Logical Methods in Computer Science
Volume 15, Issue 1, 2019, pp. 26:1–26:48
https://lmcs.episciences.org/

Submitted Aug. 10, 2018
Published Feb. 20, 2020

COAXIOMS: FLEXIBLE COINDUCTIVE DEFINITIONS BY INFERENCE SYSTEMS

FRANCESCO DAGNINO

DIBRIS, University of Genova
e-mail address: francesco.dagnino@dibris.unige.it

Abstract. We introduce a generalized notion of inference system to support more flexible interpretations of recursive definitions. Besides axioms and inference rules with the usual meaning, we allow also *coaxioms*, which are, intuitively, axioms which can only be applied "at infinite depth" in a proof tree. Coaxioms allow us to interpret recursive definitions as fixed points which are not necessarily the least, nor the greatest one, whose existence is guaranteed by a smooth extension of classical results. This notion nicely subsumes standard inference systems and their inductive and coinductive interpretation, thus allowing formal reasoning in cases where the inductive and coinductive interpretation do not provide the intended meaning, but are rather mixed together.

This is a corrected version of the paper (`https://arxiv.org/abs/1808.02943v4`) published originally on 12 March 2019.

1. Introduction

Recursive definitions are everywhere in computer science. They allow very compact and intuitive definition of several types of objects: data types, predicates and functions. Furthermore, they are also essential in programming languages, especially for declarative paradigms, to write non-trivial programs.

Assigning a formal semantics to recursive definitions is not an easy task; usually, a recursive definition is associated with a monotone function on a partially ordered set, or, more generally, with a functor on a category, and the semantics is defined to be a *fixed point* of such function/functor [JR97]. However, in general, a monotone function (a functor) has several fixed points, hence the problem is how to choose the right one, that is, the fixed point that matches the intended semantics.

The most widely known semantics for recursive definitions is the *inductive* one [Acz77], which corresponds to the least fixed point/initial algebra. This interpretation works perfectly in all cases where we can reach a base case in a finite number of steps, this is the case, for instance, when we deal with well-founded (algebraic) objects (such as natural numbers, finite lists, finite trees etc.).

Nevertheless, in some cases the inductive interpretation is not appropriate. This is the case, for instance, when we deal with circular, or more generally non-well-founded (coalgebraic) objects (graphs, infinite lists, infinite trees, etc.), where clearly we are not

Key words and phrases: recursive definitions, coinduction, inference systems, fixed points, proof trees.

DOI:10.23638/LMCS-15(1:26)2019

guaranteed to reach a base case. Here a possibility is to choose the dual to induction: the *coinductive* semantics [Acz88, Rut00, Jac16], corresponding to the greatest fixed point/final coalgebra.

Therefore we have two strongly opposite options to interpret recursive definitions: the inductive (least) semantics or the coinductive (greatest) semantics. However, as we will see, there are cases where neither of these two dual solutions is suitable, hence the need of more flexibility to choose the desired fixed point.

On the programming language side, the most widely adopted semantics for recursive definition is again the inductive one. However, the support for coinductive semantics have been provided both in logic programming [SMBG06, SBMG07, KJ15] and in functional programming [Hag87, BW88]. In both cases, the same dichotomy described above emerges, hence, recently, some operational models to support more flexible definitions on non-well-founded structures have been proposed: in the logic paradigm [Anc13], in the functional paradigm [JKS13, JKS17] and in the object-oriented paradigm [AZ12, AZ13]

In this paper, we propose a framework to interpret recursive definitions as fixed points that are not necessarily the least, nor the greatest one. More precisely, we will extend the standard and well-known framework of *inference systems*, where recursive definitions are represented as sets of *rules*, as we will formally define in the next section.

In order to illustrate the complexity of interpreting recursive definitions especially in presence of non-well-founded structures, and to introduce the idea behind our proposal, let us consider some examples on lists of integers. In the following, l will range over finite or infinite lists and x, y, z over integers, Λ is the empty list and $-::-$ is the list constructor. We start with the simple predicate $\mathit{member}(x,\, l)$ stating that the element x belongs to l, defined as follows:

$$\frac{}{\mathit{member}(x,\, x::l)} \qquad\qquad \frac{\mathit{member}(x,\, l)}{\mathit{member}(x,\, y::l)}\, x \neq y$$

The standard way to interpret an inference system is the inductive one, which consists of the set of judgements having a finite derivation. For the above definition, the inductive interpretation works perfectly in all cases, also for infinite lists. Intuitively, this is due to the fact that in all cases, in order to establish that $\mathit{member}(x,\, l)$ holds, we have to find x in l, and, if x actually belongs to l, we find it in finitely many steps.

Let us consider another example: the predicate $\mathit{allPos}(l)$ stating that l contains only strictly positive natural numbers.

$$\frac{}{\mathit{allPos}(\Lambda)} \qquad\qquad \frac{\mathit{allPos}(l)}{\mathit{allPos}(x::l)}\, x > 0$$

Here the inductive interpretation still works well on finite lists, but fails on infinite lists, since, intuitively, to establish whether l contains only positive elements, we need to inspect the whole list, and this cannot be done with a finite derivation for an infinite list.

Therefore, we have to switch to the coinductive interpretation, considering as semantics the set of judgements having an arbitrary (finite or infinite) derivation. This is indeed the correct way to get the expected semantics also on infinite lists.

We now consider a slight variation of these two examples. Let $\mathbb{B} = \{\mathsf{T}, \mathsf{F}\}$ be the set of truth values, consider the judgements $\mathit{member}(x,\, l,\, b)$ and $\mathit{allPos}(l,\, b)$ with $b \in \mathbb{B}$ such that

- $\mathit{member}(x,\, l,\, \mathsf{T})$ holds iff $\mathit{member}(x,\, l)$ holds, and otherwise $\mathit{member}(x,\, l,\, \mathsf{F})$ holds
- $\mathit{allPos}(l,\, \mathsf{T})$ holds iff $\mathit{allPos}(l)$ holds, and otherwise $\mathit{allPos}(l,\, \mathsf{F})$ holds

We can define these judgements by means of the following inference systems

$$\frac{}{\mathit{member}(x,\, \Lambda,\, \mathsf{F})} \qquad \frac{}{\mathit{member}(x,\, x{::}l,\, \mathsf{T})} \qquad \frac{\mathit{member}(x,\, l,\, b)}{\mathit{member}(x,\, y{::}l,\, b)}\, x \neq y$$

$$\frac{}{\mathit{allPos}(\Lambda,\, \mathsf{T})} \qquad \frac{}{\mathit{allPos}(x{::}l,\, \mathsf{F})}\, x \leq 0 \qquad \frac{\mathit{allPos}(l,\, b)}{\mathit{allPos}(x{::}l,\, b)}\, x > 0$$

For both definitions, neither the inductive interpretation, nor the coinductive one works well on infinite lists. For the judgement $\mathit{member}(x,\, l,\, b)$, with the inductive interpretation we cannot derive any judgement of shape $\mathit{member}(x,\, l,\, \mathsf{F})$ where l is an infinite list and x does not belong to l, while with the coinductive interpretation we get both $\mathit{member}(x,\, l,\, \mathsf{F})$ and $\mathit{member}(x,\, l,\, \mathsf{T})$. For the judgement $\mathit{allPos}(l,\, b)$, with the inductive interpretation we cannot derive any judgement of shape $\mathit{allPos}(l,\, \mathsf{T})$ where l is an infinite list containing only positive elements, while with the coinductive interpretation we get both $\mathit{allPos}(l,\, \mathsf{T})$ and $\mathit{allPos}(l,\, \mathsf{F})$.

We consider now a last example, defining the predicate $\mathit{maxElem}(l,\, x)$ stating that x is the maximum of the list l. The definition is given by the following inference system

$$\frac{}{\mathit{maxElem}(x{::}\Lambda,\, x)} \qquad \frac{\mathit{maxElem}(l,\, y)}{\mathit{maxElem}(x{::}l,\, z)}\; z = \max\{x, y\}$$

The inductive interpretation works well on finite lists, but does not allow to derive any judgement on infinite lists, again, because, to compute a maximum, we need to inspect the whole list. The coinductive interpretation still works well on finite lists, but, again, we can derive too many judgements regarding infinite lists: for instance, if l is the infinite list of 1s, we can derive both $\mathit{maxElem}(l,\, 1)$, which is correct, and $\mathit{maxElem}(l,\, 2)$, that is clearly wrong, since 2 does not belong to l.

All these examples point out that the inductive interpretation cannot properly deal with non-well-founded structure, while the coinductive one allows the derivation of too many judgements. Hence we need a way to "filter out" some infinite derivations, in order to restrict the coinductive interpretation to the intended semantics. We make this possible by introducing *coaxioms*.

Coaxioms are special rules that need to be provided together with standard rules in order to control their semantics. Intuitively, they are axioms that can be only applied "at infinite depth" in a derivation. From a model-theoretic perspective, coaxioms allow one to choose as interpretation a fixed point that is not necessarily either the least or the greatest one. For instance, in the last three examples, the intended semantics is always a fixed point that lies between the least, that is undefined on infinite lists, and the greatest one, that is undetermined on them. In addition, we will also show that inductive and coinductive interpretations are particular cases of our extension, thus proving that it is actually a generalization. Another important feature is that in this framework we can interpret also inference systems where judgements that should be defined inductively and coinductively are mixed together in the same definition.

The notion of coaxiom has been inspired by some of the operational models mentioned above [AZ12, AZ13, Anc13] and, in our intention, this generalization of inference systems will serve as an abstract framework for a better understanding of these operational models, allowing formal reasoning on them.

The rest of the paper is organized as follows. In Section 2 we will recall some basic concepts regarding inference systems and we will introduce *inference systems with coaxioms*, informally explaining their semantics with a bunch of examples. The fixed point semantics for inference systems with coaxioms is formally defined in Section 3. There we will present closure and kernel systems, which are well-known notions on the power-set, in the more general setting of complete lattices, getting the definition of *bounded fixed point*, that will represent the semantics induced by coaxioms for an inference system. In Section 4 we will first formalize the notion of proof tree, which is the object representing a derivation, then we will introduce several equivalent semantics based on proof trees, that is, proof-theoretic semantics. Particularly interesting are the two characterizations exploiting the new concept of *approximated proof tree*, that will allow us to provide the semantics in terms of sequences of well-founded trees, without considering non-well-founded derivations. Proof techniques for coaxioms to prove both completeness and soundness of definitions will be discussed in Section 5. In particular, we will introduce the *bounded coinduction principle*, that is a generalization of the standard coinduction principle, aimed to show the completeness of a definition expressed in terms of an inference system with coaxioms. In Section 6, we will illustrate weaknesses and strengths of our framework, using various, more involved, examples. A straightforward and further extension of the framework is presented in Section 7, where we introduce *corules*. Finally, in Section 8 related work is summarized and Section 9 concludes the work.

This paper is extracted from my master thesis [Dag17], and presents in more detail the work we have done in [ADZ17b]. Notably, here we discuss closures and kernels from a more general point of view (see Section 3.1), in order to better frame the bounded fixed point in lattice theory. Furthermore, thanks to a more formal treatment of proof trees, we introduce an additional proof-theoretic characterization, using approximated proof trees (see Theorem 4.17). We also present another example of application of coaxioms to graphs (see Section 6.3). With respect to [Dag17], here we briefly present a straightforward further extension of the framework, considering also *corules* (see Section 7). We only briefly mention corules, because there are still few and quite involved examples where they seem to be really needed (one can be found in [ADZ18]), hence restricting ourselves to coaxioms allows us to provide a clearer presentation.

2. Inference systems with coaxioms

First of all, we recall some standard notions about inference systems [Acz77, San11]. In the following, assume a set $\mathcal{U}$, called *universe*, whose elements are named *judgements*. An *inference system* $\mathcal{I}$ consists of a set of *inference rules*, which are pairs $\frac{Pr}{c}$, with $Pr \subseteq \mathcal{U}$ the set of *premises*, $c \in \mathcal{U}$ the *consequence* (a.k.a. *conclusion*).

The intuitive interpretation of a rule is that if the premises in Pr hold then the consequence c should hold as well. In particular, an *axiom* is (the consequence of) a rule with empty set of premises, which necessarily holds.

A *proof tree*[1] is a tree whose nodes are (labeled by) judgements in $\mathcal{U}$, and there is a node c with set of children Pr only if there exists a rule $\frac{Pr}{c} \in \mathcal{I}$. We say that a judgement $j \in \mathcal{U}$ *has a proof tree* if it is the root of some proof tree. The *inductive interpretation* of $\mathcal{I}$ is the set of judgements having a well-founded proof tree, while the *coinductive interpretation* of $\mathcal{I}$ is the set of judgements having an arbitrary (well-founded or not) proof tree. It can be

[1] See Section 4 for a more formal account of proof trees.

$$\begin{tikzcd} a \arrow[r, bend left] & b \arrow[l, bend left] & c \end{tikzcd}$$

$$\frac{b \stackrel{\star}{\to} \mathcal{N}}{a \stackrel{\star}{\to} \{a\} \cup \mathcal{N}} \qquad \frac{a \stackrel{\star}{\to} \mathcal{N}}{b \stackrel{\star}{\to} \{b\} \cup \mathcal{N}} \qquad \frac{}{c \stackrel{\star}{\to} \{c\}} \qquad \overline{\overline{a \stackrel{\star}{\to} \emptyset}} \qquad \overline{\overline{b \stackrel{\star}{\to} \emptyset}} \qquad \overline{\overline{c \stackrel{\star}{\to} \emptyset}}$$

Figure 1: Concrete graph example

shown that these definitions are equivalent to standard fixed point semantics, which we will discuss later.

2.1. **A gentle introduction.** We introduce now our generalization of inference systems. An *inference system with coaxioms* is a pair $(\mathcal{I}, \gamma)$ where $\mathcal{I}$ is an inference system and $\gamma \subseteq \mathcal{U}$ is a set of *coaxioms*. A coaxiom $c \in \gamma$ will be written as $\overline{\overline{c}}$, very much like an axiom, and, analogously to an axiom, it can be used as initial assumption to derive other judgements. However, coaxioms will be used in a special way, that is, intuitively they can be used only "at infinite depth" in a derivation. This will allow us to impose an initial assumption also to infinite proof trees, that otherwise are not required to have a base case. We will make precise this notion in next sections, now we will present some examples to illustrate how to use coaxioms to govern the semantics of an inference system.

As we are used to doing for rules, we will express sets of coaxioms by means of *meta-coaxioms* with side conditions.

Let us start with an introductory example concerning graphs, that are a widely used non-well-founded data type. Consider a graph (V, adj) where V is the set of nodes and $adj\colon V \to \wp(V)$ is the adjacency function, that is, for each node $v \in V$, $adj(v)$ is the set of nodes adjacent to v. We want to define the judgement $v \stackrel{\star}{\to} \mathcal{N}$ stating that $\mathcal{N}$ is the set of nodes reachable from v.

We define this judgement with the following (meta-)rule and (meta-)coaxiom:

$$\frac{v_1 \stackrel{\star}{\to} \mathcal{N}_1 \quad \dots \quad v_k \stackrel{\star}{\to} \mathcal{N}_k}{v \stackrel{\star}{\to} \{v\} \cup \mathcal{N}_1 \cup \ldots \cup \mathcal{N}_k}\, adj(v) = \{v_1, \ldots, v_k\} \qquad\qquad \overline{\overline{v \stackrel{\star}{\to} \emptyset}}\, v \in V$$

To be more concrete, we consider the graph drawn in Figure 1, whose corresponding rules are reported in the same figure.

Let us ignore for a moment coaxioms and reason about the standard interpretations. It is clear that, if we interpret the system inductively, we will only prove the judgement $c \stackrel{\star}{\to} \{c\}$, because it is the only axiom and other rules do not depend on it. In other words, the judgement $v \stackrel{\star}{\to} \mathcal{N}$, like other judgements on graphs, cannot be defined inductively by structural recursion, since the structure is not well-founded. In particular, the problem are cycles, where the proof may be trapped, continuously unfolding the structure of the graph without ever reaching a base case. Usual implementations of visits on graphs rely on imperative features and correct this issue by marking already visited nodes. In this way, they avoid visiting twice the same node, actually breaking cycles.

On the other hand, if we interpret the meta-rules coinductively (excluding again the coaxioms), then we get the correct judgements $a \stackrel{\star}{\to} \{a, b\}$ and $b \stackrel{\star}{\to} \{a, b\}$, but we also get the wrong judgements $a\xrightarrow{\star}\{a,b,c\}$ and $b\xrightarrow{\star}\{a,b,c\}$, as shown by the following derivations

$$\dfrac{\dfrac{\dfrac{\vdots}{a\xrightarrow{\star}\{a,b\}}}{b\xrightarrow{\star}\{a,b\}}}{a\xrightarrow{\star}\{a,b\}} \qquad \dfrac{\dfrac{\dfrac{\vdots}{b\xrightarrow{\star}\{a,b\}}}{a\xrightarrow{\star}\{a,b\}}}{b\xrightarrow{\star}\{a,b\}} \qquad \dfrac{\dfrac{\dfrac{\vdots}{a\xrightarrow{\star}\{a,b,c\}}}{b\xrightarrow{\star}\{a,b,c\}}}{a\xrightarrow{\star}\{a,b,c\}} \qquad \dfrac{\dfrac{\dfrac{\vdots}{b\xrightarrow{\star}\{a,b,c\}}}{a\xrightarrow{\star}\{a,b,c\}}}{b\xrightarrow{\star}\{a,b,c\}}$$

As already said, coaxioms allow us to impose additional conditions on proof trees to be considered valid: the semantics of an inference system with coaxioms $(\mathcal{I},\, \gamma)$ is the set of the judgements having

(1) an arbitrary (well-founded or not) proof tree t in $\mathcal{I}$ such that
(2) each node of t has a well-founded proof tree in $\mathcal{I}_{\sqcup\gamma}$

where $\mathcal{I}_{\sqcup\gamma}$ is the inference system obtained enriching $\mathcal{I}$ by judgements in γ considered as axioms. Hence, we can use coinduction thanks to 1, but we use coaxioms to restrict its usage, by *filtering out* undesired proof trees.

Note that for nodes in t, which are roots of a well-founded subtree, the second condition always holds (a well-founded proof tree in $\mathcal{I}$ is a well-founded proof tree in $\mathcal{I}_{\sqcup\gamma}$ as well), hence it is only significant for nodes which are roots of an infinite path in the proof tree.

For instance, in the example in Figure 1, the judgement $a\xrightarrow{\star}\{a,b\}$ has an infinite proof tree in $\mathcal{I}$ where each node has a finite proof tree in $\mathcal{I}_{\sqcup\gamma}$, as shown below.

$$\begin{array}{l|l} \mathcal{I} & \mathcal{I}_{\sqcup\gamma} \\ \dfrac{\dfrac{\dfrac{\vdots}{a\xrightarrow{\star}\{a,b\}}}{b\xrightarrow{\star}\{a,b\}}}{a\xrightarrow{\star}\{a,b\}} & \dfrac{\dfrac{\dfrac{}{a\xrightarrow{\star}\emptyset}}{b\xrightarrow{\star}\{b\}}}{a\xrightarrow{\star}\{a,b\}} \qquad \dfrac{\dfrac{\dfrac{}{b\xrightarrow{\star}\emptyset}}{a\xrightarrow{\star}\{a\}}}{b\xrightarrow{\star}\{a,b\}} \end{array}$$

On the other hand, the judgement $a\xrightarrow{\star}\{a,b,c\}$ has no finite proof tree in $\mathcal{I}_{\sqcup\gamma}$, because c is not reachable from a; hence such judgement is not derivable in $(\mathcal{I},\, \gamma)$, as expected.

We mentioned before an alternative view of the condition imposed by coaxioms on infinite proof trees: in an infinite derivation coaxioms can only be used "at infinite depth". The formal counterpart of this sentence is that the infinite proof tree can be approximated, for all $n \geq 0$, by a well-founded proof tree in $\mathcal{I}_{\sqcup\gamma}$ where coaxioms can only be used at depth greater than n. Hence, in a sense, the infinite proof tree is obtained by "pushing" coaxioms to infinity.

For instance, the infinite proof tree in $\mathcal{I}$ for the judgement $a\xrightarrow{\star}\{a,b\}$ shown above can be approximated, for any $n \geq 0$, by a finite proof tree in $\mathcal{I}_{\sqcup\gamma}$ where coaxioms are used at depth greater than n, as shown below.

$$\dfrac{\dfrac{\dfrac{}{a\xrightarrow{\star}\emptyset}}{b\xrightarrow{\star}\{b\}}}{a\xrightarrow{\star}\{a,b\}} \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{b\xrightarrow{\star}\emptyset}}{a\xrightarrow{\star}\{a\}}}{b\xrightarrow{\star}\{a,b\}}}{a\xrightarrow{\star}\{a,b\}} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{a\xrightarrow{\star}\emptyset}}{b\xrightarrow{\star}\{b\}}}{a\xrightarrow{\star}\{a,b\}}}{b\xrightarrow{\star}\{a,b\}}}{a\xrightarrow{\star}\{a,b\}} \qquad \ldots$$

This does not hold, instead, for $a\xrightarrow{\star}\{a,b,c\}$, since it has no finite derivation using coaxioms.

As a second example, we consider the definition of the *first* sets in a grammar. Let us represent a context-free grammar by its set of terminals T, its set of non-terminals N, and all the productions $A ::= \beta_1 \mid \ldots \mid \beta_n$ with left-hand side A, for each non-terminal A. Recall that, for each $\alpha \in (T \cup N)^+$, we can define the set $\mathit{first}(\alpha) = \{\sigma \mid \sigma \in T, \alpha \to^\star \sigma\beta\}$. Informally, $\mathit{first}(\alpha)$ is the set of the initial terminal symbols of the strings which can be derived from a string α in 0 or more steps.

We defines the judgement $\mathit{first}(\alpha, \mathcal{F})$ by the following inference system with coaxioms, where $\mathcal{F} \subseteq T$.

$$\frac{}{\mathit{first}(\sigma\alpha, \{\sigma\})}\,\sigma \in T \qquad \frac{\mathit{first}(A, \mathcal{F})}{\mathit{first}(A\alpha, \mathcal{F})}\,\begin{array}{l}A \in N\\ A \not\to^\star \epsilon\end{array} \qquad \frac{\mathit{first}(A, \mathcal{F}) \qquad \mathit{first}(\alpha, \mathcal{F}')}{\mathit{first}(A\alpha, \mathcal{F} \cup \mathcal{F}')}\,\begin{array}{l}A \in N\\ A \to^\star \epsilon\end{array}$$

$$\frac{}{\mathit{first}(\epsilon, \emptyset)} \qquad \frac{\mathit{first}(\beta_1, \mathcal{F}_1) \qquad \cdots \qquad \mathit{first}(\beta_n, \mathcal{F}_n)}{\mathit{first}(A, \mathcal{F}_1 \cup \cdots \cup \mathcal{F}_n)}\,A ::= \beta_1 \mid \ldots \mid \beta_n \qquad \overline{\overline{\mathit{first}(A, \emptyset)}}\,A \in N$$

The rules of the inference system correspond to the natural recursive definition of *first*. Note, in particular, that in a string of shape $A\alpha$, if the non-terminal A is *nullable*, that is, we can derive from it the empty string, then the *first* set for $A\alpha$ should also include the *first* set for α.

As in the previous example on graphs, the problem with this recursive definition is that, since the non-terminals in a grammar can mutually refer to each other, the function defined by the inductive interpretation can be undefined, since it may never reach a base case. That is, a naive top-down implementation might not terminate. For this reason, *first* sets are typically computed by an imperative bottom-up algorithm, or the top-down implementation is corrected by marking already encountered non-terminals, analogously to what is done for visiting graphs. Again as in the previous example, the coinductive interpretation may fail to be a function, whereas, with the coaxioms, we get the expected result.

Let us now consider some examples of judgements concerning lists. We consider arbitrary (finite or infinite) lists of integers and denote by $\mathbb{L}^\infty$ the set of such lists. We first consider the judgement $\mathit{maxElem}(l,\, x)$, with $l \in \mathbb{L}^\infty$ and $x \in \mathbb{Z}$, stating that x is the maximum element that occurs in l. This judgement has a natural definition by structural recursion we have discussed in Section 1 where we have shown that neither inductive nor coinductive interpretations are able to capture the expected semantics. Therefore, in the following definition we have added a coaxiom to the inference system from Section 1 in order to restrict the coinductive interpretation.

$$\frac{}{\mathit{maxElem}(x{::}\Lambda,\, x)} \qquad \frac{\mathit{maxElem}(l,\, y)}{\mathit{maxElem}(x{::}l,\, z)}\,z = \max\{x, y\} \qquad \overline{\overline{\mathit{maxElem}(x{::}l,\, x)}}$$

Recall that the problem with the coinductive interpretation is that it accepts all judgements $\mathit{maxElem}(l,\, x)$ where x is an upper bound of l, even if it does not occur in l. The coaxiom, thanks to the way it is used, imposes that $\mathit{maxElem}(l,\, x)$ may hold only if x appears somewhere in the list, hence undesired proofs are filtered out.

