A Coinductive Approach to Proof Search

José Espírito Santo
Centro de Matemática
Universidade do Minho
Portugal

Ralph Matthes
Institut de Recherche en Informatique de Toulouse (IRIT)
C.N.R.S. and University of Toulouse
France

Luís Pinto
Centro de Matemática
Universidade do Minho
Portugal

We propose to study proof search from a coinductive point of view. In this paper, we consider intuitionistic logic and a focused system based on Herbelin's LJT for the implicational fragment. We introduce a variant of lambda calculus with potentially infinitely deep terms and a means of expressing alternatives for the description of the "solution spaces" (called Böhm forests), which are a representation of all (not necessarily well-founded but still locally well-formed) proofs of a given formula (more generally: of a given sequent).

As main result we obtain, for each given formula, the reduction of a coinductive definition of the solution space to a effective coinductive description in a finitary term calculus with a formal greatest fixed-point operator. This reduction works in a quite direct manner for the case of Horn formulas. For the general case, the naive extension would not even be true. We need to study "co-contraction" of contexts (contraction bottom-up) for dealing with the varying contexts needed beyond the Horn fragment, and we point out the appropriate finitary calculus, where fixed-point variables are typed with sequents. Co-contraction enters the interpretation of the formal greatest fixed points - curiously in the semantic interpretation of fixed-point variables and not of the fixed-point operator.

1 Introduction

Proof theory starts with the observation that a proof is more than just the truth value of a theorem. A valid theorem can have many proofs, and several of them can be interesting. In this paper, we somehow extend this to the limit and study all proofs of a given proposition. Of course, who studies proofs can also study any of them (or count them, if there are only finitely many possible proofs, or try to enumerate them in the countable case). But we do this study somehow simultaneously: we introduce a language to express the full "solution space" of proof search. And since we focus on the generative aspects of proof search, it would seem awkward to filter out failed proof attempts from the outset. This does not mean that we pursue impossible paths in the proof search (which would hardly make sense) but that we allow to follow infinite paths. An infinite path does not correspond to a successful proof, but it is a structure of locally correct proof steps. In other words, we use coinductive syntax to model *all* locally correct proof figures. This gives rise to a not necessarily wellfounded search tree. However, to keep the technical effort simpler, we have chosen a logic where this tree is finitely branching, namely the implicational fragment of intuitionistic propositional logic (with proof system given by the cut-free fragment of the system $\overline{\lambda}$ by Herbelin [3]).

Lambda terms or variants of them (expressions that may have bound variables) are a natural means to express proofs (an observation that is called *the* Curry-Howard isomorphism) in implicational logic. Proof alternatives (locally, there are only finitely many of them since our logic has no quantifier that ranges over infinitely many individuals) can be formally represented by a finite sum of such solution space expressions, and it is natural to consider those sums up to equivalence of the *set* of the alternatives. Since infinite lambda-terms are involved and since whole solution spaces are being modeled, we call these coinductive terms *Böhm forests*.

D. Baelde and A. Carayol (Eds.): Fixed Points
in Computer Science 2013 (FICS 2013)
EPTCS 126, 2013, pp. 28–43, doi:10.4204/EPTCS.126.3

By their coinductive nature, Böhm forests are no proper syntactic objects: they can be defined by all mathematical (meta-theoretic) means and are thus not "concrete", as would be expected from syntactic elements. This freedom of definition will be demonstrated and exploited in the canonical definition (Definition 6) of Böhm forests as solutions to the task of proving a sequent (a formula A in a given context Γ). In a certain sense, nothing is gained by this representation: although one can calculate on a case-by-case basis the Böhm forest for a formula of interest and see that it is described as fixed point of a system of equations (involving auxiliary Böhm forests as solutions for the other meta-variables that appear in those equations), an arbitrary Böhm forest can only be observed to any finite depth, without ever knowing whether it is the expansion of a regular cyclic graph structure (the latter being a finite structure).

Our main result is that the Böhm forests that appear as solution spaces of sequents have such a finitary nature: more precisely, they can be interpreted as semantics of a finite term in a variant of lambda calculus with alternatives and formal greatest fixed-points. For the Horn fragment (where nesting of implications to the left is disallowed), this works very smoothly without surprises (Theorem 15). The full implicational case, however, needs some subtleties concerning the fixed-point variables over which the greatest fixed points are formed and about capturing redundancy that comes from the introduction of several hypotheses that suppose the same formula. The interpretation of the finite expressions in terms of Böhm forests needs a special operation that we call *co-contraction* (contraction bottom-up). However, this operation is already definable in terms of Böhm forests. Without this operation, certain repetitive patterns in the solution spaces due to the presence of negative occurrences of implications could not be identified. With it, we obtain the finitary representation (Theorem 24).

In the next section, we quickly recapitulate syntax and typing rules of the cut-free fragment of system $\overline{\lambda}$ and also carefully describe its restriction to Horn formulas.

Section 3 has the definition of the not necessarily well-founded proofs, corresponding to a coinductive reading of $\overline{\lambda}$ (including its typing system). This is system $\overline{\lambda}^{co}$. Elimination alternatives are then added to this system (yielding the Böhm forests), which directly allow the definition of the solution spaces for the proof search for sequents. We give several examples and then show that the defined solution spaces adequately represent all the $\overline{\lambda}^{co}$ proofs of a sequent.

In Section 4, we present first the finitary system to capture the Horn fragment and then modify it to get the main result for full implicational logic.

The paper closes with discussions on related and future work in Section 5.

