Basic Operational Preorders for Algebraic Effects in General, and for Combined Probability and Nondeterminism in Particular

Aliaume Lopez
École Normale Supérieure Paris-Saclay
Université Paris-Saclay, France
aliaume.lopez@ens-paris-saclay.fr

Alex Simpson
Faculty of Mathematics and Physics
University of Ljubljana, Slovenia
Alex.Simpson@fmf.uni-lj.si

Abstract
The “generic operational metatheory” of Johann, Simpson and Voigtländer (LiCS 2010) defines contextual equivalence, in the presence of algebraic effects, in terms of a basic operational preorder on ground-type effect trees. We propose three general approaches to specifying such preorders: (i) operational (ii) denotational, and (iii) axiomatic; coinciding with the three major styles of program semantics. We illustrate these via a nontrivial case study: the combination of probabilistic choice with nondeterminism, for which we show that natural instantiations of the three specification methods (operational in terms of Markov decision processes, denotational using a powerdomain, and axiomatic) all determine the same canonical preorder. We do this in the case of both angelic and demonic nondeterminism.

2012 ACM Subject Classification Theory of computation → Operational semantics, Theory of computation → Denotational semantics, Theory of computation → Axiomatic semantics

Keywords and phrases contextual equivalence, algebraic effects, operational semantics, domain theory, nondeterminism, probabilistic choice, Markov decision process

Digital Object Identifier 10.4230/LIPIcs.CSL.2018.29

Acknowledgements We thank Gordon Plotkin, Matija Pretnar and Niels Voorneveld for helpful discussions.

1 Introduction

Contextual equivalence, in the style of Morris, is a powerful and general method for defining program equivalence, applicable to many programming languages. Two programs are said to be contextually equivalent if they ‘behave’ equivalently when embedded in any suitable context that leads to ‘observable’ behaviour. More generally,\(^1\) one can define contextual preorder in the same manner. Let \(P_1\) and \(P_2\) be comparable programs (for example, in a typed language, \(P_1\) and \(P_2\) would have the same type in order to be comparable). Suppose further that we have some basic preorder \(\sqsubseteq\), defined on ‘observable’ computations, according to appropriate behavioural considerations. Then the contextual preorder is defined by

\[
P_1 \sqsubseteq_{\text{ctx}} P_2 \iff \text{for all observation contexts } C[-], \ C[P_1] \sqsubseteq C[P_2].
\]

\(^1\) It is more general, since every equivalence relation is a preorder.
This method of definition has important consequences. For example, the relation $\sqsubseteq_{\text{ctxt}}$ is guaranteed to be a precongruence with respect to the constructors of the programming language. However, the quantification over contexts makes the definition awkward to work with directly. So various more manageable techniques for reasoning about contextual preorder relations have been developed, including: (bi)simulations and their refinements (applicative/environmental bisimulations, bisimulations up-to), denotational interpretation in domains, game semantics, program logics, and logical relations. These techniques are all reasonably general, in the sense that they adapt to different styles of programming languages, and combinations of programming features. Nonetheless, they are usually studied on a language-by-language basis.

One direction for the systematisation of a range of programming features has been provided by Plotkin and Power through their work on algebraic effects [13, 14]. Broadly speaking, effects are interactions between a program and its environment (including the machine state), and include features such as error raising, global/local state, input/output, nondeterminism and probabilistic choice. Plotkin and Power realised that the majority of effects (including all the aforementioned ones) are algebraic, in the sense that the operations that trigger them satisfy a certain natural behavioural constraint.

The algebraic effects in a programming language can be supplied via an algebraic signature $\Sigma$ of effect-triggering operations, and the operational semantics of the language can then be defined parametrically in $\Sigma$. This is achieved by effectively splitting the semantics of the language into two steps. In the first step, operational rules specify how any program $P$ evaluates to an associated effect tree $|P|$, which documents all the effects that might potentially occur during execution. In an effect tree, the effects themselves are uninterpreted, in the sense that no specific execution behaviour is imposed upon them. As the second step, an interpretation is given to effect trees, by one means or another, from which a semantics for the whole language is extrapolated. This methodology was first followed in [13], where the operational reduction to effect trees (there called infinitary effect values) is used as a method for proving the computational adequacy of denotational semantics. In [6], effect trees (there called computation trees) are used to give a uniform definition of contextual preorder, and to characterise it as a logical relation. Effect trees also allow a general definition of applicative (bi)similarity for effects [17] (see [1] for a related approach not based on trees).

In this paper, as in [6], our aim is to exploit the notion of effect tree for the purpose of giving a unified theory of contextual preorders for programming languages with algebraic effects. In [6], this was carried out in the context of a specific polymorphically-typed call-by-name functional language with general recursion, to which algebraic effects were added. In this paper, we build on the technical work of [6], but an important departure is that we detach the development from any fixed choice of background programming language. This is based on the following general considerations. In order to define contextual preorder via (1) above, one needs to specify what constitutes an observation context, and also the basic behavioural relation $\preceq$ on the computations such contexts induce. In the case of a language with algebraic effects, we can observe two things about a computation. Firstly, we can observe any discrete return value. In any sufficiently expressive language, discrete values should be convertible to natural numbers. So it is a not unreasonable restriction to restrict observation contexts to ground contexts whose return values (if any) are natural numbers. Secondly, we can also potentially observe aspects of effectful behaviour of such computations.
with exactly what is observable very much depending on the effects in question. One general approach to taking such effectful behaviour into account is to specify a basic operational preorder \( \preceq \) on the set of effect trees with natural-number-labelled leaves, which implements a desired behavioural preorder on effectful computations with return values in \( \mathbb{N} \). We are thus led to the following general formulation of contextual preorder. Given a chosen basic operational preorder \( \preceq \), we define the induced contextual preorder on programs by:

\[
P_1 \sqsubseteq_{\text{ctxt}} P_2 \iff \text{for all ground contexts } C[-], \ |C[P_1]| \preceq |C[P_2]|.
\]

(2)

In [6], this general approach was developed in detail for a polymorphically typed call-by-name functional language with algebraic effects. The main result was that the resulting contextual preorder, defined by (2), is well behaved if the basic operational preorder satisfies two technical properties, admissibility and compositionality. In particular, it follows from these conditions that the contextual preorder is characterisable as a logical relation (and hence amenable to an important proof technique), and also that, on ground type programs \( P_1, P_2 \), the contextual and basic operational preorders coincide (i.e., \( P_1 \sqsubseteq_{\text{ctxt}} P_2 \) if and only if \( |P_1| \preceq |P_2| \)). Recently, we have carried out a similar programme for a call-by-value language, similar to the language in [13], and obtained analogous results. It seems likely that similar results hold for other language variants.

The notion of admissible and compositional basic operational preorder thus provides a uniform and well-behaved definition of contextual preorder, for different languages with algebraic effects. Furthermore, as is argued in [6, §V], it can also be given an intrinsic, more conceptually motivated justification in terms of an explicit notion of observation. Our general position is that the notion of admissible and compositional basic operational preorder is a fundamental one. For any given combination of algebraic effects, one need only define a corresponding admissible and compositional basic operational preorder. Once this has been done, one obtains, via (2), a definition of contextual preorder that can be applied to many programming languages containing those effects, and which will enjoy good properties.

