\boldsymbol{\nu} a,b)(A \mid B \mid L) \qquad\qquad MS \overset{\text{def}}{=} (\boldsymbol{\nu} a,b)(A \mid B \mid M)$$

We prove $LS \sim MS$ using the enhancements (where $\sim$ is the equivalence induced by any of the preorders in Section 3). The relation $\mathcal{R}$ we use for this is a singleton, and has just one pair, namely $(LS, MS)$. The relation is compliant; we show the progression $\mathcal{R} \rightarrowtail C^{\precsim}(\mathcal{R})$ (i.e., $\mathcal{R} \rightarrowtail {\precsim} C(\mathcal{R}) {\precsim}$), so to conclude $LS \sim MS$ by Theorem 7.3. For this, we consider the semi-progression $\mathcal{R} \rightarrowtail C^{\precsim}(\mathcal{R})$, the converse direction being similar. The only immediate transitions of the two processes are:

$$\begin{array}{rcll} LS & \xrightarrow{e} & (\boldsymbol{\nu} a,b)(A \mid B \mid L_1 \mid L) & \overset{\text{def}}{=} LS_1 \\ MS & \xrightarrow{e} & (\boldsymbol{\nu} a,b)(A \mid B \mid M_1 \mid M) & \overset{\text{def}}{=} MS_1 \end{array}$$

where

$$\begin{array}{rcl} L_1 & \overset{\text{def}}{=} & b(y).\,c(z).\,a(x).\,(\overline{e}\mid R(c,x,y,z)) \\ M_1 & \overset{\text{def}}{=} & a(x).\,c(z).\,b(y).\,(\overline{e}\mid R(c,x,y,z)) \end{array}$$

We now apply some simple and common laws, valid even for bisimilarity (the expansion law, laws for shrinking the scope of a restriction, and for duplicating a replication) so to explicitly internalise the interactions of $L_1$ and $M_1$ with the auxiliary servers and obtain:

$$LS_1 \sim L_2 \mid LS \quad \text{ for } \quad L_2 \overset{\text{def}}{=} \Sigma_{v\in\mathtt{Bool}}\tau.\,c(z).\,\Sigma_{n\in N}\tau.\,(\overline{e}\mid R(c,n,v,z))$$

Now, we use twice a basic law for the inductive preorders [Glabbeek 2001], namely

$$\lambda.\,\Sigma_i\ \mu.\,P_i \sim \Sigma_i\ \lambda.\,\mu.\,P_i$$

This law is valid for all the equivalences induced by the inductive preorders of Section 3, whereas it fails for coinductive relations such as similarity or bisimilarity. We thus derive

$$L_2 \quad\sim\quad \Sigma_{v\in\mathtt{Bool}}\Sigma_{n\in N}\tau.\,c(z).\,\tau.\,(\overline{e}\mid R(c,n,v,z)) \quad \overset{\text{def}}{=} \quad L_3$$

We can proceed similarly on $\mathit{MS}$. Thus, also rearranging the order of the summands, we have:

$$\begin{array}{cccc} \mathit{LS} & \xrightarrow{e} & \sim & L_3 \mid \mathit{LS} \\ \mathit{MS} & \xrightarrow{e} & \sim & L_3 \mid \mathit{MS} \end{array}$$

This closes the proof, up to $\precsim$ and context, since $\mathit{LS}\ \mathcal{R}\ \mathit{MS}$ and $\sim$ implies both $\precsim$ and its inverse.

The above results cannot be proved purely algebraically using standard axiom systems for the preorders or equivalences, because the systems compared are not finite or finite state (axiomatisations are complete only on these systems, and do not contain, for instance, axioms for inverting the order of prefixes in a process). One could proceed reasoning by induction on, e.g., the length of the involved traces; the proof would be tedious, for instance because $R(c,x,y,z)$ can be any process.

The result does not hold under bisimilarity, which is too fine a relation. The reason is that, intuitively, $\mathit{LS}_1$ can produce an interaction at $b$ involving the auxiliary server $B$ that makes a commitment to using a specific boolean in the communication protocol with the next client. In contrast, $\mathit{MS}_1$ makes an initial commitment on the *integer* to be used, in an interaction involving the other auxiliary server $A$. Thus the transition from $\mathit{LS}_1$ cannot be matched by $\mathit{MS}_1$, and conversely. (Indeed the two systems may not even be related by similarity.)

Consider now a variant of the first server, $L$, that throws away the boolean received at $b$ and always uses the boolean $\mathtt{t}$:

$$L \stackrel{\text{def}}{=} !e.\,b(y).\,c(z).\,a(x).\,(\overline{e}\ |R(c,x,\mathtt{t},z))$$

In this case $\mathit{LS} \sim \mathit{MS}$ does not hold. We can however take the preorder $\precsim$ and prove $\mathit{LS} \precsim \mathit{MS}$, using the semi-progression $\mathcal{R} \rightarrowtail C^{\precsim}(\mathcal{R})$. The main difference with the proof above is that we also use the law $\tau.\,P \precsim \tau.\,P + \tau.\,Q$ (again, this law is valid for all preorders discussed in Section 3). Intuitively, in a preorder, a deterministic system is 'below' a non-deterministic one. This kind of reasoning cannot be made for bisimilarity, which, as discussed in Section 1, does not have a natural associated preorder.

## 10 EXAMPLE 3: HENNESSY'S PROOF SYSTEM FOR THE TRACE PREORDER

In this section we use the theory of enhanced induction to prove the soundness of Hennessy's proof system for the trace preorder [Hennessy 2017]. The motivation for the example is twofold: it allows us to explain the peculiarities of Hennessy's proof (his proof system is a non-standard one, in the realm of behavioural preorders) using the theory of enhancements; it involves a combination of several functional enhancements.

The structure of Hennessy's proof system follows Salomaa's [Salomaa 1966] for trace equivalence (called language equivalence in [Salomaa 1966], trace equivalence being a more common terminology for behavioural equivalences); related works are also Kozen [Kozen 1994] and Rabinovich [Rabinovich 1993]. The most important difference in Hennessy's system is that Salomaa's key rule, *unique fixed-point induction* (Ufi), is replaced by rule CoRec

$$\text{CoRec}\ \frac{\Gamma, a.\,P_1 \leq a.\,P_2 \vdash P_1 \leq P_2}{\Gamma \vdash a.\,P_1 \leq a.\,P_2} \qquad (*)$$

(the rule has a coinductive flavour because the conclusion of the rule is one of its hypotheses, though the proof system is inductive). Indeed, as Hennessy deals with a preorder rather than an equivalence, it is unclear what could be a complete induction principle to be used in place of Ufi. The new 'coinductive' rule, however, makes the soundness of Hennessy system delicate; see [Hennessy 2017, section 5] for a discussion. The language is the following subset of CCS:

$$E := \mathbf{0} \ \big|\ \mu.\,E \ \big|\ E_1 + E_2 \ \big|\ X \ \big|\ \mathtt{rec}\ X.\,E$$

The variable $X$ of a recursion $\texttt{rec}\ X.E$ is guarded in the body $E$. We maintain the notations for earlier process languages, e.g., in Section 8, and let $E, F, G$ range over open expressions, and $P, Q, R$ over the closed ones. In this section, following Salomaa and others [Kozen 1994; Rabinovich 1993; Salomaa 1966] we take the ordinary trace preorder. Hennessy works in a weak semantics, which abstracts from $\tau$ actions; the proof for weak semantics is discussed in Section 14.

In the proof system, judgments are of the form $\Gamma \vdash P \leq P'$ , where $P, P'$ are closed terms, and $\Gamma$ is a finite set of assumptions of the form $P \leq P'$. The whole proof system is reported in the Appendix. Example of rules are, besides CoRec above, the following ones, where Tr is transitivity, Pl is the structural rule for sum, Hyp and W deal with the assumptions:

$$\textsc{Tr}\frac{\Gamma \vdash P_1 \leq P_2 \qquad \Gamma \vdash P_2 \leq P_3}{\Gamma \vdash P_1 \leq P_3} \quad \textsc{Pl}\frac{\Gamma \vdash P \leq P'}{\Gamma \vdash P + Q \leq P' + Q}$$

$$\textsc{Hyp}\frac{}{P \leq P' \vdash P \leq P'} \quad \textsc{W}\frac{\Gamma' \vdash P \leq P' \qquad \Gamma' \subseteq \Gamma}{\Gamma \vdash P \leq P'}$$

Hennessy's soundness proof makes use of a semantic interpretation of the judgments $\Gamma \vdash P \leq Q$ . Using $\precsim^{\mathrm{TR}}$ for the trace preorder, the obvious choice would be to interpret a judgment

$$P_1 \leq P_1', \ldots, P_m \leq P_m' \vdash P \leq P' \tag{1}$$

as valid whenever $P_i \precsim^{\mathrm{TR}} P_i'$ for each $i \in \{1, \ldots, m\}$ implies $P \precsim^{\mathrm{TR}} P'$. It turns out that this is unsound [Hennessy 2017]. Therefore Hennessy uses a *stratified* semantic interpretation inspired by the soundness proof of Brandt and Henglein's proof system for recursive types [Brandt and Henglein 1998]. Recasted in our notation, the semantic interpretation says that a judgment (1) is valid if, for all $n$, $P_i \precsim_n^{\mathrm{TR}} P_i'$ for each $i \in \{1, \ldots, m\}$ implies $P \precsim_n^{\mathrm{TR}} P'$. In the soundness proof each proof rule is shown to preserve the stratified semantics. The unsoundness of the non-stratified interpretation of judgments recalls the unsoundness of the definition of weight-preserving in Remark 5.8. Similarly, the stratification and the implication in the validity assertion recall those in the definition of weight-preserving functions (Definition 5.5 ).