2 Background

We recall below the cut-free fragment of system $\overline{\lambda}$ (a.k.a. LJT), a sequent calculus for intuitionistic implication by Herbelin [3].

Letters p,q,r are used to range over a base set of propositional variables (which we also call *atoms*). Letters A,B,C are used to range over the set of formulas (= types) built from propositional variables using the implication connective (that we write $A \supset B$) that is parenthesized to the right. Often we will use the fact that any implicational formula can be uniquely decomposed as $A_1 \supset A_2 \supset \ldots \supset A_n \supset p$ with $n \geq 0$, also written in vectorial notation as $\vec{A} \supset p$. For example, if the vector $\vec{A}$ is empty the notation means simply p, and if $\vec{A} = A_1, A_2$, the notation means $A_1 \supset (A_2 \supset p)$.

The cut-free expressions of $\overline{\lambda}$ are separated into terms and lists, and are given by:

$$\begin{array}{lrcl} \text{(terms)} & t,u & ::= & xl \mid \lambda x^A.t \\ \text{(lists)} & l & ::= & \langle\rangle \mid u :: l \end{array}$$

Figure 1: Typing rules of $\overline{\lambda}$

$$\frac{}{\Gamma|\langle\rangle : p \vdash p}\ LAx \qquad \frac{\Gamma \vdash u : A \quad \Gamma|l : B \vdash p}{\Gamma|u :: l : A \supset B \vdash p}\ LIntro$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x^A.t : A \supset B}\ RIntro \qquad \frac{\Gamma|l : A \vdash p \qquad (y : A) \in \Gamma}{\Gamma \vdash yl : p}\ App$$

where a countably infinite set of variables ranged over by letters x, y, w, z is assumed. Note that in lambda-abstractions we adopt a *domain-full* presentation, annotating the bound variable with a formula. The term constructor xl is usually called *application*. Usually in the meta-level we prefer to write $x\langle t_1, \ldots, t_n\rangle$ (with $n \in \mathbb{N}_0$) to range over application constructions, and avoid speaking about lists explicitly (where obviously, the notation $\langle t_1, \ldots, t_n\rangle$ means $\langle\rangle$ if $n = 0$ and $t_1 :: l$, if $\langle t_2, \ldots, t_n\rangle$ means l). In the meta-level, when we know $n = 0$, instead of $x\langle t_1, \ldots, t_n\rangle$, we simply write the variable x.

We will view contexts Γ as finite lists of declarations $x : A$, where no variable x occurs twice. The context $\Gamma, x : A$ is obtained from Γ by adding the declaration $x : A$, and will only be written if this yields again a valid context, i. e., if x is not declared in Γ. The system has a form of sequent for each class of expressions:

$$\Gamma \vdash t : A \qquad \Gamma|l : A \vdash p.$$

Note the restriction to *atomic sequents* (the RHS formula is an atom) in the case of list sequents.

The rules of $\overline{\lambda}$ for deriving sequents are in Figure 1. Note that, as list sequents are atomic, the conclusion of the application rule is also atomic. This is not the case in Herbelin's original system [3], where list sequents can have a non-atomic formula on the RHS. In the variant of cut-free $\overline{\lambda}$ we adopted, the only rule available for deriving a term sequent whose RHS is an implication is *RIntro*. Still, our atomic restriction will not cause loss of completeness of the system for intuitionistic implication. This restriction is typically adopted in systems tailored for proof search, as for example systems of focused proofs. In fact, $\overline{\lambda}$ corresponds to a focused backward chaining system where all atoms are *asynchronous* (see e. g. Liang and Miller [7]).

We will need the following properties of $\overline{\lambda}$.

Lemma 1 (Type uniqueness) *1. Given Γ and t, there is at most one A such that $\Gamma \vdash t : A$.*

2. Given Γ, l and A, there is at most one p such that $\Gamma|l : A \vdash p$.

Proof Simultaneous induction on derivability. □

Since the empty list $\langle\rangle$ has no type index, we need to know A in the second statement of the previous lemma.

Lemma 2 (Inversion of typing) *In $\overline{\lambda}$:*

1. $\Gamma \vdash \lambda x^A.t : B$ iff there exists C s.t. $B = A \supset C$ and $\Gamma, x : A \vdash t : C$;

2. $\Gamma \vdash x\langle t_1, \ldots, t_k\rangle : A$ iff $A = p$ and there exists $\vec{B}$ s.t. $x : \vec{B} \supset p \in \Gamma$ and $\Gamma \vdash t_i : B_i$, for any i.

Proof 1. is immediate and 2. follows with the help of the fact that: $\Gamma|\langle t_1, \ldots, t_k\rangle : B \vdash p$ iff there exist $B_1, ..., B_k$ s.t. $B = B_1 \supset ... \supset B_k \supset p$ and, for any i, $\Gamma \vdash t_i : B_i$ (proved by induction on k). □

Figure 2: Typing rules of $\overline{\lambda}_{\mathsf{Horn}}$

$$\frac{}{\Gamma|\langle\rangle : p \vdash p}\ LAx \qquad \frac{\Gamma \vdash u : p \quad \Gamma|l : H \vdash q}{\Gamma|u :: l : p \supset H \vdash q}\ LIntro$$

$$\frac{\Gamma|l : H \vdash p \qquad (y : H) \in \Gamma}{\Gamma \vdash yl : p}\ App$$

Now we identify the *Horn fragment* of cut-free $\overline{\lambda}$, that we denote by $\overline{\lambda}_{\mathsf{Horn}}$. The class of *Horn formulas* (also called *Horn clauses*) is given by the grammar:

$$\text{(Horn formulas)} \qquad H \quad ::= \quad p \,|\, p \supset H$$

where p ranges over the set of propositional variables. Note that for Horn formulas, in the vectorial notation $\vec{H} \supset p$, the vector components H_i are necessarily propositional variables, i. e., any Horn formula is of the form $\vec{q} \supset p$.