In this paper, we describe three different approaches to defining basic operational preorders. The first is an operational approach. One explicitly models the execution of the effects in question, and uses this model to determine the preorder. This is the approach that was followed in [6]. Under this approach, admissibility and compositionality do not hold automatically, and so need to be explicitly verified. The second is a denotational approach. One builds a suitable domain-based model of the relevant effect operations. This induces a basic operational preorder on effect trees that is automatically admissible and compositional. The third is axiomatic. One finds a set of (possibly infinitary) Horn-clause axioms asserting desired properties of the intended preorder. The basic operational preorder is then taken to be the smallest admissible preorder satisfying the axioms. In addition to being admissible by definition, the resulting preorder is automatically compositional.

It will not have escaped the readers attention that our three approaches to defining preorders parallel the three main styles of program semantics: operational, denotational and axiomatic. Nonetheless, irrespective of how they are defined, we view basic operational preorders themselves as a part of operational semantics, for their purpose is to define the operational notion of contextual preorder.

The general identification of these three approaches is the first main contribution of the paper. Our second contribution is more technical. We illustrate the three approaches with a nontrivial case study: the combination of (finitary) nondeterminism with probabilistic choice.

---

3 Unfortunately, there is no space to include these results, which were obtained while the first author was on an internship in Ljubljana in 2017, in this paper.
29:4 Basic Operational Preorders for Probability and Nondeterminism

which is a combination of effects that enjoys a certain notoriety for some of the technical complications it incurs [11, 12, 21, 20, 2, 3, 7]. On the operational side, we consider effect trees as Markov decision processes (MDPs), and we define a basic operational preorder based on the comparison of values of MDPs. On the denotational side, we make use of recently developed domain-theoretic models of combined nondeterministic and probabilistic choice [20, 3, 7]. On the axiomatic side, we give a simple axiomatisation, similar to axiomatisations in [12, 7]. Our main result is that the operationally, denotationally and axiomatically-defined basic operational preorders all coincide with each other. In fact, we give this result in two different versions. The first is for an angelic interpretation of nondeterminism, in which nondeterministic choices are resolved by a cooperative scheduler. The second is for demonic nondeterminism, where an antagonistic scheduler is assumed. In each case, our coincidence theorem suggests the canonicity of the preorder we obtain for the form of nondeterminism in question, with each of the three methods of definition providing a distinct perspective on it.

In Sections 2 and 3, we review the definition of effect trees and basic operational preorders, largely following [6]. Our main contribution starts in Sections 4, 5 and 6, which discuss the operational, denotational and axiomatic approaches to defining basic operational preorders. The discussion is illustrated using the example of combined nondeterminism and probabilistic choice. The main coincidence theorem, for this example, is then proved in Section 7. Finally, in Section 8, we briefly discuss related and further work.

2 Effect trees

The general scenario this paper addresses is that of a programming language whose programs may perform effects as they compute. In this paper, we assume that the available effects are specified by an effect signature: a set \( \Sigma \) of operation symbols, each with an associated finite arity. We call the operations in \( \Sigma \) effect operations. This setting is explicitly that of [13]. More general effect signatures appear in the literature, e.g., allowing parameterised operations and infinite arities [6, 19]. The technical development in this paper can be generalised to such more general signatures. Since, however, the main running example considered in this paper has only binary operations, we restrict ourselves to finite arity operations for the sake of presentational convenience.

Example 1 (Signature for combined probabilistic and non-deterministic choice). Consider a programming language that can perform two effects: probabilistic and nondeterministic choice. An appropriate signature for such a language is \( \Sigma_{pr/nd} = \{ (pr, 2), (or, 2) \} \) containing two binary operations: nondeterministic choice or, and fair probabilistic choice pr. (As is well known, in programming languages with general recursion, all computable discrete probability distributions can be simulated using fair probabilistic choice.)

During the execution of a program with effects, three different situations can arise. Firstly, the computation process may trigger an effect, represented by some \( o \in \Sigma \). The execution will then continue along one of the \( n \) possible continuation processes given as arguments to the operation \( o \). Secondly, the execution may terminate, in which case it may produce a resulting value. Thirdly, the execution may continue forever without terminating and without invoking any effects. We call this last situation silent nontermination to distinguish it from noisy nontermination, which occurs when the computation process computes for ever while performing an infinite sequence of effects along the way.

The global behaviour of such a program is captured by the notion of an effect tree: a finitely branching tree, whose internal nodes represent effect operations, and whose leaves represent either termination with a result, or silent nontermination. The branches of the tree
represent potential execution sequences of the program. Trees are allowed to be infinitely deep, with their infinite branches representing noisy nontermination. Such trees were introduced as infinitary effect values in [13], and used extensively in [6], where they are called computation trees. Two example trees, for computations that return natural number values, are drawn in Figure 1 below. The left-hand tree \text{or(pr(1, 2), 3)} represents a program that first makes a nondeterministic choice and then a potential probabilistic choice, with the choices determining the resulting number. In the second tree \text{pr(or(1, 3), or(2, 3))}, the probabilistic choice is made first, followed by the relevant nondeterministic choice.

\textbf{Definition 2.} The set \text{Trees}(X) of effect trees with values from the set \text{X} is coinductively defined so that every tree has one of the following forms.

- The root of the tree is labelled with an operation \( o \in \Sigma \), and the tree has the form \( o(t_1, \ldots, t_n) \) where \( n \) is the arity of \( o \) and \( t_1, \ldots, t_n \in \text{Trees}(X) \); or
- the tree is a leaf labelled with a value \( x \in \mathcal{X} \); or
- the tree is a leaf labelled with \( \bot \).

As this is a coinductive definition, \text{Trees}(X) contains trees of both finite and infinite depth.

We define a partial order on \text{Trees}(X) by \( t_1 \sqsubseteq t_2 \) if and only if \( t_2 \) can be obtained from \( t_1 \) by replacing (possibly infinitely many) \( \bot \)-leaves appearing in \( t_1 \) with arbitrary replacement trees (rooted where the leaves were located). With this ordering, \text{Trees}(X) is an \( \omega \)-complete partial order (\( \omega \)-CPO) with least element \( \bot \). Furthermore, by considering it as a tree constructor, every operation \( o \in \Sigma \) defines a continuous (i.e., \( \omega \)-continuous) function \( o : \text{Trees}(X)^n \rightarrow \text{Trees}(X) \), where \( n \) is the arity of \( o \). (For notational convenience, we use \( o \) for both operation symbol and function. The ambiguity can be resolved from the context.)