A similar example is given by the judgement $\mathit{elems}(l,\, xs)$ where $l \in \mathbb{L}^\infty$ and $xs \subseteq \mathbb{Z}$, stating that xs is the carrier of the list l, that is, the set of all elements appearing in l. This judgement can be defined using coaxioms as follows:

$$\frac{}{\mathit{elems}(\Lambda,\, \emptyset)} \qquad \frac{\mathit{elems}(l,\, xs)}{\mathit{elems}(x{::}l,\, \{x\} \cup xs)} \qquad \overline{\overline{\mathit{elems}(l,\, \emptyset)}}$$

If we ignore the coaxiom and interpret the system coinductively, then we can prove $\mathit{elems}(l,\, \mathit{xs})$ for any superset xs of the carrier of l if l is infinite. The coaxioms again allow us to filter out undesired derivations. For instance, for l the infinite list of 1s, any judgement $\mathit{elems}(l,\, \mathit{xs})$ with $1 \in \mathit{xs}$ can be derived. Indeed, for any such judgement we can construct an infinite proof tree which is a chain of applications of the last meta-rule. With the coaxioms, we only consider the infinite trees where the node $\mathit{elems}(l,\, \mathit{xs})$ has a finite proof tree in the inference system enriched by the coaxioms. This is only true for $\mathit{xs} = \{1\}$.

Using coaxioms, we can get the right semantics also for other examples on lists discussed in Section 1, in particular definitions for predicates $\mathit{member}(x,\, l,\, b)$ and $\mathit{allPos}(l,\, b)$ are reported below.

$$\frac{}{\mathit{member}(x,\, \Lambda,\, \mathsf{F})} \qquad \frac{}{\mathit{member}(x,\, x{::}l,\, \mathsf{T})} \qquad \frac{\mathit{member}(x,\, l,\, b)}{\mathit{member}(x,\, y{::}l,\, b)}\, x \neq y \qquad \frac{}{\mathit{member}(x,\, l,\, \mathsf{F})}$$

$$\frac{}{\mathit{allPos}(\Lambda,\, \mathsf{T})} \qquad \frac{}{\mathit{allPos}(x{::}l,\, \mathsf{F})}\, x \leq 0 \qquad \frac{\mathit{allPos}(l,\, b)}{\mathit{allPos}(x{::}l,\, b)}\, x > 0 \qquad \frac{}{\mathit{allPos}(l,\, \mathsf{T})}$$

In Section 1 we said that the standard coinductive interpretation allows us to prove too many judgements. For instance, if l is the infinite list of 1s, hence $l = 1{::}l^2$, the following are valid infinite derivations, obtained repeatedly applying the only rule with non-empty premises

$$\cfrac{\cfrac{\vdots}{\mathit{member}(2,\, l,\, \mathsf{T})}}{\mathit{member}(2,\, l,\, \mathsf{T})} \qquad \cfrac{\cfrac{\vdots}{\mathit{member}(2,\, l,\, \mathsf{F})}}{\mathit{member}(2,\, l,\, \mathsf{F})} \qquad \cfrac{\cfrac{\vdots}{\mathit{allPos}(l,\, \mathsf{T})}}{\mathit{allPos}(l,\, \mathsf{T})} \qquad \cfrac{\cfrac{\vdots}{\mathit{allPos}(l,\, \mathsf{F})}}{\mathit{allPos}(l,\, \mathsf{F})}$$

In the semantics induced by coaxioms, only the second and the third proof trees are valid, since their nodes are derivable by a finite proof tree using also coaxioms, while this fact is not true for the other derivations.

2.2. **Semantics.** We now define the model-theoretic semantics for inference systems with coaxioms. First of all we recall some notions for standard inference systems. Consider an inference system $\mathcal{I}$; the *(one step) inference operator* $F_{\mathcal{I}} : \wp(\mathcal{U}) \to \wp(\mathcal{U})$ associated with $\mathcal{I}$ is defined by

$$F_{\mathcal{I}}(S) = \left\{ c \in \mathcal{U} \mid \mathit{Pr} \subseteq S,\, \frac{\mathit{Pr}}{c} \in \mathcal{I} \right\}$$

That is, $F_{\mathcal{I}}(S)$ is the set of judgements that can be inferred (in one step) from the judgements in S using the inference rules in $\mathcal{I}$. Note that this set always includes axioms.

A set S is *closed* if $F_{\mathcal{I}}(S) \subseteq S$, and *consistent* if $S \subseteq F_{\mathcal{I}}(S)$. That is, no new judgements can be inferred from a closed set, and all judgements in a consistent set can be inferred from the set itself.

The *inductive interpretation* of $\mathcal{I}$, denoted $\mathit{Ind}(\mathcal{I})$, is the smallest closed set, that is, the intersection of all closed sets, and the *coinductive interpretation* of $\mathcal{I}$, denoted $\mathit{CoInd}(\mathcal{I})$, is the largest consistent set, that is, the union of all consistent sets. Both interpretations are well-defined and can be equivalently expressed as the least (respectively, the greatest) fixed point of the inference operator. These definitions can be shown to be equivalent to the proof-theoretic characterizations introduced before, see [LG09, Dag17].

[2]It is well-known that an infinite term can be represented by a set of recursive equations, see, e.g., [AMV06].

For particular inference systems, we can also compute $Ind(\mathcal{I})$ and $CoInd(\mathcal{I})$ iteratively, see e.g. [San11]. More precisely, if all rules in $\mathcal{I}$ have a finite set of premises, then $Ind(\mathcal{I}) = \bigcup\{F_{\mathcal{I}}^n(\emptyset) \mid n \in \mathbb{N}\}$, and, if for each judgement c there is a finite set of rules having c as conclusion, then $CoInd(\mathcal{I}) = \bigcap\{F_{\mathcal{I}}^n(\mathcal{U}) \mid n \in \mathbb{N}\}$. This happens because, under the former condition, $F_{\mathcal{I}}$ is upward continuous, and, under the latter condition, $F_{\mathcal{I}}$ is downward continuous (see page 14 for a formal definition of upward/downward continuity).

Given an inference systems with coaxioms $(\mathcal{I}, \gamma)$, we can construct the *interpretation generated by coaxioms*, denoted by $Gen(\mathcal{I}, \gamma)$, by the following two steps:

(1) First, we consider the inference system $\mathcal{I}_{\sqcup\gamma}$ obtained enriching $\mathcal{I}$ by judgements in γ considered as axioms, and we take its inductive interpretation $Ind(\mathcal{I}_{\sqcup\gamma})$.
(2) Then, we take the coinductive interpretation of the inference system obtained from $\mathcal{I}$ by keeping only rules with consequence in $Ind(\mathcal{I}_{\sqcup\gamma})$, that is, we define

$$Gen(\mathcal{I}, \gamma) = CoInd(\mathcal{I}_{\sqcap Ind(\mathcal{I}_{\sqcup\gamma})})$$

where $\mathcal{I}_{\sqcap S}$, with $\mathcal{I}$ inference system and $S \subseteq \mathcal{U}$, denotes the inference system obtained from $\mathcal{I}$ by keeping only rules with consequence in S, that is, $\mathcal{I}_{\sqcap S} = \{\frac{Pr}{c} \in \mathcal{I} \mid c \in S\}$.

If we consider again the example of the graph in Figure 1, since the universe is finite, every monotone function is continuous, hence we can compute fixed points iteratively. Therefore, in the first phase, we obtain the following judgements (the number at the beginning of each line indicates the iteration step):

(1) $a \xrightarrow{\star} \emptyset$, $b \xrightarrow{\star} \emptyset$, $c \xrightarrow{\star} \emptyset$, $c \xrightarrow{\star} \{c\}$
(2) $a \xrightarrow{\star} \emptyset$, $b \xrightarrow{\star} \emptyset$, $c \xrightarrow{\star} \emptyset$, $c \xrightarrow{\star} \{c\}$, $a \xrightarrow{\star} \{a\}$, $b \xrightarrow{\star} \{b\}$
(3) $a \xrightarrow{\star} \emptyset$, $b \xrightarrow{\star} \emptyset$, $c \xrightarrow{\star} \emptyset$, $c \xrightarrow{\star} \{c\}$, $a \xrightarrow{\star} \{a\}$, $b \xrightarrow{\star} \{b\}$, $a \xrightarrow{\star} \{a, b\}$, $b \xrightarrow{\star} \{a, b\}$

The last set is closed, hence it is $Ind(\mathcal{I}_{\sqcup\gamma})$.

For the second phase, first of all we have to construct the inference system $\mathcal{I}_{\sqcap Ind(\mathcal{I}_{\sqcup\gamma})}$, whose rules are those of $\mathcal{I}$ (in Figure 1) with conclusion in $Ind(\mathcal{I}_{\sqcup\gamma})$, computed above. Hence, they are the following:

$$\frac{b \xrightarrow{\star} \mathcal{N}}{a \xrightarrow{\star} \{a\} \cup \mathcal{N}}\ c \notin \mathcal{N} \qquad \frac{a \xrightarrow{\star} \mathcal{N}}{b \xrightarrow{\star} \{b\} \cup \mathcal{N}}\ c \notin \mathcal{N} \qquad \frac{}{c \xrightarrow{\star} \{c\}}$$

These rules have to be interpreted coinductively, hence each iteration of the inference operator removes judgements which cannot be inferred from the set obtained from the previous iteration step, that is, we get:

(1) $c \xrightarrow{\star} \{c\}$, $a \xrightarrow{\star} \{a\}$, $b \xrightarrow{\star} \{b\}$, $a \xrightarrow{\star} \{a, b\}$, $b \xrightarrow{\star} \{a, b\}$
(2) $c \xrightarrow{\star} \{c\}$, $a \xrightarrow{\star} \{a, b\}$, $b \xrightarrow{\star} \{a, b\}$

This last set is consistent, hence it is $Gen(\mathcal{I}, \gamma)$, and it is indeed the expected result.

In next sections, we will study properties of $Gen(\mathcal{I}, \gamma)$ in a more formal way, notably, we will show that it is actually a fixed point of the inference operator $F_{\mathcal{I}}$ as expected (see Section 3). Such a fixed point will be constructed by taking the greatest consistent subset of the smallest closed superset of the set of coaxioms. Then, we will also prove that such semantics corresponds to the proof-theoretic characterization informally introduced at the beginning of this section (see Section 4).

3. Fixed point semantics for coaxioms

As mentioned in Section 2.2, we can associate with an inference system $\mathcal{I}$ a monotone function $F_{\mathcal{I}}$ on the power-set lattice. We always require the semantics of $\mathcal{I}$ to be a fixed point of $F_{\mathcal{I}}$, hence we aim to show that this property indeed holds for $Gen(\mathcal{I}, \gamma)$.

In this section, we will develop the theory needed for this result and some important consequences. In order to construct the fixed point we need, we work in the general framework of *lattice theory* [Nat98, DP02], so that we can highlight only the essential structure. More precisely, in Section 3.1 we discuss closure and kernel operators, presenting almost standard results, for which, however, we have not found a complete enough discussion in literature [Nat98, DP02]. Then, in Section 3.2 and Section 3.3, we define the *bounded fixed point*, showing it corresponds to the interpretation generated by coaxioms and it subsumes both inductive and coinductive interpretations.

Let us start by recalling some basic definitions about lattices. A *complete lattice* is a partially ordered set $(L, \sqsubseteq)$ where all subsets $A \subseteq L$ have a least upper bound (a.k.a. *join*), denoted by $\bigsqcup A$. In particular, in L there are both a top element $\top = \bigsqcup L$ and a bottom element $\bot = \bigsqcup \emptyset$. Furthermore, it can be proved that in L all subsets $A \subseteq L$ have also a greatest lower bound (a.k.a. *meet*), denoted by $\bigsqcap A$. In the following, for all $x, y \in L$, we will write $x \sqcup y$ for the binary join and $x \sqcap y$ for the binary meet, that is, respectively, $\bigsqcup\{x, y\}$ and $\bigsqcap\{x, y\}$, respectively.

Given a function $F : L \to L$ and an element $x \in L$, we say that

- x is a *pre-fixed point* if $F(x) \sqsubseteq x$
- x is a *post-fixed point* if $x \sqsubseteq F(x)$
- x is a *fixed point* if $x = F(x)$

We will denote by $\mathsf{pre}(F)$, $\mathsf{post}(F)$ and $\mathsf{fix}(F)$, respectively, the sets of pre-fixed, post-fixed and fixed points of F.

We also say that F is *monotone* if, for all $x, y \in L$, if $x \sqsubseteq y$ then $F(x) \sqsubseteq F(y)$. Monotone functions over a complete lattice are particularly interesting since, thanks to the Knaster-Tarski theorem [Tar55, LNS82], we know that they have both the least and the greatest fixed point, that we denote by μF and νF, respectively.

In the following we will assume a complete lattice $(L, \sqsubseteq)$ and a monotone function $F : L \to L$.

3.1. Closures and kernels.

We start by introducing some notions which are slight generalizations of concepts that can be found in [AJ94, Nat98].

Definition 3.1. Let $(L, \sqsubseteq)$ be a complete lattice. Then

(1) a subset $C \subseteq L$ is a *closure system* if, for any subset $X \subseteq C$, $\bigsqcap X \in C$
(2) a subset $K \subseteq L$ is a *kernel system* if, for any subset $X \subseteq K$, $\bigsqcup X \in K$

Note that, with the usual convention that $\bigsqcup \emptyset = \bot$ and $\bigsqcap \emptyset = \top$, we have that, for all closure systems $C \subseteq L$, $\top \in C$, and, for all kernel systems $K \subseteq L$, $\bot \in K$.

This definition provides a general order-theoretic account of a kind of structures that are very frequent in mathematics, in particular considering the complete lattice carried by the power-set. For instance, given a group G, the set $\mathsf{Sub}(G) \subseteq \wp(G)$ of all subgroups of G is closed under arbitrary intersections, that is, under the meet operation in the power-set lattice $(\wp(G), \subseteq)$. Hence, $\mathsf{Sub}(G)$ is a closure system in the complete lattice $(\wp(G), \subseteq)$, according to the above definition. It is easy to see that this fact also holds for any algebraic structure. Another example comes from topology. Indeed, given a topological space (X, τ), by definition $\tau \subseteq \wp(X)$ and is closed under arbitrary unions, hence τ is a kernel system with respect to the complete lattice $(\wp(X), \subseteq)$. Moreover, the set of closed sets in the topological space (X, τ), that is, the set $\{X \setminus A \mid A \in \tau\}$, is closed under arbitrary intersections, hence it is a closure system in $(\wp(X), \subseteq)$. Actually this is a general fact: if $K \subseteq \wp(X)$ is a kernel system, then $\{X \setminus A \mid A \in K\}$ is a closure system. Also the converse is true.

It is quite easy to check that the following proposition holds

Proposition 3.2. *Let $(L, \sqsubseteq)$ be a complete lattice and $F : L \to L$ a monotone function. Then*
(1) *$\mathsf{pre}(F)$ is a closure system*
(2) *$\mathsf{post}(F)$ is a kernel system*

Proof. We only prove 1, since 2 can be obtained by duality.
Let $A \subseteq \mathsf{pre}(F)$ be a set of pre-fixed points of F. We have that, for all $x \in A$, $\bigsqcap A \sqsubseteq x$ (by definition of greatest lower bound), then $F(\bigsqcap A) \sqsubseteq F(x) \sqsubseteq x$ (since F is monotone and x is pre-fixed), hence, finally, $F(\bigsqcap A) \sqsubseteq \bigsqcap A$ (by definition of greatest lower bound). □

This observation provides us with a canonical way to associate a closure and a kernel system with a monotone function. Let us introduce another notion.

Definition 3.3. Let $(L, \sqsubseteq)$ be a complete lattice. Then
(1) A monotone function $\nabla : L \to L$ is a *closure operator* if it satisfies the following conditions:
   - for all $x \in L$, $x \sqsubseteq \nabla(x)$
   - for all $x \in L$, $\nabla(\nabla(x)) = \nabla(x)$
(2) A monotone function $\Delta : L \to L$ is a *kernel operator* if it satisfies the following conditions:
   - for all $x \in L$, $\Delta(x) \sqsubseteq x$
   - for all $x \in L$, $\Delta(\Delta(x)) = \Delta(x)$

Note that, since a closure operator $\nabla : L \to L$ is a monotone function, we can associate with it both a closure and a kernel system, $\mathsf{pre}(\nabla)$ and $\mathsf{post}(\nabla)$. However, by the first condition of the definition of closure operator, we get that $\mathsf{post}(\nabla) = L$, hence it is not interesting, and $\mathsf{pre}(\nabla) = \mathsf{fix}(\nabla)$. Dually, for a kernel operator $\Delta : L \to L$, only $\mathsf{post}(\Delta) = \mathsf{fix}(\Delta)$ is interesting, because $\mathsf{pre}(\Delta) = L$. Therefore, we can say that every closure operator naturally induces a closure system and every kernel operator naturally induces a kernel system.

The next result shows how we can build, in a canonical way, a closure/kernel operator from a closure/kernel system.

Theorem 3.4. *Let $(L, \sqsubseteq)$ be a complete lattice. Then*
(1) *given a closure system $C \subseteq L$ the function*

$$\nabla_C(x) = \bigsqcap\{y \in C \mid x \sqsubseteq y\}$$

is a closure operator such that $\mathsf{fix}(\nabla_C) = C$
(2) *given a kernel system $K \subseteq L$ the function*

$$\Delta_K(x) = \bigsqcup\{y \in K \mid y \sqsubseteq x\}$$

is a kernel operator such that $\mathsf{fix}(\Delta_K) = K$

Proof. We prove only point 1, the point 2 can be obtained by duality. We first prove that ∇_C is monotone. Consider $x, y \in L$ such that $x \sqsubseteq y$, hence $\{z \in C \mid y \sqsubseteq z\} \subseteq \{z \in C \mid x \sqsubseteq z\}$, thus $\nabla_C(x) \sqsubseteq \nabla_C(y)$, since the greatest lower bound is a monotone operator.[3]
The fact that $x \sqsubseteq \nabla_C(x)$ for all $x \in L$ follows from the fact that x is a lower bound of the set $\{y \in C \mid x \sqsubseteq y\}$.
Finally, note that by definition, for all $x \in L$, $\nabla_C(x) \in C$, because C is a closure system, hence in order to show that $\nabla_C(\nabla_C(x)) = \nabla_C(x)$ it is enough to show that, for all $z \in C$, $\nabla_C(z) = z$, namely, $C \subseteq \mathsf{fix}(\nabla_C)$. So consider $z \in C$, we have already shown that $z \sqsubseteq \nabla_C(z)$, thus we have only to show the other inequality. Since $z \in C$, $z \in \{y \in C \mid z \sqsubseteq y\}$, and this implies that $\nabla_C(z) \sqsubseteq z$.

This shows that ∇_C is a closure operator. Actually we have also proved that $C \subseteq \mathsf{fix}(\nabla_C)$. Therefore, to conclude the proof it remains to show that $\mathsf{fix}(\nabla_C) \subseteq C$, but this is trivial, since if $z = \nabla_C(z)$, then $z \in C$ by definition. $\square$

The above theorem, considered for instance for closure systems, states that each closure system induces a closure operator having as (pre-)fixed points exactly the elements in the closure system. Actually, we can say even more: each closure system induces a unique closure operator, that is, each closure operator is uniquely determined by its (pre-)fixed points.

Theorem 3.5. *Let $(L, \sqsubseteq)$ be a complete lattice. Then*
- *if $\nabla : L \to L$ is a closure operator then $\nabla_{\mathsf{fix}(\nabla)} = \nabla$*
- *if $\Delta : L \to L$ is a kernel operator, then $\Delta_{\mathsf{fix}(\Delta)} = \Delta$.*

Proof. We prove only point 1, the point 2 can be obtained by duality.
We have to show that $\nabla(x) = \nabla_{\mathsf{fix}(\nabla)}(x)$ for all $x \in L$. By definition, $\nabla_{\mathsf{fix}(\nabla)} = \bigsqcap A$ with $A = \{y \in \mathsf{fix}(\nabla) \mid x \sqsubseteq y\}$, hence, since $x \sqsubseteq \nabla(x)$, $\nabla(x) \in A$. In order to conclude the proof we have to show that $\nabla(x)$ is the least element of A. To this aim, consider $y = \nabla(y) \in A$ and prove that it is above $\nabla(x)$. Note that $x \sqsubseteq y$, hence, by monotonicity of ∇, $\nabla(x) \sqsubseteq \nabla(y) = y$, as needed. $\square$

In other words, the above theorem states that to define a closure or kernel operator it is enough to specify a closure or a kernel system. For instance, the closure system $\mathsf{Sub}(G)$, where G is a group, induces the closure operator $\langle - \rangle : \wp(G) \to \wp(G)$, that computes for any set $X \subseteq G$ the subgroup generated by X. For a topological space (X, τ) we have that the topology τ induces a kernel operator that, for any set $A \subseteq X$, computes its interior, and the set of closed sets $\{X \setminus A \mid A \in \tau\}$ induces the topological closure operator.

3.2. The bounded fixed point.

Let us now consider a monotone function $F : L \to L$. As we have seen, we can associate with F both a closure and a kernel system, $\mathsf{pre}(F)$ and $\mathsf{post}(F)$ respectively. Thanks to Theorem 3.4 and Theorem 3.5 we know that these systems induce a unique closure and kernel operator respectively, defined below

$$\nabla_F(x) = \nabla_{\mathsf{pre}(F)} = \bigsqcap\{y \in \mathsf{pre}(F) \mid x \sqsubseteq y\}$$

$$\Delta_F(x) = \Delta_{\mathsf{post}(F)} = \bigsqcup\{y \in \mathsf{post}(F) \mid y \sqsubseteq x\}$$

We call ∇_F the *closure* of F and Δ_F the *kernel* of F. Intuitively, $\nabla_F(x)$ is the best pre-fixed approximation of x (the least pre-fixed point above x), while $\Delta_F(x)$ is the best post-fixed

[3]We are considering the function $A \mapsto \bigsqcap A$ from $\wp(L)$ to L

approximation of x (the greatest post-fixed point below x). In this part of the section we will study some properties of these operators related to fixed points constructions.

First of all, we note that from the definitions of the closure and the kernel of F we can immediately derive a generalization of both the induction and the coinduction principles. Given $\gamma, \beta \in L$, for all $x \in L$ we have

(Ind) if $F(x) \sqsubseteq x$ (x pre-fixed) and $\gamma \sqsubseteq x$, then $\nabla_F(\gamma) \sqsubseteq x$
(CoInd) if $x \sqsubseteq F(x)$ (x post-fixed) and $x \sqsubseteq \beta$, then $x \sqsubseteq \Delta_F(\beta)$

These two principles are a generalization of standard induction and coinduction principles, because we can retrieve them through particular choices for γ and β. Indeed, if $\gamma = \bot$, the condition $\gamma \sqsubseteq x$ is trivially always true, and we have $\nabla_F(\bot) = \bigsqcap \mathsf{pre}(F) = \mu F$ by Knaster-Tarski fixed point theorem [Tar55], hence (Ind) allows us to conclude $\mu F \sqsubseteq x$ like standard induction, requiring the same hypothesis. Dually, if $\beta = \top$, again the condition $x \sqsubseteq \beta$ is trivially always true, and $\Delta_F(\top) = \bigsqcup \mathsf{post}(F) = \nu F$, again by Knaster-Tarski, hence (CoInd) allows us to conclude $x \sqsubseteq \nu F$ like standard coinduction, requiring the same hypothesis.

We now prove a result ensuring us that under suitable hypotheses we can use the closure and the kernel of a monotone function to build a fixed point of that function. We will denote by $\downarrow x$ and $\uparrow x$ respectively the set of lower bounds of x and the set of upper bounds of x.

Proposition 3.6. *Let $\gamma, \beta \in L$. Then*

(1) *if β is a pre-fixed point of F, then $\Delta_F(\beta)$ is a fixed point*
(2) *if γ is post-fixed point of F, then $\nabla_F(\gamma)$ is a fixed point*

Proof. We will prove only point 1, the point 2 can by obtained by duality.
Note that $\downarrow \beta$ is a complete lattice and the function $F : \downarrow \beta \to \downarrow \beta$ (obtained by restricting F to $\downarrow \beta$) is well-defined and monotone, since β is a pre-fixed point. Therefore, $\Delta_F(\beta)$ is the join of all post-fixed points of F in the complete lattice $\downarrow \beta$, hence by Knaster-Tarski it is a fixed point. $\square$

Therefore, we now know that if β is pre-fixed $\Delta_F(\beta)$ is the greatest fixed point below β, and, if γ is post-fixed, then $\nabla_F(\gamma)$ is the least fixed point above γ.