Indeed, reformulated in our setting, each proof rule corresponds to a function that preserves weights. Such functions include: the transitive-closure function $\mathcal{T}$, the identity function $\mathcal{I}$, the context-closure function $C$, the reflexive function $\mathcal{X}$, the constant-to-$\precsim^{\mathrm{TR}}$ function $\mathcal{U}_{\precsim^{\mathrm{TR}}}$. Call $\widetilde{\mathcal{F}}$ the set of all such functions. Then the crux of the proof consists in showing that any given proof tree $\delta$ yields a relation $\mathcal{R}_\delta$ on processes that semi-progresses to the closure $\mathcal{L}^{\widetilde{\mathcal{F}}}(\mathcal{R})$. That is, we use the semi-progression $\mathcal{R}_\delta \rightarrowtail \mathcal{L}^{\widetilde{\mathcal{F}}}(\mathcal{R}_\delta)$, which, by Corollary 6.4, preserves weights.

More precisely, $\mathcal{R}_\delta$ is the set all pairs $(a.P_1, a.P_2)$ for which there is a CoRec node in $\delta$ of the form $(*)$ above, for some $\Gamma$. For the semi-progression to hold, we then have to show that for any such $(a.P_1, a.P_2) \in \mathcal{R}_\delta$ we have $(P_1, P_2) \in \mathcal{L}^{\widetilde{\mathcal{F}}}(\mathcal{R}_\delta)$. We derive this from the following property:

$$\text{if } \Delta \vdash P \leq Q \text{ is a node in } \delta \text{ then } (P, Q) \text{ is in the closure } \mathcal{L}^{\widetilde{\mathcal{F}}}(\mathcal{R}_\delta) \text{ of } \mathcal{R}_\delta.$$

The proof proceeds by induction of the depth of $\delta$, using a case analysis on the rule whose conclusion is the root of $\delta$. As an example, if the last rule is Tr, then the node is of the form

$$\textsc{Tr}\ \frac{\Delta \vdash P \leq R \qquad \Delta \vdash R \leq Q}{\Delta \vdash P \leq Q}$$

By induction, $(P, R)$ and $(R, Q)$ are in the closure $\mathcal{L}^{\widetilde{\mathcal{F}}}(\mathcal{R})$. Hence also $(P, Q)$ is in the closure, as the transitive-closure function is in $\widetilde{\mathcal{F}}$. Similarly, if the rule is Pl, the node is of the form

$$\textsc{Pl}\ \frac{\Delta \vdash P \leq Q}{\Delta \vdash P + R \leq Q + R}$$

By induction, $(P, Q)$ is in the closure $\mathcal{L}^{\widetilde{\mathcal{F}}}(\mathcal{R})$; hence, also $(P + R, Q + R)$, as the context-closure function is in $\widetilde{\mathcal{F}}$.

## 11 WEAK SEMANTICS

By *weak* semantics we refer to behavioural relations that distinguish between internal process activities and visible actions. In this section and up to Section 14 we discuss the extension of the theory presented to behavioural relations obtained from weak inductive observables.

### 11.1 Differences between Strong and Weak Semantics

There are two major differences with respect to the theory of strong semantics in earlier sections. The first difference concerns weights and relational progressions. We begin by examining weights for weak observables similar to those used for the strong observables. However, as some useful enhancements are not captured, we then also consider more sophisticated weights, namely composite weights that separate the contributions of internal actions from that of the external ones. This in turn yields progressions in which different functions may be employed according to whether the action performed is internal or external.

The other main difference is that the observables used in weak semantics may refer to state predicates such as *stability*, that did not appear in the observables for the strong semantics. For instance, all the preorders discussed in Section 3 except trace equivalence mention ready or refusal sets of processes. In weak semantics, the observations of such sets are only made on stable processes.

The second of the two differences above is specific to the inductive setting, as with bisimulation state predicates like stability are usually ignored. In contrast, the first difference is related to the soundness problems of weak coinductive enhancements — due to which theories of weak coinductive enhancements tend to be rather more involved than the 'strong' theories. Some of the technical solutions that we adopt for the inductive setting are inspired from those used in the coinductive setting, others are specific to the weight-based conditions of the inductive setting.

### 11.2 Weak Observables

The set of actions *Act* now contains a special element $\tau$ that represents internal activities in a process. As before, $\mu$ ranges over *Act*. We now use $\ell$ to range over $Act - \{\tau\}$, the *visible* (or *external*) actions. Some standard notations for weak transitions: $\Longrightarrow$ is the reflexive and transitive closure of $\xrightarrow{\tau}$, and $\overset{\mu}{\Longrightarrow}$ is $\Longrightarrow\xrightarrow{\mu}\Longrightarrow$ (the composition of the three relations). Moreover, $P \xrightarrow{\widehat{\mu}} P'$ holds if $P \xrightarrow{\mu} P'$ or ($\mu = \tau$ and $P = P'$); similarly $P \overset{\widehat{\mu}}{\Longrightarrow} P'$ holds if $P \overset{\mu}{\Longrightarrow} P'$ or ($\mu = \tau$ and $P = P'$). We also use weighted transitions, writing $P \Longrightarrow_n P'$ if $P$ may evolve into $P'$ by performing $n$ $\tau$-transitions, and $P \overset{\mu}{\Longrightarrow}_n P'$ if $P \Longrightarrow_m\xrightarrow{\mu}\Longrightarrow_s P'$, for some $m, s$ with $m + s + 1 = n$; similarly for $P \overset{\widehat{\mu}}{\Longrightarrow}_n P'$.

A process $P$ is *stable* if $P \overset{\tau}{\not\longrightarrow}$, i.e., $P$ cannot perform internal actions. In the grammars for observables, the weak diamond $\langle\!\langle \mu \rangle\!\rangle. \theta$, insensitive to $\tau$-transitions, replaces the strong diamond $\langle \mu \rangle. \theta$.

*Definition 11.1.* A *family of weak observables (over Act and $\mathcal{A}$)* is a subset $O$ of the formulas inductively generated by the following grammar:

$$\theta \quad := \quad \langle\!\langle \mu \rangle\!\rangle. \theta \;\Big|\; \textstyle\bigwedge_{j\in J} \theta_j \;\Big|\; \mathsf{a}$$

where, as before, $\mu \in Act$, $\mathsf{a} \in \mathcal{A}$, $J$ is a countable set, and:

- (downward-closure) if $\langle\!\langle \mu \rangle\!\rangle. \theta \in O$ then also $\theta \in O$, and similarly if $\bigwedge_{j\in J} \theta_j \in O$ then also $\theta_j \in O$ for each $j$; if $\langle\!\langle \mu \rangle\!\rangle. \theta \in O$ then also $\langle\!\langle \tau \rangle\!\rangle. \theta \in O$.

With respect to the strong case, the additional requirement $\langle\!\langle \tau \rangle\!\rangle.\,\theta \in O$ is needed in the soundness proofs; intuitively it is part of the downward-closure property because a process with an observable $\langle\!\langle \mu \rangle\!\rangle.\,\theta$ will also have the observable $\langle\!\langle \mu \rangle\!\rangle.\,\langle\!\langle \tau \rangle\!\rangle.\,\theta$.

Satisfaction is defined as in the strong case; the only difference is in the weak diamond where abstraction from internal transition is expressed.

$$\textsc{w-act}\ \frac{P \overset{\widehat{\mu}}{\Longrightarrow} P' \qquad P' \models \theta}{P \models \langle\!\langle \mu \rangle\!\rangle.\,\theta} \qquad \textsc{w-conj}\ \frac{P \models \theta_j \qquad \text{for all } j \in J}{P \models \bigwedge_{j\in J} \theta_j} \qquad \textsc{w-at}\ \frac{P \in [\![\mathsf{a}]\!]}{P \models \mathsf{a}}$$

In w-act, when $\mu = \tau$ process $P$ need not move, hence we may have $P' = P$.