The properties described above state that \text{Trees}(X) is a continuous \( \Sigma \)-algebra. In general, a \textit{continuous} \( \Sigma \)-\textit{algebra} is a pointed (i.e., with least element) \( \omega \text{-CPO} \) \( A \) with associated continuous functions \( o_A : A^n \rightarrow A \) for every \( o \in \Sigma \) of arity \( n \). As morphisms between continuous \( \Sigma \)-algebras \( A \) and \( B \), we consider functions \( h : A \rightarrow B \) that are strict (i.e., preserve least element) continuous homomorphisms with respect to the \( \Sigma \)-algebra structure. We refer to such functions \( h : A \rightarrow B \) as \textit{continuous homomorphisms}, leaving the strictness property implicit. We write \text{ContAlg}_\Sigma \text{ for the category of continuous } \Sigma \text{-algebras and continuous homomorphisms. The characterisation of Trees(X) below is standard.}

\textbf{Proposition 3.} Trees(X) is the free continuous \( \Sigma \)-algebra over the set \text{X}. 

\[ X \xrightarrow{f} A \]

\[ \xrightarrow{f} \text{Trees}(X) \]
Basic Operational Preorders for Probability and Nondeterminism

That is, for every function \( f : X \to A \), where \( A \) is a continuous \( \Sigma \)-algebra, there exists a unique continuous homomorphism
\[
\hat{f} : \text{Trees}(X) \to A
\]
such that \( f = \hat{f} \circ i \), where \( i : X \to \text{Trees}(X) \) is the function mapping every \( x \in X \) to the leaf-tree labelled \( x \).

We use the above proposition to define a substitution operation on trees. For any tree \( t \in \text{Trees}(X) \), every function \( f : X \to \text{Trees}(Y) \) determines a tree \( t[f] \) in \( \text{Trees}(Y) \) defined by substitution, viz: \( t[f] := \hat{f}(t) \).

\section{Basic operational preorders}

As discussed in Section 1, our interest in effect trees is that they provide a uniform template for defining contextual preorders for programming languages with algebraic effect operations specified by signature \( \Sigma \). As in [6], the crucial data is provided by a preorder \( \preceq \) on \( \text{Trees}(N) \), called the basic operational preorder. In order for the resulting contextual preorder to be well behaved, we ask for the the basic operational preorder satisfy two properties: admissibility and compositionality. In this section, we review the definitions of these and related notions.

\begin{definition}[Admissibility]
A binary relation \( R \) on \( \text{Trees}(X) \) is admissible if, for every ascending chain \( (t_i)_{i \geq 0} \) and \( (t'_i)_{i \geq 0} \), we have:
\[
(t_i R t'_i \text{ for all } i) \implies \left( \bigsqcup_{i \geq 0} t_i \right) R \left( \bigsqcup_{i \geq 0} t'_i \right).
\]
\end{definition}

\begin{definition}[Compatibility]
A binary relation \( R \) on \( \text{Trees}(X) \) is compatible if, for every \( o \in \Sigma \) of arity \( n \), and for all trees \( t_1, \ldots, t_n \) and \( t'_1, \ldots, t'_n \), we have:
\[
(t_i R t'_i \text{ for all } i = 1, \ldots, n) \implies o(t_1, \ldots, t_n) R o(t'_1, \ldots, t'_n).
\]
If a compatible relation is a preorder then it is called a precongruence. If it is an equivalence relation it is called a congruence.

The next two definitions make use of the substitution operation on trees defined at the end of Section 2.

\begin{definition}[Substitutivity]
A binary relation \( R \) on \( \text{Trees}(X) \) is substitutive if, for all trees \( t \), \( t' \) and \( \{t_x\}_{x \in X} \), we have:
\[
t R t' \implies t[x \mapsto t_x] R t'[x \mapsto t_x].
\]
\end{definition}

\begin{definition}[Compositionality]
A binary relation \( R \) on \( \text{Trees}(X) \) is compositional if, for all trees \( t \), \( t' \), \( \{t_x\}_{x \in X} \), and \( \{t'_x\}_{x \in X} \), we have:
\[
(t R t' \text{ and } t_x R t'_x \text{ for all } x \in X) \implies t[x \mapsto t_x] R t'[x \mapsto t'_x].
\]
\end{definition}

\begin{proposition}
Let \( \preceq \) be a preorder on \( \text{Trees}(N) \).
1. If \( \preceq \) is compositional then it is a substitutive precongruence.
2. If \( \preceq \) is an admissible substitutive precongruence then it is compositional.
\end{proposition}
demonic
interpretation, models the certainty with which a goal can be achieved, however
We would like to use compatibility to derive that also
t'[n ↦ t_n] ≲ t'[n ↦ t_n]', however this is only possible if t' is finite. The solution is to use finite approximations (s') of
t' satisfying [ ], s' = t'. For each finite tree s we have that s'[n ↦ t_n] ≲ s'[n ↦ t_n'], by compatibility. Hence, by admissibility, t'[n ↦ t_n] ≲ t'[n ↦ t_n]', whence t[n ↦ t_n] ≲ t'[n ↦ t_n]' by transitivity.

4 Operationally-defined preorders

In this section, we consider our first approach to defining an admissible and compositional basic operational preorder ≲ on Trees(ℕ). We call this method operational. Its characteristic is that the preorder ≲ is directly defined using a mathematical model of the way that an effect tree in Trees(ℕ) will be executed. There is not much to say in general about this approach, since such execution models vary enormously from one effect to another. The main point to emphasise is that there is no general reason for admissibility and compositionality to hold for such operationally defined preorders. Accordingly, these properties need to be established on a case-by-case basis.

The operational approach to defining basic preorders is illustrated for several examples of effects in [6]. The main goal of the section is to demonstrate the approach using a different example, the signature Σ_{pr/nd} = {pr, or} from Example 1, which is of interest because of the interplay between probabilistic and nondeterministic effects. In this case, trees in Trees(ℕ) have both probabilistic and nondeterministic branching nodes, as in Figure 1.

It is natural to consider such trees as (countable state) Markov decision processes, with the leaves representing nodes which either carry an observable value from ℕ, or which represent nontermination ⊥. Nondeterministic choices may be thought of as being resolved by an external agent, the scheduler. We model the actions of the scheduler by a function s : {l,r}∗ → {l,r}. The idea is that a word w ∈ {l,r}∗ represents a finite path of left/right choices from the root of a tree t ∈ Trees(ℕ). If the computation reaches a nondeterministic choice at the node indexed by w then it takes the left/right branch according to the value of s(w). This way of representing choices has some redundancy (in every tree that is not a complete infinite binary tree, there will be words w that do not index nodes in the tree; if s(ε) = l then the value of s on words beginning with r is immaterial; the value of s(w) on words w that index probabilistic nodes in t is irrelevant, etc.), but it is simple and convenient for future purposes. For any given t ∈ Trees(ℕ), such a function s : {l,r}∗ → {l,r} can be thought of as a (deterministic) strategy for the scheduler, in which the choice of direction at a nondeterministic node can respond to the outcomes of probabilistic nodes higher up the tree.

A strategy s and a tree t in combination determine a subtree t ≤ s, defined by removing, at every nondeterministic node in t with index w, the child tree that is not selected by s(w). So t ≤ s is a tree that has binary branching at probabilistic nodes, and unary branching at nondeterministic nodes. It is thus, in effect, a purely probabilistic tree, with leaves in ℕ ∪ {⊥}, and so may be viewed as a Markov chain, in which the branching nodes are fair binary choices, determining a subprobability distribution over ℕ. Specifically, each n ∈ ℕ is assigned the probability that a run of the Markov chain will end at a leaf labelled with n. This is a subprobability distribution in general because there can be a positive probability of nontermination (either at a ⊥ leaf, or along an infinite branch).