We are now able to define the *bounded fixed point*.

Definition 3.7 (Bounded fixed point)**.** Let $\gamma \in L$. The *bounded fixed point of F generated by γ*, denoted by $Gen(F, \gamma)$, is the greatest fixed point of F below the closure of γ, that is, $Gen(F, \gamma) = \Delta_F(\nabla_F(\gamma))$.

The bounded fixed point is well-defined since, thanks to Proposition 3.6, there exists the greatest fixed point below β, provided that the bound β is a pre-fixed point. Since in general γ might not be pre-fixed, we need to construct a pre-fixed point from γ, and this is done by the closure operator ∇_F. Note that the first step of this construction *cannot* be expressed as the least fixed point of F on the complete lattice $\uparrow \gamma$, since in general F may fail to be well-defined (e.g., if F is the function which maps any element to $\bot \sqsubseteq \gamma$ with $\gamma \neq \bot$). Indeed, $\nabla_F(\gamma)$ is *not* a fixed point in general, but only a pre-fixed point: we need the two steps to obtain a fixed point.

Note also that the definition of bounded fixed point is asymmetric, that is, we take the greatest fixed point bounded from above by a least pre-fixed point, rather than the other way round. This is motivated by the intuition, explained in Section 2, that we essentially need a greatest fixed point, since we want to deal with non-well-founded structures, but we want to “constrain” in some way such greatest fixed point. Investigating the dual construction $(\nabla_F(\Delta_F(\gamma)))$ is a matter of further work.

The following proposition states some immediate properties of the bounded fixed point.

Proposition 3.8.
(1) *If $z \in L$ is a fixed point of F, then $Gen(F, z) = z$.*
(2) *For all $\gamma_1, \gamma_2 \in L$, if $\gamma_1 \sqsubseteq \gamma_2$, then $Gen(F, \gamma_1) \sqsubseteq Gen(F, \gamma_2)$.*

Proof.
(1) If z is a fixed point, then it is both pre-fixed and post-fixed, hence $\nabla_F(z) = z$ and $\Delta_F(z) = z$. Thus, we get that $Gen(F, z) = \Delta_F(\nabla_F(z)) = \Delta_F(z) = z$.
(2) The statement can be rephrased saying that the function $Gen(F, -) : L \to L$ is monotone, and this trivially holds since it is a composition of the monotone function ∇_F and Δ_F. □

Therefore, by Proposition 3.6 we already know that $Gen(F, \gamma)$ is a fixed point for any $\gamma \in L$; the first point of the above proposition says that all fixed points of F can be generated as bounded fixed points. In other words, considering $Gen(F, -)$ as a function from L into itself, the first point implies that the range of this function is exactly $\mathsf{fix}(F)$. Moreover, the second point states that $Gen(F, -)$ is a monotone function on L.

An important fact is that bounded fixed points are a generalization of both least and greatest fixed points, since they can be obtained by taking particular generators, as stated in the following proposition.

Proposition 3.9.
(1) *$Gen(F, \top)$ is the greatest fixed point of F*
(2) *$Gen(F, \bot)$ is the least fixed point of F*

Proof.
(1) Note that $\nabla_F(\top) = \top$, since the only pre-fixed point above $\top$ is $\top$ itself, hence we get $Gen(F, \top) = \Delta_F(\top) = \bigsqcup \mathsf{post}(F) = \nu F$,
(2) As already noted $\nabla_F(\bot) = \mu F$, in particular $\nabla_F(\bot)$ is post-fixed, therefore we get $Gen(F, \bot) = \Delta_F(\nabla_F(\bot)) = \nabla_F(\bot)$, namely it is the least fixed point of F. □

An alternative proof for the above proposition is possible by exploiting Proposition 3.8. We preferred to give the above proof, since this follows the asymmetry of the definition of the bounded fixed point.

We now present a result that will be particularly useful to develop proof techiques for the bounded fixed point (see Section 5).

We first recall some standard notions. A *chain* C, is a totally ordered sequence $(x_i)_{i\in\mathbb{N}}$, we say that C is *ascending* if for all $i \in \mathbb{N}$, $x_i \sqsubseteq x_{i+1}$, and that C is *descending* if for all $i \in \mathbb{N}$, $x_{i+1} \sqsubseteq x_i$. A function $F : L \to L$ is said to be *upward continuous* if for any chain C, $F(\bigsqcup C) = \bigsqcup F(C)$ and *downward continuous* if for any chain C, $F(\bigsqcap C) = \bigsqcap F(C)$.

We will denote by $I_{F,x}$ the set $\{F^n(x) \mid n \in \mathbb{N}\}$ where $F^0 = \mathsf{id}_L$ and $F^{n+1} = F \circ F^n$. It is easy to check that if x is either pre-fixed or post-fixed, $I_{F,x}$ is a chain and in particular a descending chain if x is pre-fixed.

Proposition 3.10. *Let $\beta \in L$ be a pre-fixed point of F. Then*
(1) *for all $n \in \mathbb{N}$, $\Delta_F(\beta) = \Delta_F(F^n(\beta))$*
(2) $\Delta_F(\beta) = \Delta_F(\bigsqcap I_{F,\beta})$

Proof. Note that, since β is pre-fixed, $I_{F,\beta}$ is a descending chain, hence for all $n \in \mathbb{N}$ we have $F^{n+1}(\beta) \sqsubseteq F^n(\beta)$, that is, $F^n(\beta)$ is a pre-fixed point for all $n \in \mathbb{N}$.

(1) We prove the statement by induction on n. If $n = 0$ there is nothing to prove. Now, assume the thesis for n. By definition, $\Delta_F(F^n(\beta))$ is a post-fixed point, hence $\Delta_F(F^n(\beta)) \sqsubseteq F(\Delta_F(F^n(\beta)))$. Since Δ_F is a kernel operator, by Definition 3.3, we have $\Delta_F(F^n(\beta)) \sqsubseteq F^n(\beta)$, hence by the monotonicity of F, we get $F(\Delta_F(F^n(\beta))) \sqsubseteq F^{n+1}(\beta)$. Now, by transitivity of $\sqsubseteq$ we get $\Delta_F(F^n(\beta)) \sqsubseteq F^{n+1}(\beta)$. Therefore, by (CoInd) we conclude $\Delta_F(F^n(\beta)) \sqsubseteq \Delta_F(F^{n+1}(\beta))$.
On the other hand, since $F^n(\beta)$ is pre-fixed, we have $F^{n+1}(\beta) \sqsubseteq F^n(\beta)$. Thus, by the monotonicity of Δ_F we get the other inequality, and this implies $\Delta_F(F^n(\beta)) = \Delta_F(F^{n+1}(\beta))$. Finally, thanks to the induction hypothesis we get the thesis.

(2) By point 1 we have $\Delta_F(\beta) \sqsubseteq F^n(\beta)$ for all $n \in \mathbb{N}$, hence $\Delta_F(\beta) \sqsubseteq \bigsqcap I_{F,\beta}$. Therefore, by (CoInd) we get $\Delta_F(\beta) \sqsubseteq \Delta_F(\bigsqcap I_{F,\beta})$. On the other hand, we have $\bigsqcap I_{F,\beta} \sqsubseteq \beta$, hence, by monotonicity of Δ_F, we get the other inequality, and this implies the thesis. $\square$

Another way to read the above proposition is that, given a bound β which is pre-fixed, we obtain the same greatest fixed point below β if we take as bound any element $F^n(\beta)$ of the descending chain $I_{F,\beta}$. Moreover, Proposition 3.10 says also that we obtain the same greatest fixed point induced by β if we take as bound the greatest lower bound of that chain, namely, $\bigsqcap I_{F,\beta}$.

We conclude this part of the section with a result that characterizes the closure and the kernel of respectively a post-fixed and a pre-fixed point using chains in analogy with the Kleene theorem [LNS82]

Proposition 3.11. *Let $\beta, \gamma \in L$ be a pre-fixed and a post-fixed point respectively. Then*

(1) *if F is downward continuous, then $\Delta_F(\beta) = \bigsqcap I_{F,\beta}$*
(2) *if F is upward continuous, then $\nabla_F(\gamma) = \bigsqcup I_{F,\gamma}$*

Proof. We prove only point 1, the point 2 can by obtained by duality.
Note that $\downarrow\beta$ is a complete lattice with top element β and the function $F : \downarrow\beta \to \downarrow\beta$ (obtained by restricting F to $\downarrow\beta$) is well-defined and monotone, since β is a pre-fixed point. In this case it is also downward continuous, because so is F. Therefore, by Proposition 3.6, $\Delta_F(\beta)$ is the greatest fixed point of F in the complete lattice $\downarrow\beta$, hence, since F is downward continuous, we get the thesis by the Kleene theorem. $\square$

Note that the above proposition requires an additional hypothesis on F, that is required to be continuous, as happens for the Kleene theorem [LNS82]. Under this assumption the above result immediately applies to the bounded fixed point, providing us with an iterative characterization of it, as the following corollary shows.

Corollary 3.12. *Let $\gamma \in L$ and set $\beta = \nabla_F(\gamma)$. If F is downward continuous, then $Gen(F, \gamma) = \bigsqcap I_{F,\beta}$.*

Proof. By Definition 3.7 we have $Gen(F, \gamma) = \Delta_F(\beta)$. Since F is downward continuous, by Proposition 3.11 we get the thesis. $\square$

3.3. **Coaxioms as generators.** In this part of the section we come back to inference systems and we show that the interpretation generated by coaxioms of an inference system is indeed a fixed point of the inference operator. In Section 2 we have described two steps to construct $\mathit{Gen}(\mathcal{I},\gamma)$, the interpretation generated by coaxioms γ of an inference system $\mathcal{I}$:

(1) First, we consider the inference system $\mathcal{I}_{\sqcup\gamma}$ obtained enriching $\mathcal{I}$ by judgements in γ considered as axioms, and we take its inductive interpretation $\mathit{Ind}(\mathcal{I}_{\sqcup\gamma})$.
(2) Then, we take the coinductive interpretation of the inference system obtained from $\mathcal{I}$ by keeping only rules with consequence in $\mathit{Ind}(\mathcal{I}_{\sqcup\gamma})$, that is, we define
$$\mathit{Gen}(\mathcal{I},\gamma) = \mathit{CoInd}(\mathcal{I}_{\sqcap \mathit{Ind}(\mathcal{I}_{\sqcup\gamma})})$$

The definition of the bounded fixed point is the formulation of these two steps in the general setting of complete lattices. Indeed, the inference operator $F_{\mathcal{I}}$ is a monotone function on the complete lattice $(\wp(\mathcal{U}),\subseteq)$ obtained by taking set inclusion as order, and specifying the coaxioms γ corresponds to fixing an arbitrary element of the lattice as generator. Then:

(1) First, we construct the closure of γ, that is, the best closed approximation of γ. This closure plays the role of the bound for the next step.
(2) Then we construct the greatest fixed point below such bound.

To show the correspondence in a precise way, we give an alternative and equivalent characterization of both the closure and the kernel of an element in L.

Proposition 3.13. *Let $\gamma,\beta\in L$.*

(1) *Consider the function $F_{\sqcup\gamma} : L\to L$ defined by $F_{\sqcup\gamma}(x) = F(x)\sqcup\gamma$. Then, $\nabla_F(\gamma) = \mu F_{\sqcup\gamma}$.*
(2) *Consider the function $F_{\sqcap\beta} : L\to L$ defined by $F_{\sqcap\beta}(x) = F(x)\sqcap\beta$. Then, $\Delta_F(\beta) = \nu F_{\sqcap\beta}$.*

Proof. We prove only point 1, point 2 can be obtained by duality. First of all note that $F_{\sqcup\gamma}$ is a monotone function. By definition of ∇_F, we have that $F(\nabla_F(\gamma))\sqsubseteq\nabla_F(\gamma)$ and $\gamma\sqsubseteq\nabla_F(\gamma)$, hence $\nabla_F(\gamma)$ is a pre-fixed point of $F_{\sqcup\gamma}$. Then, by (Ind), $\nabla_F(\gamma)$ is the least pre-fixed point of $F_{\sqcup\gamma}$, hence, by Knaster-Tarski, $\nabla_F(\gamma) = \mu F_{\sqcup\gamma}$. □

By this alternative characterization we can formally state the correspondence with the two steps for defining $\mathit{Gen}(\mathcal{I},\gamma)$.

Theorem 3.14. *Let $\mathcal{I}$ be an inference system and $\gamma,\beta\in\wp(\mathcal{U})$, then the following facts hold:*

(1) $(F_{\mathcal{I}})_{\sqcup\gamma} = F_{(\mathcal{I}_{\sqcup\gamma})}$ *(so we can safely omit brackets)*
(2) $(F_{\mathcal{I}})_{\sqcap\beta} = F_{(\mathcal{I}_{\sqcap\beta})}$ *(so we can safely omit brackets)*
(3) $\nabla_{F_{\mathcal{I}}}(\gamma) = \mathit{Ind}(\mathcal{I}_{\sqcup\gamma})$
(4) $\Delta_{F_{\mathcal{I}}}(\beta) = \mathit{CoInd}(\mathcal{I}_{\sqcap\beta})$

Proof.

(1) We have to show that, for $S\subseteq\mathcal{U}$, $(F_{\mathcal{I}})_{\sqcup\gamma}(S) = F_{(\mathcal{I}_{\sqcup\gamma})}(S)$. If $c\in(F_{\mathcal{I}})_{\sqcup\gamma}(S)$, then either $c\in\gamma$ or $c\in F_{\mathcal{I}}(S)$; in the former case, there exists $\frac{}{c}\in\mathcal{I}_{\sqcup\gamma}$ by definition of $\mathcal{I}_{\sqcup\gamma}$, in the latter, there exists $\frac{Pr}{c}\in\mathcal{I}$ such that $Pr\subseteq S$, and this implies $\frac{Pr}{c}\in\mathcal{I}_{\sqcup\gamma}$. Therefore, in both cases $c\in F_{(\mathcal{I}_{\sqcup\gamma})}(S)$.
Conversely, if $c\in F_{(\mathcal{I}_{\sqcup\gamma})}(S)$, then there exists $\frac{Pr}{c}\in\mathcal{I}_{\sqcup\gamma}$ such that $Pr\subseteq S$. By definition of $\mathcal{I}_{\sqcup\gamma}$, either $\frac{Pr}{c}\in\mathcal{I}$ or $c\in\gamma$ and $Pr=\emptyset$, therefore, in the former case $c\in F_{\mathcal{I}}(S)$ and in the latter $c\in\gamma$, thus in both cases $c\in(F_{\mathcal{I}})_{\sqcup\gamma}(S)$.

(2) We have to show that, for $S \subseteq \mathcal{U}$, $(F_{\mathcal{I}})_{\sqcap\beta}(S) = F_{(\mathcal{I}_{\sqcap\beta})}(S)$. If $c \in (F_{\mathcal{I}})_{\sqcap\beta}(S)$, then we have $c \in \beta$ and $c \in F_{\mathcal{I}}(S)$, hence there is $\frac{Pr}{c} \in \mathcal{I}$ such that $Pr \subseteq S$; therefore, by definition of $\mathcal{I}_{\sqcap\beta}$, we get $\frac{Pr}{c} \in \mathcal{I}_{\sqcap\beta}$, and this implies that $c \in F_{(\mathcal{I}_{\sqcap\beta})}(S)$.
Conversely, if $c \in F_{(\mathcal{I}_{\sqcap\beta})}(S)$, then there exists $\frac{Pr}{c} \in \mathcal{I}_{\sqcap\beta}$ such that $Pr \subseteq S$. By definition of $\mathcal{I}_{\sqcap\beta}$, we have that $\frac{Pr}{c} \in \mathcal{I}$ and $c \in \beta$, therefore $c \in F_{\mathcal{I}}(S)$ and $c \in \beta$, thus $c \in (F_{\mathcal{I}})_{\sqcap\beta}(S)$.

(3) By Proposition 3.13 we get that $\nabla_{F_{\mathcal{I}}}(\gamma) = \mu F_{\mathcal{I}_{\sqcup\gamma}}$, that corresponds to the *inductive interpretation* of $\mathcal{I}_{\sqcup\gamma}$, $\mathit{Ind}(\mathcal{I}_{\sqcup\gamma})$, by point 1 of this theorem.

(4) By Proposition 3.13 we get that $\Delta_{F_{\mathcal{I}}}(\beta) = \nu F_{\mathcal{I}_{\sqcap\beta}}$, that corresponds to the *coinductive interpretation* of $\mathcal{I}_{\sqcap\beta}$, $\mathit{CoInd}(\mathcal{I}_{\sqcap\beta})$, by point 2 of this theorem. □

Thanks to Theorem 3.14, we can conclude that, given an inference system with coaxioms $(\mathcal{I}, \gamma)$:

$$\mathit{Gen}(\mathcal{I},\gamma) = \mathit{CoInd}(\mathcal{I}_{\sqcap \mathit{Ind}(\mathcal{I}_{\sqcup\gamma})}) = \Delta_{F_{\mathcal{I}}}(\nabla_{F_{\mathcal{I}}}(\gamma)) = \mathit{Gen}(F_{\mathcal{I}},\gamma)$$

that is, the interpretation generated by coaxioms γ of the inference system $\mathcal{I}$ is exactly the bounded fixed point of $F_{\mathcal{I}}$ generated by γ.

Finally, applying Proposition 3.9 we get that the inductive and the coinductive interpretations of $\mathcal{I}$ are particular cases of the interpretation generated by coaxioms. Indeed, we get the inductive interpretation when $\gamma = \emptyset$ and we get the coinductive interpretation when $\gamma = \mathcal{U}$, as shown below.

$$\mathit{Gen}(\mathcal{I},\emptyset) = \mathit{Gen}(F_{\mathcal{I}},\emptyset) = \mu F_{\mathcal{I}} = \mathit{Ind}(\mathcal{I})$$
$$\mathit{Gen}(\mathcal{I},\mathcal{U}) = \mathit{Gen}(F_{\mathcal{I}},\mathcal{U}) = \nu F_{\mathcal{I}} = \mathit{CoInd}(\mathcal{I})$$

4. Proof trees for coaxioms

In this section we formalize several proof-theoretic characterizations of the semantics of inference systems with coaxioms, and prove their equivalence with the fixed point semantics presented in Section 3. In order to discuss such proof-theoretic semantics in a rigorous way, we need a more explicit and mathematically precise notion of proof tree than the one we introduced in Section 2; therefore, we start by fixing some concepts on trees.

4.1. **A digression on graphs and trees.** Here we report some results about trees and graphs. We essentially follow the approach adopted in [AAV01, AAMV03, ALM+15], with few differences in the definition of trees: for us a tree will be labelled and unordered as in [MP00, vdBM07]. The main theorem of this subsection (Theorem 4.3) is a weaker form of results presented in [AAMV03, ALM+15], which, however, require additional conditions[4] on trees, which we can ignore.

Along this section we denote by $A^\star$ the set of finite *strings* on the *alphabet* A, which is an arbitrary set of *symbols*. We use Greek letters $\alpha, \beta, \ldots$ to range over strings and Roman letters $a, b, \ldots$ to range over symbols in A and we implicitly identify strings of length one and symbols. Moreover, we denote by juxtaposition *string concatenation*, and by $|\alpha|$ the *length* of the string α. Finally, ε is the empty string. We also extend string concatenation to sets of strings, denoting, for $X, Y \subseteq A^\star$, by XY the set $\{\alpha\beta \in A^\star \mid \alpha \in X, \beta \in Y\}$; moreover if either X or Y are singletons we will omit curly braces, namely $\alpha Y = \{\alpha\}Y$.

[4] These conditions are needed since they want a final coalgebra for suitable power-set functors.

On the set $A^\star$ we can define the *prefixing relation* $\prec$ as follows: for any $\alpha, \beta \in A^\star$, $\alpha \prec \beta$ if and only if there exists $\gamma \in A^\star$ such that $\alpha\gamma = \beta$. It can be shown that $\prec$ is a partial order and thus, for any $X \subseteq A^\star$, the restriction of $\prec$ to X is well-defined and still a partial order. We say that a subset $X \subseteq A^\star$ is *well-founded* with respect to prefixing if any chain $C \subseteq X$ is finite.

A non-empty subset $L \subseteq A^\star$ is a *tree language* if it satisfies the *prefix property*, that is, if $\alpha a \in L$ then $\alpha \in L$. In particular, $\varepsilon \in L$ for any tree language $L \subseteq A^\star$. Now we are able to define trees following [Cou83].

Definition 4.1. Let A be an alphabet, $L \subseteq A^\star$ a tree language and $\mathcal{L}$ a set. A *tree* labelled in $\mathcal{L}$ is a function $t : L \to \mathcal{L}$. The element $t(\varepsilon)$ is called the *root* of t.

The notion of tree in Definition 4.1 is essentially the same as standard ones, see, e.g., [Cou83, AAV01]. The main difference is that we allow an arbitrary set to be taken as alphabet. This is important because, as we will see, the branching of the tree is bounded by the cardinality of the alphabet, and, since we have to use trees in the context of inference systems, this cardinality cannot be bounded a priori.

If $t : L \to \mathcal{L}$ is a tree, then, for any $\alpha \in L$, the *subtree* rooted at α is the function $t_{|\alpha} : L_{|\alpha} \to \mathcal{L}$, where $L_{|\alpha} = \{\beta \in A^\star \mid \alpha\beta \in L\}$ and $t_{|\alpha}(\beta) = t(\alpha\beta)$. This notion is well-defined since $L_{|\alpha}$ is a tree language, hence $t_{|\alpha}$ is a tree. Note that t is itself a subtree, rooted at ε. Subtrees rooted at α with $|\alpha| = 1$ are called *direct subtrees* of t. Finally, a tree t is *well-founded* if $\mathit{dom}(t)$ is well-founded with respect to $\prec$.

The notion of tree introduced in Definition 4.1 is mathematically precise, but not very intuitive. A usual, and perhaps more natural, way to introduce trees is as particular graphs. Intuitively, using a graph-like terminology, that we will make precise below, we can see the elements in the tree language $\mathit{dom}(t)$ as nodes. Actually, thanks to the prefix property, a node $\alpha \in \mathit{dom}(t)$ represents also all nodes (its prefixes) we have to traverse to reach α starting from the root ε. For instance, if $\alpha = abc$, we know that $\varepsilon, a, ab, abc \in \mathit{dom}(t)$, hence they are nodes of t and they form the path from the root to α. Therefore, requiring t to be well-founded is equivalent to require that any sequence of prefixes is finite, hence it is equivalent to require that all paths in t are finite.

To formally show that indeed trees can be seen as particular graphs, we start by giving a definition of graph.

Definition 4.2. A *graph* is a pair (V, adj) where V is the set of *nodes* and $\mathit{adj} : V \to \wp(V)$ is the *adjacency function*.

With this definition it is easy to assign a graph structure to (the domain of) a tree. Let $t : L \to \mathcal{L}$ be a tree, we can represent it as a graph with set of nodes L and adjacency function $\mathit{chl}_t(\alpha) = \{\beta \in L \mid \exists a \in A.\beta = \alpha a\}$ returning the *children* of a node α. In the following we will omit the reference to t when it is clear from the context. Thanks to this graph structure we justify terminology like node and adjacent for trees: a node is a string $\alpha \in \mathit{dom}(t)$ and, given a node α, the set of its adjacents is $\mathit{chl}(\alpha)$. Furthermore, the *depth* of a node $\alpha \in \mathit{dom}(t)$ is its distance, in a graph-theoretic sense, from the root, that is, $|\alpha|$, hence it is alway finite; the depth of the tree t is the least upper bound of the depth of all its nodes, hence it can be infinite.

We now analyse the role of the alphabet A in the definition of tree (Definition 4.1). First, note that its elements are essentially not relevant. What actually matters is the cardinality of A, that determines the maximum *branching* of the tree, that is, the maximum number of children (hence subtrees) for each node α. In other words, we have $|chl(\alpha)| \leq |A|$ for all $\alpha \in L$. For instance, we can build essentially the same trees if A is either $\{1, 2, 3\}$ or $\{a, b, c\}$. However, the fact that they have both cardinality 3 is relevant, since trees built on A have for each node at most 3 children. Therefore, each cardinality λ determines a class of trees, called λ-*branching trees*, that is, trees built on an alphabet of cardinality λ.