The Horn fragment is obtained by restricting sequents as follows:

1. contexts are restricted to *Horn contexts*, i. e., contexts where all formulas are Horn formulas;
2. term sequents are restricted to atomic sequents, i. e., term sequents are of the form $\Gamma \vdash t : p$.

As a consequence, the λ-abstraction construction and the rule *RIntro*, that types it, are no longer needed. The restricted typing rules are presented in Figure 2.

3 Coinductive representation of proof search in lambda-bar

We want to represent the whole search space for cut-free proofs in $\overline{\lambda}$. This is profitably done with coinductive structures. Of course, we only consider locally correct proofs. Since proof search may fail when infinite branches occur (depth-first search could be trapped there), we will consider such infinite proofs as proofs in an extended sense and represent them as well, thus we will introduce expressions that comprise all the possible well-founded and non-wellfounded proofs in cut-free $\overline{\lambda}$.

The raw syntax of these possibly non-wellfounded proofs is presented as follows

$$N ::=_{co} \lambda x^A.N \,|\, x\langle N_1, \ldots, N_k\rangle \ ,$$

yielding the (co)terms of system $\overline{\lambda}^{co}$ (read coinductively, as indicated by the index *co*). Note that instead of a formal class of lists l as in the $\overline{\lambda}$-system, we adopt here the more intuitive notation $\langle N_1, \ldots, N_k\rangle$ to represent finite lists.

Since the raw syntax is interpreted coinductively, also the typing rules have to be interpreted coinductively, which is symbolized by the double horizontal line in Figure 3, a notation that we learnt from Nakata, Uustalu and Bezem [9]. (Of course, the formulas/types stay inductive.) As expected, the restriction of the typing relation to the finite $\overline{\lambda}$-terms coincides with the typing relation of the $\overline{\lambda}$ system:

Lemma 3 *For any $t \in \overline{\lambda}$, $\Gamma \vdash t : A$ in $\overline{\lambda}$ iff $\Gamma \vdash t : A$ in $\overline{\lambda}^{co}$.*

Proof By induction on t, with the help of Lemma 2. □

Figure 3: Typing rules of $\overline{\lambda}^{co}$

$$\frac{\Gamma, x:A \vdash t:B}{\Gamma \vdash \lambda x^A.t : A \supset B}\ \mathit{RIntro} \qquad \frac{(x:B_1,\ldots,B_k \supset p) \in \Gamma \quad \Gamma \vdash N_i : B_i, i=1,\ldots,k}{\Gamma \vdash x\langle N_1,\ldots,N_k\rangle : p}\ \mathit{LVecIntro}$$

Figure 4: Extra typing rule of $\overline{\lambda}^{co}_{\Sigma}$ w. r. t. $\overline{\lambda}^{co}$

$$\frac{\Gamma \vdash E_i : p, i=1,\ldots,n}{\Gamma \vdash E_1+\cdots+E_n : p}\ \mathit{Alts}$$

Example 4 *Consider $\omega := \lambda f^{p\supset p}.\lambda x^p.N$ with $N = f\langle N\rangle$ of type p. This infinite term N is also denoted f^∞.*

It is quite common to describe elements of coinductive syntax by (systems of) fixed point equations. As a notation on the *meta-level* for unique solutions of fixed-point equations, we will use the binder ν for the solution, writing $\nu N.M$, where N typically occurs in the term M. Intuitively, $\nu N.M$ is the N s. t. $N = M$. (The letter ν indicates interpretation in coinductive syntax.)

Example 5 *ω of Example 4 can be written as $\lambda f^{p\supset p}.\lambda x^p.\nu N.f\langle N\rangle$. $\Gamma, f : p\supset p, x : p \vdash \nu N.f\langle N\rangle : p$ is seen coinductively, so we get $\Gamma \vdash \omega : (p\supset p)\supset p\supset p$.*

We now come to the representation of whole search spaces. The set of coinductive cut-free $\overline{\lambda}$-terms with finite numbers of elimination alternatives is denoted by $\overline{\lambda}^{co}_{\Sigma}$ and is given by the following grammar:

$$\begin{array}{lllll} \text{(co-terms)} & N & ::=_{co} & \lambda x^A.N \mid E_1+\cdots+E_n \\ \text{(elim. alternatives)} & E & ::=_{co} & x\langle N_1,\ldots,N_k\rangle \end{array}$$

where both $n,k \geq 0$ are arbitrary. Note that summands cannot be lambda-abstractions.[1] We will often use $\sum_i E_i$ instead of $E_1+\cdots+E_n$ if the dependency of E_i on i is clear, as well as the number of elements. Likewise, we write $\langle N_i\rangle_i$ instead of $\langle N_1,\ldots,N_k\rangle$. If $n=0$, we write $\mathbb{O}$ for $E_1+\cdots+E_n$. If $n=1$, we write E_1 for $E_1+\cdots+E_n$ (in particular this injects the category of elimination alternatives into the category of co-terms) and do as if $+$ was a binary operation on (co)terms. However, this will always have a unique reading in terms of our raw syntax of $\overline{\lambda}^{co}_{\Sigma}$. In particular, this reading makes $+$ associative and $\mathbb{O}$ its neutral element.