The angelic interpretation of nondeterminism takes into account the possibility of a nondeterministic computation achieving a specified goal, given a cooperative scheduler. The demonic interpretation, models the certainty with which a goal can be achieved, however
adversarial the scheduler. This suggests the two basic operational preorders below. In each case, we consider functions $h: \mathbb{N} \to [0, \infty]$ assigning desirability weightings to possible results of a run of the computation. We then define $t \preceq t'$ if, for any $h$, the ‘expected’ desirability weighting of $t'$ exceeds that of $t$. Here, ‘expected’ is in inverted commas, because we have to take into account the actions of the scheduler, so this is not just a probabilistic expectation. In the case of angelic nondeterminism, the scheduler will help us, whereas, under demonic nondeterminism, it will impede us. Technically this is taken account of by considering suprema of probabilistic expectations in the angelic case, and infima in the demonic case.

$$ t \preceq_{\text{pr/ang}} t' \iff \forall h: \mathbb{N} \to [0, \infty] \quad \sup_s E_{t|s}(h) \leq \sup_s E_{t'|s}(h) $$

$$ t \preceq_{\text{pr/dem}} t' \iff \forall h: \mathbb{N} \to [0, \infty] \quad \inf_s E_{t|s}(h) \leq \inf_s E_{t'|s}(h) $$

Here $E_{t|s}(h)$ means the expectation of the function $h$ under the subprobability distribution on $\mathbb{N}$ induced by the Markov chain $t|s$. In Markov-decision-process terminology, each preorder $\preceq$ is continuous (since it is obviously monotone).

**Proposition 9.** The preorders $\preceq_{\text{pr/ang}}$ and $\preceq_{\text{pr/dem}}$ are admissible and compositional.

We outline the proof of this proposition in the case of $\preceq_{\text{pr/dem}}$. The proof for $\preceq_{\text{pr/ang}}$ is easier, largely because the analogue of the lemma below is trivial in the case of angelic nondeterminism.

**Lemma 10.** Consider $\text{Trees}(\mathbb{N})$ and $[0, +\infty]$ as $\omega$CPOs. Then, for any $h: \mathbb{N} \to [0, \infty]$, the value-finding function $F_h$ is continuous:

$$ F_h : t \mapsto \inf_s E_{t|s}(h) : \text{Trees}(\mathbb{N}) \to [0, +\infty] $$

**Proof.** The set $S = \{l, r\}^{t(r)}$ of strategies is a countably-based compact Hausdorff space under the product topology. (It is Cantor space.) It is easy to see that the function

$$ G_h : (s, t) \mapsto E_{t|s}(h) : S \times \text{Trees}(\mathbb{N}) \to [0, +\infty] $$

is continuous. Essentially, it follows that $F_h$ is continuous because it is defined from $G_h$ by taking an infimum over a compact set. This can be made precise using, e.g., the general machinery in Section 7.3 of [16]. For completeness, we give a self-contained argument.

Suppose $(t_i)$ is an ascending chain of trees. Because $S$ is compact, there is $s \in S$ with

$$ \inf_s G_h(s, t_i) = G_h(s, t_i), $$

and we can then extract a convergent subsequence $(s_{t_i})$ of $(s_i)$ such that $s_{t_i} \to s$ in $S$. Then:

$$ \sup_i \inf_s G_h(s, t_i) = \sup_i G_h(s_i, t_i) = \sup_i G_h(s_{t_i}, t_{t_i}) = G_h(s, \bigcup_i t_i) \geq \inf_s G_h(s, \bigcup_i t_i), $$

where the second equality holds because $G_h(s_i, t_i)$ is an ascending sequence, and the third by the continuity of $G_h$. We have shown that $\sup_i \inf_s G_h(s, t_i) \geq \inf_s G_h(s, \bigcup_i t_i)$, i.e.,

$$ \sup_i F_h(t_i) \geq F_h(\bigcup_i t_i). $$

Therefore $F_h$ is continuous (since it is obviously monotone).

The admissibility of $\preceq_{\text{pr/dem}}$ follows easily from the lemma. Suppose $t_i \preceq_{\text{pr/dem}} t'_i$, for ascending chains $(t_i)$ and $(t'_i)$. Then $F_h(t_i) \leq F_h(t'_i)$, for all $i$ and $h$. By the lemma, $F_h(\bigcup_i t_i) \leq F_h(\bigcup_i t'_i)$, for all $h$. So indeed $\bigcup_i t_i \preceq_{\text{pr/dem}} \bigcup_i t'_i$.

For compositionality, by Proposition 8, it suffices to show that $\preceq_{\text{pr/dem}}$ is a substitutive precongruence. The compatibility properties of a precongruence are easily shown. Substitutivity follows from the lemma below.
Lemma 11. Suppose \( t \) and \( \{ t_n \}_{n \in \mathbb{N}} \) are trees in \( \text{Trees}(\mathbb{N}) \) then, for any weighting \( h \),

\[
\inf_s E_{t,(u \rightarrow s_t,u)}(h) = \inf_s E_{t_s}(\tilde{h}) \quad \text{where} \quad \tilde{h}(u) = \inf_s E_{t,u}(h).
\]

This lemma is proved first for finite trees, by induction on their height. It is then extended to infinite trees by expressing them as suprema of finite trees, and applying Lemma 10.

We end this section by observing that a natural attempt to simplify the definitions of \( \leq_{\text{pr/ang}}^\text{op} \) and \( \leq_{\text{pr/dem}}^\text{op} \) does not work. Instead of considering arbitrary weightings \( h : \mathbb{N} \rightarrow [0, \infty] \), one might restrict to functions \( h : \mathbb{N} \rightarrow \{0, 1\} \), which can be viewed as specifying goal subsets \( H \subseteq \mathbb{N} \). Proceeding analogously to above, we compare suprema of probabilities of landing in \( H \) in the angelic case, and infima in the demonic case. For both the angelic and demonic versions, the desired compositionality property fails.

Proposition 12. Neither of the formulas below defines a compositional relation \( \preceq \neq t' \).

\[
\forall H \subseteq \mathbb{N} \quad \sup_s P_{t|s}(H) \leq \sup_s P_{t'|s}(H)
\]

\[
\forall H \subseteq \mathbb{N} \quad \inf_s P_{t|s}(H) \leq \inf_s P_{t'|s}(H)
\]

Proof. We use the two trees in Figure 1, representing the expressions \( A = 3 \text{ or } (1 \text{ pr } 2) \) and \( B = (3 \text{ or } 1) \text{ pr } (3 \text{ or } 2) \). It is easily checked that, for every subset \( H \subseteq \{1, 2, 3\} \), it holds that \( \sup_s P_{A|s}(H) = \sup_s P_{B|s}(H) \) and \( \inf_s P_{A|s}(H) = \inf_s P_{B|s}(H) \). Thus \( A \) is equivalent to \( B \) under both preorders.