In the strong case, to derive the preorders of Section 3 we used the atomic observable $\mathtt{ref}_\mu$ and the 'barb' observable $\langle \mu \rangle.\,\mathtt{true}$. Their weak counterpart, $\mathtt{Ref}_\mu$ and $\mathtt{Acc}_\mu$, are both atomic since employed on stable processes:

$$P \in \mathtt{Ref}_\ell \ \text{ if } (P \text{ stable and } P \overset{\ell}{\not\longrightarrow}) \qquad\qquad P \in \mathtt{Acc}_\ell \ \text{ if } (P \text{ stable and } P \overset{\ell}{\longrightarrow})$$

More generally, as in the strong case, the atomic observables should be *local*: they can be checked by looking only at the immediate outgoing transitions emanating from a state. The preorder induced by a family $O$ of weak observables is written $\precapprox^{O}$.

When the atomic observables refer to stability, the preorder may distinguish between stable and non-stable processes such as $\tau.\,a$ and $a$ (however $\mu.\,\tau.\,P$ and $\mu.\,P$ are always equated). Such distinctions are often found in the literature (the resulting preorders are sometimes called *rooted*), and are needed to guarantee substitutivity of the preorder on operators such as summation.

Remark 11.2 (Non-local atomic observables and divergence). Non-local atomic observables can be accommodated in the theory, but care would be required in the context-closure function, which may become troublesome. In the literature for weak semantics, a non-local observable that is sometimes found is *divergence* (a process is *divergent* if it can perform an infinite sequente of $\tau$-transitions). There are many ways in which divergence is taken into account in the literature. For instance, in failure equivalence sometimes divergence is taken as *chaotic* [Brookes et al. 1984], in that a divergent process is supposed to have all possible failures; similarly, in ready semantics, when testing processes, sometimes divergence is taken to yield a separate outcome (besides the ordinary success and failure outcomes). Indeed, a number of variations are possible, see e.g., discussions in [Bergstra et al. 1987; Glabbeek 1993]. Most important, divergence is a coinductive predicate, whereas the theory developed in this paper deals with observables that are inductively defined.

A comprehensive treatment of divergence, capable of covering its various uses in inductive equivalences in the literature, or perhaps pinpointing which instances can be handled, or are easier to handle, requires a dedicated study, related to the extension of the theory with coinductive observables, which we leave for future work (Section 16).

## 11.3 Weak-preorder Instances

On LTSs that do not contain divergences, the weak versions of the strong preorders of Section 3 (e.g., trace, failure, failure trace, ready, ready trace preorders) are obtained by abstracting, in the definition of traces, from the silent transitions performed (e.g., an arbitrary number of silent transitions may be performed between two visible actions), and by imposing that refusal and ready sets are only observed on *stable* states. And, as in the strong case, the characterisation results mentioned (e.g., failure versus must testing) require image-finite LTSs. Similarly, in the formulas for observables, the main difference is the replacement of the strong diamond $\langle \mu \rangle.\,\theta$ with the weak diamond $\langle\!\langle \mu \rangle\!\rangle.\,\theta$, and the use of the stable-sensitive (atomic) observables $\mathtt{Acc}_\mu$ and $\mathtt{Ref}_\mu$.

## 11.4 Weak Weights

As in strong semantics, so in weak semantics $P \models^n \theta$ means that the observable $\theta$ can be checked by exploring the transitions emanating from $P$ up to a depth at most $n$. The rules for weak weighed satisfaction are:

$$\text{W-AT+}\ \frac{P \in [\![\mathsf{a}]\!] \qquad 1 \leq n}{P \models^n \mathsf{a}} \qquad\qquad \text{W-ACT+}\ \frac{P \overset{\widehat{\mu}}{\Longrightarrow}_s P' \qquad P' \models^m \theta \qquad s + m \leq n}{P \models^n \langle\!\langle \mu \rangle\!\rangle.\, \theta}$$

$$\text{W-CONJ+}\ \frac{P \models^{m_j} \theta_j \qquad m_j \leq n \qquad \text{for all } j \in J}{P \models^n \bigwedge_{j \in J} \theta_j}$$

The definitions and results here are the expected modifications of those in the strong case. Proof details may be found in the Appendix

*Definition 11.3.* A family of weak observables $O$ and the corresponding preorder $\precapprox^O$ are *measurable* if, for any LTS (over the same set of actions) and process $P$ of the LTS, and for any observable $\theta \in O$, whenever $P \models \theta$ then also $P \models^n \theta$, for some $n$.

All preorders of Section 11.3 are measurable.

*Definition 11.4 (weak stratification).* Suppose $\precapprox^O$ is the preorder induced by a family $O$ of weak observables. We write $P \precapprox^O_n Q$ if $P \models^n \theta$ implies $Q \models \theta$, for all $\theta \in O$.

The relations $\precapprox^O_n$ need not be transitive. For instance, in the trace preorder we have $a \precapprox^{\mathrm{TR}}_1 \tau.\, a$ and $\tau.\, a \precapprox^{\mathrm{TR}}_1 b$ but not $a \precapprox^{\mathrm{TR}}_1 b$.

Lemma 11.5. *If $O$ is measurable, then $\precapprox^O = \cap_n \precapprox^O_n$.*

*Definition 11.6 (weak semi-progression).* Given two relations $\mathcal{R}$ and $\mathcal{S}$, we say that $\mathcal{R}$ *weakly semi-progresses to* $\mathcal{S}$, written $\mathcal{R} \rangle\!\rangle\!\rightharpoonup \mathcal{S}$, if $P \mathrel{\mathcal{R}} Q$ implies:

- whenever $P \xrightarrow{\mu} P'$, there exists $Q'$ such that $Q \overset{\widehat{\mu}}{\Longrightarrow} Q'$ and $P' \mathrel{\mathcal{S}} Q'$.

We recall that $\mathcal{R}$ is *compliant (with $O$)* if for all $P \mathrel{\mathcal{R}} Q$ and $\mathsf{a} \in O$, $P \in [\![\mathsf{a}]\!]$ implies $Q \in [\![\mathsf{a}]\!]$.

*Definition 11.7 (soundness).* A function $\mathcal{F}$ is *sound for* $\precapprox^O$ if $\mathcal{R} \rangle\!\rangle\!\rightharpoonup \mathcal{F}(\mathcal{R})$ implies $\mathcal{R} \subseteq \precapprox^O$, for any compliant $\mathcal{R}$.

*Definition 11.8 (weak-weight-preserving function).* A function $\mathcal{F}$ *preserves weak weights on* $\precapprox^O$ when, for all $n$ and for all $\mathcal{R}$, if $\mathcal{R} \subseteq \precapprox^O_n$, then also $\mathcal{F}(\mathcal{R}) \subseteq \precapprox^O_n$. Function $\mathcal{F}$ *preserves weak weights* if it preserves weak weights on $\precapprox^O$, for any $\precapprox^O$.

Theorem 11.9. *If $O$ is measurable and $\mathcal{F}$ preserves weak weights on $\precapprox^O$, then $\mathcal{F}$ is sound for $\precapprox^O$.*

## 11.5 Functions and Constructors that Preserve Weak Weights

All the functions and constructors from the strong case (Section 6) behave well also in the weak case, with the exception of chaining. In the strong case, the main usefulness of chaining was in deriving the soundness of the up-to-preorder function. In Section 11.6 we study ways of replacing this function. Whenever non ambiguous, we abbreviate $\precapprox^O$ as $\precapprox$.

### 11.5.1 *Functions.*

The notion of 'weak-weight preserving', for functions and constructors, and related results, are transported to the weak case in the expected manner. For instance, we have:

Lemma 11.10. *The composition and union constructors preserve weak weights.*

The only result from Section 6 that does not carry over concerns the chaining constructor and all functions derived from it. We show that chaining does not preserve weak weights on the up-to-preorder function $\mathcal{P}_{\precapprox}$, that maps a relation $\mathcal{R}$ onto

$$\{(P,Q) \mid P \precapprox P', Q' \precapprox Q \text{ and } P' \mathcal{R} Q' \text{ for some } P', Q'\}$$

This function is obtained (as in the strong case) by chaining the identity and constant-to-$\precapprox$ functions (which preserve weights). When $\precapprox$ is the trace preorder and $\mathcal{R} \overset{\text{def}}{=} \{(\tau.\, a, \mathbf{0})\}$, we have $\mathcal{R} \subseteq \precapprox_1$ (the only observable for $\tau.\, a$ in $\precapprox_1$ is TRUE). However, as $a \precapprox \tau.\, a$ and $\precapprox$ is reflexive, we also have $a\ \mathcal{P}_{\precapprox}(\mathcal{R})\ \mathbf{0}$, which shows $\mathcal{P}_{\precapprox}(\mathcal{R}) \not\subseteq \precapprox_1$ because $a \models^1 \langle\!\langle a \rangle\!\rangle.\,\text{true}$ whereas not $\mathbf{0} \models^1 \langle\!\langle a \rangle\!\rangle.\,\text{true}$. It would be indeed unsound to use $\mathcal{P}_{\precapprox}$ in progressions. For this, we can use a similar counterexample: take $\mathcal{R} \overset{\text{def}}{=} \{(\tau.\, a, \mathbf{0})\}$; then $\mathcal{R} \rightarrowtail \mathcal{P}_{\precapprox}(\mathcal{R})$, yet $\tau.\, a$ and $\mathbf{0}$ are not in the trace preorder. (The same $\mathcal{R}$ is used in the literature to show the unsoundness of "weak bisimulation up-to weak bisimilarity" [Sangiorgi and Milner 1992], though the reasoning is different.)