In the following, we will denote by $\mathcal{T}_\lambda(\mathcal{L})$ the set of all λ-branching trees, where λ is a given cardinal. We will omit λ when not relevant. Since, as we have noticed, the elements of the alphabet are irrelevant, we will say that two λ-branching trees are equal if, considering them as functions, they are equal up to isomorphism on the alphabet[5]. Furthermore, a tree $t \in \mathcal{T}_\kappa(\mathcal{L})$ with $\kappa \leq \lambda$ can be regarded as an element of $\mathcal{T}_\lambda(\mathcal{L})$ up to an inclusion of the alphabet in a set of cardinality λ. We will leave implicit this inclusion and hence write $\mathcal{T}_\kappa(\mathcal{L}) \subseteq \mathcal{T}_\lambda(\mathcal{L})$.

We now consider a special class of trees, suitable to model proof trees. As we will see, proof trees are labelled by judgements, notably nodes are (labelled by) consequences of rules and their children correspond to sets of premises, hence each child has a distinct label. Trees of this kind can be represented in a more compact way, and enjoy an important property.

Let us introduce these trees formally. We say that a tree $t : L \to \mathcal{L}$ is *children injective* if, for all $\alpha \in dom(t)$, the restriction of t to the set $chl(\alpha)$ is injective; more explicitly, for all $\alpha \in dom(t)$, if $\alpha a, \alpha b \in dom(t)$ and $t(\alpha a) = t(\alpha b)$, then $a = b$. In other words, all children of a node must have different labels. Note that all subtrees of a children injective tree are themselves children injective.

The first property we observe is that a children injective tree is completely determined by the label of its root and by the set of all paths of labels in it. Indeed, if $t : L \to \mathcal{L}$ is children injective, then we can define the following function:

$$f_t : L \to \mathcal{L}^\star \quad \begin{cases} f_t(\varepsilon) & = \varepsilon \\ f_t(\alpha a) & = f_t(\alpha) t(\alpha a) \end{cases}$$

Intuitively, the function f maps each node $\alpha \in L$ to the string of labels encountered in the path from the root to α. It is easy to see that f is injective and $f_t(L)$ is a tree language. Hence, the pair $(t(\varepsilon),\ f_t(L))$ conveys a complete description of t, that is, starting from it, we can reconstruct t, up to a change of the alphabet. More precisely, we can define a tree $t_\mathcal{L} : f_t(L) \to \mathcal{L}$ such that $t(\alpha) = t_\mathcal{L}(f_t(\alpha))$ as follows:

$$\begin{cases} t_\mathcal{L}(\varepsilon) & = t(\varepsilon) \\ t_\mathcal{L}(\alpha a) & = a \end{cases}$$

As a consequence, a children injective tree t can always be equivalently represented as $t_\mathcal{L}$. Finally, note that the subtree of t rooted at α, $t_{|\alpha}$, is represented by $\big(t(\alpha),\ f(L)_{|f(\alpha)}\big)$.

We denote by $\mathcal{T}^{\mathsf{ci}}(\mathcal{L})$ the set of children injective trees labelled in $\mathcal{L}$. From the construction just presented, we also get that the branching of a children injective tree t is bounded by the cardinality λ of $\mathcal{L}$, hence we have that $\mathcal{T}^{\mathsf{ci}}(\mathcal{L}) \subseteq \mathcal{T}_\lambda(\mathcal{L})$.

The main result of this subsection concerning children injective trees is Theorem 4.3. Before stating it we need to briefly say something about *paths* in a graph. Let $G = (V,\ adj)$ be a graph, a path in G is a non-empty string $v_0 \cdots v_n \in V^\star$ such that, for all $i \in \{0, \ldots, n-1\}$, $v_{i+1} \in adj(v_i)$, that is, for all pairs of subsequent nodes the latter is adjacent to the former.

[5]Formally, we should define an equivalence relation on trees and then work in the quotient. Two trees $t, t' \in \mathcal{T}_\lambda(\mathcal{L})$ are equivalent iff $t = t' \circ b$ where b is a bijection between the alphabet of t and that of t'.

We say that $v_0 \cdots v_n$ is a path *from* v_0 *to* v_n. Note that the string v_0 of length 1 is also a path, from v_0 to v_0, that does not traverse any edge. We denote by $\mathsf{path}(G)$ the set of paths in G.

Note that $\mathsf{path}(G)$ is closed under non-empty prefixes, that is, if αa is a path and α is not empty, then α is a path too, and more generally, if $\alpha\beta \in \mathsf{path}(G)$ and α and β are not empty, then $\alpha, \beta \in \mathsf{path}(G)$. Therefore we can easily lift $\mathsf{path}(G)$ to a tree language, by adding to it the empty string. From these observations immediately follows that, for each $\alpha \in \mathsf{path}(G)$, the set $\{\beta \in V^\star \mid \alpha\beta \in \mathsf{path}(G)\} \subseteq \mathsf{path}(G) \cup \{\varepsilon\}$ is a tree language.

It is also important to note that the sets $\mathcal{T}_\lambda(\mathcal{L})$ and $\mathcal{T}^{\mathsf{ci}}(\mathcal{L})$ both carry a graph structure with the following adjacency function:

$$dsub(t) = \{t_{|\alpha} \mid \alpha \in dom(t),\, |\alpha| = 1\}$$

which returns the direct subtrees of t.

Thanks to this observation, we can now prove the following theorem, that will be essential to give our proof of equivalence between the proof-theoretic and the fixed point semantics for coaxioms (Theorem 4.6). Intuitively, this result allows us to associate with any node in a graph, in a canonical way, a tree rooted in it, preserving the graph structure.

Theorem 4.3. *Let $G = (V,\, adj)$ be a graph, then there exists a function $\mathsf{P}\colon V \to \mathcal{T}^{\mathsf{ci}}(V)$ such that the following diagram commutes:*

$$\begin{tikzcd} V \arrow[r, "\mathsf{P}"] \arrow[d, "adj"] & \mathcal{T}^{\mathsf{ci}}(V) \arrow[d, "dsub"] \\ \wp(V) \arrow[r, "\wp(\mathsf{P})"] & \wp(\mathcal{T}^{\mathsf{ci}}(V)) \end{tikzcd}$$

Proof. The function P computes for each node the *path expansion* starting from this node, that is, it maps each node v to the set of all paths starting with v. More precisely, the set of paths we compute for each node v is the following:

$$L_v = \{\alpha \in V^\star \mid v\alpha \in \mathsf{path}(G)\}$$

Hence, using the representation of children injective trees as pairs $(r,\, L)$ where r is a label and L is a tree language, using nodes as alphabet, we have that

$$\mathsf{P}(v) = (v,\, L_v)$$

Now we have to show that, for each node v, $\wp(\mathsf{P})(adj(v)) = dsub(\mathsf{P}(v))$, that is, $(u,\, L) \in dsub(\mathsf{P}(v))$ if and only if $u \in adj(v)$ and $\mathsf{P}(u) = (u,\, L_u) = (u,\, L)$. The implication $\Rightarrow$ holds by construction. On the other hand, if $(u,\, L) \in dsub(\mathsf{P}(v))$ then $L = \{\alpha \in V^\star \mid u\alpha \in L_v\}$, hence $L = L_u$, that is, $(u,\, L) = \mathsf{P}(u)$. Moreover, for all $\alpha \in L$, $u\alpha \in L_v$ implies that $vu\alpha$ is a path in G, hence $u \in adj(v)$, and this shows the equality. $\square$

In the end, note that, if $t_v = \mathsf{P}(v)$ for each $v \in V$, then P is the unique map making the diagram commute and such that $t_v(\varepsilon) = v$.

4.2. **Combining non-well-founded and well-founded proof trees.** Before providing the first proof-theoretic characterization, we give a more precise definition of proof tree, which is a generalization of the notion of rule graph proposed in [Bro05].

Definition 4.4. Let $\mathcal{I}$ be an inference system, a *proof tree* in $\mathcal{I}$ is a children injective tree $t : L \to \mathcal{U}$ such that, for all $\alpha \in L$, there is a rule $\frac{Pr}{c} \in \mathcal{I}$ such that $t(\alpha) = c$ and $t(chl(\alpha)) = Pr$.

In other words, a proof tree t is a tree labelled in $\mathcal{U}$ where each node $\alpha \in dom(t)$ is labelled by the conclusion of a rule $\frac{Pr}{c} \in \mathcal{I}$ and children of α are bijectively labelled by judgements in Pr. Since a proof tree t is children injective by definition, we can also represent it as $(t(\varepsilon),\ f_t(dom(t)))$.

In the following, we will often represent proof trees using stacks of rules, that is, if $\frac{Pr}{c} \in \mathcal{I}$ and $\mathcal{T}$ is a set of proof trees in bijection with Pr and such that for all $t \in \mathcal{T}$, $t(\varepsilon) \in Pr$, we denote by $\frac{\mathcal{T}}{c}$ the proof tree t_c given by

$$dom(t_c) = \{\varepsilon\} \cup \bigcup_{t\in\mathcal{T}} t(\varepsilon) f_t(dom(t)) \qquad \begin{cases} t_c(\varepsilon) & = c \\ t_c(\alpha j) & = j \end{cases}$$

We say that a tree t is a proof tree for a judgement $j \in \mathcal{U}$ if it is a proof tree rooted in j. Finally, note that all subtrees of a proof tree t are proof trees themselves for their roots. With this terminology we can define our proof-theoretic semantics.

The first proof-theoretic characterization of the semantics of inference systems with coaxioms is based on the following theorem which slightly generalizes the standard result about the correspondence between the fixed point and the proof-theoretic semantics of inference systems in the coinductive case (see [LG09]). We choose to do the proof from scratch, even if it can be done relying on the standard equivalence (see [ADZ17b]), since we have not found in literature a detailed enough proof for the standard equivalence, and, in addition, the proof helps us to understand what happens on the proof-theoretic side, in particular when we prove that a set is consistent. We begin proving a lemma.

Lemma 4.5. *Let $\mathcal{I}$ be an inference system and $\mathcal{S}$ a consistent subset of $\mathcal{U}$, then for each $j \in \mathcal{S}$ there is a proof tree t for j such that, for all $\alpha \in dom(t)$, $t(\alpha) \in \mathcal{S}$*

Proof. By hypothesis $\mathcal{S}$ is consistent, so for each judgement $j \in \mathcal{S}$ we can choose a rule $\frac{Pr_j}{j} \in \mathcal{I}$ such that $Pr_j \subseteq \mathcal{S}$. In other words, we can define the map $adj : \mathcal{S} \to \wp(\mathcal{S})$ given by $adj(j) = Pr_j$, that turns $\mathcal{S}$ into a graph as in Definition 4.2. By Theorem 4.3, there exists a map $\mathsf{P} : \mathcal{S} \to \mathcal{T}^{\mathsf{ci}}(\mathcal{S})$ making the following diagram commute.

$$\begin{array}{ccc} \mathcal{S} & \xrightarrow{\mathsf{P}} & \mathcal{T}^{\mathsf{ci}}(\mathcal{S}) \\ \downarrow{\scriptstyle adj} & & \downarrow{\scriptstyle dsub} \\ \wp(\mathcal{S}) & \xrightarrow{\wp(\mathsf{P})} & \wp(\mathcal{T}^{\mathsf{ci}}(\mathcal{S})) \end{array}$$

Therefore, for each $j \in \mathcal{S}$, $\mathsf{P}(j)$ is a tree rooted in j and labelled in $\mathcal{S}$, that is, for all $\alpha \in dom(\mathsf{P}(j))$, $\mathsf{P}(j)(\alpha) \in \mathcal{S} \subseteq \mathcal{U}$. Set $t_j = \mathsf{P}(j)$ and note that by construction, for all $\alpha \in dom(t_j)$, we have $chl(\alpha) = \alpha\, adj(t_j(\alpha)) = \alpha Pr_{t_j(\alpha)}$ and

$$\frac{Pr_{t_j(\alpha)}}{t_j(\alpha)} \in \mathcal{I}$$

hence t_j is a proof tree in $\mathcal{I}$ as needed. $\square$

This lemma essentially ensures that all judgements in a consistent set $\mathcal{S}$ have an arbitrary proof tree whose nodes are all (labelled) in $\mathcal{S}$. The next theorem is a slight generalization of the standard equivalence between proof-theoretic and fixed point semantics.

Theorem 4.6. *Let $\mathcal{I}$ be an inference system and $\beta \subseteq \mathcal{U}$ a set of judgements. Then for all $j \in \mathcal{U}$ the following are equivalent:*
(1) $j \in \Delta_{F_{\mathcal{I}}}(\beta)$
(2) *there exists a proof tree t for j in $\mathcal{I}$ such that each node of t is (labelled) in β*

Proof. We prove separately the two implications.

1 ⇒ 2: By construction $\Delta_{F_{\mathcal{I}}}(\beta)$ is a consistent set and $\Delta_{F_{\mathcal{I}}}(\beta) \subseteq \beta$, hence by Lemma 4.5 each judgement $j \in \Delta_{F_{\mathcal{I}}}(\beta)$ has a proof tree in $\mathcal{I}$ whose nodes are all (labelled) in $\Delta_{F_{\mathcal{I}}}(\beta)$, hence they are (labelled) in β as needed.

2 ⇒ 1: Let $\mathcal{S} \subseteq \mathcal{U}$ be the set of all judgements having a proof tree in $\mathcal{I}$ whose nodes are all (labelled) in β. As a consequence, we immediately have that $\mathcal{S} \subseteq \beta$, hence if we show that $\mathcal{S}$ is consistent, we get the thesis by (CoInd). Consider $j \in \mathcal{S}$, hence there is a proof tree t_j for j whose nodes are all (labelled) in β. Note that each $t \in \mathit{dsub}(t_j)$ is a proof tree like t, hence $t(\varepsilon) \in \mathcal{S}$ and, since t_j is a proof tree,

$$\frac{\{t(\varepsilon) \mid t \in \mathit{dsub}(t_j)\}}{j} \in \mathcal{I}$$

and this shows that $\mathcal{S}$ is consistent as needed. $\square$

As a particular case we get our first proof-theoretic characterization of $\mathit{Gen}(\mathcal{I}, \gamma)$.

Corollary 4.7. *Let $(\mathcal{I}, \gamma)$ be an inference system with coaxioms. Then the following are equivalent:*
(1) $j \in \mathit{Gen}(\mathcal{I}, \gamma)$
(2) *there exists a proof tree t for j in $\mathcal{I}$ such that each node of t has a well-founded proof tree in $\mathcal{I}_{\sqcup\gamma}$*

Proof. We have that $\mathit{Gen}(\mathcal{I}, \gamma) = \Delta_{F_{\mathcal{I}}}(\mathit{Ind}(\mathcal{I}_{\sqcup\gamma}))$, hence by Theorem 4.6 we get that $j \in \mathit{Gen}(\mathcal{I}, \gamma)$ iff there is a proof tree t for j in $\mathcal{I}$ whose nodes are all (labelled) in $\mathit{Ind}(\mathcal{I}_{\sqcup\gamma})$. Therefore, all nodes of t have a well-founded proof tree in $\mathcal{I}_{\sqcup\gamma}$ by the standard equivalence for the inductive case (see e.g., [LG09]). $\square$

4.3. **Approximated proof trees.**

For the second proof-theoretic characterization, we need to define *approximated proof trees* in an inference system with coaxioms.

Definition 4.8. Let $(\mathcal{I}, \gamma)$ be an inference system with coaxioms, the sets $\mathcal{T}_n$ of *approximated proof trees of level n in $(\mathcal{I}, \gamma)$*, for $n \in \mathbb{N}$, are inductively defined as follows:

$$t \in \mathcal{T}_0 \qquad \text{if } t \text{ well-founded proof tree in } \mathcal{I}_{\sqcup\gamma}$$

$$\frac{\mathcal{T}}{c} \in \mathcal{T}_{n+1} \quad \text{if } \frac{Pr}{c} \in \mathcal{I} \text{ and } \mathcal{T} = (t_j)_{j \in Pr} \text{ and } \forall j \in Pr.\, t_j \in \mathcal{T}_n \text{ and } t_j(\varepsilon) = j$$

Note that an approximated proof tree is a proof tree, since the set $\mathcal{T}$ and the set Pr in the inductive step are in bijection: elements t_j in $\mathcal{T}$ are indexed by judgements in Pr, hence there is a surjective map from Pr to $\mathcal{T}$; moreover, if $t_j = t_{j'}$, then $j = t_j(\varepsilon) = t_{j'}(\varepsilon) = j'$, hence this map is also injective.

In other words, an approximated proof tree of level n in $(\mathcal{I}, \gamma)$ is a well-founded proof tree in $\mathcal{I}_{\sqcup\gamma}$ where coaxioms can only be used at depth $\geq n$. Therefore, if $t \in \mathcal{T}_n$ is an approximated proof tree of level n, then, for all $\alpha \in \mathit{dom}(t)$ with $|\alpha| < n$, $t(\alpha)$ is the consequence of a rule in $\mathcal{I}$, more precisely

$$\frac{\{t(\beta) \mid \beta \in \mathit{chl}(\alpha)\}}{t(\alpha)} \in \mathcal{I}$$

Another simple property of approximated proof trees is stated in the following proposition.

Proposition 4.9. *If $t \in \mathcal{T}_n$, $\alpha \in \mathit{dom}(t)$ and $|\alpha| = k \leq n$, then $t_{|\alpha} \in \mathcal{T}_{n-k}$.*

Proof. We proceed by induction on $|\alpha|$. If $|\alpha| = 0$, then $\alpha = \varepsilon$, hence $t_{|\varepsilon} = t \in \mathcal{T}_n$. Assume $|\alpha| = k + 1$, hence $\alpha = \beta a$, hence $\beta \in \mathit{dom}(t)$ and $|\beta| = k$. Therefore, by induction hypothesis $t_{|\beta} \in \mathcal{T}_{n-k}$, hence $t_{|\alpha} = (t_{|\beta})_{|a} \in \mathit{dsub}(t_{|\beta})$, and this implies, by Definition 4.8, that $t_{|\alpha} \in \mathcal{T}_{n-k-1}$. □

The following theorem states that approximated proof trees of level n correspond to the n-th element of the descending chain $I_{F_{\mathcal{I}},\beta} = \{F_{\mathcal{I}}^n(\beta) \mid n \in \mathbb{N}\}$, with $\beta = \nabla_{F_{\mathcal{I}}}(\gamma) = \mathit{Ind}(\mathcal{I}_{\sqcup\gamma})$.

Theorem 4.10. *Let $(\mathcal{I}, \gamma)$ be an inference system with coaxioms, and $j \in \mathcal{U}$ a judgement. We have that, for all $n \in \mathbb{N}$, the following are equivalent:*
(1) $j \in F_{\mathcal{I}}^n(\nabla_{F_{\mathcal{I}}}(\gamma))$
(2) j has an approximated proof tree of level n in $(\mathcal{I}, \gamma)$

Proof. Let β be $\nabla_{F_{\mathcal{I}}}(\gamma)$. We prove the thesis by induction on n.

Base: If $n = 0$, then, by Theorem 3.14, $\beta = \nabla_{F_{\mathcal{I}}}(\gamma)$ corresponds to the inductive interpretation of $\mathcal{I}_{\sqcup\gamma}$, hence the the thesis reduces to the standard equivalence between proof-theoretic and fixed point semantics in the inductive case (see [LG09]).

Induction: We assume the equivalence for n and prove it for $n + 1$. We prove separately the two implications.

$1 \Rightarrow 2$**:** If $c \in F_{\mathcal{I}}^{n+1}(\beta)$, then there exists $\frac{Pr}{c} \in \mathcal{I}$ such that $Pr \subseteq F_{\mathcal{I}}^n(\beta)$. Hence, by induction hypothesis, each judgement in Pr has an approximated proof tree of level n, that is, for all $j \in Pr$ there is an approximated proof tree $t_j \in \mathcal{T}_n$ rooted in j. Set $\mathcal{T} = \{t_j \in \mathcal{T}_n \mid j \in Pr\}$. Hence, $t = \frac{\mathcal{T}}{c}$ is a proof tree for c, and by Definition 4.8, $t \in \mathcal{T}_{n+1}$.

$2 \Rightarrow 1$**:** If $t \in \mathcal{T}_{n+1}$ is an approximated proof tree for $c \in \mathcal{U}$, then, by definition, there exists $\frac{Pr}{c} \in \mathcal{I}$ such that $t = \frac{\mathcal{T}}{c}$, $\mathcal{T} = (t_j)_{j \in Pr}$ and for all $j \in Pr$, $t_j \in \mathcal{T}_n$ and $t_j(\varepsilon) = j$. By induction hypothesis we have $Pr \subseteq F_{\mathcal{I}}^n(\beta)$, and, by definition of $F_{\mathcal{I}}$, this implies $c \in F_{\mathcal{I}}^{n+1}(\beta)$ as needed. □

The second proof-theoretic characterization of the interpretation generated by coaxioms is an immediate consequence of the above theorem.

Corollary 4.11. *Let $(\mathcal{I}, \gamma)$ be an inference system with coaxioms, and $j \in \mathcal{U}$ a judgement. Then the following are equivalent:*

(1) $j \in \mathit{Gen}(\mathcal{I}, \gamma)$
(2) *there exists a proof tree t for j in $\mathcal{I}$ such that each node of t has an approximated proof tree of level n in $(\mathcal{I}, \gamma)$, for all $n \in \mathbb{N}$.*

Proof. By Theorem 3.14, Proposition 3.10, and Theorem 4.6, we get that, for all $j \in \mathcal{U}$, $j \in \mathit{Gen}(\mathcal{I}, \gamma)$ iff there exists a proof tree t for j in $\mathcal{I}$ such that each node j' of t is in $\bigcap I_{F_\mathcal{I},\beta}$ with $\beta = \nabla_{F_\mathcal{I}}(\gamma)$. By Theorem 4.10, $j' \in \bigcap I_{F_\mathcal{I},\beta}$ iff has an approximated proof tree of level n, for all $n \in \mathbb{N}$. $\square$

If the hypotheses of Corollary 3.12 are satisfied, namely, if the inference operator is downward continuous, then we get a simpler equivalent proof-theoretic characterization.

Corollary 4.12. *Let $(\mathcal{I}, \gamma)$ be an inference system with coaxioms, and $j \in \mathcal{U}$ a judgement. If $F_\mathcal{I}$ is downward continuous, then the following are equivalent:*
(1) $j \in \mathit{Gen}(\mathcal{I}, \gamma)$
(2) *j has an approximated proof tree of level n in $(\mathcal{I}, \gamma)$, for all $n \in \mathbb{N}$.*

Proof. Let β be the set $\nabla_{F_\mathcal{I}}(\gamma)$. By Theorem 3.14 and Corollary 3.12, we get that $\mathit{Gen}(\mathcal{I}, \gamma) = \bigcap I_{F_\mathcal{I},\beta}$, therefore the thesis follows immediately from Theorem 4.10. $\square$

In order to define the last proof-theoretic characterization (Theorem 4.17), we need to introduce a richer structure on trees. In particular, we will consider the partial order on trees defined by Courcelle in [Cou83], adapted to our context.

Consider trees $t, t' \in \mathcal{T}(\mathcal{L})$ we define

$$t \triangleleft t' \iff \mathit{dom}(t) \subseteq \mathit{dom}(t') \text{ and } \forall \alpha \in \mathit{dom}(t).\ t(\alpha) = t'(\alpha)$$

It is easy to see that $\triangleleft$ is a partial order, actually it is function inclusion. Indeed, reflexivity and transitivity follow from the same properties of $\subseteq$ and $=$, and antisymmetry can be proved noting that if $t \triangleleft t'$ and $t' \triangleleft t$ we have that $\mathit{dom}(t) = \mathit{dom}(t')$ (by antisymmetry of $\subseteq$) and $t(\alpha) = t'(\alpha)$ for all $\alpha \in \mathit{dom}(t)$, hence $t = t'$.

Intuitively, $t \triangleleft t'$ means that t can be obtained from t' by pruning some branches. Alternatively, considering trees as graphs, $t \triangleleft t'$ means that t is a subgraph of t'. In any case, $\triangleleft$ expresses a very strong relation among trees, actually too strong for our aims, hence we need to relax it a little bit.

We relax the order relation by considering what we call its *n-th approximation*, defined below. Given a tree t, we denote by $\mathit{dom}_n(t)$ the set $\{\alpha \in \mathit{dom}(t) \mid |\alpha| \leq n\}$. The n-th approximation of $\triangleleft$, denoted by $\triangleleft_n$, is defined as follows:

$$t \triangleleft_n t' \iff \mathit{dom}_n(t) \subseteq \mathit{dom}_n(t') \text{ and } \forall \alpha \in \mathit{dom}_n(t).\ t(\alpha) = t'(\alpha)$$

Intuitively $\triangleleft_n$ is identical to $\triangleleft$, but limited to nodes at level $\leq n$. We call it the n-th approximation of $\triangleleft$ since $\triangleleft_n$ is coarser than $\triangleleft$, namely, if $t \triangleleft t'$ then $t \triangleleft_n t'$ for all $n \in \mathbb{N}$. Actually we can say even more: $t \triangleleft t'$ if and only if $t \triangleleft_n t'$ for all $n \in \mathbb{N}$. Moreover, if $t \triangleleft_n t'$ then for all $k \leq n$ we have $t \triangleleft_k t'$, that is, $\triangleleft_n$ is a finer approximation than $\triangleleft_k$. Finally, note that $\triangleleft_n$ is reflexive and transitive, but it fails to be antisymmetric, because we compare only nodes until level n, hence we cannot conclude an equality between the whole trees.