However, one can build a family \( \{ t_1, t_2, t_3 \} \) such that \( A[i \mapsto t_1] = t_3 \text{ or } (t_1 \text{ pr } t_2) = C \) is not equivalent to \( B[i \mapsto t_1] = (t_3 \text{ or } t_1) \text{ pr } (t_3 \text{ or } t_2) = D \), which contradicts substitutivity. Let \( t_1 = 0 \text{ pr } (0 \text{ pr } (0 \text{ pr } 1)) \), \( t_2 = 1 \) and \( t_3 = 0 \text{ pr } (0 \text{ pr } 1) \). The distinguishing factor will be the probability associated with the subset \( \{1\} \).

A simple calculation shows that \( \sup_s P_{C|s}(\{1\}) = 9/16 \neq 5/8 = \sup_s P_{D|s}(\{1\}) \). Similarly \( \inf_s P_{C|s}(\{1\}) = 1/4 \neq 3/16 = \inf_s P_{D|s}(\{1\}) \). This contradicts the substitutivity and hence also the compositionality of both preorders. □

The necessity of using quantitative properties to obtain a compositional preorder is consistent with a general need for quantitative concepts that can be found in the literature on probabilistic computation. For example, in [8, 9], quantitative logics are required to obtain compositional reasoning methods. Similarly, in [10], quantitative observations are needed to distinguish non-bisimilar processes combining probabilistic and nondeterministic choice.

5 Denotationally-defined preorders

Our second approach to defining an admissible and compositional basic denotational preorder \( \preceq \) on \( \text{Trees}(\mathbb{N}) \) is to make use of established constructions from domain theory. Under this approach, admissibility and compositionality of the defined preorder \( \preceq \) hold for general reasons. Since this approach essentially amounts to giving a denotational semantics to effect trees, we call it the denotational method of defining a basic operational preorder.

In order to define a basic operational preorder using the denotational method, one needs to merely provide a continuous \( \Sigma \)-algebra \( D \) (see Section 2), together with a function \( j : \mathbb{N} \rightarrow D \). Define \( \llbracket \cdot \rrbracket : \text{Trees}(\mathbb{N}) \rightarrow D \) to be the unique continuous homomorphism that makes the diagram below commute.

\[
\begin{array}{ccc}
\mathbb{N} & \xrightarrow{j} & D \\
\downarrow & & \\
\text{Trees}(\mathbb{N}) & \xrightarrow{\llbracket \cdot \rrbracket} & D
\end{array}
\]
The map \[\cdot\] : Trees(\(\mathbb{N}\)) \(\rightarrow\) \(D\) is used to induce the basic operational preorder \(\preceq_D\) from the partial order relation on the \(\omega\)CPO \(D\).
\[
t \preceq_D t' \iff \llbracket t \rrbracket \subseteq \llbracket t' \rrbracket.
\]

**Proposition 13.** The relation \(\preceq_D\) is admissible precongruence.

The proof is immediate: admissibility follows from the continuity of \(\cdot\), and compatibility because \(\cdot\) is a homomorphism.

In order to obtain substitutivity, hence compositionality, a further property is required.

**Definition 14** (Factorisation property). The map \(j : \mathbb{N} \rightarrow D\) is said to have the factorisation property if, for every function \(f : \mathbb{N} \rightarrow D\), there exists a continuous homomorphism \(h_f : D \rightarrow D\) such that \(f = h_f \circ j\).

\[
\mathbb{N} \xrightarrow{j} D \xrightarrow{h_f} D.
\]

**Proposition 15.** If \(j : \mathbb{N} \rightarrow D\) has the factorisation property then the relation \(\preceq_D\) is substitutive, hence it is an admissible compositional precongruence.

**Proof.** Suppose \(\sigma : \mathbb{N} \rightarrow\) Trees(\(\mathbb{N}\)) is any substitution. Let \(\hat{\sigma} : \text{Trees}(\mathbb{N}) \rightarrow\) Trees(\(\mathbb{N}\)) be the continuous homomorphism such that \(\hat{\sigma} \circ i = \sigma\). Consider the map \(g := \llbracket \cdot \rrbracket \circ \hat{\sigma} \circ i : \mathbb{N} \rightarrow D\).

By the factorisation property, there exists \(h_g : D \rightarrow D\) such that \(g = h_g \circ j\). Expanding this, and using the definition of \(\cdot\), we have:
\[
\llbracket \cdot \rrbracket \circ \hat{\sigma} = h_g \circ \llbracket \cdot \rrbracket.
\]

It then follows from the uniqueness property of Proposition 3 that
\[
\llbracket \cdot \rrbracket \circ \hat{\sigma} = h_g \circ \llbracket \cdot \rrbracket,
\]
because both maps are continuous homomorphisms.

Now, for substitutivity, suppose that \(t \preceq_D t'\), i.e., \(\llbracket t \rrbracket \subseteq \llbracket t' \rrbracket\). Then \(h_g(\llbracket t \rrbracket) \leq h_g(\llbracket t' \rrbracket)\) by monotonicity. That is \(\hat{\sigma}(\llbracket t \rrbracket) \leq \hat{\sigma}(\llbracket t' \rrbracket)\), by (3). This says that \(\llbracket t[\sigma] \rrbracket \subseteq \llbracket t'[\sigma] \rrbracket\). That is \(t[\sigma] \preceq_D t'[\sigma]\), as required. \(\blacksquare\)

In practice, it is usually not necessary to prove the factorisation property directly. Instead it holds as a consequence of the continuous algebra \(D\) and map \(j : \mathbb{N} \rightarrow D\) being derived from a suitable monad. The next result establishes general conditions under which this holds.

**Proposition 16.** Let \(\mathbf{S}\) be a category with a faithful functor \(U : \mathbf{S} \rightarrow \text{Set}\). Suppose also that \(\mathbf{S}\) has an object \(N\) such that \(UN = \mathbb{N}\), and every hom set \(\mathbf{S}(N, X)\) is mapped bijectively by \(U\) to \(\text{Set}(\mathbb{N}, UX)\). Suppose also that \((T, \eta, \mu)\) is a monad on \(\mathbf{S}\) with the following properties: there is a continuous \(\Sigma\)-algebra structure on \(UTN\); and, for every map \(f : N \rightarrow TN\), the induced function \(Uf^*\), where \(f^* : TN \rightarrow TN\) is the Kleisli lifting, is a continuous homomorphism. Then defining \(D\) to be the continuous \(\Sigma\)-algebra on \(UTN\), and \(j\) to be \(U\eta : \mathbb{N} \rightarrow UTN\), it follows that \(j\) has the factorisation property.

We omit the easy proof. Although the statement of the proposition is verbose, the result is relatively easy to apply in practice, as the examples we consider next will illustrate.

In the remainder of the section, we return to our main example, and again define basic operational preorders for the combination of probabilistic choice and nondeterminism (both angelic and demonic), but this time we use the denotational method. Accordingly, we call the defined preorders \(\preceq_{\text{pr/ang}}\) and \(\preceq_{\text{pr/dem}}\).
We use the powerdomains combining probabilistic choice and nondeterminism defined in [7, §3.4], although our setting is simpler because we only need to apply them to sets. The basic idea of these constructions is that a computation with probabilistic and nondeterministic choice is modelled as a set of subprobability distributions, where the set collects the possible nondeterministic outcomes, each of which is probabilistic in nature.