Co-terms of $\overline{\lambda}^{co}_{\Sigma}$ will also be called Böhm forests. Their coinductive typing rules are the ones of $\overline{\lambda}^{co}$, together with the rule given in Figure 4, where the sequents for (co)terms and elimination alternatives are not distinguished notationally.

Notice that $\Gamma \vdash \mathbb{O} : p$ for all Γ and p.

Below we consider sequents $\Gamma \Rightarrow A$ with Γ a context and A an implicational formula (corresponding to term sequents of $\overline{\lambda}$ without proof terms – in fact, $\Gamma \Rightarrow A$ is nothing but the pair consisting of Γ and A, but which is viewed as a problem description: to prove formula A in context Γ).

[1] The division into two syntactic categories also forbids the generation of an infinite sum (for which $n=2$ would suffice had the categories for N and E been amalgamated).

Definition 6 *The function $\mathscr{S}$, which takes a sequent $\Gamma \Rightarrow A$ and produces a Böhm forest which is a coinductive representation of the sequent's solution space, is given corecursively as follows: In the case of an implication,*

$$\mathscr{S}(\Gamma \Rightarrow A \supset B) := \lambda x^A.\mathscr{S}(\Gamma, x : A \Rightarrow B) \ ,$$

since RIntro is the only way to prove the implication.

In the case of an atom p, for the definition of $\mathscr{S}(\Gamma \Rightarrow p)$, let $y_i : A_i$ be the i-th variable in Γ with A_i of the form $\vec{B}_i \supset p$. Let $\vec{B}_i = B_{i,1}, \ldots, B_{i,k_i}$. Define $N_{i,j} := \mathscr{S}(\Gamma \Rightarrow B_{i,j})$. Then, $E_i := y_i\langle N_{i,j}\rangle_j$, and finally,

$$\mathscr{S}(\Gamma \Rightarrow p) := \sum_i E_i \ .$$

This is more sloppily written as

$$\mathscr{S}(\Gamma \Rightarrow p) := \sum_{y:\vec{B}\supset p \in \Gamma} y\langle \mathscr{S}(\Gamma \Rightarrow B_j)\rangle_j \ .$$

In this manner, we can even write the whole definition in one line:

$$\mathscr{S}(\Gamma \Rightarrow \vec{A} \supset p) := \lambda \vec{x} : \vec{A}. \sum_{y:\vec{B}\supset p \in \Delta} y\langle \mathscr{S}(\Delta \Rightarrow B_j)\rangle_j \quad \textit{with } \Delta := \Gamma, \vec{x} : \vec{A}$$

This is a well-formed definition: for every Γ and A, $\mathscr{S}(\Gamma \Rightarrow A)$ is a Böhm forest and as such rather a semantic object.

Lemma 7 *Given Γ and A, the typing $\Gamma \vdash \mathscr{S}(\Gamma \Rightarrow A) : A$ holds in $\overline{\lambda}^{co}_{\Sigma}$.*

Let us illustrate the function $\mathscr{S}$ at work with some examples.

Example 8 *We consider first the formula $A = (p \supset p) \supset p \supset p$ and the empty context. We have:*

$$\mathscr{S}(\Rightarrow (p \supset p) \supset p \supset p) = \lambda f^{p\supset p}.\lambda x^p.\mathscr{S}(f : p \supset p, x : p \Rightarrow p)$$

Now, observe that $\mathscr{S}(f : p \supset p, x : p \Rightarrow p) = f\langle \mathscr{S}(f : p \supset p, x : p \Rightarrow p)\rangle + x$. We identify $\mathscr{S}(f : p \supset p, x : p \Rightarrow p)$ as the solution for N of the equation $N = f\langle N\rangle + x$. Using ν as means to communicate solutions of fixed-point equations on the meta-level *as for $\overline{\lambda}^{co}$, we have*

$$\mathscr{S}(\Rightarrow (p \supset p) \supset p \supset p) = \lambda f^{p\supset p}.\lambda x^p.\nu N.f\langle N\rangle + x$$

By unfolding of the fixpoint and by making a choice at each of the elimination alternatives, we can collect *from this coterm as the finitary solutions of the sequent all the Church numerals ($\lambda f^{p\supset p}.\lambda x^p.f^n\langle x\rangle$ with $n \in \mathbb{N}_0$), together with the infinitary solution $\lambda f^{p\supset p}.\lambda x^p.f^\infty$, studied before as example for $\overline{\lambda}^{co}$ (corresponding to always making the f-choice at the elimination alternatives).*

Example 9 *We consider now an example in the Horn fragment. Let $\Gamma = x : p \supset q \supset p, y : q \supset p \supset q, z : p$ (again with $p \neq q$). Note that the solution spaces of p and q relative to this sequent are mutually dependent and they give rise to the following system of equations:*

$$\begin{array}{rcl} N_p & = & x\langle N_p, N_q\rangle + z \\ N_q & = & y\langle N_q, N_p\rangle \end{array}$$

Figure 5: Membership relations

$$\frac{\mathsf{mem}(M,N)}{\mathsf{mem}(\lambda x^A.M,\lambda x^A.N)} \qquad \frac{\mathsf{mem}_E(M,E_i)}{\mathsf{mem}(M,E_1+\cdots+E_n)}\ \text{(for some } i\text{)}$$

$$\frac{\mathsf{mem}(M_1,N_1) \quad \ldots \quad \mathsf{mem}(M_k,N_k)}{\mathsf{mem}_E(x\langle M_1,\ldots,M_k\rangle, x\langle N_1,\ldots,N_k\rangle)}$$

and so we have

$$\begin{array}{rcl} \mathscr{S}(\Gamma \Rightarrow p) & = & \nu N_p.x\langle N_p, \nu N_q.y\langle N_q, N_p\rangle\rangle + z \\ \mathscr{S}(\Gamma \Rightarrow q) & = & \nu N_q.y\langle N_q, \nu N_p.x\langle N_p, N_q\rangle + z\rangle \end{array}$$

Whereas for p we can collect one finite solution (z), for q we can only collect infinite solutions. Because in the Horn case the recursive calls of the $\mathscr{S}$ function are all relative to the same (initial) context, in this fragment the solution space of a sequent can always be expressed as a finite system of equations (one for each atom occurring in the sequent), see Theorem 15.