We now state a result that is crucial for our proof-theoretic characterization (Theorem 4.17). Indeed, the following theorem shows that a collection of trees, behaving like a sequence of more and more precise approximations, uniquely determines a tree, which can be regarded as the limit of such sequence.

Theorem 4.13. *Let $(t_n)_{n\in\mathbb{N}}$ be a sequence of trees, such that, for all $n \in \mathbb{N}$, $t_n \lhd_n t_{n+1}$. Then, there exists a tree t such that $\forall n \in \mathbb{N}.\ t_n \lhd_n t$, and, for any other tree t' such that $\forall n \in \mathbb{N}.\ t_n \lhd_n t'$, we have $t \lhd t'$.*

Proof. We define the function $t : L \to \mathcal{L}$ where $L = \bigcup_{n\in\mathbb{N}} dom_n(t_n)$ and for all $\alpha \in L$, $t(\alpha) = t_{k(\alpha)}(\alpha)$, where $k(\alpha) = \min D_\alpha$ with $D_\alpha = \{n \in \mathbb{N} \mid \alpha \in dom_n(t_n)\}$. Note that $k(\alpha)$ is well-defined, because $D_\alpha \neq \emptyset$, since $\alpha \in L$ and, by construction of L, there is at least an index $n \in \mathbb{N}$ such that $\alpha \in dom_n(t_n)$. Moreover, L is a tree language, since if $\alpha a \in L$, then $\alpha a \in dom_n(t_n)$ for some $n \in \mathbb{N}$, that is a tree language, hence $\alpha \in dom_n(t_n) \subseteq L$. Therefore, t is a tree.
Fix now $n \in \mathbb{N}$, we have to show that $t_n \lhd_n t$. By construction we have $dom_n(t_n) \subseteq dom_n(t)$, and if $\alpha \in dom_n(t_n)$, then by construction $k(\alpha) \leq n$. Therefore, we have that $t_{k(\alpha)} \lhd_{k(\alpha)} t_n$, hence $t_{k(\alpha)}(\alpha) = t_n(\alpha)$, thus $t(\alpha) = t_n(\alpha)$ and this implies $t_n \lhd_n t$.

Consider now a tree t' such that $\forall n \in \mathbb{N}.\ t_n \lhd_n t'$. Therefore we have that for all $n \in \mathbb{N}$, $dom_n(t_n) \subseteq dom_n(t') \subseteq dom(t')$, hence $dom(t) \subseteq dom(t')$. Then, if $\alpha \in dom(t)$, there is $n \in \mathbb{N}$ such that $\alpha \in dom_n(t_n)$ and $t(\alpha) = t_n(\alpha)$. Since $t_n \lhd_n t'$, we have that $t_n(\alpha) = t'(\alpha)$, hence $t(\alpha) = t'(\alpha)$ and this implies $t \lhd t'$. □

It is easy to see that a tree t having the property expressed in the above theorem is unique. Indeed, if t and t' have that property for a sequence $(t_n)_{n\in\mathbb{N}}$, then we have both $t \lhd t'$ and $t' \lhd t$, hence $t = t'$. Therefore we denote such a tree by $\bigvee_{n\in\mathbb{N}} t_n$.

The above theorem ensures the existence of a sort of least upper bound of an ascending chain of trees: $\bigvee_{n\in\mathbb{N}} t_n$ behaves like a least upper bound, but for approximations of a partial order. However, since $\lhd_n$ is an approximation of $\lhd$, it can be shown that if $(t_n)_{n\in\mathbb{N}}$ is a chain with respect to $\lhd$, then $\bigvee_{n\in\mathbb{N}} t_n$ is indeed the least upper bound of the chain, as stated in the following corollary.

Corollary 4.14. *Let $(t_n)_{n\in\mathbb{N}}$ be a sequence of trees, such that for all $n \in \mathbb{N}$, $t_n \lhd t_{n+1}$. Then, $\bigvee_{n\in\mathbb{N}} t_n$ is the least upper bound of the sequence $(t_n)_{n\in\mathbb{N}}$ with respect to $\lhd$.*

Proof. Since $\lhd_n$ is an approximation of $\lhd$ we have that $t_n \lhd_n t_{n+1}$ for all $n \in \mathbb{N}$. Setting $t = \bigvee_{n\in\mathbb{N}} t_n$, by Theorem 4.13, we get $t_n \lhd_n t$ for all $n \in \mathbb{N}$. We have to show that t is an upper bound of $(t_n)_{n\in\mathbb{N}}$, hence consider $\alpha \in dom(t_n)$ and suppose $|\alpha| = k$. We have two cases:

- if $k \leq n$, then $\alpha \in dom_n(t_n)$, hence $\alpha \in dom_n(t) \subseteq dom(t)$ and $t_n(\alpha) = t(\alpha)$
- if $k > n$, then, since $t_n \lhd t_k$, $\alpha \in dom_k(t_k) \subseteq dom_k(t) \subseteq dom(t)$, and $t_n(\alpha) = t_k(\alpha) = t(\alpha)$

Therefore we get $t_n \lhd t$.

To show that t is the least upper bound, consider an upper bound t', hence $t_n \lhd t'$ for all $n \in \mathbb{N}$, and this implies that $t_n \lhd_n t'$ for all $n \in \mathbb{N}$. Therefore, by Theorem 4.13, we get $t \lhd t'$. □

We now consider the equivalence relations induced by each $\lhd_n$, defined as follows:

$$t \bowtie_n t' \iff t \lhd_n t' \text{ and } t' \lhd_n t$$

or, more explicitly:

$$t \bowtie_n t' \iff dom_n(t) = dom_n(t') \text{ and } \forall \alpha \in dom_n(t).\ t(\alpha) = t'(\alpha)$$

These equivalence relations are an approximation of the equality relation, indeed $t = t'$ if and only if $t \bowtie_n t'$ for all $n \in \mathbb{N}$.

The relations $\lhd_n$ and $\bowtie_n$ look very similar: they are both an approximation of another relation, they are both reflexive and transitive and they both do not care about levels higher than n. However, the fact that $\bowtie_n$ is an equivalence relation makes it different. Indeed, if $t \bowtie_n t'$, then the first n levels of t and t' are forced to be equal, while, if $t \lhd_n t'$, then the first n levels of t' must contain also those of t, but can have also additional branches. In other words, with $\bowtie_n$ we can change only the depth of the trees, while with $\lhd_n$ we can change both the depth and the breadth.

As a consequence of Theorem 4.13, we get the following theorem.

Theorem 4.15. *Let $(t_n)_{n\in\mathbb{N}}$ be a sequence of trees, such that, for all $n \in \mathbb{N}$, $t_n \bowtie_n t_{n+1}$. Then, there exists a unique tree t such that $\forall n \in \mathbb{N}.\ t_n \bowtie_n t$.*

Proof. By definition of $\bowtie_n$ we have that $t_n \lhd_n t_{n+1}$ and $t_{n+1} \lhd_n t_n$ for all $n \in \mathbb{N}$. Therefore, by Theorem 4.13, we get that there is a tree $t = \bigvee_{n\in\mathbb{N}} t_n$ such that, for all $n \in \mathbb{N}$, $t_n \lhd_n t$, hence we have only to prove that $t \lhd_n t_n$ and that t is unique. Since we know that $dom_n(t_n) \subseteq dom_n(t)$ and for all $\alpha \in dom_n(t_n)$, $t_n(\alpha) = t(\alpha)$, it is enough to show that $dom_n(t) \subseteq dom_n(t_n)$. Thus, consider $\alpha \in dom_n(t)$. By construction of t (see the proof of Theorem 4.13), there is an index $k \in \mathbb{N}$ such that $\alpha \in dom_k(t_k)$. We have two cases:

- If $k \le n$, then by hypothesis $dom_k(t_k) \subseteq dom_k(t_n) \subseteq dom_n(t_n)$, hence $\alpha \in dom_n(t_n)$.
- Otherwise, that is, if $n < k$, since $\alpha \in dom_n(t)$, we have $|\alpha| \le n < k$, hence $\alpha \in dom_n(t_k) \subseteq dom_n(t_n)$, because, by hypothesis $t_n \bowtie_n t_k$.

Therefore we have $dom_n(t) \subseteq dom_n(t_n)$ as needed.

To prove that t is unique, consider a tree t' such that $t_n \bowtie_n t'$ for all $n \in \mathbb{N}$, by transitivity we get $t \bowtie_n t'$ for all $n \in \mathbb{N}$, and this implies $t = t'$. ☐

It is well known that trees carry a complete metric space structure [AN80, Cou83] and, even if our notion of tree is more general than that adopted in these works, we can recover the same metric on our trees, using the equivalence relations introduced earlier. The metric is defined as follows:

$$d(t, t') = 2^{-h} \qquad\qquad h = \min\{n \in \mathbb{N} \mid t \not\bowtie_n t'\}$$

with assumptions $\min\emptyset = \infty$ and $2^{-\infty} = 0$. It is easy to see that a sequence $(t_n)_{n\in\mathbb{N}}$ such that $t_n \bowtie_n t_{n+1}$, like that considered in Theorem 4.15, is a Cauchy sequence in the metric space; indeed $d(t_n, t_{n+1}) \le 2^{-n}$. Therefore, such sequences converge also in the metric space, and the limit is the same. However, our notion of convergence seems to be more general: sequences like those considered in Theorem 4.13 are not necessarily Cauchy sequences, but they admit a limit in our framework. For instance, consider the sequence $(t_n)_{n\in\mathbb{N}}$ of children injective trees labelled on $\mathbb{N}$ and rooted in 0 defined[6] by

$$dom(t_0) = \{\varepsilon\} \qquad\qquad dom(t_{n+1}) = dom(t_n) \cup \{n\}$$

It is easy to check that $t_n \lhd_n t_{n+1}$, hence, by Theorem 4.13, it converges to $\bigvee_{n\in\mathbb{N}} t_n$. However, it is not a Cauchy sequence, since $d(t_n, t_{n+1}) = 2^{-1}$ for all $n \in \mathbb{N}$, and $\bigvee_{n\in\mathbb{N}} t_n$ is not a limit of the sequence in the metric space. A deeper comparison between these relations and the standard metric structure on trees will be matter of further work.

We can now introduce the concept that will allow the last proof-theoretic characterization.

Definition 4.16. Let $(\mathcal{I}, \gamma)$ be an inference system with coaxioms and $j \in \mathcal{U}$ a judgement. Then:

[6] It is enough to provide a definition for the domains since trees are children injective.

(1) An *approximating proof sequence* for j is a sequence of proof trees $(t_n)_{n\in\mathbb{N}}$ for j such that $t_n \in \mathcal{T}_n$ and $t_n \lhd_n t_{n+1}$ for all $n \in \mathbb{N}$.
(2) A *strongly approximating proof sequence* for j is a sequence of proof trees $(t_n)_{n\in\mathbb{N}}$ for j such that $t_n \in \mathcal{T}_n$ and $t_n \bowtie_n t_{n+1}$ for all $n \in \mathbb{N}$.

Obviously every strongly approximating proof sequence is also an approximating proof sequence. Note also that all trees in these sequences are well-founded proof trees in $\mathcal{I}_{\sqcup\gamma}$. Intuitively, both notions represent the growth of a proof for j in $\mathcal{I}$ approximated using coaxioms. The difference is that trees in an approximating proof sequence can grow both in depth and in breadth, while in a strongly approximating proof sequence they can grow only in depth. We now prove our last theorem, characterizing $\mathit{Gen}(\mathcal{I},\gamma)$ in terms of (strongly) approximating proof sequences.

Theorem 4.17. *Let $(\mathcal{I}, \gamma)$ be an inference system with coaxioms and $j \in \mathcal{U}$ a judgement. Then the following are equivalent*
(1) $j \in \mathit{Gen}(\mathcal{I},\gamma)$
(2) *j has a strongly approximating proof sequence*
(3) *j has an approximating proof sequence $(t_n)_{n\in\mathbb{N}}$ such that $\bigvee_{n\in\mathbb{N}} t_n$ is a proof tree in $\mathcal{I}$.*

Proof. We assume the canonical representation for children injective trees.

$1 \Rightarrow 2$**:** We define trees $t_{j,n}$ for $j \in \mathit{Gen}(\mathcal{I},\gamma)$ and $n \in \mathbb{N}$ such that $t_{j,n}(\varepsilon) = j$ by induction on n. By Corollary 4.7, we know that every judgement $j \in \mathit{Gen}(\mathcal{I},\gamma)$ has a well-founded proof tree in $\mathcal{I}_{\sqcup\gamma}$, that is, a proof tree in $\mathcal{T}_0$ rooted in j: we select one of these trees and call it $t_{j,0}$. Furthermore, since $\mathit{Gen}(\mathcal{I},\gamma)$ is a post-fixed point, for any $j \in \mathit{Gen}(\mathcal{I},\gamma)$ we can select a rule $\frac{Pr_j}{j} \in \mathcal{I}$ with $Pr_j \subseteq \mathit{Gen}(\mathcal{I},\gamma)$; hence $t_{j,n+1}$ can be defined as follows:

$$t_{j,n+1} = \frac{\{t_{j',n} \mid j' \in Pr_j\}}{j}$$

Clearly, by construction for all $j \in \mathit{Gen}(\mathcal{I},\gamma)$ and for all $n \in \mathbb{N}$, $t_{j,n} \in \mathcal{T}_n$. We show by induction on n that for all $n \in \mathbb{N}$ and for all $j \in \mathit{Gen}(\mathcal{I},\gamma)$, $t_{j,n} \bowtie_n t_{j,n+1}$.

Base: If $n = 0$, then $\mathit{dom}_0(t_{j,0}) = \mathit{dom}_0(t_{j,1}) = \{\varepsilon\}$ and by construction $t_{j,0}(\varepsilon) = t_{j,1}(\varepsilon) = j$, hence $t_{j,0} \bowtie_0 t_{j,1}$.

Induction: We assume the thesis for $n-1$ and prove it for n, hence we have to show that $t_{j,n} \bowtie_n t_{j,n+1}$. By construction, we have

$$t_{j,n} = \frac{\{t_{j',n-1} \mid j' \in Pr_j\}}{j} \qquad\qquad t_{j,n+1} = \frac{\{t_{j',n} \mid j' \in Pr_j\}}{j}$$

By induction hypothesis, we get $t_{j',n-1} \bowtie_{n-1} t_{j',n}$ for all $j' \in Pr_j$. Therefore we have

$$\begin{aligned} \mathit{dom}_n(t_{j,n}) &= \{\varepsilon\} \cup \bigcup_{j'\in Pr_j} j'\,\mathit{dom}_{n-1}(t_{j',n-1}) \\ &= \{\varepsilon\} \cup \bigcup_{j'\in Pr_j} j'\,\mathit{dom}_{n-1}(t_{j',n}) \\ &= \mathit{dom}_n(t_{j,n+1}) \end{aligned}$$

Consider now $\alpha \in \mathit{dom}_n(t_{j,n})$, we have two cases:

$$\begin{array}{ll} \alpha = \varepsilon & t_{j,n}(\alpha) = j = t_{j,n+1}(\alpha) \\ \alpha = j'\beta,\ j' \in Pr_j & t_{j,n}(\alpha) = t_{j',n-1}(\beta) = t_{j',n}(\beta) = t_{j,n+1}(\alpha) \end{array}$$

and this shows $t_{j,n} \bowtie_n t_{j,n+1}$ as needed.

$2 \Rightarrow 3$: By hypothesis, j has a strongly approximating proof sequence $(t_n)_{n\in\mathbb{N}}$ and by Definition 4.16 it is also an approximating proof sequence. We set $t = \bigvee_{n\in\mathbb{N}} t_n$ and prove that t is a proof tree in $\mathcal{I}$ for j. By Theorem 4.15, we have that $t_n \bowtie_n t$ for all $n \in \mathbb{N}$, hence, we get $j = t_0(\varepsilon) = t(\varepsilon)$. Consider $\alpha \in \mathit{dom}(t)$ and set $n = |\alpha|+1$. By construction of t, we have that $\alpha \in \mathit{dom}_n(t_n)$ and $\mathit{chl}_t(\alpha) = \mathit{chl}_{t_n}(\alpha) \subseteq \mathit{dom}_n(t_n)$, as $t_n \bowtie_n t$. Then, since $t_n \in \mathcal{T}_n$ and $|\alpha| < n$ the rule

$$\frac{\{t(\beta) \mid \beta \in \mathit{chl}_t(\alpha)\}}{t(\alpha)}$$

is a rule in $\mathcal{I}$ by Definition 4.8, thus t is a proof tree in $\mathcal{I}$.

$3 \Rightarrow 1$: Set $t = \bigvee_{n\in\mathbb{N}} t_n$, which, by hypothesis, is a proof tree in $\mathcal{I}$ for j. Consider a node $\alpha \in \mathit{dom}(t)$, then there is $k \in \mathbb{N}$ such that $\alpha \in \mathit{dom}_k(t_k)$, and so $|\alpha| \leq k$ and $\alpha \in \mathit{dom}_m(t_m)$ for all $m \geq k$, by Definition 4.16. We define the sequence $(t^\alpha_n)_{n\in\mathbb{N}}$ such that $t^\alpha_n = t_{n+k|\alpha}$. By Proposition 4.9, we get $t^\alpha_n \in \mathcal{T}_{n+k-|\alpha|} \subseteq \mathcal{T}_n$. This observation shows that every node in t has an approximated proof tree of level n for all $n \in \mathbb{N}$, hence by Theorem 4.10 we get $j \in \mathit{Gen}(\mathcal{I}, \gamma)$. □

The last condition in Theorem 4.17 requires to build an approximating proof sequence $(t_n)_{n\in\mathbb{N}}$ and to check its limit $\bigvee_{n\in\mathbb{N}} t_n$ is a proof tree. However, for a large class of inference systems, this last requirement is always true, as formally proved below.

We say an inference system $\mathcal{I}$ is *bounded* if there exists a natural number $b \in \mathbb{N}$ such that for each rule $\dfrac{\mathit{Pr}}{c} \in \mathcal{I}$, the cardinality of Pr is less than or equal to b. Note that most common inference systems are bounded, since they are specified by a finite set of finitary meta-rules. An inference system with coaxioms $(\mathcal{I}, \gamma)$ with $\mathcal{I}$ bounded enjoys the following key property:

Proposition 4.18. *Let $(\mathcal{I}, \gamma)$ be an inference system with coaxioms where $\mathcal{I}$ is bounded, and consider an approximating proof sequence $(t_n)_{n\in\mathbb{N}}$ in $(\mathcal{I}, \gamma)$. Then $\bigvee_{n\in\mathbb{N}} t_n$ is a proof tree.*

Proof. Set $t = \bigvee_{n\in\mathbb{N}}$ and consider $\alpha \in \mathit{dom}(t)$. By definition of t, there exists $n > |\alpha|$ such that $\alpha \in \mathit{dom}_n(t_n)$ and so $\mathit{chl}_t(\alpha) = \bigcup_{k\geq n} \mathit{chl}_{t_k}(\alpha)$. Since $\mathit{dom}_k(t_k) \subseteq \mathit{dom}_k(t_{k+1})$ we have $\mathit{chl}_{t_k}(\alpha) \subseteq \mathit{chl}_{t_{k+1}}(\alpha)$, and, since all t_k are proof trees and $\mathcal{I}$ is bounded, we get that $\mathit{chl}_t(\alpha)$ is finite. As a consequence, we have that $\mathit{chl}_t(\alpha) = \mathit{chl}_{t_k}(\alpha)$ for some $k \geq n$, hence, since $|\alpha| < k$, the rule

$$\frac{\{t(\beta) \mid \beta \in \mathit{chl}_t(\alpha)\}}{t(\alpha)}$$

is a rule in $\mathcal{I}$, as $t_k \in \mathcal{T}_k$, thus t is a proof tree. □

Therefore, thanks to this property, we can rephrase Theorem 4.17 under the boundedness hypothesis as follows:

Corollary 4.19. *Let $(\mathcal{I}, \gamma)$ be an inference system with coaxioms where $\mathcal{I}$ is bounded, and $j \in \mathcal{U}$ a judgement. Then the following are equivalent*

(1) $j \in \mathit{Gen}(\mathcal{I}, \gamma)$
(2) *j has a strongly approximating proof sequence*
(3) *j has an approximating proof sequence*

5. Reasoning with coaxioms

In this section we discuss proof techniques for inference systems with coaxioms.

Assume that $\mathcal{D} = \mathit{Gen}(\mathcal{I}, \gamma)$ (for "defined") is the interpretation generated by coaxioms for some $(\mathcal{I}, \gamma)$, and that $\mathcal{S}$ (for "specification") is the intended set of judgements, called *valid* in the following.

Typically, we are interested in proving $\mathcal{S} \subseteq \mathcal{D}$ (*completeness*, that is, each valid judgement can be derived) and/or $\mathcal{D} \subseteq \mathcal{S}$ (*soundness*, that is, each derivable judgement is valid). Proving both properties amounts to say that the inference system with coaxioms actually defines the intended set of judgements.

For what follows, recall that, given an inference system with coaxioms $(\mathcal{I}, \gamma)$, the following identities hold:

$$\mathit{Gen}(\mathcal{I}, \gamma) = \mathit{CoInd}(\mathcal{I}_{\sqcap \mathit{Ind}(\mathcal{I}_{\sqcup\gamma})}) = \Delta_{F_{\mathcal{I}}}(\mathit{Ind}(\mathcal{I}_{\sqcup\gamma}))$$

Completeness proofs. To show completeness, we can use (CoInd). Indeed, since $\mathcal{D} = \Delta_{F_{\mathcal{I}}}(\mathit{Ind}(\mathcal{I}_{\sqcup\gamma}))$, if $\mathcal{S} \subseteq \mathit{Ind}(\mathcal{I}_{\sqcup\gamma})$ and $\mathcal{S}$ is a post-fixed point of $F_{\mathcal{I}}$, by (CoInd) we get that $\mathcal{S} \subseteq \mathcal{D}$. That is, using the notations of inference systems, to prove completeness it is enough to show that:

- $\mathcal{S} \subseteq \mathit{Ind}(\mathcal{I}_{\sqcup\gamma})$ and
- $\mathcal{S} \subseteq F_{\mathcal{I}}(\mathcal{S})$

We call this principle the *bounded coinduction principle*.

We illustrate the technique on the inference system with coaxioms $(\mathcal{I}, \gamma)$ which defines the judgement $\mathit{allPos}(l,\, b)$. We report here the definition from Section 2, for the reader's convenience.

$$\frac{}{\mathit{allPos}(\Lambda,\, \mathsf{T})} \qquad \frac{}{\mathit{allPos}(x{::}l,\, \mathsf{F})}\, x \leq 0 \qquad \frac{\mathit{allPos}(l,\, b)}{\mathit{allPos}(x{::}l,\, b)}\, x > 0 \qquad \overline{\overline{\mathit{allPos}(l,\, \mathsf{T})}}$$

Let $\mathcal{S}^{\mathit{allPos}}$ be the set of judgements $\mathit{allPos}(l,\, b)$ where b is T if all the elements in l are positive, F otherwise. Completeness means that the judgement $\mathit{allPos}(l,\, b)$ can be proved, for all $\mathit{allPos}(l,\, b) \in \mathcal{S}^{\mathit{allPos}}$. By the bounded coinduction principle, it is enough to show that

- $\mathcal{S}^{\mathit{allPos}} \subseteq \mathit{Ind}(\mathcal{I}_{\sqcup\gamma})$
- $\mathcal{S}^{\mathit{allPos}} \subseteq F_{\mathcal{I}}(\mathcal{S}^{\mathit{allPos}})$

To prove the first condition, we have to show that, for each $\mathit{allPos}(l,\, b) \in \mathcal{S}^{\mathit{allPos}}$, $\mathit{allPos}(l,\, b)$ has a *finite* proof tree in $\mathcal{I}_{\sqcup\gamma}$. This can be easily shown, indeed:

- If l contains a (first) non-positive element, hence
  $l = x_1{::}\ldots{::}x_n{::}x{::}l'$ with $x_i > 0$ for $i \in [1..n]$, $x \leq 0$, and $b = F$
  then we can reason by arithmetic induction on n. Indeed, for $n = 0$, $\mathit{allPos}(l,\, b)$ is the consequence of the second rule with no premises, and for $n > 0$ it is the consequence of the third rule where we can apply the induction hypothesis to the premise.
- If l contains only positive elements, hence $b = T$, then $\mathit{allPos}(l,\, b)$ is a coaxiom, hence it is the consequence of a rule with no premises in $\mathcal{I}_{\sqcup\gamma}$.