As is standard, the sets relevant to angelic nondeterminism are the closed sets in the Scott topology, whereas those relevant to demonic nondeterminism are the compact upper-closed sets, see [18]. Due to the combination with probabilistic choice, sets are further required to be convex; see, for example, the discussion in [7].

Let \( \mathcal{V}_{\leq 1} X \) be the \( \omega \)CPO of (discrete) subprobability distributions on a set \( X \). We write \( \mathcal{H} \mathcal{V}_{\leq 1} X \) for the \( \omega \)CPO of nonempty Scott-closed convex subsets of \( \mathcal{V}_{\leq 1} X \) ordered by subset inclusion \( \subseteq \). We write \( \mathcal{S} \mathcal{V}_{\leq 1} X \) for the \( \omega \)CPO of nonempty Scott-compact convex upper-closed subsets of \( \mathcal{V}_{\leq 1} X \) ordered by reverse inclusion \( \supseteq \). The \( \omega \)CPOs \( \mathcal{H} \mathcal{V}_{\leq 1} X \) and \( \mathcal{S} \mathcal{V}_{\leq 1} X \) are both continuous algebras for \( \Sigma_{\text{pr/nd}} \). In both cases, the operations are defined by:

\[
\begin{align*}
or(A, B) &= \text{conv}(A \cup B) \\
p(A, B) &= \{\frac{1}{2}a + \frac{1}{2}b \mid a \in A, b \in B\} ,
\end{align*}
\]

where \( \text{conv} \) is the convex closure operation. We remark that these straightforward uniform definitions are possible because of the simple structure of the domains \( \mathcal{H} \mathcal{V}_{\leq 1} X \) and \( \mathcal{S} \mathcal{V}_{\leq 1} X \), over a set \( X \). For the more general lower and upper ‘Kegelspitze’ considered in [7], additional order-theoretic and topological closure operations need to be applied.

To apply the above in the angelic case, we use the fact that \( \mathcal{H} \mathcal{V}_{\leq 1} X \) is the free Kegelspitze join semilattice over a set \( X \) [7, Corollary 3.15] (where the result is proved more generally for domains). It follows that \( \mathcal{H} \mathcal{V}_{\leq 1} \) is a monad on \( \text{Set} \) itself satisfying the conditions of Proposition 16. Thus defining \( D_{\text{pr/ang}} = \mathcal{H} \mathcal{V}_{\leq 1} \mathbb{N} \), and \( j(n) = \downarrow \delta(n) \) (where \( \delta(n) \) is the Dirac probability distribution that assigns probability 1 to \( n \) and 0 to all other numbers, and \( \downarrow x \) is the down-closure \( \{y \mid y \leq x\} \)), the induced \( \llbracket \cdot \rrbracket_{\text{pr/ang}} : \text{Trees}(\mathbb{N}) \rightarrow D_{\text{pr/ang}} \) defines an admissible and compositional preorder

\[
t \xleftarrow{\text{pr/ang}} t' \iff \llbracket t \rrbracket_{\text{pr/ang}} \leq \llbracket t' \rrbracket_{\text{pr/ang}} .
\]

Similarly, in the demonic case, we use [7, Corollary 3.16], which characterises \( \mathcal{S} \mathcal{V}_{\leq 1} X \) as the free Kegelspitze meet semilattice over \( X \). Again \( \mathcal{S} \mathcal{V}_{\leq 1} \) is a monad on \( \text{Set} \) to which Proposition 16 applies. In this case, we define \( D_{\text{pr/dem}} = \mathcal{S} \mathcal{V}_{\leq 1} \mathbb{N} \), and \( j(n) = \{\delta(n)\} \). Then the induced \( \llbracket \cdot \rrbracket_{\text{pr/dem}} : \text{Trees}(\mathbb{N}) \rightarrow D_{\text{pr/dem}} \) defines an admissible and compositional preorder

\[
t \xleftarrow{\text{pr/dem}} t' \iff \llbracket t \rrbracket_{\text{pr/dem}} \leq \llbracket t' \rrbracket_{\text{pr/dem}} .
\]

6 Axiomatically-defined preorders

In this section, we look at the definition of basic operational preorders by axiomatising properties of the operations in the effect signature \( \Sigma \). Since we are defining a preorder, it is appropriate for the axiomatisation to involve inequalities specifying desired properties of the operational preorder. As the technical framework for this, we allow axiomatisations involving infinitary Horn-clauses of inequalities between infinitary terms. This provides a flexible general setting for axiomatising admissible and compositional preorders on \( \text{Trees}(\mathbb{N}) \).

Let \( \text{Vars} \) be a set of countably many distinct variables. By an expression, we mean a tree \( e \in \text{Trees}(\text{Vars}) \). The use of trees incorporates infinitary non-well-founded terms alongside...
the usual finite algebraic terms. By an inequality we mean a statement $e_1 \leq e_2$, where $e_1, e_2$ are expressions. By an (infinitary) Horn clause we mean an implication of the form:

$$\left( \bigwedge_{i \in I} e_i \leq e_i' \right) \implies e \leq e' ,$$

(4)

An effect theory $T$ is a set of Horn clauses.

A precongruence $\preceq$ on Trees$(X)$ is said to satisfy a Horn clause (4) if, for every environment $\rho : \text{Vars} \rightarrow \text{Trees}(X)$, the implication below holds (recall the notation for tree substitution from Section 2).

$$\left( \bigwedge_{i \in I} e_i[\rho] \preceq e_i' [\rho] \right) \implies e[\rho] \preceq e'[\rho] .$$

We say that a precongruence $\preceq$ is a model of a Horn clause theory $T$ if it satisfies every Horn clause in $T$. We consider models as subsets of Trees$(X) \times \text{Trees}(X)$, partially ordered by inclusion. Note that models are precongruences by assumption.

\textbf{Proposition 17.} Every Horn clause theory $T$ has a smallest admissible model

$$\preceq_T \subseteq \text{Trees}(X) \times \text{Trees}(X) ,$$

for any set $X$. The model $\preceq_T$ is substitutive. In the case that $X = \mathbb{N}$, the smallest admissible model is thus an admissible compositional preorder.

\textbf{Proof.} It is easily seen that the intersection of any set of admissible models is itself an admissible model. Thus the intersection of the set of all admissible models is the required smallest admissible model $\preceq_T$. For substitutivity, define

$$t \preceq_S t' \iff \forall \sigma : X \rightarrow \text{Trees}(X) . \ t[\sigma] \preceq_T t'[\sigma] .$$

(5)

Using the substitution $\sigma(x) = x$, we see that $\preceq_S \subseteq \preceq_T$. Conversely, it is easily shown that the relation $\preceq_S$ is itself an admissible model of $T$. Thus $\preceq_T \subseteq \preceq_S$. Since $\preceq_T$ and $\preceq_S$ coincide, (5) asserts the substitutivity of $\preceq_T$. The statement about compositionality now follows from Proposition 8. \hfill ▶

Given the proposition, we can use any effect theory to define an admissible and compositional basic operational preorder, namely the smallest admissible model over $\mathbb{N}$. We now apply this method to our running example of combined nondeterminism and probabilistic choice. The axioms are given in Figure 2.