## 11.6 Contraction Preorders

We introduce some useful functions that, while more restrictive than the (unsound) $\mathcal{P}_{\precapprox}$ above, may often be used in places where $\mathcal{P}_{\precapprox}$ would be needed.

In the coinductive case a useful auxiliary preorder that is used for similar purposes is the *expansion preorder* [Sangiorgi and Milner 1992], a refinement of which is the *contraction preorder* [Sangiorgi 2017]. We import the idea into the inductive setting.

*Definition 11.11.* For any $O$, the *contraction preorder* $\leqq^{O}$ is defined thus:

- $P \leqq^{O} Q$ if $P \models^{n} \theta$ implies $Q \models^{n} \theta$, for all $\theta \in O$.

Thus, while in the ordinary preorder $\precapprox^{O}$ the answer from $Q$ uses $Q \models \theta$, here we find $Q \models^{n} \theta$; that is, the observable for $Q$ must be at least as efficient as that for $P$. Thus $\leqq^{O} \subseteq \precapprox^{O}$. Still, $\leqq$ is coarser than the strong preorder $\precsim$; for instance only the former may relate $a.\, \tau.\, b$ and $a.\, b$.

We recall that $\mathcal{U}_{\mathcal{R}}$ is the constant function mapping all relations onto $\mathcal{R}$, for any relation $\mathcal{R}$, that $\mathcal{I}$ is the identity function, and that $\frown$ is the chaining constructor (Section 6).

Lemma 11.12. *Suppose $\mathcal{F}$ preserves weak weights on $\precapprox$. Then also $\mathcal{U}_{\leqq} \frown \mathcal{F}$ and $\mathcal{F} \frown \mathcal{U}_{\precapprox}$ preserve weak weights on $\precapprox$.*

We write $\mathcal{P}_{\leqq\precapprox}$ for the function mapping a relation $\mathcal{R}$ onto

$$\{(P,Q) \mid \text{ there are } P', Q' \text{ and } P \leqq P', P' \mathcal{R} Q', Q' \precapprox Q\}$$

Lemma 11.13. *Function $\mathcal{P}_{\leqq\precapprox}$ preserves weak weights on $\precapprox$.*

We derive Lemma 11.13 from Lemmas 11.12 and 11.10, as $\mathcal{P}_{\leqq\precapprox}$ is obtained by composition of the functions $\mathcal{U}_{\leqq} \frown \mathcal{I}$ and $\mathcal{I} \frown \mathcal{U}_{\precapprox}$.

## 11.7 Context Closure

The analogous of the results in Section 6 holds for weak weights; i.e., the up-to context function preserves weak weights for all families of observables in Section 11.3. As in the strong case, the key property for contexts is *measure faithfulness* (Definition 6.6, the same definition applies to the weak case). The proofs are similar to those for the strong case; some care however is needed with stability.

Theorem 11.14. *In the language CCS$^{!}$, function $C$ preserves weak weights on $\precapprox^{O}$, for all the families of weak observables $O$ in Section 11.3.*

We can thus combine the context-closure function with other functions that preserve weak weights. For instance, combining $\mathcal{P}_{\leqq \precsim}$ (Lemma 11.13) with $C$ we derive the function $\mathcal{P}_{\leqq \precsim} \circ C$ whereby

$$(\mathcal{P}_{\leqq \precsim} \circ C)(\mathcal{R}) \quad \overset{\text{def}}{=} \quad \{(P, Q) \ \mid \ \text{there are } P', Q', C \text{ with } P \leqq C[P'], C[Q'] \precsim Q \text{ and } P' \ \mathcal{R} \ Q'\}$$

We will use this function in some of the examples.

## 12 COMPOSITE WEIGHTS AND COMPOSITE SEMI-PROGRESSIONS

In this section we examine a different form of weights for the weak enhancements. We replace weak weights with *composite weights* and *visible weights*. The former are the important ones, the latter are derived from the former. The reason for the two kinds of weights is that we can then define semi-progressions with two functions, one to be used after a silent-action challenge, the other after a visible-action challenge. Two functions yield more flexibility and allow us to obtain more powerful enhancements.

### 12.1 Composite and Visible Weights

Composite weights are pairs $(n, m)$ of integers. We use the lexicographical order to compare composite weights, in which $(n, m) < (n', m')$ if $m < m'$ or ($m = m'$ and $n < n'$). Addition and subtraction on composite weights are computed componentwise.

Roughly, $P \models^{(n,m)} \theta$ says that $P \models \theta$ has a proof in which the sequences of transitions from $P$ that are unrolled have at most $n$ silent transitions and $m$ visible transitions. More precisely, a proof of $P \models^{(n,m)} \theta$ explores a finite tree of transitions with $P$ as a root; and if $n'$ and $m'$ are the number of silent and visible actions in a path of the tree, then $(n', m') \leq (n, m)$. We require however a bound on the number of silent actions in any path of the tree; as a consequence, the meaning of '$O$ measurable' is the same, under weak or composite weights. We use $u, v, w$ to range over composite weights.

Visible weights are written $\widehat{m}$, where $m$ is an integer. We define satisfaction under visible weights on top of satisfaction under composite weights, writing $P \models^{\widehat{m}} \theta$ if $P \models^{(n,m)} \theta$, for some $n$ (defining $P \models^{\widehat{m}} \theta$ in terms of $P \models^{(n,m)} \theta$, rather than with dedicated rules, is convenient in proofs in which the two forms of weights are intertwined, notably the proof of Theorem 12.4). Thus, roughly $P \models^{\widehat{m}} \theta$ implies that the observable $\theta$ can be checked on $P$ by exploring a tree of transitions in whose paths there are at most $m$ visible actions.

Concerning the rules that assign composite weights to weak observables, the stratifications of a preorder $\precsim^{O}$ for composite and visible weights, respectively written $\precsim^{O}_{u}$ and $\precsim^{O}_{\widehat{m}}$, the meaning of functions that preserve composite and visible weights: these notions are defined by modifying those for weak weights in Section 11 as expected.

Lemma 12.1. *For any $\precsim^{O}$, if a function preserves composite weights, then it also preserves weak weights and visible weights.*

We only present here the soundness of semi-progressions using two functions, one to be used after a silent-action challenge, the other after a visible-action challenge, and similarly for semi-progressions on stable relations, in which only functions preserving visible weights are involved.

*Definition 12.2 (semi-progression, with two relations).* Given three relations $\mathcal{R}$ and $\mathcal{S}, \mathcal{T}$, we say that $\mathcal{R}$ *semi-progresses to* $\langle \mathcal{S}, \mathcal{T} \rangle$, written $\mathcal{R} \rangle\!\rangle\!\!\rightharpoonup \langle \mathcal{S}, \mathcal{T} \rangle$, if $P \ \mathcal{R} \ Q$ implies:

(1) whenever $P \xrightarrow{\tau} P'$, there exists $Q'$ such that $Q \Longrightarrow Q'$ and $P' \ \mathcal{S} \ Q'$;
(2) whenever $P \xrightarrow{\ell} P'$, there exists $Q'$ such that $Q \overset{\ell}{\Longrightarrow} Q'$ and $P' \ \mathcal{T} \ Q'$.

*Definition 12.3 (soundness, with a pair of functions).* A pair of functions $\langle \mathcal{F}, \mathcal{G} \rangle$ is *sound for* $\precapprox^{O}$ if $\mathcal{R} \rangle\!\rangle\!\rightharpoonup \langle \mathcal{F}(\mathcal{R}), \mathcal{G}(\mathcal{R}) \rangle$ implies $\mathcal{R} \subseteq \precapprox^{O}$, for any compliant $\mathcal{R}$.

Theorem 12.4. *If $O$ is measurable, $\mathcal{F}$ preserves composite weights on $\precapprox^{O}$ and $\mathcal{G}$ preserves visible weights on $\precapprox^{O}$, then $\langle \mathcal{F}, \mathcal{G} \rangle$ is sound for $\precapprox^{O}$.*

See the Appendix for a discussion on the hypothesis of Theorem 12.4.