Example 10 *Let us consider one further example where $A = ((((p \supset q) \supset p) \supset p) \supset q) \supset q$ (a formula that can be viewed as double negation of Pierce's law, when q is viewed as absurdity). We have the following (where in sequents we omit formulas on the LHS)*

$$\begin{array}{rcl} N_0 & = & \mathscr{S}(\Rightarrow A) = \lambda x^{(((p\supset q)\supset p)\supset p)\supset q}.N_1 \\ N_1 & = & \mathscr{S}(x \Rightarrow q) = x\langle N_2\rangle \\ N_2 & = & \mathscr{S}\big(x \Rightarrow ((p \supset q) \supset p) \supset p\big) = \lambda y^{(p\supset q)\supset p}.N_3 \\ N_3 & = & \mathscr{S}(x,y \Rightarrow p) = y\langle N_4\rangle \\ N_4 & = & \mathscr{S}(x,y \Rightarrow p \supset q) = \lambda z^p.N_5 \\ N_5 & = & \mathscr{S}(x,y,z \Rightarrow q) = x\langle N_6\rangle \\ N_6 & = & \mathscr{S}\big(x,y,z \Rightarrow ((p \supset q) \supset p) \supset p\big) = \lambda y_1^{(p\supset q)\supset p}.N_7 \\ N_7 & = & \mathscr{S}(x,y,z,y_1 \Rightarrow p) = y\langle N_8\rangle + z + y_1\langle N_8\rangle \\ N_8 & = & \mathscr{S}(x,y,z,y_1 \Rightarrow p \supset q) = \lambda z_1^p.N_9 \\ N_9 & = & \mathscr{S}(x,y,z,y_1,z_1 \Rightarrow q) \end{array}$$

Now, in N_9 observe that y, y_1 both have type $(p \supset q) \supset p$ and z, z_1 both have type p, and we are back at N_5 but with the duplicates y_1 of y and z_1 of z. Later, we will call this duplication phenomenon co-contraction*, and we will give a finitary description of N_0 and, more generally, of all $\mathscr{S}(\Gamma \Rightarrow A)$, see Theorem 24. Of course, by taking the middle alternative in N_7, we obtain a finite proof, showing that A is provable in $\overline{\lambda}$.*

We now define a membership semantics for co-terms and elimination alternatives of $\overline{\lambda}_\Sigma^{co}$ in terms of sets of (co)terms in $\overline{\lambda}^{co}$.

The *membership relations* $\mathsf{mem}(M,N)$ and $\mathsf{mem}_E(M,E)$ are contained in $\overline{\lambda}^{co} \times \overline{\lambda}_\Sigma^{co}$ and $\overline{\lambda}^{co} \times E\overline{\lambda}_\Sigma^{co}$ respectively (where $E\overline{\lambda}_\Sigma^{co}$ stands for the set of elimination alternatives of $\overline{\lambda}_\Sigma^{co}$) and are given coinductively by the rules in Fig. 5.

Proposition 11 *For any $N \in \overline{\lambda}^{co}$, $\mathsf{mem}(N, \mathscr{S}(\Gamma \Rightarrow A))$ iff $\Gamma \vdash N : A$ in $\overline{\lambda}^{co}$.*

Proof "If". Consider the relations

$$\begin{array}{l} R := \{(N, \mathscr{S}(\Gamma \Rightarrow A)) \mid \Gamma \vdash N : A\} \\ R_E := \{(x\langle N_i\rangle_i, x\langle \mathscr{S}(\Gamma \Rightarrow B_i)\rangle_i) \mid (x : B_1,\ldots,B_k \supset p) \in \Gamma \wedge \Gamma \vdash x\langle N_1,\ldots,N_k\rangle : p\} \end{array}$$

It suffices to show that $R \subseteq \mathsf{mem}$, but this cannot be proven alone since mem and mem_E are defined simultaneously. We also prove $R_E \subseteq \mathsf{mem}_E$, and to prove both by coinduction on the membership relations, it suffices to show that the relations R, R_E are *backwards closed*, i. e.:

1. $(\lambda x^A.M, \lambda x^A.N) \in R$ implies $(M,N) \in R$;
2. $(M, E_1 + \cdots + E_n) \in R$ implies for some i, $(M,E_i) \in R_E$;
3. $(x\langle M_1,\ldots,M_k\rangle, x\langle N_1,\ldots,N_k\rangle) \in R_E$ implies for all i, $(M_i,N_i) \in R$

We illustrate one case. Consider $(N, \mathscr{S}(\Gamma \Rightarrow A)) \in R$, with $\mathscr{S}(\Gamma \Rightarrow A) = E_1 + \cdots + E_n$. We must show that, for some i, $(N,E_i) \in R_E$. From $\mathscr{S}(\Gamma \Rightarrow A) = E_1 + \cdots + E_n$, we must have $A = p$. Now, from $\Gamma \vdash N : p$, there must exist $(x : B_1,\ldots,B_k \supset p) \in \Gamma$ and $N_1,...,N_k$ s. t. $N = x\langle N_1,\ldots,N_k\rangle$. By definition of $\mathscr{S}(\Gamma \Rightarrow A)$, there is i s. t. $E_i = x\langle \mathscr{S}(\Gamma \Rightarrow B_1),\ldots,\mathscr{S}(\Gamma \Rightarrow B_k)\rangle$.