The axioms include a special axiom for $\bot$, which is legitimate since $\bot$ is a tree, hence an expression. The axioms for probability include three standard equalities (each of which is given officially as two inequalities), and one Horn approximation axiom, $\text{Appr}$, which is separated out for the sake of Proposition 19 below. The axioms for nondeterminism are split into a neutral list, followed by further axioms for angelic and demonic nondeterminism respectively. Finally, there is a distributivity axiom that relates probabilistic and nondeterministic choice. Our two effect theories of interest are:

$$T_{\text{pr/ang}} = \text{Bot, Prob, Appr, Nondet, Ang, Dist}$$

$$T_{\text{pr/dem}} = \text{Bot, Prob, Appr, Nondet, Dem, Dist} .$$
model satisfies the Appr axiom if and only if it satisfies the equation

We outline the proof of the theorem in the demonic case, which we split into three lemmas.

Whence, by the Horn axiom,

We then define $\preceq_{\text{pr/ang}}^\text{ax}$ as an equational axiom, albeit one involving an infinitary expression.

It is easily checked that

Proof. We first derive $(1-2^{-\infty})x = x$, from the axioms with Appr. It is clear that $(1-2^{-n})x \leq x$ for every $n < \infty$, and therefore $(1-2^{-\infty})x \leq x$ by admissibility. We also have $x \text{pr}(1-2^{-n})x \leq (1-2^{-(n+1)})x$, and so, again by admissibility, $x \text{pr}(1-2^{-\infty})x \leq (1-2^{-\infty})x$.

Whence, by the Horn axiom, $x \leq (1-2^{-\infty})x$. We have thus derived $(1-2^{-\infty})x = x$.

For the converse, we assume $(1-2^{-\infty})x = x$ and derive Appr. Suppose $x \text{pr} y \leq y$. Then also $x \text{pr}(x \text{pr} y) \leq y$, and $x \text{pr}(x \text{pr}(x \text{pr} y)) \leq y$, etc. So also $x \text{pr} \bot \leq y$, and $x \text{pr}(x \text{pr} \bot) \leq y$, and $x \text{pr}(x \text{pr}(x \text{pr} \bot)) \leq y$, etc. That is, $(1-2^{-\infty})x \leq y$, for every $n < \infty$. By admissibility, $(1-2^{-\infty})x \leq y$. Whence, by the assumed axiom, $x \leq y$ as required.

7 The coincidence theorem

Our main theorem is that our operational, denotational and axiomatic preorders for combined probability and nondeterminism all coincide, in both the angelic and demonic cases.

Theorem 20 (Coincidence theorem).
1. The three preorders $\preceq_{\text{pr/ang}}^\text{op}$, $\preceq_{\text{pr/ang}}^\text{den}$ and $\preceq_{\text{pr/ang}}^\text{ax}$, for mixed probability and angelic nondeterminism, coincide.
2. Similarly, the preorders $\preceq_{\text{pr/dem}}^\text{op}$, $\preceq_{\text{pr/dem}}^\text{den}$ and $\preceq_{\text{pr/dem}}^\text{ax}$, for mixed probability and demonic nondeterminism, coincide.

We outline the proof of the theorem in the demonic case, which we split into three lemmas. The proof for the angelic case is similar.

Lemma 21. $\preceq_{\text{pr/dem}}^\text{ax} \subseteq \preceq_{\text{pr/dem}}^\text{op}$.

Proof. It is easily checked that $\preceq_{\text{pr/dem}}^\text{op}$ satisfies the axioms of $T_{\text{pr/dem}}$. Since $\preceq_{\text{pr/dem}}^\text{op}$ is admissible and $\preceq_{\text{pr/dem}}^\text{ax}$ is the smallest admissible model, $\preceq_{\text{pr/dem}}^\text{ax} \subseteq \preceq_{\text{pr/dem}}^\text{op}$.
We remark on the following aspect of the above result. The distributivity axiom Dist of Figure 2 is sometimes discussed as expressing that nondeterministic choices are resolved before probabilistic ones; see, e.g., [12, 7]. Such statements need careful interpretation. The definition of \( \preceq_{\text{pr/dem}} \), which is based on implementing nondeterministic schedulers as strategies for MDPs, explicitly allows the scheduler’s choices to take account of the outcomes of probabilistic choices that precede it. Nevertheless, the distributivity axiom is sound.

\[ \textbf{Lemma 22.} \quad \preceq_{\text{pr/dem}}^{\text{op}} = \preceq_{\text{pr/dem}}^{\text{den}}. \]

\[ \textbf{Proof.} \] We make use of the functional representation of \( \mathcal{SV}_{\leq 1} \mathbb{N} \) from [7] (see also [3]). For any topological space \( X \), we write \( \mathcal{L}(X) \) for the space of all lower semicontinuous functions from \( X \) to \([0, \infty]\) (i.e., functions that are continuous with respect to the Scott topology on \([0, \infty]\), and we endow \( \mathcal{L}(X) \) itself with the Scott topology. The space \( D' = \mathcal{L}(\mathcal{L}(\mathbb{N})) \) carries a continuous \( \Sigma_{\text{pr/nd}} \)-algebra structure

\[ (F \lor G)(f) = \min(F(f), G(f)) \quad (F \land G)(f) = \frac{1}{2}F(f) + \frac{1}{2}G(f). \]

(There is another \( \Sigma_{\text{pr/nd}} \)-algebra structure, relevant to angelic nondeterminism, in which \( \min \) is replaced with \( \max \).) Define \( j': \mathbb{N} \to D' \) by \( j'(n)(f) = f(n) \). This induces \( \|\cdot\|_{\text{pr/dem}} : \text{Trees}([\mathbb{N}]) \to D' \) satisfying \( \|\cdot\|_{\text{pr/dem}} \circ \Delta = j' \), as in Section 5. We show that \( \|h\|_{\text{pr/dem}}(t) = F_h(t) \), where \( F_h \) is defined as in Lemma 10. For this, the function \( t \mapsto (h \mapsto F_h(t)) \) is easily shown to be a \( \Sigma_{\text{pr/nd}} \)-algebra homomorphism satisfying \( F_h(i(n)) = j'(n) \). Moreover, it is continuous by Lemma 10. Thus it indeed coincides with \( \|\cdot\|_{\text{pr/dem}}' \). By the definition of \( F_h \), if follows that \( t \preceq_{\text{pr/dem}}^{\text{op}} t' \) if and only if \( \|t\|_{\text{pr/dem}}' \preceq \|t'\|_{\text{pr/dem}}' \).