A relation is *stable* if all the processes in the pairs of the relation are stable. If $\mathcal{R}$ is stable, then in a semi-progression $\mathcal{R} \rangle\!\rangle\!\rightharpoonup \langle \mathcal{F}(\mathcal{R}), \mathcal{G}(\mathcal{R}) \rangle$, the relation $\mathcal{F}(\mathcal{R})$ is irrelevant. In this case we therefore write the semi-progression as $\mathcal{R} \rangle\!\rangle\!\rightharpoonup \langle -, \mathcal{G}(\mathcal{R}) \rangle$

*Definition 12.5 (soundness on stable relations).* A function $\mathcal{G}$ is *sound for* $\precapprox^{O}$ *under stability* if, whenever $\mathcal{R}$ is stable and compliant, then $\mathcal{R} \rangle\!\rangle\!\rightharpoonup \langle -, \mathcal{G}(\mathcal{R}) \rangle$ implies $\mathcal{R} \subseteq \precapprox^{O}$.

Corollary 12.6 (soundness of functions that preserve visible weights under stability). *If $O$ is measurable, and $\mathcal{G}$ preserves visible weights on $\precapprox^{O}$, then $\mathcal{G}$ is sound for $\precapprox^{O}$ under stability.*

Corollary 12.6 only refers to *visible* weights (i.e., natural numbers) and to a *single* weight-preserving function. It is unclear, however, how to derive the result without going through the *composite* weights and the *two* kinds of functions employed in Theorem 12.4.

## 12.2 Instances for Composite and Visible Weights

We revisit the functions and constructors examined in Section 11.5 for weak weights, this time with respect to the property of preserving composite and visible weigths. We only mention the two results that differ; everything else is maintained, including the results about contractions.

- The chaining constructor preserves visible weights (we saw that it does not preserve weak weights).
- The (full) context-closure function of CCS$^{!}$ does not preserve composite and visible weights.

The reason for the failure of the context-closure function is that a context may transform a visible action of a process into a silent action. This interferes with the definition of composite and visible weights, in which visible and silent actions are separated. We show below that, on certain subsets of contexts, the closure may be recovered.

Intuitively, a context is *light for a process P* if the context and the process will never interact. For instance, the following grammar trivially defines light CCS contexts, as a process in the hole will never run with components of the context in parallel:

$$C := [\cdot] \;\big|\; \mu.C \;\big|\; C + R \tag{2}$$

where $R$ is any process. We write $\mathcal{C}_{\mathrm{L}}$ for the context closure of a relation with respect to such light contexts:

$$\mathcal{C}_{\mathrm{L}}(\mathcal{R}) \overset{\mathrm{def}}{=} \{(C[P], C[Q]) \;\mid\; P \;\mathcal{R}\; Q \text{ and } C \text{ is a context of grammar (2)}\}$$

The light contexts defined by the grammar above are sufficient for illustrating the concept and for the examples in this paper. It is possible to define richer classes of light contexts, for instance requiring that the free names of a context do not appear as free names of the process that fills its hole.

Theorem 12.7. *For any $\precapprox^{O}$, the light-context function $\mathcal{C}_{\mathrm{L}}$ preserves composite weights and visible weights.*

## 13 WEAK SEMANTICS: A SUMMARY

For weak semantics we have defined two main forms of enhancements, corresponding to semi-progressions of the form $\mathcal{R} \rightarrowtail\!\!\rightharpoonup \mathcal{F}(\mathcal{R})$ and $\mathcal{R} \rightarrowtail\!\!\rightharpoonup \langle \mathcal{F}(\mathcal{R}), \mathcal{G}(\mathcal{R})\rangle$ (a third form, namely, $\mathcal{R} \rightarrowtail\!\!\rightharpoonup \langle -, \mathcal{G}(\mathcal{R})\rangle$, is a special case of the latter). We regard the semi-progressions $\mathcal{R} \rightarrowtail\!\!\rightharpoonup \mathcal{F}(\mathcal{R})$, which are the simplest, as the most important ones. The main problem for them is that chaining, and hence the up-to-preorder enhancement, become unsound. A limited form of up-to-preorder, using contractions, is however sound (Section 11.6).

Whenever chaining and the full up-to-preorder enhancement are needed, one may use the 'composite' semi-progressions $\mathcal{R} \rightarrowtail\!\!\rightharpoonup \langle \mathcal{F}(\mathcal{R}), \mathcal{G}(\mathcal{R})\rangle$. Here, however, the context-closure function may not be used (though we do not know if it is unsound); one may however use the closure under light contexts. The final example of the next section shows an application of this form of semi-progression (actually $\mathcal{R} \rightarrowtail\!\!\rightharpoonup \langle -, \mathcal{G}(\mathcal{R})\rangle$ is sufficient), where we take full advantage of chaining.

## 14 WEAK SEMANTICS: THE EXAMPLES, REVISITED

*Congruence for recursion.* The schema of the proof of congruence for recursion in Section 8 can be adapted to weak semantics. For this, however, we need an advanced form of enhancement, involving contractions.

As before we assume that the syntax of the language includes an action prefix operator $\mu. P$ and a recursion operator $\mathtt{rec}\ X. E$ in which $X$ is guarded in the body $E$. The guard need not be a visible prefix – it may also be $\tau$. Thus, here $\precapprox$ is any measurable weak preorder, as defined in Section 11.4, and $\leqq$ its corresponding contraction preorder. The assumptions we need are similar to those for the strong case: **Guard** is actually the same; in **Cong** and **Canc** we simply replace the strong preorder $\precsim$ with the weak one $\precapprox$.

Theorem 14.1. *Assume $X$ guarded in $E$ and $F$. If $E \precapprox F$, then* $\mathtt{rec}\ X. E \precapprox \mathtt{rec}\ X. F$.

The proof is similar to the strong case. The relation to be used, for $P \overset{\text{def}}{=} \mathtt{rec}\ X. E$ and $Q \overset{\text{def}}{=} \mathtt{rec}\ X. F$, is

$$\mathcal{R} \overset{\text{def}}{=} \{(G\{P/Y\}, G\{Q/Y\}) \mid Y \text{ guarded in } G\}$$

We derive $\mathcal{R} \subseteq \precapprox$ using the function $\mathcal{P}_{\leqq\precapprox}$ (Lemma 11.13, the 'up-to $\leqq$ and $\precapprox$' technique). We show that a transition $G\{P/Y\} \xrightarrow{\mu} G^*\{P/Y\}$ is matched by $G\{Q/Y\} \xrightarrow{\mu} G^*\{Q/Y\}$ by finding a $G^\#$ guarded such that $G^*\{P/Y\} \leqq G^\#\{P/Y\}$ and $G^*\{Q/Y\} \succapprox G^\#\{Q/Y\}$, using the hypothesis of the theorem, assumptions **Guard**, **Cong**, and **Canc**, and the inclusion $\precsim \subseteq \leqq$ when folding or unfolding the recursions.

*Two servers.* We revisit the example of the two servers, in Section 9. The same results and proofs in that section can be repeated under weak semantics. However a weak semantics allows us some greater flexibility. Consider for instance an eager variant of the first server, $L$, that interrogates *both* auxiliary servers $B$ and $A$, at $b$ and $a$, before accepting a request from a client at $c$:

$$L^{\mathsf{E}} \quad \overset{\text{def}}{=} \quad !e. b(y). a(x). c(z). (\overline{e} \mid R(c, x, y, z))$$

and let $LS^{\mathsf{E}}$ be the system defined as $LS$ but with $L^{\mathsf{E}}$ in place of $L$. In $LS^{\mathsf{E}}$ a visible action at $c$ is only possible after *two* $\tau$ actions (the two interrogations with the auxiliary servers), whereas in $MS$ only *one* $\tau$ action is produced. Hence $LS^{\mathsf{E}}$ and $MS$ may not be related by a strong equivalence. Let $\precapprox$ be any of the weak preorders in Section 11.3, $\leqq$ its corresponding contraction, and $\approx$ the equivalence induced by $\precapprox$. (The systems have no divergences, assuming the protocol $R(c, x, y, z)$

does not produce divergences, hence the characterisation results in Section 11.3 hold). We can prove $LS^{\mathsf{E}} \approx MS$ using a weak progression $\mathcal{R} \rangle\!\rangle\!\!\rightarrowtail (\mathcal{P}_{\leqq \underset{\approx}{\precsim}} \circ \mathcal{C})(\mathcal{R})$ (i.e., $\mathcal{R} \rangle\!\rangle\!\!\rightarrowtail \leqq \mathcal{C}(\mathcal{R}) \underset{\approx}{\precsim}$, Section 11.7).

The proof is similar to those in Section 9; $\mathcal{R}$ is the singleton $\{(LS^{\mathsf{E}}, MS)\}$. The only addition is the application of the law $\mu.\,\tau.\,P = \mu.\,P$. Precisely, we need $\mu.\,\tau.\,P \leqq \mu.\,P$ and $\mu.\,\tau.\,P \underset{\approx}{\succsim} \mu.\,P$ (these laws are valid for all the contractions and preorders discussed).