"Only if". By coinduction on the typing relation of $\overline{\lambda}^{co}$. This is conceptually easier than the other direction since $\vdash$ is a single coinductively defined notion. We define a relation R for which it is sufficient to prove $R \subseteq \vdash$:

$$R := \{(\Gamma,N,A) \mid \mathsf{mem}(N, \mathscr{S}(\Gamma \Rightarrow A))\}$$

Proving $R \subseteq \vdash$ by coinduction amounts to showing that R is backwards closed – with respect to the typing relation of $\overline{\lambda}^{co}$, i. e., we have to show:

1. $(\Gamma, \lambda x^A.t, A \supset B) \in R$ implies $((\Gamma, x : A), t, B) \in R$;
2. $(\Gamma, x\langle N_1,\ldots,N_k\rangle, p) \in R$ implies the existence of $B_1,\ldots,B_k$ s. t. $(x : B_1,\ldots,B_k \supset p) \in \Gamma$ and, for all $i = 1,\ldots,k$, $(\Gamma, N_i, B_i) \in R$.

We show the second case (relative to rule *LVecIntro*). So, we have $\mathsf{mem}(N, \mathscr{S}(\Gamma \Rightarrow A))$ with $N = x\langle N_1,\ldots,N_k\rangle$ and $A = p$, and we need to show that, for some $(x : B_1,\ldots,B_k \supset p) \in \Gamma$, we have, for all i, $\mathsf{mem}(N_i, \mathscr{S}(\Gamma \Rightarrow B_i))$. Since $A = p$, $\mathscr{S}(\Gamma \Rightarrow A) = E_1 + \cdots + E_n$. Hence, the second rule for mem was used to infer $\mathsf{mem}(N, \mathscr{S}(\Gamma \Rightarrow A))$, i. e., there is a j s. t. $\mathsf{mem}_E(N,E_j)$. Therefore, $E_j = x\langle M_1,\ldots,M_k\rangle$ with terms $M_1, \ldots, M_k$, and, for all i, $\mathsf{mem}(N_i,M_i)$. By the definition of $\mathscr{S}(\Gamma \Rightarrow A)$, this means that there are formulas $B_1, \ldots, B_k$ s. t. $(x : B_1,\ldots,B_k \supset p) \in \Gamma$ and, for all i, $M_i = \mathscr{S}(\Gamma \Rightarrow B_i)$. □

Example 12 *Let us consider the case of Pierce's law that is not valid intuitionistically. We have (for $p \neq q$):*

$$\mathscr{S}(\Rightarrow ((p \supset q) \supset p) \supset p) = \lambda x^{(p \supset q) \supset p}.x\langle \lambda y^p.\mathbb{O}\rangle$$

The fact that we arrived at $\mathbb{O}$ and found no elimination alternatives on the way annihilates *the co-term and implies there are no terms in the solution space of $\Rightarrow ((p \supset q) \supset p) \supset p$ (hence no proofs, not even infinite ones).*

Corollary 13 (Adequacy of the co-inductive representation of proof search in $\overline{\lambda}$) *For any $t \in \overline{\lambda}$, we have $\mathsf{mem}(t, \mathscr{S}(\Gamma \Rightarrow A))$ iff $\Gamma \vdash t : A$ (where the latter is the inductive typing relation of $\overline{\lambda}$).*

Proof By the proposition above and Lemma 3. □

4 Finitary representation of proof search in lambda-bar

In the first section we define a calculus of finitary representations. In the third section we obtain our main result (Theorem 24): given $\Gamma \Rightarrow C$, there is a finitary representation of $\mathscr{S}(\Gamma \Rightarrow C)$ in the finitary calculus. To make the proof easier to understand, we first develop in the second section the particular case of the Horn fragment.

4.1 The finitary calculus

The set of inductive cut-free $\overline{\lambda}$-terms with finite numbers of elimination alternatives, and a fixpoint operator is denoted by $\overline{\lambda}^{\mathsf{gfp}}_{\Sigma}$ and is given by the following grammar (read inductively):

$$\begin{array}{lrcl} \text{(terms)} & N & ::= & \lambda x^A.N \,|\, \mathsf{gfp}X.E_1 + \cdots + E_n \,|\, X \\ \text{(elim. alternatives)} & E & ::= & x\langle N_1,\ldots,N_k\rangle \end{array}$$

where X is assumed to range over a countably infinite set of *fixpoint variables* (letters Y, Z will also be used to range over fixpoint variables that may also be thought of as meta-variables), and where both $n,k \geq 0$ are arbitrary. Below, when we refer to *finitary terms* we have in mind the terms of $\overline{\lambda}^{\mathsf{gfp}}_{\Sigma}$. The fixed-point operator is called gfp ("greatest fixed point") to indicate that its semantics is (now) defined in terms of infinitary syntax, but there, fixed points are unique. Hence, the reader may just read this as "the fixed point".