To prove the second condition, we have to show that, for each $\mathit{allPos}(l,\, b) \in \mathcal{S}^{\mathit{allPos}}$, $\mathit{allPos}(l,\, b)$ is the consequence of a rule with premises in $\mathcal{S}^{\mathit{allPos}}$. This can be easily shown, indeed:

- If $l = \Lambda$, hence $b = T$, then $allPos(\Lambda, \mathsf{T})$ is the consequence of the first rule with no premises.
- If $l = x{::}l'$ with $x \leq 0$, hence $b = F$, then $allPos(l, \mathsf{F})$ is the consequence of the second rule with no premises.
- If $l = x{::}l'$ with $x > 0$, and $b = T$, hence $allPos(l', \mathsf{T}) \in \mathcal{S}^{allPos}$, then $allPos(l, \mathsf{T})$ is the consequence of the third rule with premise $allPos(l', \mathsf{T})$, and analogously if $b = F$.

Soundness proofs. To show soundness, it is convenient to use the alternative characterization in terms of approximated proof trees given in Section 4, as detailed below. First of all, from Proposition 3.10, $\mathcal{D} \subseteq \bigcap\{F^n_{\mathcal{I}}(Ind(\mathcal{I}_{\sqcup\gamma})) \mid n \geq 0\}$. Hence, to prove $\mathcal{D} \subseteq \mathcal{S}$, it is enough to show that $\bigcap\{F^n_{\mathcal{I}}(Ind(\mathcal{I}_{\sqcup\gamma})) \mid n \geq 0\} \subseteq \mathcal{S}$. Moreover, by Theorem 4.10, for all $n \in \mathbb{N}$, judgements in $F^n_{\mathcal{I}}(Ind(\mathcal{I}_{\sqcup\gamma}))$ are those which have an approximated proof tree of level n. Hence, to prove the above inclusion, we can show that all judgements, which have an approximated proof tree of level n for each n, are in $\mathcal{S}$ or equivalently, by contraposition, that judgements, which are not in $\mathcal{S}$, that is, non-valid judgements, fail to have an approximated proof tree of level n for some n.

We illustrate the technique again on the example of *allPos*. We have to show that, for each $allPos(l, b) \notin \mathcal{S}^{allPos}$, there exists $n \geq 0$ such that $allPos(l, b)$ cannot be proved by using coaxioms at level greater than n. By cases:

- If l contains a (first) non-positive element, hence $l = x_1{::}\ldots{::}x_n{::}x{::}l'$ with $x_i > 0$ for $i \in [1..n]$, $x \leq 0$, then, assuming that coaxioms can only be used at a level greater than $n+1$, $allPos(l, b)$ can only be derived by instantiating n times the third rule, and once the second rule, hence b cannot be T.
- If l contains only positive elements, then it is immediate to see that there is no finite proof tree for $allPos(l, \mathsf{F})$.

6. Taming coaxioms: advanced examples

In this section we will present some more examples of situations where coaxioms can help to define judgements on non-well-founded structures. These more involved examples will serve for explaining how to use coaxioms, which kind of problems they can cope with, and how complex can be the interaction between coaxioms and standard rules.

6.1. Mutual recursion.

Circular definitions involving inductive and coinductive judgements have no semantics in standard inference systems, because all judgements have to be interpreted either inductively, or coinductively. The same problem arises in the context of coinductive logic programming [SBMG07], where a logic program has a well-defined semantics only if inductive and coinductive predicates can be stratified: each stratum defines only inductive or coinductive predicates (possibly defined in a mutually recursive way), and cannot depend on predicates defined in upper strata. Hence, it is possible to define the semantics of a logic program only if there are no mutually recursive definitions involving both inductive and coinductive predicates.

We have already seen that an inductive inference system corresponds to an inference system with coaxioms where there are no coaxioms, while a coinductive one corresponds to the case where coaxioms consist of all judgements in $\mathcal{U}$; however, between these two extremes, coaxioms offer many other possibilities thus allowing a finer control on the semantics of the defined judgements.

There exist cases where two or more related judgements need to be defined recursively, but for some of them the correct interpretation is inductive, while for others is coinductive [SMBG06, SBMG07, Anc13, BK16]. In such cases, coaxioms may be employed to provide the correct definition in terms of a single inference system with no stratification. However, the interaction between coaxioms and standard rules is not that easy, hence special care is required to get from the inference system the intended meaning of judgements. In order to see this, let us consider the judgement $\mathit{path0}(t)$, where t is an infinite tree[7] over $\{0, 1\}$, which holds iff there exists a path starting from the root of t and containing just 0s. Trees are represented as infinite terms of shape $\mathit{tree}(n, l)$, where $n \in \{0, 1\}$ is the root of the tree, and l is the infinite list of its direct subtrees. For instance, if t_1 and t_2 are the trees defined by the syntactic equations

$$t_1 = \mathit{tree}(0, l_1) \qquad l_1 = t_2{::}t_1{::}l_1 \qquad t_2 = \mathit{tree}(0, l_2) \qquad l_2 = \mathit{tree}(1, l_1){::}l_2$$

then we expect $\mathit{path0}(t_1)$ to hold, but not $\mathit{path0}(t_2)$.

To define $\mathit{path0}$, we introduce an auxiliary judgement $\mathit{is\_in0}(l)$ testing whether an infinite list l of trees contains a tree t such that $\mathit{path0}(t)$ holds. Intuitively, we expect $\mathit{path0}$ and $\mathit{is\_in0}$ to be interpreted coinductively and inductively, respectively; this reflects the fact that $\mathit{path0}$ checks a property universally quantified over an infinite sequence (a *safety* property in the terminology of concurrent systems): all the elements of the path must be equal to 0; on the contrary, $\mathit{is\_in0}$ checks a property existentially quantified over an infinite sequence (a *liveness* property in the terminology of concurrent systems): the list must contain a tree t with a specific property (that is, $\mathit{path0}(t)$ must hold). Driven by this intuition, one could be tempted to define the following inference system with coaxioms for all judgements of shape $\mathit{path0}(t)$, and no coaxioms for judgements of shape $\mathit{is\_in0}(l)$:

$$\frac{\mathit{is\_in0}(l)}{\mathit{path0}(\mathit{tree}(0, l))} \qquad \overline{\overline{\mathit{path0}(t)}} \qquad \frac{\mathit{path0}(t)}{\mathit{is\_in0}(t{::}l)} \qquad \frac{\mathit{is\_in0}(l)}{\mathit{is\_in0}(t{::}l)}$$

Unfortunately, because of the mutual recursion between $\mathit{is\_in0}$ and $\mathit{path0}$, the inference system above does not capture the intended behaviour: $\mathit{is\_in0}(l)$ is derivable for every infinite list of trees l, even when l does not contain a tree t with an infinite path starting from its root and containing just 0s. Indeed, the coaxiom we added is not really restrictive, because it allows the predicate $\mathit{path0}$ to be coinductive, but, since $\mathit{is\_in0}$ directly depends on $\mathit{path0}$, it is allowed to be coinductive as well.

To overcome this problem, we can break the mutual dependency between judgements, replacing the judgement $\mathit{is\_in0}$ with the more general one $\mathit{is\_in}$, such that $\mathit{is\_in}(t, l)$ holds iff the infinite list l contains the tree t. Consequently, we can define the following inference system with coaxioms:

$$\frac{\mathit{is\_in}(t, l) \quad \mathit{path0}(t)}{\mathit{path0}(\mathit{tree}(0, l))} \qquad \overline{\overline{\mathit{path0}(t)}} \qquad \frac{}{\mathit{is\_in}(t, t{::}l)} \qquad \frac{\mathit{is\_in}(t, l)}{\mathit{is\_in}(t, t'{::}l)}$$

Now the semantics of the system corresponds to the intended one, since now $\mathit{is\_in}$ does not depend on $\mathit{path0}$, hence the coaxioms do not influence the semantics of $\mathit{is\_in}$, which remains inductive as expected. Nevertheless, the semantics is well-defined without the need of stratifying the definitions into two separate inference systems.

[7] For the purpose of this example, we only consider trees with infinite depth and branching.

Following the characterization in terms of proof trees and the proof techniques provided in Section 4 and Section 5, we can sketch a proof of correctness. Let $\mathcal{S}$ be the set where elements have either shape $\mathit{path0}(t)$, where t represents a tree with an infinite path of just 0s starting from its root, or $\mathit{is\_in}(t, l)$, where l represents an infinite list containing the tree t; then a judgement belongs to $\mathcal{S}$ iff it can be derived in the inference system with coaxioms defined above.

Completeness. We first show that the set $\mathcal{S}$ is a post-fixed point, that is, it is consistent w.r.t. the inference rules, coaxioms excluded. Indeed, if t has an infinite path of 0s, then it has necessarily shape $\mathit{tree}(0, l)$, where l must contain a tree t' with an infinite path of 0s. Hence, the inference rule for $\mathit{path0}$ can be applied with premises $\mathit{is\_in}(t', l) \in \mathcal{S}$, and $\mathit{path0}(t') \in \mathcal{S}$. If an infinite list contains a tree t, then it has necessarily shape $t'{::}l$ where, either $t = t'$, and hence the axiom for $\mathit{is\_in}$ can be applied, or $t \neq t'$ and t is contained in l, and hence the inference rule for $\mathit{is\_in}$ can be applied with premise $\mathit{is\_in}(t, l) \in \mathcal{S}$.

We then show that $\mathcal{S}$ is bounded by the closure of the coaxioms. For the elements of shape $\mathit{path0}(t)$ it suffices to directly apply the corresponding coaxiom; for the elements of shape $\mathit{is\_in}(t, l)$ we can show that there exists a finite proof tree built possibly also with the coaxioms by induction on the first position (where the head of the list corresponds to 0) in the list where t occurs. If the position is 0 (base case), then $l = t{::}l'$, and the axiom can be applied; if the position is $n > 0$ (inductive step), then $l = t'{::}l'$ and t occurs in l' at position $n - 1$, therefore, by induction hypothesis, there exists a finite proof tree for $\mathit{is\_in}(t, l')$, therefore we can build a finite proof tree for $\mathit{is\_in}(t, l)$ by applying the inference rule for $\mathit{is\_in}$.

Soundness. We first observe that the only finite proof trees that can be derived for $\mathit{is\_in}(t, l)$ are obtained by application of the axiom for $\mathit{is\_in}$, hence $\mathit{is\_in}(t, l)$ holds iff there exists a finite proof tree for $\mathit{is\_in}(t, l)$ built with the inference rules for $\mathit{is\_in}$. Then, we can prove that, if $\mathit{is\_in}(t, l)$ holds, then t is contained in l by induction on the inference rules for $\mathit{is\_in}$. For the axiom (base case) the claim trivially holds, while for the other inference rule we have that if t belongs to l, then trivially t belongs to $t'{::}l$.

For the elements of shape $\mathit{path0}(t)$ we first observe that by the coaxioms they all trivially belong to the closure of the coaxioms. Then, any proof tree for $\mathit{path0}(t)$ must be infinite, because there are no axioms but only one inference rule for $\mathit{path0}$ where $\mathit{path0}$ is referred in the premises; furthermore, such a rule is applicable only if the root of the tree is 0. We have already proved that if $\mathit{is\_in}(t, l)$ is derivable, then t belongs to l, therefore we can conclude that if $\mathit{path0}(t)$ is derivable, then t contains an infinite path starting from its root, and containing just 0s.

6.2. **A numerical example.** It is well-known that real numbers in the closed interval $[0, 1]$ can be represented by infinite sequences $(d_i)_{i \in \mathbb{N}^+}$ of decimal[8] digits, where $\mathbb{N}^+$ denotes the set of all positive natural numbers. Indeed, $(d_i)_{i \in \mathbb{N}^+}$ represents the real number which is the limit of the series $\sum_{i=1}^{\infty} 10^{-i} d_i$ in the standard complete metric space of real numbers (such a limit always exists by completeness, because the associated sequence of partial sums is always a Cauchy sequence). Such a representation is not unique for all rational numbers in $[0, 1]$ (except for the bounds 0 and 1) that can be represented by a finite sequence of digits

[8] Of course the example can be generalized to any base $B \geq 2$.

followed by an infinite sequence of 0s; for instance, 0.42 can be represented either by the sequence $42\bar{0}$, or by the sequence $41\bar{9}$, where $\bar{d}$ denotes the infinite sequence containing just the digit d.

For brevity, for $r = (d_i)_{i\in\mathbb{N}^+}$, $[\![r]\!]$ denotes $\sum_{i=1}^{\infty} 10^{-i} d_i$ (that is, the real number represented by r). We want to define the judgement $add(r_1, r_2, r, c)$ which holds iff $[\![r_1]\!] + [\![r_2]\!] = [\![r]\!] + c$ with c an integer number; that is, $add(r_1, r_2, r, c)$ holds iff the addition of the two real numbers represented by the sequences r_1 and r_2 yields the real number represented by the sequence r with carry c. We will soon discover that, to get a complete definition for add, c is required to range over a proper superset of the set $\{0, 1\}$, differently from what one could initially expect.

We can define the judgement add by an inference system with coaxioms as follows. We represent a real number in $[0, 1]$ by an infinite list of decimal digits, which, therefore, can always be decomposed as $d{::}r$, where d is the first digit (corresponding to the exponent -1), and r is the rest of the list of digits. Hence, in the definition below, r, r_1, r_2 range over infinite lists of digits, d_1, d_2 range over decimal digits (between 0 and 9), c is an integer and $\div$ and mod denote the integer division, and the remainder operator, respectively.

$$\frac{add(r_1, r_2, r, c)}{add(d_1{::}r_1, d_2{::}r_2, (s \bmod 10){::}r, s \div 10)}\, s = d_1 + d_2 + c \qquad \frac{}{add(r_1, r_2, r, c)}\, c \in \{-1, 0, 1, 2\}$$

As clearly emerges from the proof of completeness provided below, besides the obvious values 0 and 1, the values -1 and 2 have to be considered for the carry to ensure a complete definition of add because both $add(\bar{0}, \bar{0}, \bar{9}, -1)$ and $add(\bar{9}, \bar{9}, \bar{0}, 2)$ hold, and, hence, should be derivable; these two judgements allow the derivation of an infinite number of other valid judgements, as, for instance, $add(1\bar{0}, 1\bar{0}, 1\bar{9}, 0)$ and $add(1\bar{9}, 1\bar{9}, 4\bar{0}, 0)$, respectively, as shown by the following infinite derivations:

$$\cfrac{\cfrac{\cfrac{\vdots}{add(\bar{0}, \bar{0}, \bar{9}, -1)}}{add(\bar{0}, \bar{0}, \bar{9}, -1)}}{add(1\bar{0}, 1\bar{0}, 1\bar{9}, 0)} \qquad \cfrac{\cfrac{\cfrac{\vdots}{add(\bar{9}, \bar{9}, \bar{0}, 2)}}{add(\bar{9}, \bar{9}, \bar{0}, 2)}}{add(1\bar{9}, 1\bar{9}, 4\bar{0}, 0)}$$

Also in this case we can sketch a proof of correctness: for all infinite sequences of decimal digits r_1, r_2 and r, and all $c \in \{-1, 0, 1, 2\}$, $add(r_1, r_2, r, c)$ is derivable iff $[\![r_1]\!] + [\![r_2]\!] = [\![r]\!] + c$.

Completeness. By the coaxioms we trivially have that each element of shape $add(r_1, r_2, r, c)$ such that $[\![r_1]\!] + [\![r_2]\!] = [\![r]\!] + c$ with $c \in \{-1, 0, 1, 2\}$ belongs to the closure of the coaxioms.

To show that the unique inference rule of the system is consistent with the set of all valid judgements, let us assume that $[\![r'_1]\!] + [\![r'_2]\!] = [\![r']\!] + c'$ with $r'_1 = d_1{::}r_1$, $r'_2 = d_2{::}r_2$, $r' = d{::}r$ and $c' \in \{-1, 0, 1, 2\}$. Let us set $s = 10c' + d$, and $c = s - d_1 - d_2$, then $s \bmod 10 = d$ and $s \div 10 = c'$, and we get the desired conclusion of the inference rule, and the side condition holds; it remains to show that $[\![r_1]\!] + [\![r_2]\!] = [\![r]\!] + c$ with $c \in \{-1, 0, 1, 2\}$.

We first observe that by the properties of limits w.r.t. the usual arithmetic operations, and by definition of $[\![-]\!]$, for all infinite sequence r of decimal digits, if $r = d{::}r'$, then $[\![r]\!] = 10^{-1}(d + [\![r']\!])$; then, from the hypotheses we get the equality $d_1 + [\![r_1]\!] + d_2 + [\![r_2]\!] = d + [\![r]\!] + 10c'$, hence $d_1 + [\![r_1]\!] + d_2 + [\![r_2]\!] = [\![r]\!] + s$, and, therefore, $[\![r_1]\!] + [\![r_2]\!] = [\![r]\!] + c$; finally,

c is an integer because it is an algebraic sum of integers, and, since $c = [\![r_1]\!] + [\![r_2]\!] - [\![r]\!]$, and $0 \le [\![r_1]\!], [\![r_2]\!], [\![r]\!] \le 1$, we get $c \in \{-1, 0, 1, 2\}$.

Soundness. Let $r'_1 = d_1{::}r_1$, $r'_2 = d_2{::}r_2$, and $r' = d{::}r$ be infinite sequences of decimal digits, and $c' \in \{-1, 0, 1, 2\}$; we note that the judgement $add(r'_1, r'_2, r', c')$ can only be derived from the unique inference rule where the premise is the judgement $add(r_1, r_2, r, c)$ with c equal to $10c' + d - d_1 - d_2$ and must range over $\{-1, 0, 1, 2\}$.

To prove soundness we show that if $[\![r'_1]\!] + [\![r'_2]\!] \neq [\![r']\!] + c'$, then the judgement $add(r'_1, r'_2, r', c')$ cannot be derived in the inference system. Let us set $\delta' = |\, [\![r']\!] + c' - [\![r'_1]\!] - [\![r'_2]\!]\,|$; obviously, under the hypothesis $[\![r'_1]\!] + [\![r'_2]\!] \neq [\![r']\!] + c'$, we get $\delta' > 0$. In particular, the following fact holds: if $\delta' \ge 4 \cdot 10^{-1}$, then $10c' + d - d_1 - d_2 \notin \{-1, 0, 1, 2\}$. Indeed, by the identity $[\![r]\!] = 10^{-1}(d + [\![r']\!])$ already used for the proof of completeness, we have that $\delta' = 10^{-1}\delta$ with $\delta = |\, [\![r]\!] + c - [\![r_1]\!] - [\![r_2]\!]\,|$, with $c = 10c' + d - d_1 - d_2$; $10^{-1}([\![r]\!] + c - [\![r_1]\!] - [\![r_2]\!]) \ge 4 \cdot 10^{-1}$ implies $c \ge 3$ ($[\![r_1]\!], [\![r_2]\!], [\![r]\!] \in [0, 1]$), and, hence, $c = 10c' + d - d_1 - d_2 \notin \{-1, 0, 1, 2\}$. On the other hand, $10^{-1}([\![r]\!] + c - [\![r_1]\!] - [\![r_2]\!]) \le -4 \cdot 10^{-1}$ implies $c \le -2$, hence $c = 10c' + d - d_1 - d_2 \notin \{-1, 0, 1, 2\}$.

By virtue of this fact, and thanks to the hypotheses, we can prove by arithmetic induction over n that for all $n \ge 1$, if $\delta' \ge 4 \cdot 10^{-n}$, then there exist only finite proof trees for $add(r'_1, r'_2, r', c')$ where the coaxioms are applied at most at depth $n - 1$. The base case is directly derived from the previously proven fact. Indeed, for $n = 1$, we can only derive $add(r'_1, r'_2, r', c')$ by directly applying the coaxiom. For the inductive step we observe that all derivation of depth greater than 1 are built applying the inference rule as first step. If such rule is applicable for deriving the conclusion $add(r'_1, r'_2, r', c')$, then we can apply the inductive hypothesis for the premise $add(r_1, r_2, r, c)$ since we have already shown that $\delta' = 10^{-1}\delta$, therefore $\delta \ge 4 \cdot 10^{-(n-1)}$.

We can now conclude by observing that if $[\![r'_1]\!] + [\![r'_2]\!] \neq [\![r']\!] + c'$, then there exists n such that $\delta' \ge 4 \cdot 10^{-n}$, therefore, by the previous result, we deduce that it is not possible to build a finite tree for $add(r'_1, r'_2, r', c')$ where the coaxioms are applied at arbitrary depth k (in particular, k is bounded by $n - 1$); therefore $add(r'_1, r'_2, r', c')$ cannot be derived in the inference system.

From the proof of soundness we can also deduce that if we let c range over $\mathbb{Z}$, then the inference system becomes unsound; for instance, it would be possible to build the following infinite proof for $add(\bar{0}, \bar{0}, \bar{0}, 1)$ where all nodes clearly belong to the closure of the coaxioms, and, hence, $add(\bar{0}, \bar{0}, \bar{0}, 1)$ would be derivable, but $[\![\bar{0}]\!] + [\![\bar{0}]\!] \neq [\![\bar{0}]\!] + 1$:

$$\dfrac{\dfrac{\vdots}{add(\bar{0}, \bar{0}, \bar{0}, 10^1)}}{add(\bar{0}, \bar{0}, \bar{0}, 10^0)}$$

6.3. **Distances and shortest paths on weighted graphs.** As we already said, another widespread non-well-founded structure are graphs. In Section 2, we have shown a first examples concerning graphs, defining the judgements $v \xrightarrow{\star} \mathcal{N}$, stating that $\mathcal{N}$ is the set of nodes reachable from v in the graph. Essentially, the proposed definition performs a visit of the graph, keeping track of all encountered nodes. The same pattern can be adopted to solve more complex problems. For instance, in this section we will deal with distances between nodes in a *weighted graph*.

$$\dfrac{\dfrac{\dfrac{}{dist(e,\, e,\, 0)} \quad \dfrac{\vdots}{dist(b,\, e,\, \delta_1)}}{dist(a,\, e,\, \delta_1)}\ \delta_1 \leq 5}{dist(c,\, e,\, 1+\delta_1)} \qquad\qquad \dfrac{\dfrac{}{dist(e,\, d,\, \infty)} \quad \dfrac{\vdots}{dist(b,\, e,\, \delta_2)}}{dist(a,\, d,\, \delta_2)}$$

Figure 2: Infinite proof trees for $dist(c,\, e,\, 1+\delta_1)$ and $dist(a,\, d,\, \delta_2)$

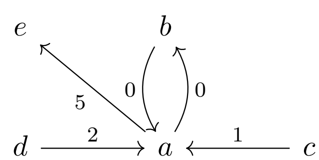
Let us introduce the notion of weights for graphs. In a graph $(V,\, adj)$ the set of edges is the set $E \subseteq V \times V$ defined by $E = \{(v,\, u) \in V \times V \mid u \in adj(v)\}$. We will often write vu for an edge $(v,\, u) \in E$. A weight function is a function $w : E \to \mathbb{N}$. Here we consider natural numbers as codomain, however we could have considered any other set of non-negative numbers. Hence, a weighted graph is a graph $(V,\, adj)$ together with a weight function w.

In a weighted graph G, the weight of a path α is the sum of the weights of the edges (counting repetitions) determined by α, we denote this by $w(\alpha)$. Note that in general the weight of a path α is different from its length, defined as the number of edges (counting repetitions) determined by the path and denoted by $\|\alpha\|$. The distance between nodes v and u is defined as the minimum weight of a path connecting v to u, it is infinite if no such path exists. Below we show the inference system with coaxioms defining the judgement $dist(v,\, u,\, \delta)$ on a weighted graph, where $\delta \in \mathbb{N} \cup \{\infty\}$.

$$\dfrac{}{dist(v,\, v,\, 0)} \qquad \dfrac{}{dist(v,\, u,\, \infty)}\ \begin{array}{l} v \neq u \\ adj(v) = \emptyset \end{array} \qquad \dfrac{}{dist(v,\, u,\, \infty)}\ v \neq u$$

$$\dfrac{dist(v_1,\, u,\, \delta_1) \quad \ldots \quad dist(v_k,\, u,\, \delta_k)}{dist(v,\, u,\, \delta)}\ \begin{array}{l} v \neq u \\ adj(v) = \{v_1, \ldots, v_k\} \neq \emptyset \\ \delta = \min\{w(vv_1) + \delta_1, \ldots, w(vv_k) + \delta_k\} \end{array}$$

In order to show that we cannot simply consider the coinductive interpretation of the above inference system, and therefore we need coaxioms, let us consider the following weighted graph:



If we would interpret the inference system coinductively we can derive judgements like $dist(c,\, e,\, \delta)$ for any $\delta \in [1..5]$ or $dist(a,\, d,\, \delta)$ for any $\delta \in \mathbb{N} \cup \{\infty\}$, as shown in Figure 2. The issue here is the cycle that, having total weight equal to 0, allows us to build cyclic proofs without increasing the value of δ. Therefore, the coaxiom is needed to filter out such proofs. Indeed, it is easy to see that it is not possible to build a finite proof tree for judgements proved in Figure 2 starting from the coaxiom.