*Hennessy's proof system for trace preorder.* Moving to *weak* trace preorder, the only difference in Hennessy's proof system discussed in Section 10 is the addition of the pairs of axioms $\tau.\,P \leq P$ and $P \leq \tau.\,P$. The soundness of the proof system therefore requires the (straightforward) soundness proof of these two extra axioms.

An important technical remark, however, is mandatory. In the strong case our proof employed a sophisticated semi-progression, involving the closure with respect to various functions, including up-to preorder, up-to context, up-to transitivity. Some of these functions are derived from the chaining constructor. In the weak case, chaining is not sound for weak weights; it is only sound for the visible weights of Section 12. We therefore appeal to enhancement functions that preserve visible weights and to the stability condition of Corollary 12.6. We can do so because the relation on which the semi-progression is applied has, as in the strong case, only pairs of processes of the form $(a.\,P, a.\,Q)$, which are stable, and because the contexts that are needed in the closure are the light contexts of Section 12.2. In other words, the semi-progression employed uses the weak version of the closure function in the proof for strong semantics in Section 10, tailored to functions that preserve visible weights.

## 15 COMPARISONS AND RELATED WORK

We are not aware of theories of enhancements for 'inductive behavioural relations' (i.e., relations defined from inductive observables). In this respect, all the material in the paper, beginning at Section 2, is new. We have been inspired by theories of enhancements for coinductive behavioural equivalences; these are however technically quite different, because they are not based on weights and weight-preserving functions (see the discussion below). A key ingredient for our enhancements are inductive stratifications, or approximations, of behavioural relations. This is a well-know tool in concurrency and more generally in semantics (e.g., [Hennessy 1988; Milner 1989; Roscoe 1998]); our use of stratification, to define weights and then the weight-preserving functions, in turn used to derive the inductive enhancements, is novel.

*Coinductive enhancements.* In the coinductive case, the condition on functions used to yield sound enhancements is *respectfulness* or *compatibility* [Pous and Sangiorgi 2012]. In both cases, the key requirement is the lifting of a progression $\mathcal{R} \rightarrowtail \mathcal{S}$ to $\mathcal{F}(\mathcal{R}) \rightarrowtail \mathcal{F}(\mathcal{S})$, where $\mathcal{F}$ is the function. For compatibility, this is the only condition; respectfulness has the additional hypothesis $\mathcal{R} \subseteq \mathcal{S}$ and then requires $\mathcal{F}(\mathcal{R}) \subseteq \mathcal{F}(\mathcal{S})$. These conditions are the coinductive counterpart of our weight-preserving condition (Definition 5.5). The remarks below on respectfulness also hold for compatibility.

As sets of functions, the weight-preserving and the respectful ones are incomparable. This already shows up in strong semantics. An easy way to obtain a respectful function that is not weight-preserving for a given inductive equivalence exploits languages in which bisimilarity is a congruence whereas the inductive equivalence is not. As an example for trace equivalence, consider the language of processes

$$P := a.\,P \;\big|\; P_1 + P_2 \;\big|\; \mathbf{0} \;\big|\; f(P)$$

where the SOS rule for the unary operator $f$ is:

$$\frac{P \xrightarrow{a} P'}{f(P) \xrightarrow{a} f(P')} a \neq b \qquad \frac{P \xrightarrow{b} P' \qquad P \xrightarrow{c} P''}{f(P) \xrightarrow{b} P'}$$

The up-to context function for this language is respectful, yet it is not weight-preserving for trace equivalence (indeed trace equivalence is not preserved by the operator $f(-)$ above). Similar counterexamples can be exhibited for other inductive equivalences.

This may appear puzzling at first, because respectfulness ensures soundness for bisimilarity and, as bisimilarity is more demanding than any of the inductive relations in the paper, soundness for bisimilarity implies soundness for the inductive relations. What the above counterexample rather shows is that adding the set of respectful functions to the set of weight-preserving functions would yield a set of functions without good closure properties. For instance the composition of two functions might not be sound. We show a counterexample in the language above, using the up-to context function $C$ and the constant-to-$\sim$ function $\mathcal{U}_\sim$ ($C$ is respectful and $\mathcal{U}_\sim$ is weight-preserving). Consider $P \overset{\text{def}}{=} a.\, f(a.\,(b+c))$ and $Q \overset{\text{def}}{=} a.\, f(a.\, b + a.\, c))$, and let $\sim$ be trace equivalence. Then $P \sim Q$ does not hold, as only the former processes has the trace $a; a; b$. However, if $\mathcal{R} \overset{\text{def}}{=} \{P, Q\}$ then $\mathcal{R} \rightarrowtail C(\mathcal{U}_\sim(\mathcal{R})) = C(\sim)$, as

$$P \xrightarrow{a} f(a.\,(b+c)) \overset{\text{def}}{=} P' \qquad\qquad Q \xrightarrow{a} f(a.\, b + a.\, c)) \overset{\text{def}}{=} Q'$$

and $(P', Q') \in C(\sim)$.

On the other hand, the 'up-to context' function $C$ of the language CCS$^!$ in Section 6 is not respectful (though it is contained in a larger respectful function, for instance, the function $\mathcal{T} \circ C$ that yields 'up-to polyadic contexts') while, as shown in that Section 6, $C$ is weight-preserving for all families of observables there discussed. Moreover, for any equivalence $\sim$ induced by a measurable preorder, function $\mathcal{P}_\sim$ (the 'up to $\sim$' in Corollary 6.3, for an equivalence in place of a preorder) is not sound for any equivalence finer than $\sim$; hence, in particular, $\mathcal{P}_\sim$ is not sound for bisimilarity (let alone respectful). None of the enhancements used in Sections 8-10 and 14 corresponds to a respectful function.

*Unique solutions of equations.* In [Durier et al. 2019; Sangiorgi 2017] the authors study proof techniques for weak bisimilarity based on *unique solutions of equations* and special inequations called *contractions*. These techniques give one the power of some bisimulation enhancements such as 'up-to context'. Moreover, the authors transport them onto trace equivalence and trace preorder. Such techniques seem to have a limited applicability to preorders (they can only be used to show that a given process is related to the syntactic solution of an equation, that is, the process whose syntactic definition is the equation itself), and adapting them to a new equivalence seems to require considerable work. Milner [Milner 1989] presents unique solution of equations as a proof technique alternative to the bisimulation proof method. In this sense, we may say that [Durier et al. 2019; Sangiorgi 2017] follow the former style of proof technique, whereas in this paper we follow the latter.

*Further related work.* The idea of developing a theory of enhancements, for the bisimulation proof method, is put forward in [Sangiorgi 1998], introducing the notion of progression and of respectful function. Later [Pous 2007, 2016] the theory has been both generalised to the coinduction proof method, using complete lattices, and refined, first replacing respectfulness with compatibility (simpler though more restrictive), then focusing on the largest compatible function, called the *companion* (that also coincides with the largest respectful function). The companion can also be obtained using Kleene's construction of the greatest fixed-point [Parrow and Weber 2016].

Abstract formulations of the meaning of coinductive enhancements have also been given using category theory. The main technical tools are final semantics, coalgebras, spans of coalgebra homomorphisms, fibrations, and corecursion schemes. See, e.g., [Basold et al. 2017; Pous and Rot 2017; Rot et al. 2017] (based on earlier works such as [Bartels 2004; Jacobs 2006; Lenisa et al. 2000; Milius et al. 2013; Uustalu et al. 2001]), and [Bartels 2003; Bonchi et al. 2018b, 2017b; Lenisa 1999]. Enhancements of corecursion schemes may also be examined using the generalised powerset construction [Silva et al. 2010]. For the more details on coinductive enhancements, we refer to the technical survey [Pous and Sangiorgi 2012] and to the historical review [Pous and Sangiorgi 2019].

Bisimilarity (and similar coinductive relations) has been employed to reason about inductive relations by relying on transformations of the transitions systems that modify the nondeterminism and the set of states, in such a way that a given equivalence on the original systems corresponds to bisimilarity on the altered systems. For instance, trace equivalence on nondeterministic processes can be reduced to bisimilarity on deterministic processes, following the powerset construction for automata [Hopcroft et al. 2006]; a similar reduction can be made for testing equivalence [Cleaveland and Hennessy 1993]. In [de Frutos-Escrig and Gregorio-Rodríguez 2009], related processes may be rewritten using a (possibly inductive) equivalence so to characterise, using a bisimulation-like game, the equivalence itself, and similarly for preorders; the main goal is to exploit such a homogeneous presentation to derive axiomatisations for a broad spectrum of behavioural relations.