We now give a straightforward interpretation of the formal fixed points (built with gfp) of $\overline{\lambda}^{\mathsf{gfp}}_{\Sigma}$ in terms of the coinductive syntax of $\overline{\lambda}^{co}_{\Sigma}$ (using the ν operation on the meta-level).

Definition 14 *We call* environment *a function from the set of fixpoint variables into the set of (co)terms of $\overline{\lambda}^{co}_{\Sigma}$. The interpretation of a finitary term (relative to an environment) is a (co)term of $\overline{\lambda}^{co}_{\Sigma}$ given via a family of functions $[\![-]\!]_\xi : \overline{\lambda}^{\mathsf{gfp}}_{\Sigma} \to \overline{\lambda}^{co}_{\Sigma}$ indexed by environments, which is recursively defined as follows:*

$$\begin{array}{rcl} [\![X]\!]_\xi & = & \xi(X) \\ [\![\lambda x^A.N]\!]_\xi & = & \lambda x^A.[\![N]\!]_\xi \\ [\![\mathsf{gfp}X.\sum_i E_i]\!]_\xi & = & \nu N.\sum_i [\![E_i]\!]_{\xi\cup[X\mapsto N]} \\ [\![x\langle N_1,\ldots,N_k\rangle]\!]_\xi & = & x\langle [\![N_1]\!]_\xi,\ldots,[\![N_k]\!]_\xi\rangle \end{array}$$

where the notation $\xi \cup [X \mapsto N]$ stands for the environment obtained from ξ by setting X to N.

Remark that the recursive definition above has an embedded corecursive case (pertaining to the gfp-operator). Its definition is well-formed since every elimination alternative starts with a head/application variable and the occurrences of N are thus guarded.

When a finitary term N has no free occurrences of fixpoint variables, all environments determine the same coterm, and in this case we simply write $[\![N]\!]$ to denote that coterm.

4.2 Equivalence of the representations: Horn case

Theorem 15 (Equivalence for the Horn fragment) *Let Γ be a Horn context. Then, for any atom r, there exists $N_r \in \overline{\lambda}^{\mathsf{gfp}}_{\Sigma}$ with no free occurrences of fixpoint variables such that $[\![N_r]\!] = \mathscr{S}(\Gamma \Rightarrow r)$.*

Proof

Let us assume there are k atoms occurring in $\Gamma \Rightarrow r$. We define simultaneously k functions $N_p(\overrightarrow{X:q})$ (one for each atom p occurring in $\Gamma \Rightarrow r$), parameterized by a vector of declarations of the form $X:q$. The vector is written $\overrightarrow{X:q}$ and is such that no fixpoint variable and no atom occurs twice. The simultaneous definition is by recursion on the number of atoms of $\Gamma \Rightarrow r$ not occurring in $\overrightarrow{X:q}$, and is as follows:

$$N_p(\overrightarrow{X:q}) = \begin{cases} X_i & \text{if } p = q_i \\ \mathsf{gfp}X_p.\sum_{(y:\overrightarrow{r}\supset p)\in\Gamma} y\langle N_{r_j}(\overrightarrow{X:q}, X_p:p)\rangle_j & \text{otherwise} \end{cases}$$

where vector $\overrightarrow{X:q}, X_p : p$ is obtained by adding the component $X_p : p$ to the vector $\overrightarrow{X:q}$. Observe that only fixpoint variables among the fixpoint variables declared in the vector have free occurrences in $N_p(\overrightarrow{X:q})$.

By induction on the number of atoms of (the fixed sequent) $\Gamma \Rightarrow r$ not in (the variable) $\overrightarrow{X:q}$, we prove that:

$$[\![N_p(\overrightarrow{X:q})]\!]_\xi = \mathscr{S}(\Gamma \Rightarrow p) \text{ if } \xi(X_i) = \mathscr{S}(\Gamma \Rightarrow q_i), \text{ for any } i. \tag{1}$$

Case $p = q_i$, for some i. Then,

$$LHS = [\![X_i]\!]_\xi = \xi(X_i) = \mathscr{S}(\Gamma \Rightarrow q_i) = RHS.$$

Otherwise,

$$LHS = [\![\mathsf{gfp}\, X_p. \sum_{(y:\overrightarrow{r}\supset p)\in\Gamma} y\langle N_{r_j}(\overrightarrow{X:q}, X_p : p)\rangle_j]\!]_\xi = N^\infty$$

where N^∞ is given as the unique solution of the following equation:

$$N^\infty = \sum_{(y:\overrightarrow{r}\supset p)\in\Gamma} y\langle [\![N_{r_j}(\overrightarrow{X:q}, X_p : p)]\!]_{\xi\cup[X_p\mapsto N^\infty]}\rangle_j \tag{2}$$

Now observe that, by I.H., the following equations (3) and (4) are equivalent.

$$\mathscr{S}(\Gamma \Rightarrow p) = \sum_{(y:\overrightarrow{r}\supset p)\in\Gamma} y\langle [\![N_{r_j}(\overrightarrow{X:q}, X_p : p)]\!]_{\xi\cup[X_p\mapsto \mathscr{S}(\Gamma\Rightarrow p)]}\rangle_j \tag{3}$$

$$\mathscr{S}(\Gamma \Rightarrow p) = \sum_{(y:\overrightarrow{r}\supset p)\in\Gamma} y\langle \mathscr{S}(\Gamma \Rightarrow r_j)\rangle_j \tag{4}$$

By definition of $\mathscr{S}(\Gamma \Rightarrow p)$, (4) holds; hence – because of (3) – $\mathscr{S}(\Gamma \Rightarrow p)$ is the solution N^∞ of (2), concluding the proof that $LHS = RHS$.