Corollary 4.7 of [7] provides a functional representation of \( \mathcal{SV}_{\leq 1} X \) inside \( \mathcal{L}(\mathcal{L}(X)) \). In the case \( X = \mathbb{N} \), consider the function

\[ \Lambda : A \mapsto \left( f \mapsto \inf_{p \in A} E_p f \right) : \mathcal{SV}_{\leq 1} \mathbb{N} \to D'. \]

It is shown in [7] that \( \Lambda \) is a continuous \( \Sigma_{\text{pr/nd}} \)-algebra homomorphism, and also an order embedding (i.e., \( \Lambda(A) \leq \Lambda(B) \) implies \( A \leq B \)). By the uniqueness property of Proposition 3, it thus holds that \( \Lambda \circ \|\cdot\|_{\text{pr/dem}} = \|\cdot\|_{\text{pr/dem}}' \). We therefore have

\[ t \preceq_{\text{pr/dem}}^{\text{den}} t' \iff \|t\|_{\text{pr/dem}} \preceq \|t'\|_{\text{pr/dem}} \iff \|t\|_{\text{pr/dem}}' \leq \|t'\|_{\text{pr/dem}}' \iff t \preceq_{\text{pr/dem}}^{\text{ax}} t', \]

where the middle equivalence holds because \( \Lambda \) is an order embedding.

\[ \textbf{Lemma 23.} \quad \preceq_{\text{pr/dem}}^{\text{den}} \subseteq \preceq_{\text{pr/dem}}^{\text{ax}}. \]

\[ \textbf{Proof.} \] The proof proceeds in three steps.

1. Prove that both preorders coincide on probability trees (i.e., trees without or nodes).
2. Prove the inclusion of preorders for trees with a finite number of or nodes.
3. Use finite approximations and admissibility to conclude the general case.

We omit discussion of the first step, which is comparatively straightforward, cf. [4].

For step 2, suppose \( t \preceq_{\text{pr/dem}}^{\text{den}} t' \) where \( t, t' \) are trees with finitely many or nodes. For each of \( t, t' \), we use the distributivity axiom to rewrite the tree as an or-combination of finitely many (possibly infinite) probability trees. We then establish the following.

(a) If for every probability tree \( t'_i \) in \( t' \) there exists a corresponding tree \( t_i \) in \( t \) such that \( t_i \preceq_{\text{pr/dem}}^{\text{den}} t'_i \), then we have that \( t \preceq_{\text{pr/dem}}^{\text{ax}} t' \), using the Dem axiom, and step 1 above.
(b) The tree $t$ is equivalent in both preorder to $t \mathbin{or} k$, where $k = \lambda_1 t_1 + \cdots + \lambda_n t_n$ is any tree representing a convex combination of the probability trees of $t$. The tree $k$ is defined using infinite combinations of $pr$ nodes to assign the correct weight to each $t_i$.

(c) Making direct use of the definition of $SV \subseteq_1 N$, it follows from $t \mathbin{\leq_{\text{pr/dem}}} t'$ that, for every probability tree $t'_i$ of $t'$, there is a convex combination $k_i := \lambda_1 t'_1 + \cdots + \lambda_n t'_n$ of probability trees of $t$, such that $k_i \mathbin{\leq_{\text{pr/dem}}} t'_i$. To complete the argument for step 2, the tree $t'$ has the form $t'_1 \mathbin{or} \ldots \mathbin{or} t'_m$. By (c), there exist corresponding $k_1, \ldots, k_m$. By (b), $t$ is equivalent to $t \mathbin{or} k_1 \mathbin{or} \ldots \mathbin{or} k_m$. It now follows from (a) that $t \mathbin{\leq_{\text{pr/dem}}} t'$, by the property of the $k_j$ given by (c).

For step 3, suppose $t \mathbin{\leq_{\text{pr/dem}}} t'$, where $t, t'$ are arbitrary. Take approximating sequences $t = \bigcup_i t_i$ and $t' = \bigcup_i t'_i$, where both ascending sequences are composed of finite trees.

We use Definition 18 to further restrict the approximations of $t$. Using the finiteness of $t_i$, we have $[(1 - 2^{-i})t_i] \mathbin{\leq_{\text{pr/dem}}} t_i \mathbin{\leq_{\text{pr/dem}}} t'_i \mathbin{\leq_{\text{pr/dem}}} [(1 - 2^{-i})t'_i]$ in the way-below relation on $SV \subseteq_1 N$, via the explicit characterisation of this relation in [7]. Also, $(1 - 2^{-i})t_i$ is an ascending sequence of finite trees with $\bigcup_i (1 - 2^{-i})t_i = (1 - 2^{-\infty})t$.

For every $i$, we have $[(1 - 2^{-i})t_i] \mathbin{\leq_{\text{pr/dem}}} t_i \mathbin{\leq_{\text{pr/dem}}} t'_i \mathbin{\leq_{\text{pr/dem}}} [(1 - 2^{-i})t'_i]$. Since $[(1 - 2^{-i})t_i] \mathbin{\leq_{\text{pr/dem}}} [(1 - 2^{-i})t'_i]$, it follows from the way-below property that, for every $i$, $[(1 - 2^{-i})t_i] \mathbin{\leq_{\text{pr/dem}}} [(1 - 2^{-i})t'_i]$. By (c), there exist corresponding $k_1, \ldots, k_m$. By (b), $t$ is equivalent to $t \mathbin{or} k_1 \mathbin{or} \ldots \mathbin{or} k_m$. It now follows from (a) that $t \mathbin{\leq_{\text{pr/dem}}} t'$, by the property of the $k_j$ given by (c).

8 Related and future work

The results in this paper concern three methods of defining basic operational preorders on effect trees, which we claim to be a useful abstraction for defining contextual preorder for programming languages with algebraic effects. This has been verified for simple call-by-name [6] and call-by-value languages, but needs further substantiation.

The axiomatic approach to defining basic operational preorders in Section 6 is close in spirit to the algebraic axiomatisation of effects of Plotkin and Power [14], but with a different focus. In [14], (in)equational axiomatisations are required in order to determine a free-algebra monad modelling denotational equality of programs. Such axiomatisations have also been used to combine effects [5], and to induce a logic of effects [15]; but they have not hitherto been explicated as a method for defining contextual preorder/equivalence. In this paper, we have used infinitary Horn clause axioms between infinitary terms for this purpose, with the notion of admissible model playing an important role.

The main coincidence theorem in Section 7 has some precursors in the literature. The characterisations of $HV \subseteq_1 D$ and $SV \subseteq_1 D$ as free Kegelspitze in [7] can be viewed as completeness theorems for (in)equational axiomatisations with respect to domains $D$. In the special case $D = N$, this is implied by our results, for it can be derived from Lemma 23 that the partial-order quotients of Trees($N$) by $ax_{\text{pr/ang}}$ and $ax_{\text{pr/dem}}$ are isomorphic to $HV \subseteq_1 N$ and $SV \subseteq_1 N$. Another related completeness result is given in [12], where inequational axioms for a simple process algebra with nondeterministic and probabilistic choice are proved complete with respect to a domain-theoretic semantics. Translated into our setting, this process algebra corresponds to regular trees in a signature that combines the operations $or$, $pr$ with an additional prefix operation and zero constant. In [12], the semantics uses the convex powerdomain, rather than the upper $S$ and lower $H$ that we consider. In the present paper, we have not considered convex powerdomains and the associated neutral (as opposed to angelic or demonic) nondeterminism. However, it would be a natural extension to do so.
The main limitation we see of the present paper is the restriction throughout to admissible basic operational preorders. The admissibility condition plays a fundamental role in almost everything we do. It is, however, violated by some natural operational preorders; for example, for countable demonic nondeterminism. It is an open question how to incorporate such more general preorders into our theory.

References