The bisimulation enhancements have been shown to be useful not only for Labeled Transition Systems. For example, efficient algorithms for equivalence (and inclusion) of automata and streams have been obtained, e.g., [Bonchi et al. 2017a; Bonchi and Pous 2013; Rot et al. 2016]. In the powerset construction from non-deterministic to deterministic automata, up-to techniques are employed to avoid the construction of the whole automata. The algorithms exploit the property that language equivalence coincides with bisimilarity on deterministic systems. Up-to techniques have also been proposed for streams to facilitate coinductive definitions in Coq, e.g., [Endrullis et al. 2013].

Proofs of soundness for the coinductive enhancements have been related to proofs of completeness for domains of abstract interpretations [Bonchi et al. 2018a].

## 16 CONCLUSIONS AND FUTURE WORK

In this paper we have studied how to transport the well-known enhancements of the proof method for coinductively-defined behavioural relations, notably bisimilarity, onto inductive behaviour relations, i.e., behaviour relations defined from inductive observables. We have formalised the observables by means of operators from modal logics, so to capture the most common inductive behavioural relations in the literature. Following the coinductive setting, our theory makes use of semi-progressions of the form $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ where $\mathcal{F}$ is a function on relations. The functions *sound* for a given preorder are those for which $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ implies that $\mathcal{R}$ is contained in the preorder; and similarly for equivalences. We have introduced *weights* on the observables, and a *weight-preserving* condition on functions that guarantees soundness. We have shown that the class of weight-preserving functions contains non-trivial functions and that it enjoys closure properties with respect to desirable function constructors, so to be able to derive sophisticated sound functions (and hence sophisticated proof techniques) from simpler ones. For weak semantics we have considered two forms of weights. (We have summarised the differences between the theories for strong and weak semantics in Section 11.1, and those between the two forms of weights in Section 13.) We have shown our enhancements at work on a few non-trivial examples, both in the strong and in the weak case: the examples about recursion illustrate the possibility of deriving proofs that are parametric on the behavioural relation of interest; the proofs about the servers show our techniques at work in cases where the bisimulation enhancements may not be used because bisimilarity is too fine or does not have a natural associated preorder; the examples about

Hennessy's proof system use the enhancements to explain Hennessy's non-standard proof, and bring up a sophisticated combination of enhancements.

We would like to see if and how the theory of inductive enhancements in the paper can be formulated in a more abstract setting, e.g., fixed-point theory or category theory. Proposals along these lines exist for the coinductive enhancements (c.f., references in Section 15). Their meaning in our setting is unclear, both because the observables are inductive and because of the coinductive flavour of the semi-progressions at the heart of the theory. For this, one may consider variant formulation of the weight-preserving condition. For instance (as pointed out by one of the anonymous referees), requiring in Definition 5.5 the function $\mathcal{F}$ to be monotone and with $\mathcal{F}(\precsim_n^{O}) \subseteq \precsim_n^{O}$, for all $n$. We would also to see if the theory presented can be lifted to a probabilistic setting and labelled Markov processes.

A powerful enhancement is up-to context. We have studied it in a CCS-like language. We would like to see if there are general conditions that guarantee its soundness, more precisely the weight-preserving property. Such conditions could, for instance, look at the format of the rules defining the operational behaviour of the operators of the language. In coinduction, up-to-context has been shown very effective in *higher-order languages*, such as $\lambda$-calculi or languages enriched with functional features, and *nominal languages* such as the $\pi$-calculus. More generally, here the goal would be transferring the inductive enhancements in the paper to these classes of languages.

The up-to-context enhancement could also be used to derive general congruence properties for inductive behavioural relations; e.g., for CCS$^!$ one exploits Corollary 6.9.

As enhancement functions, we have considered functions inspired by the most common ones used in coinductive enhancements. This suggests another direction to investigate, namely the search of other useful functions that preserve weights.

We have studied enhancements for behavioural relations defined from inductive observables. We have not considered observables that are naturally defined coinductively, such as infinite traces and divergence. It would be interesting to see if the theory developed can be extended so to accommodate both inductive and coinductive observables. This would be particularly relevant for divergence in weak semantics (c.f., Remark 11.2).

## ACKNOWLEDGMENTS

I benefitted from exchanges and comments from Matthew Hennessy. I would also like to thank Rob van Glabbeek, for discussions about inductive equivalences and for suggesting Example 1 about recursion, and the anonymous referees, for various useful comments.

This material is based upon work supported by the MIUR-PRIN project 'Analysis of Program Analyses' (ASPRA, ID 201784YSZ5_004).

## REFERENCES

Jos C. M. Baeten, Jan A. Bergstra, and Jan Willem Klop. 1987. Ready-Trace Semantics for Concrete Process Algebra with the Priority Operator. *Comput. J.* 30, 6 (1987), 498–506.

F. Bartels. 2003. Generalised Coinduction. *Math. Struct. in Computer Science* 13, 2 (2003), 321–348. https://doi.org/10.1017/S0960129502003900

F. Bartels. 2004. *On generalised coinduction and probabilistic specification formats.* Ph. D. Dissertation. CWI, Amsterdam.

Henning Basold, Damien Pous, and Jurriaan Rot. 2017. Monoidal company for accessible functors. In *Proc. CALCO (LIPIcs, Vol. 72)*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik. https://doi.org/10.4230/LIPIcs.CALCO.2017.5

J.A. Bergstra, J.W. Klop, and E.-R. Olderog. 1987. Failures without Chaos: a Process Semantics for Fair Abstraction. In *IFIP Formal Description of Programming Concepts – III*, M. Wirsing (Ed.). Elsevier Science Publishers B.V., 77–101.

B. Bloom, S. Istrail, and A.R. Meyer. 1995. Bisimulation can't be Traced. *Journal of the ACM* 42, 1 (1995), 232–268. https://doi.org/10.1145/200836.200876

Filippo Bonchi, Pierre Ganty, Roberto Giacobazzi, and Dusko Pavlovic. 2018a. Sound up-to techniques and Complete abstract domains. In *LICS 2018*, Anuj Dawar and Erich Grädel (Eds.). ACM, 175–184. https://doi.org/10.1145/3209108.3209169

Filippo Bonchi, Barbara König, and Sebastian Küpper. 2017a. Up-To Techniques for Weighted Systems. In *TACAS 2017 (Lecture Notes in Computer Science, Vol. 10205)*, Axel Legay and Tiziana Margaria (Eds.). 535–552. https://doi.org/10.1007/978-3-662-54577-5_31

Filippo Bonchi, Barbara König, and Daniela Petrisan. 2018b. Up-To Techniques for Behavioural Metrics via Fibrations. In *CONCUR'18, (LIPIcs, Vol. 118)*, Sven Schewe and Lijun Zhang (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 17:1–17:17. https://doi.org/10.4230/LIPIcs.CONCUR.2018.17

Filippo Bonchi, Daniela Petrisan, Damien Pous, and Jurriaan Rot. 2017b. A general account of coinduction up-to. *Acta Inf.* 54, 2 (2017), 127–190. https://doi.org/10.1007/s00236-016-0271-4

Filippo Bonchi and Damien Pous. 2013. Checking NFA equivalence with bisimulations up to congruence. In *Proc. POPL'13*, Roberto Giacobazzi and Radhia Cousot (Eds.). ACM, 457–468. https://doi.org/10.1145/2429069.2429124

Michael Brandt and Fritz Henglein. 1998. Coinductive Axiomatization of Recursive Type Equality and Subtyping. *Fundam. Inform.* 33, 4 (1998), 309–338. https://doi.org/10.3233/FI-1998-33401

Stephen D. Brookes, C. A. R. Hoare, and A. W. Roscoe. 1984. A Theory of Communicating Sequential Processes. *J. ACM* 31, 3 (1984), 560–599.

Stephen D. Brookes and A. W. Roscoe. 1984. An Improved Failures Model for Communicating Processes. In *Seminar on Concurrency (Lecture Notes in Computer Science, Vol. 197)*, Stephen D. Brookes, A. W. Roscoe, and Glynn Winskel (Eds.). Springer Verlag, 281–305.

Rance Cleaveland and Matthew Hennessy. 1993. Testing Equivalence as a Bisimulation Equivalence. *Formal Asp. Comput.* 5, 1 (1993), 1–20. https://doi.org/10.1007/BF01211314

David de Frutos-Escrig and Carlos Gregorio-Rodríguez. 2009. (Bi)simulations up-to characterise process semantics. *Inf. Comput.* 207, 2 (2009), 146–170. https://doi.org/10.1016/j.ic.2007.12.003

R. De Nicola and R. Hennessy. 1984. Testing Equivalences for Processes. *Theoretical Computer Science* 34 (1984), 83–133.