Now we will sketch a proof of correctness. We can formulate the correctness statement as follows: $dist(v,\, u,\, \delta)$ is derivable iff δ is the minimum of $w(\alpha)$ for all paths α from v to u.

Completeness. Let us consider a judgement $dist(v,\, u,\, \delta)$ where δ is the minimum of $w(\alpha)$ for α a path from v to u. If $v = u$, then $\delta = 0$ and so the judgement is the consequence of the first axiom. If $adj(v) = \emptyset$, then $\delta = \infty$ and so the judgement is the consequence of the second axiom. Otherwise, note that $\alpha = v\beta$ where β is a path from a node $v' \in adj(v)$ to

u, hence $w(\alpha) = w(vv') + w(\beta)$. Furthermore, if there were another path β' from the node v' to u with $w(\beta') < w(\beta)$, then the path $v\beta'$ would be such that $w(v\beta') < w(\alpha) = \delta$, that is absurd, hence $\mathit{dist}(v', u, w(\beta))$ is a valid judgement. Moreover, note that for any other $v_i \in \mathit{adj}(v)$, with $\mathit{dist}(v_i, u, \delta_i)$ a valid judgement, we have $\delta \leq w(vv_i) + \delta_i$, since, otherwise, we could build a path from v to u with weight smaller than δ, that is absurd. Therefore, $\mathit{dist}(v, u, \delta)$ is the consequence of the inference rule and its premises are valid judgements, and this shows that the specification is a consistent set.

In order to show the boundedness condition, we have to build a finite proof tree for $\mathit{dist}(v, u, \delta)$ (chosen as before) using coaxioms as axioms. If there is no path from v to u, then $v \neq u$ and $\delta = \infty$, hence we can apply the coaxiom. Otherwise, there is a path α from v to u with $w(\alpha) = \delta$. We proceed by induction on the length of α. If $\|\alpha\| = 0$, then $v = u$ and $\delta = 0$, hence we can apply the first axiom. If $\|\alpha\| = n + 1$, then $\alpha = vv'\beta$ with $\|v'\beta\| = n$, $v' \in \mathit{adj}(v)$, $w(v'\beta) = \delta'$ and $\delta = w(vv') + \delta'$. By induction hypothesis, we get that $\mathit{dist}(v', u, \delta')$ is derivable, then we get a proof tree for $\mathit{dist}(v, u, \delta)$ by applying the inference rule with consequence $\mathit{dist}(v, u, \delta)$ and for each $v'' \in \mathit{adj}(v)$ a premise $\mathit{dist}(v'', u, \infty)$ if $v'' \neq v'$ and $v'' \neq u$, which is derivable by the coaxiom, $\mathit{dist}(v'', u, 0)$ if $v'' = u$, which is derivable by the first axiom, and $\mathit{dist}(v'', u, \delta')$ if $v'' = v'$, which is derivable by induction hypothesis.

Soundness. To proove soundness, we first show some useful facts.

Fact 6.1. For all proof trees t for a judgement $\mathit{dist}(v, u, \delta)$, there exists a path α from v to u with $\|\alpha\| = n$ iff there exists a node in t at depth n labelled by $\mathit{dist}(u, u, 0)$.

Proof. Let t be a proof tree for $\mathit{dist}(v, u, \delta)$. We prove separately the two implications.

- **⇒:** Let α be a path from v to u. We proceed by induction on the length of α. If $\|\alpha\| = 0$ (base case), then $v = u$, hence $\mathit{dist}(v, u, \delta)$ has been derived by applying the first axiom, and this implies $\delta = 0$. Therefore, the root of t (at depth 0) is labelled by $\mathit{dist}(u, u, 0)$. If $\|\alpha\| = n+1$ (inductive step), then $\alpha = v\beta$ where β is a path from a node v' to u of length n. Therefore, $\mathit{dist}(v, u, \delta)$ has been derived by applying the inference rule, hence there is a direct subtree of t rooted in $\mathit{dist}(v', u, \delta')$, where, by induction hypothesis, $\mathit{dist}(u, u, 0)$ occurs at depth n. Thus, in t that judgement occurs at depth $n + 1$ as needed.
- **⇐:** We proceed by induction on the depth n. If $\mathit{dist}(u, u, 0)$ occurs at depth 0 (base case), then it is the root of t, hence $v = u$ and the searched path is the singleton path u. If it occurs at depth $n + 1$ (inductive step), then the depth of t is greater than 0, hence $\mathit{dist}(v, u, \delta)$ has been derived by applying the inference rule. Therefore, $\mathit{dist}(u, u, 0)$ belongs to a direct subtree t' of t rooted in $\mathit{dist}(v', u, \delta')$ with $v' \in \mathit{adj}(v)$, and it occurs in t' at depth n. Thus, by induction hypothesis, there is a path β from v' to u of length n, hence the path $v\beta$ of length $n + 1$ connects v to u. □

Fact 6.2. For all proof trees t, t is rooted in $\mathit{dist}(v, u, \infty)$ iff all nodes in t are of shape $\mathit{dist}(v', u, \infty)$.

Proof. Consider a proof tree t. The implication ⇐ is trivial. Let us prove the other one. We can rephrase the thesis as follows: if the root of t is $\mathit{dist}(v, u, \infty)$, then, for all $n \in \mathbb{N}$, all nodes of t at depth n have shape $\mathit{dist}(v', u, \infty)$. Thus, we can proceed by induction on the depth n. If the depth is 0 (base case), then there is only one node at depth 0, which is the root $\mathit{dist}(v, u, \infty)$, hence the thesis follows immediately by hypothesis. If the depth is $n + 1$ (inductive step), then consider a node $\mathit{dist}(v', u, \delta)$ at depth $n + 1$. By definition, it is the child of a node at depth n, that, by induction hypothesis, is of shape $\mathit{dist}(v'', u, \infty)$. Therefore, the inference rule has been applied, and, since the conclusion is $\mathit{dist}(v'', u, \infty)$, all premises $\mathit{dist}(v'_i, u, \delta_i)$ with $i \in \{1, \ldots, k\}$ are such that $\delta_i = \infty$, since $\min\{\delta_1, \ldots, \delta_k\} \geq \infty$. Then, by construction, we have $v' = v'_j$ and $\delta = \delta_j$ for some $j \in \{1, \ldots, k\}$, hence we get the thesis. □

Fact 6.3. If $\mathit{dist}(v, u, \delta)$ with $\delta \in \mathbb{N}$ has an approximated proof tree (of any level), then there exists a path α from v to u such that $w(\alpha) = \delta$.

Proof. Since approximated proof trees are well-founded by Definition 4.8, we can proceed by induction on the tree structure. If the tree has a single node (base case), then, since $\delta \in \mathbb{N}$, we have necessarily applied the first axiom, hence $v = u$ and the searched path is the trivial one, which has weight 0. If the tree is compound (inductive step), we have necessarily applied the inference rule, hence there is a direct subtree rooted in $\mathit{dist}(v', u, \delta')$ with $v' \in \mathit{adj}(v)$ and $\delta = w(vv') + \delta'$. Then, by induction hypothesis, there is a path α' from v' to u with $w(\alpha') = \delta'$, hence the path $\alpha = v\alpha'$ from v to u is such that $w(\alpha) = w(vv') + w(\alpha') = \delta$, as needed. □

Fact 6.4. If $\mathit{dist}(v, u, \delta)$ has an approximated proof tree of level n, then $\delta \leq w(\alpha)$ for all paths α from v to u with $\|\alpha\| \leq n$.

Proof. First note that, if $v = u$, then the only applicable rule is the first axiom, hence $\delta = 0$ and the thesis trivially holds, since 0 is the least possible weight. So, let us assume $v \neq u$ and proceed by induction on the level n. If the level is 0 (base case), then there is no path from v to u with length 0, hence we have to show $\delta \leq \infty$, which is always true. If the level is $n + 1$ (inductive step), then, since the level is greater than 0, we have applied either the second axiom or the inference rule. In the former case, there is no path from v to u since $\mathit{adj}(v) = \emptyset$, hence the thesis trivially holds. In the latter case, assume that $\mathit{adj}(v) = \{v_1, \ldots, v_k\}$, hence the premises of the rule are $\mathit{dist}(v_i, u, \delta_i)$ for $i \in \{1, \ldots, k\}$ and $\delta = \min\{w(vv_1) + \delta_1, \ldots, w(vv_k) + \delta_k\}$. Now, consider a path α from v to u with $\|\alpha\| \leq n + 1$, hence $\alpha = v\alpha_i$ with α_i a path from v_i to u for some $v_i \in \mathit{adj}(v)$ with $\|\alpha_i\| = n$. Therefore, $w(\alpha) = w(vv_i) + w(\alpha_i)$, and, by induction hypothesis, $\delta_i \leq w(\alpha_i)$, hence we get $\delta \leq w(vv_i) + \delta_i \leq w(vv_i) + w(\alpha_i) = w(\alpha)$ as needed. □

To prove soundness, we have to show that each derivable judgement is valid. For judgements of shape $\mathit{dist}(v, u, \infty)$ the thesis follows immediately from Fact 6.1 and Fact 6.2. Hence, let us assume $\delta \in \mathbb{N}$. By Corollary 4.11, the judgement has an approximated proof tree for each level $n \in \mathbb{N}$. Hence, by Fact 6.4, $\delta \leq w(\alpha)$ for all paths α from v to u with $\|\alpha\| \leq n$ for each $n \in \mathbb{N}$, that is, simply $\delta \leq w(\alpha)$ for all paths α from v to u. Furthermore, by Fact 6.3, $\delta = w(\beta)$ for some path β from v to u, thus $\mathit{dist}(v, u, \delta)$ is valid.

The notion of distance is tightly related to paths in a graph G. Actually, from the above proofs, it is easy to see that a proof tree for a judgement $\mathit{dist}(v, u, \delta)$ explores all possible paths from v to u in the graph in order to compute the distance. Therefore, in some sense, it also finds the shortest path from v to u. Hence, with a slight variation of the inference system for the distance, we can get an inference system for the judgement $\mathit{spath}(v, u, \alpha, \delta)$ stating that α is the shortest path from v to u with weight δ. We add to paths a special value $\bot$ that represents the absence of paths between two nodes, with the assumption that $v\bot = \bot$. The definition is reported in Figure 3.

$$\frac{}{spath(v,\ v,\ v,\ 0)} \qquad \frac{}{spath(v,\ u,\ \bot,\ \infty)}\ \begin{array}{l} v \neq u \\ adj(v) = \emptyset \end{array} \qquad \overline{\overline{spath(v,\ u,\ \bot,\ \infty)}}$$

$$\frac{spath(v_1,\ u,\ \alpha_1,\ \delta_1) \quad \ldots \quad spath(v_k,\ u,\ \alpha_k,\ \delta_k)}{spath(v,\ u,\ v\alpha_i,\ w(vv_i) + \delta_i)}\ \begin{array}{l} v \neq u \\ adj(v) = \{v_1, \ldots, v_k\} \neq \emptyset \\ i = \arg\min\{w(vv_1)+\delta_1, \ldots, w(vv_k)+\delta_k\} \end{array}$$

Figure 3: Inference system with coaxioms for $spath(v,\ u,\ \alpha,\ \delta)$.

Syntax of terms and values

$$e ::= v \mid x \mid e\ e \qquad v ::= \lambda x.e \qquad v_\infty ::= v \mid \infty$$

Semantic rules

$$(\text{coax})\ \overline{\overline{e \Rightarrow \infty}} \qquad (\text{val})\ \frac{}{v \Rightarrow v} \qquad (\text{app})\ \frac{e_1 \Rightarrow \lambda x.e \quad e_2 \Rightarrow v \quad e[x \leftarrow v] \Rightarrow v_\infty}{e_1\ e_2 \Rightarrow v_\infty}$$

$$(\text{l-inf})\ \frac{e_1 \Rightarrow \infty}{e_1\ e_2 \Rightarrow \infty} \qquad (\text{r-inf})\ \frac{e_1 \Rightarrow v \quad e_2 \Rightarrow \infty}{e_1\ e_2 \Rightarrow \infty}$$

Figure 4: Call-by-value big-step semantics of λ-calculus with divergence

6.4. **Big-step operational semantics with divergence.** It is well-known that divergence cannot be captured by the big-step operational semantics of a programming language when semantic rules are interpreted inductively (that is, in the standard way) [LG09, Anc12, Anc14]. When rules are interpreted coinductively some partial result can be obtained under suitable hypotheses, but a practical way to capture divergence with a big-step operational semantics is to introduce two different forms of judgement [CC92, LG09]: one corresponds to the standard big-step evaluation relation, and is defined inductively, while the other one captures divergence, and is defined coinductively in terms of the inductive judgement, thus requiring stratification.

With coaxioms a unique judgement can be defined in a more direct and compact way. Here we show how this is possible for the standard call-by-value operational semantics of the λ-calculus, but other and more complex applications of coaxioms to model infinite behaviour of programs can be found in [ADZ17c, ADZ18]. For soundness and completeness proofs of this example we refer to [ADZ17c].

Figure 4 defines syntax, values, and semantic rules. The meta-variable v ranges over standard values, that is, lambda abstractions, while v_∞ includes also divergence, represented by ∞. The evaluation judgement has the general shape $e \Rightarrow v_\infty$, meaning that either e evaluates to a value v (when $v_\infty \neq \infty$) or diverges (when $v_\infty = \infty$).

For what concerns the semantic rules, only a coaxiom is needed, stating that every expression may diverge. This ensures that ∞ is the only allowed outcome for the evaluation of an expression which diverges; this can only happen when the corresponding derivation tree is infinite. Rule (val) is standard. Rule (app) deals with the evaluation of application when both expressions e_1 and e_2 do not diverge; the meta-variable v is required for the judgement $e_2 \Rightarrow v$ to guarantee convergence of e_2, while v_∞ is used for the result of the whole application, since the evaluation of the body of the lambda abstraction could diverge.

As usual, $e[x \leftarrow v]$ denotes capture-avoiding substitution modulo α-renaming. Rules (l-inf) and (r-inf) cover the cases when either e_1 or e_2 diverges when trying to evaluate application, assuming that a left-to-right evaluation strategy has been imposed.

As a paradigmatic example, we consider the expression $e_\Delta = (\lambda x.x\ x)\lambda x.x\ x$ and show that the only judgement derivable for it is $e_\Delta \Rightarrow \infty$. To this end, we first disregard the coaxiom and exhibit an infinite derivation tree for the judgement $e_\Delta \Rightarrow v_\infty$, which is valid for all v_∞:

$$\text{(app)}\,\frac{\text{(val)}\,\dfrac{}{\lambda x.x\ x \Rightarrow \lambda x.x\ x} \quad \text{(val)}\,\dfrac{}{\lambda x.x\ x \Rightarrow \lambda x.x\ x} \quad \text{(app)}\,\dfrac{\vdots}{(x\ x)[x \leftarrow \lambda x.x\ x] \Rightarrow v_\infty}}{(x\ x)[x \leftarrow \lambda x.x\ x] = e_\Delta \Rightarrow v_\infty}$$

In this particular case the derivation tree is also regular, but of course there are examples of divergent computations whose derivation tree is not regular. The vertical dots indicate that the derivation continues with the same repeated pattern. This derivation shows that the coinductive interpretation of the rules in Figure 4 has a non-deterministic behaviour, as happens for the coinductive interpretation of the standard big-step semantics rules [LG09, Anc12]. However, here the coaxiom plays a crucial role: it allows us to filter out all undesired values, leaving only the value ∞, which represents divergence. Indeed, if we take into acount the coaxiom, we have also to construct finite derivations, where the coaxiom can be used as an axiom.

For the expression e_Δ, we can build such finite derivations only for the judgement $e_\Delta \Rightarrow \infty$. More precisely, we can easily prove by induction that, if $e_\Delta \Rightarrow v_\infty$ has a finite proof tree, then $v_\infty = \infty$. Indeed, there are only two cases: if we have applied the coaxiom, the thesis is immediate, and, if we have applied the rule (app), then there is a premise $e_\Delta \Rightarrow v_\infty$, hence $v_\infty = \infty$ holds by induction hypothesis.

As a consequence, in the inference system with the coaxiom, the only derivable judgement for e_Δ is $e_\Delta \Rightarrow \infty$.

7. From coaxioms to corules

As already mentioned, the notion of coaxiom presented in this work has been inspired by operational models for object-oriented and logic programming proposed in [AZ12, AZ13, Anc13]. Intuitions behind such models lead us to develop a theory where rules added to an inference system in order to control its semantics have no premises. In addition, this restriction to coaxioms (rules with no premises) is also motivated by the fact that, in all examples we have considered, they are enough to get the intended semantics.

However, as we will briefly sketch in this section, all the notions presented until now smoothly generalize to the case where we can add to an inference system arbitrary rules, with a meaning analogous to the one of coaxioms. For this reason such rules are named *corules*, and are denoted in the same way as coaxioms. Furthermore, this extension seems to be needed to deal with more complex examples like those we consider in [ADZ18].

Let us introduce the concept more formally.

Definition 7.1. An *inference system with corules* is a pair $(\mathcal{I}, \mathcal{I}^{\mathsf{co}})$ where $\mathcal{I}$ and $\mathcal{I}^{\mathsf{co}}$ are inference systems. Elements of $\mathcal{I}^{\mathsf{co}}$ are called *corules*.

The semantics is defined in two steps in analogy with coaxioms:

(1) first we consider the inference system $\mathcal{I} \cup \mathcal{I}^{\mathsf{co}}$ and take its inductive interpretation $\mathit{Ind}(\mathcal{I} \cup \mathcal{I}^{\mathsf{co}})$
(2) then, we take the coinductive interpretation of $\mathcal{I}$ restricted to rules having consequence in $\mathit{Ind}(\mathcal{I} \cup \mathcal{I}^{\mathsf{co}})$

Using a notation similar to the one used for coaxioms we have that

$$\mathit{Gen}(\mathcal{I}, \mathcal{I}^{\mathsf{co}}) = \mathit{CoInd}(\mathcal{I}_{\sqcap \mathit{Ind}(\mathcal{I} \cup \mathcal{I}^{\mathsf{co}})})$$

It is easy to see that an inference systems with coaxioms is a inference system with corules where all corules have no premises.

As we have done for coaxioms, in order to characterize $\mathit{Gen}(\mathcal{I}, \mathcal{I}^{\mathsf{co}})$ as a fixed point of $F_{\mathcal{I}}$, we study the analogous construction in the general framework of complete lattices.

Consider two monotone functions $F, G : L \to L$ defined on a complete lattice $(L, \sqsubseteq)$. We can consider the monotone function $F \sqcup G$ defined as the pointwise join of F and G. Then, we define the *bounded fixed point of F generated by G*, as $\mathit{Gen}(F, G) = \Delta_F(\mu(F \sqcup G))$. This is a fixed point of F thanks to Proposition 3.6, since $\mu(F \sqcup G)$ is the least (pre-)fixed point of $F \sqcup G$ and it is easy to check that all pre-fixed points of $F \sqcup G$ are pre-fixed point of F.

Note that, in the case where G is the constant function $x \mapsto \gamma$, we get $F \sqcup G = F_{\sqcup\gamma}$, hence we have $\mathit{Gen}(F, G) = \mathit{Gen}(F, \gamma)$, that is, this construction is a generalization of the bounded fixed point generated by an element.

Then, it is easy to see that $\mathit{Gen}(\mathcal{I}, \mathcal{I}^{\mathsf{co}}) = \mathit{Gen}(F_{\mathcal{I}}, F_{\mathcal{I}^{\mathsf{co}}})$, because $F_{\mathcal{I} \cup \mathcal{I}^{\mathsf{co}}} = F_{\mathcal{I}} \cup F_{\mathcal{I}^{\mathsf{co}}}$. Therefore $\mathit{Gen}(\mathcal{I}, \mathcal{I}^{\mathsf{co}})$ is really a fixed point of $F_{\mathcal{I}}$ as expected.

On the proof-theoretic side all notions smoothly generalize to this case, indeed, we have that $j \in \mathit{Gen}(\mathcal{I}, \mathcal{I}^{\mathsf{co}})$ if and only if there is an arbitrary proof tree in $\mathcal{I}$ for j, whose nodes have a well-founded derivation in $\mathcal{I} \cup \mathcal{I}^{\mathsf{co}}$. Also the construction of approximated proof trees is the same, only the starting point changes: this time we start from the set of well-founded proof tree in $\mathcal{I} \cup \mathcal{I}^{\mathsf{co}}$.

Proof techniques introduced for coaxioms can be applied also to this more general case, in particular the bounded coinduction principle can be formulated as follows: if $\mathcal{S} \subseteq \mathcal{U}$ and
(1) $\mathcal{S} \subseteq \mathit{Ind}(\mathcal{I} \cup \mathcal{I}^{\mathsf{co}})$ and
(2) $\mathcal{S} \subseteq F_{\mathcal{I}}(\mathcal{S})$

then, $\mathcal{S} \subseteq \mathit{Gen}(\mathcal{I}, \mathcal{I}^{\mathsf{co}})$.

At this point a natural question arises: are corules more expressive than coaxioms? Here more expressive means that they are able to capture more fixed points than coaxioms. However, considering monotone functions $F, G : L \to L$, we know from Proposition 3.8, that all fixed points of F can be expressed as bounded fixed points generated by themselves, that is, if $z \in L$ is a fixed point, then $z = \mathit{Gen}(F, z)$. Therefore, since $\mathit{Gen}(F, G)$ is a fixed point of F, there must be $z \in L$, such that $\mathit{Gen}(F, G) = \mathit{Gen}(F, z)$, in particular we can choose $z = \mu(F \sqcup G)$.

Therefore at this level, adding corules does not change the expressive power of our framework. However, it seems that there are cases where corules are fundamental for expressing some definitions, as in [ADZ18]. We think that this apparently inconsistency is due to the fact that, in common practice, definitions are expressed through a finite set of finitary meta-rules, while the theory is developed for plain rules (with no variables), and the translation from meta-rules to rules is always left implicit. Hence, in order to better understand the relationship between coaxioms and corules we need a formal treatment of definitions given by meta-rules, like in [MT03, BS11], that is matter of further work.

8. Related work

Inference systems [Acz77, San11] are widely adopted to formally define operational semantics, language translations, type systems, subtyping relations, deduction calculi, and many other relevant judgements. Although inference systems have been introduced to deal with inductive definitions, in the last two decades several authors have focused on their coinductive interpretation.

Cousot and Cousot [CC92] define divergence of programs by coinductive interpretation of an inference system that extends the big-step operational semantics. The same approach is followed by other authors [HM95, Sch98, LG09]. Leroy and Grall [LG09] analyse two kinds of coinductive big-step operational semantics for the call-by-value λ-calculus, and study their relationships with the small-step and denotational semantics, and their suitability for compiler correctness proofs. Coinductive big-step semantics is used as well to reason about cyclic objects stored in memory [MT91, LR98], and to prove type soundness in Java-like languages [Anc12, Anc14]. Coinductive inference systems are also considered in the context of type analysis and subtyping for object-oriented languages [AL09, AC14].

On the programming language side, coinduction is adopted to provide primitives helping the programmer dealing with infinite objects. Examples can be found both in logic programming [SMBG06, SBMG07, KJ15] and in functional programming [Hag87, BW88]. Recently, other approaches have been proposed to support coinduction in a more flexible way. We can find contributions in all most popular paradigms: logic [Anc13, MRM14], functional [JKS13, JKS17] and object-oriented [AZ12, AZ13]. As a consequence, these proposals are more focused on operational aspects, and their corresponding implementation issues. Here we discuss those approaches most closely related to coaxioms.

The logic paradigm naturally supports coinduction. Indeed, a logic program, like an inference system, has an associated monotone function on a suitable power-set lattice, and its declarative semantics is defined as a fixed point of such function [Llo87]; hence, considering the greatest fixed point enables coinduction. To support non-well-founded objects, the semantics is defined in the power-set of the *complete Herbrand basis*, which is the set of all ground atoms built on finite and infinite terms for the program signature [Llo87].