Adrien Durier, Daniel Hirschkoff, and Davide Sangiorgi. 2019. Divergence and unique solution of equations. *Logical Methods in Computer Science* 15, 3 (2019). https://doi.org/10.23638/LMCS-15(3:12)2019

Jörg Endrullis, Dimitri Hendriks, and Martin Bodin. 2013. Circular Coinduction in Coq Using Bisimulation-Up-To Techniques. In *ITP 2013 (Lecture Notes in Computer Science, Vol. 7998)*, Sandrine Blazy, Christine Paulin-Mohring, and David Pichardie (Eds.). Springer, 354–369. https://doi.org/10.1007/978-3-642-39634-2_26

R.J. van Glabbeek. 1993. The linear time—branching time spectrum II (The semantics of sequential systems with silent moves), In Proc. CONCUR '93, E. Best (Ed.). *Lecture Notes in Computer Science* 715. https://doi.org/10.1007/3-540-57208-2_6

R.J. van Glabbeek. 2001. The linear time—branching time spectrum I. In *Handbook of Process Algebra*, A. Ponse J. Bergstra and S. Smolka (Eds.). Elsevier, 3–99. https://doi.org/10.1016/b978-044482830-9/50019-9

J.F. Groote and F.W. Vaandrager. 1992. Structured Operational Semantics and Bisimulation as a Congruence. *Information and Computation* 100 (1992), 202–260. https://doi.org/10.1016/0890-5401(92)90013-6

M. Hennessy. 1988. *Algebraic Theory of Processes.* The MIT Press, Cambridge, Mass.

Matthew Hennessy. 2017. A Coinductive Equational Characterisation of Trace Inclusion for Regular Processes. In *Models, Algorithms, Logics and Tools (Lecture Notes in Computer Science, Vol. 10460)*, L. Aceto, G. Bacci, G. Bacci, A. Ingólfsdóttir, A. Legay, and R. Mardare (Eds.). Springer, 449–465. https://doi.org/10.1007/978-3-319-63121-9_22

M. Hennessy and R. Milner. 1985. Algebraic Laws for Nondeterminism and Concurrency. *Journal of the ACM* 32 (1985), 137–161.

John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. 2006. *Introduction to Automata Theory, Languages, and Computation (3rd Edition).* Addison-Wesley, Boston, MA, USA.

Bart Jacobs. 2006. Distributive laws for the coinductive solution of recursive equations. *Information and Computation* 204, 4 (2006), 561–587. https://doi.org/10.1016/j.ic.2005.03.006

Dexter Kozen. 1994. A Completeness Theorem for Kleene Algebras and the Algebra of Regular Events. *Inf. Comput.* 110, 2 (1994), 366–390. https://doi.org/10.1006/inco.1994.1037

Marina Lenisa. 1999. From Set-theoretic Coinduction to Coalgebraic Coinduction: some results, some problems. *Electronical Notes in Computer Science* 19 (1999), 2–22. https://doi.org/10.1016/S1571-0661(05)80265-8

Marina Lenisa, John Power, and Hiroshi Watanabe. 2000. Distributivity for endofunctors, pointed and co-pointed endofunctors, monads and comonads. *Electronical Notes in Computer Science* 33 (2000), 230–260. https://doi.org/10.1016/S1571-0661(05)80350-0

S. Milius, L. S. Moss, and D. Schwencke. 2013. Abstract GSOS Rules and a Modular Treatment of Recursive Definitions. *Logical Methods in Computer Science* 9, 3 (2013). https://doi.org/10.2168/LMCS-9(3:28)2013

R. Milner. 1989. *Communication and Concurrency.* Prentice Hall.

Ernst-Rüdiger Olderog and C. A. R. Hoare. 1986. Specification-Oriented Semantics for Communicating Processes. *Acta Inf.* 23, 1 (1986), 9–66.

D. Park. 1981. A New Equivalence notion for Communicating Systems. In *Bulletin EATCS*, G. Maurer (Ed.), Vol. 14. 78–80. Abstract of the talk presented at the Second Workshop on the Semantics of Programming Languages, Bad Honnef, March

16–20 1981. Abstracts collected in the Bulletin by B. Mayoh.

Joachim Parrow and Tjark Weber. 2016. The Largest Respectful Function. *Logical Methods in Computer Science* 12, 2 (2016). https://doi.org/10.2168/LMCS-12(2:11)2016

Iain Phillips. 1987. Refusal Testing. *Theor. Comput. Sci.* 50 (1987), 241–284. A preliminary version in Proc. ICALP'86, Lecture Notes in Computer Science 226, Springer Verlag..

A. Pnueli. 1985. Linear and Branching Structures in the Semantics and Logics of Reactive Systems. In *12th ICALP (Lecture Notes in Computer Science, Vol. 194)*, W. Brauer (Ed.). Springer Verlag, 15–32.

Lucia Pomello. 1985. Some equivalence notions for concurrent systems. An overview. In *Advances in Petri Nets 1985 (Lecture Notes in Computer Science, Vol. 222)*, Grzegorz Rozenberg (Ed.). Springer, 381–400. https://doi.org/10.1007/BFb0016202

D. Pous. 2007. Complete Lattices and Up-To Techniques. In *Proc. APLAS '07 (Lecture Notes in Computer Science, Vol. 4807)*. Springer Verlag, 351–366. http://dx.doi.org/10.1007/978-3-540-76637-7_24

Damien Pous. 2016. Coinduction all the way up. In *Proc. LICS*. ACM, 307–316. https://doi.org/10.1145/2933575.2934564

Damien Pous and Jurriaan Rot. 2017. Companions, Codensity, and Causality. In *Proc. FoSSaCS (Lecture Notes in Computer Science, Vol. 10203)*. Springer Verlag, 106–123. https://doi.org/10.1007/978-3-662-54458-7_7

Damien Pous and Davide Sangiorgi. 2012. Enhancements of the bisimulation proof method. In *Advanced Topics in Bisimulation and Coinduction*, Davide Sangiorgi and Jan Rutten (Eds.). Cambridge University Press.

Damien Pous and Davide Sangiorgi. 2019. Bisimulation and Coinduction Enhancements: A Historical Perspective. *Formal Asp. Comput.* 31, 6 (2019), 733–749. https://doi.org/10.1007/s00165-019-00497-w

Alexander Moshe Rabinovich. 1993. A Complete Axiomatisation for Trace Congruence of Finite State Behaviors. In *Proc. 9th MFPS (Lecture Notes in Computer Science, Vol. 802)*, Stephen D. Brookes, Michael G. Main, Austin Melton, Michael W. Mislove, and David A. Schmidt (Eds.). Springer, 530–543. https://doi.org/10.1007/3-540-58027-1_25

A. W. Roscoe. 1998. *The theory and practice of concurrency.* Prentice Hall. http://www.cs.ox.ac.uk/people/bill.roscoe/publications/68b.pdf

Jurriaan Rot, Filippo Bonchi, Marcello M. Bonsangue, Damien Pous, Jan Rutten, and Alexandra Silva. 2017. Enhanced coalgebraic bisimulation. *Mathematical Structures in Computer Science* 27, 7 (2017), 1236–1264. https://doi.org/10.1017/S0960129515000523

Jurriaan Rot, Marcello M. Bonsangue, and Jan Rutten. 2016. Proving language inclusion and equivalence by coinduction. *Inf. Comput.* 246 (2016), 62–76. https://doi.org/10.1016/j.ic.2015.11.009

Arto Salomaa. 1966. Two Complete Axiom Systems for the Algebra of Regular Events. *J. ACM* 13, 1 (1966), 158–169. https://doi.org/10.1145/321312.321326

D. Sangiorgi. 1998. On the bisimulation proof method. *Journal of Mathematical Structures in Computer Science* 8 (1998), 447–479. https://doi.org/10.1017/S0960129598002527

Davide Sangiorgi. 2012. *Introduction to Bisimulation and Coinduction.* Cambridge University Press. https://doi.org/10.1017/CBO9780511777110

Davide Sangiorgi. 2017. Equations, Contractions, and Unique Solutions. *ACM Trans. Comput. Log.* 18, 1 (2017), 4:1–4:30. https://doi.org/10.1145/2971339

D. Sangiorgi and R. Milner. 1992. The problem of "Weak Bisimulation up to". In *Proc. CONCUR '92 (Lecture Notes in Computer Science, Vol. 630)*, W.R. Cleveland (Ed.). Springer Verlag, 32–46. https://doi.org/10.1007/BFb0084781

A. Silva, F. Bonchi, M. Bonsangue, and J. Rutten. 2010. Generalizing the powerset construction, coalgebraically. In *FSTTCS (LIPIcs)*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 272–283. https://doi.org/10.4230/LIPIcs.FSTTCS.2010.272

Colin Stirling. 2001. *Modal and temporal properties of processes.* Springer Verlag.

Tarmo Uustalu, Varmo Vene, and Alberto Pardo. 2001. Recursion Schemes from Comonads. *Nord. J. Comput.* 8, 3 (2001), 366–390. http://www.cs.helsinki.fi/njc/References/uustaluvp2001:366.html