In order to support at the same time both inductive and coinductive predicates, in [SMBG06, SBMG07] a stratified semantics is proposed: essentially the semantics is well-defined only if there is no mutual dependency beween inductive and coinductive predicates.

On the operational side, two sound resolution strategies have been proposed: *CoSLD resolution* [SMBG06, SBMG07, AD15] and *structural resolution* [KJ15]. The former strategy represents infinite objects through regular terms, that is, terms that can be represented, through unification, as a finite set of syntactic equations [AMV06], hence only cyclic objects are supported. Then, the resolution is essentially based on a loop detection mechanism and accepts all cyclic derivations. On the other hand, the latter adopts a lazy approach, working with finite approximation of infinite objects, hence it requires programs to be *productive*, in order to ensure it is able to construct such finite approximation. Differently from CoSLD, structural resolution can accept also non-cyclic derivations, but cannot deal with non-productive programs, while coSLD can.

Since coSLD aims to capture all coinductively derivable atoms, it accepts all cyclic proofs, but in some cases this behaviour is too rigid. To allow more flexible behaviours, in [Anc13, MRM14] other operational models are provided. In particular, the notion of finally clause, introduce by Ancona [Anc13], allows the programmer to specify a fact that should be resolved when a cycle is detected, instead of simply accepting the atom. In this way, predicates that are neither purely inductive nor purely coinductive can be defined and used in a logic program.

The notion of finally clause has inspired coaxioms as described in the introduction. However, despite the existing strong correlation with coaxioms, the semantics of finally clauses does not always coincide with a fixed point of the inference operator induced by the program. This is a relevant difference with coaxioms, that, instead, always generate a fixed point.

To overcome this issue of the finally clause, we have designed an extension of coinductive logic programming supporting coaxioms, in this context called *cofacts* [ADZ17a]. Here the declarative semantics is based on the bounded fixed point, and the resolution procedure is a combination of standard SLD and coSLD resolutions: when the latter discovers a loop, then a standard SLD resolution is triggered, which takes into account also cofacts. We have also implemented a prototype meta-interpreter in SWI-Prolog[9].

In the object-oriented paradigm cyclic objects are usually managed relying on imperative features, thus the language does not provide any native support for computing with such objects. The programmer has to implement ad-hoc machinery to deal with cyclic objects in an appropriate way, and this is often involved and error-prone.

In order to overcome these difficulties, Ancona and Zucca [AZ12, AZ13] have proposed an extension of Featherweight Java (FJ) [IPW99]: corecursive Featherweight Java (coFJ). This is a purely functional core calculus for Java-like languages supporting cyclic objects and corecursive methods.

Cyclic objects are represented by syntactic equations. They cannot be directly written by the programmer, but only built during the execution by corecursive methods. Analogously to the coSLD, each corecursive call is evaluated in an environment associating to already encountered calls a unique label. If the call is in the environment, then the associated label is returned as result, otherwise a fresh label is associated to the current call, and the method body is evaluated in the extended environment; finally, an equation for this new label is returned as result.

To make the mechanism more flexible, like in the logic paradigm, the authors introduce a with clause, which is an expression that will be evaluated when a cycle is detected, instead of simply returning the label, and this provides support for methods that are neither purely recursive nor purely corecursive. Again like in the logic paradigm, this feature has inspired coaxioms and is strongly related to them, however the semantics of with clauses may not always correspond to a fixed point, while coaxioms always generate a fixed point.

9. Conclusions

Inference systems are a general and versatile framework that is well-known and widely used. They allow to define several kinds of judgements, from operational semantics to type systems, from deduction calculi to language translations. They can also serve as theory to reason about recursive definitions, providing a rigorous semantics in a quite simple way.

However, standard inference systems suffer from a strong rigidity: their interpretation is dichotomous, either inductive (the least one) or coinductive (the greatest one), but what

[9]Available at `http://www.disi.unige.it/person/AnconaD/Software/co-facts.zip`

can we do if we need something in the middle? One may wonder whether this is a real issue, but the examples we have provided shows that there are many interesting cases in which we need a fixed point that is neither the least nor the greatest one, and standard inference systems are not able to provide such flexibility. Therefore, in this paper we have proposed an extension of inference systems, aimed to provide more flexibility in such cases, without affecting standard behaviour.

The core of this paper is the concept of inference system with coaxioms, introduced in Section 2: a generalized notion of inference system, that subsumes the standard one, supporting more flexible interpretations.

Our work originates from the operational models, closely related to each other, introduced by Ancona and Zucca [AZ12, AZ13] and Ancona [Anc13]. As already discussed, these operational semantics introduce some flexibility for interpreting predicates and functions recursively defined on non-well-founded data types. The initial objective of our work was to provide a more abstract semantics for such operational models, hence we developed a first model in [ADZ16] focused on this aim. However, the result was not satisfactory, since we managed to capture the semantics of a restricted class of definitions, with a model that was quite tricky.

Then, we decided to take a more abstract perspective, considering inference systems as reference framework. In this context we discovered the notion of coaxioms, that convinced us to be the right one. We firstly proposed it in [ADZ17b] and discussed it in more detail in the master thesis [Dag17], from which this paper is extracted.

In order to finely describe coaxioms, we have generalized the meta-theory of inference systems by providing two equivalent semantics, one based on fixed points in a complete lattice, and the other on the notion of proof tree.

On the model-theoretic side (Section 3), we have defined the *bounded fixed point* of a monotone function on a complete lattice generated by an element, which is the greatest (post-)fixed point of the corresponding one step inference operator, below the least pre-fixed point above the generator; this turned out to capture the semantics of inference systems with coaxioms (Theorem 3.14). An important property is that the bounded fixed point can be obtained as a combination of a least and a greatest fixed point of suitable functions (Proposition 3.13).

From the proof-theoretic perspective, we have provided three different equivalent semantics. All of them essentially impose a condition on coinductive proof trees[10] to be accepted, induced by coaxioms. In other words, all these conditions allow us to filter out undesired derivations. Since in literature we have not found a rigorous enough (for our aims) treatment of the standard proof-theoretic semantics of inference systems, we have developed our proof-theoretic model in more detail, starting from a very precise notion of tree (see Section 4.1).

The first characterization (Section 4.2) requires that each judgement in the tree is derivable with a well-founded proof tree in the extended inference system (the inference system where coaxioms are considered as axioms). The other two characterizations (Section 4.3) are based on the notion of *approximated proof trees of level n*, that are well-founded proof trees in the extended inference system where coaxioms can only be used at depth greater than n. We have proved that all these proof-theoretic semantics are equivalent to each other and to the fixed point semantics.

[10]Here we mean proof trees valid for the coinductive interpretation, hence both well-founded and non-well-founded proof trees.

We have also developed proof techniques to reason with coaxioms (Section 5). For completeness proofs we have generalized the standard coinduction principle, taking into account also coaxioms, while for soundness proofs we have described a technique based on approximated proof trees and reasoning by contraposition.

Finally, in Section 7 we have defined a further extension of our framework, allowing also corules, that is, rules used in the same way as coaxioms, but that can have non-empty premises.

Further work. Starting from this work, we identify three main directions for further investigations:

(1) deepening the theory of coaxioms,
(2) defining language constructs to support flexible (co)inductive definitions of data types, predicates and functions,
(3) using coaxioms/corules to model and reason about infinite behaviours of programs and systems.

In the first direction, a compelling topic for further developments is exploring other proof techniques for coaxioms, trying to extend proof techniques known for coinduction to this generalized framework. Possible examples are techniques based on parametrization [HNDV13], or up-to techniques [Pou07].

Another important goal we would like to pursue is to provide the support for coaxioms in a proof assistant, such as Agda [Thea] or Coq [Theb], to have a tool to mechanize proofs. In type theories supporting inductive and coinductive types [Hag87, APTS13, AP13, Bas18], like the one at the basis of Agda, we can implement inference systems with coaxioms, representing proof trees as a coinductive type, where each node is annotated by a finite proof tree (given by an inductive type). What would be interesting is to hide this construction, in such a way that the programmer has only to care about specifying rules and coaxioms, leaving everything else to the engine.

An open problem concerning the interpretation generated by coaxioms is its computability. It is quite obvious that in general this set is not decidable, however it could be interesting to study conditions and/or restrictions that ensure at least that it is semi-decidable. To this aim, it could be useful trying to provide another proof-theoretic characterization based on partial proof trees, that are proof trees with assumptions, and form a complete partial order.

Another question concerns the expressive power of this framework. Here for expressive power we mean how many fixed points of the inference operator we manage to capture using coaxioms. As we noticed in Section 3, all fixed points can be generated by a set of coaxioms: it is enough to take as generator the fixed point itself (Proposition 3.8). However, this sounds not very relevant, since we get something that we already have. Actually inference systems, and hence inference systems with coaxioms, are never used in the form they are regarded in the development of the meta-theory, but, rather, they are expressed using a *finite set of meta-rules*, leaving implicit the step from meta-rules to plain rules, which, instead, are considered in the meta-theory. At this level, the above question becomes more interesting, however, to deal with this problem, we first need to clarify what is an inference system in terms of meta-rules, filling in the gap between them and plain rule. To this aim, interesting starting points could be [MT03, BS11], which discuss proof systems for first-order logics with a notion of inductive and/or coinductive definition. Then, in that setting, we would be able to reason about the expressive power of the resulting framework.

Another interesting development is to investigate a variant of the model able to directly capture the definition of functions, rather than representing them as functional relations. This would be relevant to more appropriately model language constructs to support flexible corecursion in functional languages. This variant could also imply a change of framework, moving from lattice theory to domain or category theory, where the semantics of (co)recursive definitions of functions is better supported. Therefore a deeper comparison between coaxioms and category-theoretic or type-theoretic models could be useful.

In the second direction, considering language support for flexible coinduction, we have already taken the first steps by prividing a support for coaxioms in the logic paradigm [ADZ17a].

Extending the notion of coaxioms to support more flexible semantics for recursively defined functions in the object-oriented and functional paradigms is more challenging, due to the gap between the underlying theories. The simplest idea would be to view functions as relations, which are the entities managed by inference systems with coaxioms, however we have always to ensure that the generated fixed point is actually a function, and this is not always guaranteed.

For the object-oriented paradigm a starting point could be the revision of the operational semantics of coFJ [AZ12, AZ13] on the basis of the abstract model provided by coaxioms; in particular, to guarantee that the function denoted by a function definition in coFJ is actually a fixed point of the induced monotone operator. For the functional paradigm the situation is even more challenging, since we have to deal with more complex constructs such as higher order functions and pattern matching. A similar problem is addressed in [JKS13, JKS17], which could be an interesting starting point.

In the last direction, starting from the example in Section 6.4, it could be interesting to better study the capabilities of coaxioms to model non-termination. We have already done a first step in this direction in [ADZ17c], where we apply the approach sketched in Section 6.4 to an imperative FJ-like language, studying in particular application of proof techniques for coaxioms to prove the soundness of predicates (such as typing relations) with respect to the operational semantics.

A further extension in this direction would be applying coaxioms to define trace-based operational semantics [NU09], that allow to capture finer characterizations of the behaviour of non-terminating programs. In this context, corules seem to be needed to properly define the semantics, as we started studying in [ADZ18].

Acknowledgements

Coaxioms have been initially designed together with professors Davide Ancona and Elena Zucca from University of Genova, who have supervised the master thesis from which this paper is extracted; their suggestions have been essential to better structure this work and to make it clearer and more readable. Special thanks go also to professor Eugenio Moggi, who has reviewed the thesis, providing several useful comments. I also would like to thank all the reviewers for their remarks, which have been really important to improve this work.

References

[AAMV03] Peter Aczel, Jirí Adámek, Stefan Milius, and Jiri Velebil. Infinite trees and completely iterative theories: a coalgebraic view. *Theoretical Computer Science*, 300(1-3):1–45, 2003.

[AAV01] Peter Aczel, Jirí Adámek, and Jiri Velebil. A coalgebraic view of infinite trees and iteration. *Electronic Notes in Theoretical Computer Science*, 44(1):1–26, 2001.

[AC14] Davide Ancona and Andrea Corradi. Sound and complete subtyping between coinductive types for object-oriented languages. In Richard E. Jones, editor, *ECOOP 2014 - 28th European Conference on Object-Oriented Programming*, volume 8586 of *Lecture Notes in Computer Science*, pages 282–307. Springer, 2014.

[Acz77] Peter Aczel. An introduction to inductive definitions. In Jon Barwise, editor, *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic and the Foundations of Mathematics*, pages 739 – 782. Elsevier, 1977.

[Acz88] Peter Aczel. *Non-well-founded Sets*, volume 14 of *Center for the Study of Language and Information Lecture Notes*. Cambridge University Press, 1988.

[AD15] Davide Ancona and Agostino Dovier. A theoretical perspective of coinductive logic programming. *Fundamenta Informaticae*, 140(3-4):221–246, 2015.

[ADZ16] Davide Ancona, Francesco Dagnino, and Elena Zucca. Towards a model of corecursion with default. In Vladimir Klebanov, editor, *FTfJP'16 - 18th Workshop on Formal Techniques for Java-like Programs*, pages 9–14. ACM Press, 2016.

[ADZ17a] Davide Ancona, Francesco Dagnino, and Elena Zucca. Extending coinductive logic programming with co-facts. In Ekaterina Komendantskaya and John Power, editors, *CoALP-Ty'16 - First Workshop on Coalgebra, Horn Clause Logic Programming and Types*, volume 258 of *Electronic Proceedings in Theoretical Computer Science*, pages 1–18. Open Publishing Association, 2017.

[ADZ17b] Davide Ancona, Francesco Dagnino, and Elena Zucca. Generalizing inference systems by coaxioms. In Hongseok Yang, editor, *ESOP 2017 - 26th European Symposium on Programming*, volume 10201 of *Lecture Notes in Computer Science*, pages 29–55. Springer, 2017.

[ADZ17c] Davide Ancona, Francesco Dagnino, and Elena Zucca. Reasoning on divergent computations with coaxioms. *Proceedings of ACM on Programming Languages*, 1(OOPSLA), 2017.

[ADZ18] Davide Ancona, Francesco Dagnino, and Elena Zucca. Modeling infinite behaviour by corules. In Todd D. Millstein, editor, *ECOOP 2018 - 32nd European Conference on Object-Oriented Programming*, volume 109 of *LIPIcs*, pages 21:1–21:31. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

[AJ94] Samson Abramsky and Achim Jung. *Domain Theory*, volume 3 of *Handbook of Logic in Computer Science*, pages 1–168. Oxford University Press, 1994.

[AL09] Davide Ancona and Giovanni Lagorio. Coinductive type systems for object-oriented languages. In Sophia Drossopoulou, editor, *ECOOP 2009 - 23th European Conference on Object-Oriented Programming*, volume 5653 of *Lecture Notes in Computer Science*, pages 2–26. Springer, 2009.

[ALM$^+$15] Jirí Adámek, Paul Blain Levy, Stefan Milius, Lawrence S. Moss, and Lurdes Sousa. On final coalgebras of power-set functors and saturated trees. *Applied Categorical Structures*, 23(4):609–641, 2015.

[AMV06] Jirí Adámek, Stefan Milius, and Jiri Velebil. Iterative algebras at work. *Mathematical Structures in Computer Science*, 16(6):1085–1131, 2006.

[AN80] André Arnold and Maurice Nivat. The metric space of infinite trees. algebraic and topological properties. *Fundamenta Informaticae*, 3(4):445–476, 1980.

[Anc12] Davide Ancona. Soundness of object-oriented languages with coinductive big-step semantics. In James Noble, editor, *ECOOP 2012 - 26th European Conference on Object-Oriented Programming*, volume 7313 of *Lecture Notes in Computer Science*, pages 459–483. Springer, 2012.

[Anc13] Davide Ancona. Regular corecursion in Prolog. *Computer Languages, Systems & Structures*, 39(4):142–162, 2013.

[Anc14] Davide Ancona. How to prove type soundness of Java-like languages without forgoing big-step semantics. In David J. Pearce, editor, *FTfJP'14 - 16th Workshop on Formal Techniques for Java-like Programs*, pages 1:1–1:6. ACM Press, 2014.

[AP13] Andreas Abel and Brigitte Pientka. Wellfounded recursion with copatterns: a unified approach to termination and productivity. In Greg Morrisett and Tarmo Uustalu, editors, *ICFP'13 - 18th ACM International Conference on Functional Programming*, pages 185–196. ACM Press, 2013.

[APTS13] Andreas Abel, Brigitte Pientka, David Thibodeau, and Anton Setzer. Copatterns: programming infinite structures by observations. In Roberto Giacobazzi and Radhia Cousot, editors, *POPL'13*

- *40th Annual ACM Symposium on Principles of Programming Languages*, pages 27–38. ACM Press, 2013.

[AZ12] Davide Ancona and Elena Zucca. Corecursive Featherweight Java. In Wei-Ngan Chin and Aquinas Hobor, editors, *FTfJP'12 - 14th Workshop on Formal Techniques for Java-like Programs*, pages 3–10. ACM Press, 2012.

[AZ13] Davide Ancona and Elena Zucca. Safe corecursion in coFJ. In Werner Dietl, editor, *FTfJP'13 - 15th Workshop on Formal Techniques for Java-like Programs*, pages 2:1–2:7. ACM Press, 2013.

[Bas18] Henning Basold. Mixed inductive-coinductive reasoning: Types, programs and logic. Phd thesis, Radboud University Nijmegen, 2018.

[BK16] Henning Basold and Ekaterina Komendantskaya. Models of inductive-coinductive logic programs. In *CoALP-Ty'16 - First Workshop on Coalgebra, Horn Clause Logic Programming and Types*, 2016.

[Bro05] James Brotherston. Cyclic proofs for first-order logic with inductive definitions. In *TABLEAUX 2005 - International Conference on Automated Reasoning with Analytic Tableaux and Related Methods*, pages 78–92, 2005.

[BS11] James Brotherston and Alex Simpson. Sequent calculi for induction and infinite descent. *Journal of Logic and Computation*, 21(6):1177–1216, 2011.

[BW88] Richard S. Bird and Philip Wadler. *Introduction to functional programming*. Prentice Hall International series in computer science. Prentice Hall, 1988.

[CC92] Patrick Cousot and Radhia Cousot. Inductive definitions, semantics and abstract interpretations. In Ravi Sethi, editor, *POPL'92 - 19th Annual ACM Symposium on Principles of Programming Languages*, pages 83–94. ACM Press, 1992.

[Cou83] Bruno Courcelle. Fundamental properties of infinite trees. *Theoretical Computer Science*, 25:95–169, 1983.

[Dag17] Francesco Dagnino. Generalizing inference systems by coaxioms. Technical report, DIBRIS, University of Genova, 2017. Master thesis supervised by Davide Ancona and Elena Zucca.

[DP02] Brian A. Davey and Hilary A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2 edition, 2002.

[Hag87] Tatsuya Hagino. A typed lambda calculus with categorical type constructors. In David H. Pitt, Axel Poigné, and David E. Rydeheard, editors, *Category Theory and Computer Science*, volume 283 of *Lecture Notes in Computer Science*, pages 140–157. Springer, 1987.

[HM95] John Hughes and Andrew Moran. Making choices lazily. In John Williams, editor, *FPCA'95 - 7th International Conference on Functional Programming Languages and Computer Architecture*, pages 108–119. ACM Press, 1995.

[HNDV13] Chung-Kil Hur, Georg Neis, Derek Dreyer, and Viktor Vafeiadis. The power of parameterization in coinductive proof. In Roberto Giacobazzi and Radhia Cousot, editors, *POPL'13 - 40th Annual ACM Symposium on Principles of Programming Languages*, pages 193–206. ACM Press, 2013.

[IPW99] Atsushi Igarashi, Benjamin C. Pierce, and Philip Wadler. Featherweight Java: A minimal core calculus for Java and GJ. In Brent Hailpern, Linda M. Northrop, and A. Michael Berman, editors, *OOPSLA'99 - 14th ACM Conference on Object-Oriented Programming, Systems, Languages and Applications*, pages 132–146. ACM Press, 1999.

[Jac16] Bart Jacobs. *Introduction to Coalgebra: Towards Mathematics of States and Observation*, volume 59 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2016.

[JKS13] Jean-Baptiste Jeannin, Dexter Kozen, and Alexandra Silva. Language constructs for non-well-founded computation. In Matthias Felleisen and Philippa Gardner, editors, *ESOP 2013 - 22nd European Symposium on Programming*, volume 7792 of *Lecture Notes in Computer Science*, pages 61–80. Springer, 2013.

[JKS17] Jean-Baptiste Jeannin, Dexter Kozen, and Alexandra Silva. Cocaml: Functional programming with regular coinductive types. *Fundamenta Informaticae*, 150(3-4):347–377, 2017.

[JR97] Bart Jacobs and Jan Rutten. A tutorial on (co)algebras and (co)induction. *EATCS Bulletin*, 62:62–222, 1997.

[KJ15] Ekaterina Komendantskaya and Patricia Johann. Structural resolution: a framework for coinductive proof search and proof construction in horn clause logic. *CoRR*, abs/1511.07865, 2015.

[LG09] Xavier Leroy and Hervé Grall. Coinductive big-step operational semantics. *Information and Computation*, 207(2):284–304, 2009.

[Llo87] John W. Lloyd. *Foundations of Logic Programming, 2nd Edition*. Springer, 1987.

[LNS82] Jean-Louis Lassez, V. L. Nguyen, and Liz Sonenbergs. Fixed point theorems and semantics: A folk tale. *Inf. Process. Lett.*, 14(3):112–116, 1982.

[LR98] Xavier Leroy and François Rouaix. Security properties of typed applets. In David B. MacQueen and Luca Cardelli, editors, *POPL'98 - 25th Annual ACM Symposium on Principles of Programming Languages*, pages 391–403. ACM Press, 1998.

[MP00] Ieke Moerdijk and Erik Palmgren. Wellfounded trees in categories. *Annals of Pure and Applied Logic*, 104(1-3):189–218, 2000.

[MRM14] Theofrastos Mantadelis, Ricardo Rocha, and Paulo Moura. Tabling, rational terms, and coinduction finally together! *Theory and Practice of Logic Programming*, 14(4-5):429–443, 2014.

[MT91] Robin Milner and Mads Tofte. Co-induction in relational semantics. *Theoretical Computer Science*, 87(1):209–220, 1991.

[MT03] Alberto Momigliano and Alwen Fernanto Tiu. Induction and co-induction in sequent calculus. In *TYPES*, volume 3085 of *Lecture Notes in Computer Science*, pages 293–308. Springer, 2003.

[Nat98] James B. Nation. Notes on lattice theory. *Cambridge studies in advanced mathematics*, 60, 1998.

[NU09] Keiko Nakata and Tarmo Uustalu. Trace-based coinductive operational semantics for while. In Stefan Berghofer, Tobias Nipkow, Christian Urban, and Makarius Wenzel, editors, *TPHOLs 2009 - Theorem Proving in Higher Order Logics*, volume 5674 of *Lecture Notes in Computer Science*, pages 375–390. Springer, 2009.

[Pou07] Damien Pous. Complete lattices and up-to techniques. In Zhong Shao, editor, *APLAS 2007 - 5th Asian Symposium on Programming Languages and Systems*, volume 4807 of *Lecture Notes in Computer Science*, pages 351–366. Springer, 2007.

[Rut00] Jan Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249(1):3–80, 2000.

[San11] Davide Sangiorgi. *Introduction to Bisimulation and Coinduction*. Cambridge University Press, 2011.

[SBMG07] Luke Simon, Ajay Bansal, Ajay Mallya, and Gopal Gupta. Co-logic programming: Extending logic programming with coinduction. In Lars Arge, Christian Cachin, Tomasz Jurdzinski, and Andrzej Tarlecki, editors, *ICALP 2007 - 34th International Colloquium on Automata, Languages and Programming*, volume 4596 of *Lecture Notes in Computer Science*, pages 472–483. Springer, 2007.

[Sch98] David A. Schmidt. Trace-based abstract interpretation of operational semantics. *Lisp and Symbolic Computation*, 10(3):237–271, 1998.

[SMBG06] Luke Simon, Ajay Mallya, Ajay Bansal, and Gopal Gupta. Coinductive logic programming. In Sandro Etalle and Miroslaw Truszczynski, editors, *ICLP 2006 - 22nd International Conference on Logic Programming*, volume 4079 of *Lecture Notes in Computer Science*, pages 330–345. Springer, 2006.

[Tar55] Alfred Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5(2):285–309, 1955.

[Thea] The Agda Team. *The Agda Reference Manual*.

[Theb] The Coq Development Team. *The Coq Reference Manual*. INRIA.

[vdBM07] Benno van den Berg and Federico De Marchi. Non-well-founded trees in categories. *Annals of Pure and Applied Logic*, 146(1):40–59, 2007.