Finally, the theorem follows as the particular case of (1) where $p = r$ and the vector of fixpoint variable declarations is empty. □

4.3 Equivalence of the representations: full implicational case

The main difference with exhaustive proof search in the case of Horn formulas is that the backwards application of *RIntro* brings new variables into the context that may have the same type as an already existing declaration, and so, for the purpose of proof search, they should be treated the same way.

We illustrate this phenomenon with the following definition and lemma and then generalize it to the form that will be needed for the main theorem (Theorem 24).

Definition 16 *For N and E in $\overline{\lambda}^{co}_\Sigma$, we define $[x_1 + \cdots + x_n/y]N$ and $[x_1 + \cdots + x_n/y]E$ by simultaneous corecursion as follows:*

$$\begin{array}{lcll}
[x_1+\cdots+x_n/y](\lambda x^A.N) & = & \lambda x^A.[x_1+\cdots+x_n/y]N & \\
[x_1+\cdots+x_n/y]\sum_i E_i & = & \sum_i [x_1+\cdots+x_n/y]E_i & \\
[x_1+\cdots+x_n/y]\big(z\langle N_i\rangle_i\big) & = & z\langle [x_1+\cdots+x_n/y]N_i\rangle_i & \text{if } z \neq y \\
[x_1+\cdots+x_n/y]\big(y\langle N_i\rangle_i\big) & = & \sum_{1\leq j\leq n} x_j\langle [x_1+\cdots+x_n/y]N_i\rangle_i &
\end{array}$$

Lemma 17 (Co-contraction: invertibility of contraction) *If $x_1, x_2, y \notin \Gamma$, then*

$$\mathscr{S}(\Gamma, x_1 : A, x_2 : A \Rightarrow C) = [x_1 + x_2/y]\mathscr{S}(\Gamma, y : A \Rightarrow C) \ .$$

Proof The proof is omitted since Lemma 20 below is essentially a generalization of this result. □

We now capture when a context Γ' is an inessential extension of context Γ:

Definition 18 *1.* $|\Gamma| = \{A : \exists x \ s.t. (x : A) \in \Gamma\}$.

2. $\Gamma \leq \Gamma'$ *if* $\Gamma \subseteq \Gamma'$ *and* $|\Gamma| = |\Gamma'|$.

3. $(\Gamma \Rightarrow p) \leq (\Gamma' \Rightarrow p')$ *if* $\Gamma \leq \Gamma'$ *and* $p = p'$.

Let σ range over sequents of the form $\Gamma \Rightarrow p$. Thus, the last definition clause defines in general when $\sigma \leq \sigma'$.

Definition 19 *1. Let $\Gamma \leq \Gamma'$. For N and E in $\overline{\lambda}^{co}_{\Sigma}$, we define $[\Gamma'/\Gamma]N$ and $[\Gamma'/\Gamma]E$ by simultaneous corecursion as follows:*

$$\begin{array}{lcll}
[\Gamma'/\Gamma](\lambda x^A.N) & = & \lambda x^A.[\Gamma', (x : A)/\Gamma, (x : A)]N & \\
[\Gamma'/\Gamma]\sum_i E_i & = & \sum_i [\Gamma'/\Gamma]E_i & \\
[\Gamma'/\Gamma]\big(z\langle N_i\rangle_i\big) & = & z\langle [\Gamma'/\Gamma]N_i\rangle_i & \text{if } z \notin dom(\Gamma) \\
[\Gamma'/\Gamma]\big(z\langle N_i\rangle_i\big) & = & \sum_{(w:\Gamma(z))\in\Gamma'} w\langle [\Gamma'/\Gamma]N_i\rangle_i & \text{if } z \in dom(\Gamma)
\end{array}$$

2. Let $\sigma \leq \sigma'$. $[\sigma'/\sigma]N = [\Gamma'/\Gamma]N$ where $\sigma = (\Gamma \Rightarrow p)$ and $\sigma' = (\Gamma' \Rightarrow p)$. Similarly for $[\sigma'/\sigma]E$.

Lemma 20 (Co-contraction) *If $\Gamma \leq \Gamma'$ then $\mathscr{S}(\Gamma' \Rightarrow C) = [\Gamma'/\Gamma](\mathscr{S}(\Gamma \Rightarrow C))$.*

Proof Let $R := \{(\mathscr{S}(\Gamma' \Rightarrow C), [\Gamma'/\Gamma](\mathscr{S}(\Gamma \Rightarrow C))) \mid \Gamma \leq \Gamma', C \text{ arbitrary}\}$. We prove that R is backward closed relative to the canonical equivalence $=$ generated by the coinductive definition of terms of $\overline{\lambda}^{co}_{\Sigma}$ (but see the comments following the proof), whence $R \subseteq =$.

$$\mathscr{S}(\Gamma' \Rightarrow C) = \lambda z_1^{A_1} \cdots z_n^{A_n}. \sum_{(z:\vec{B}\supset p)\in\Delta'} z\langle \mathscr{S}(\Delta' \Rightarrow B_j)\rangle_j \tag{5}$$

and

$$[\Gamma'/\Gamma](\mathscr{S}(\Gamma \Rightarrow C)) = \lambda z_1^{A_1} \cdots z_n^{A_n}. \sum_{(y:\vec{B}\supset p)\in\Delta} \sum_{(w:\Delta(y))\in\Delta'} w\langle [\Delta'/\Delta]\mathscr{S}(\Delta \Rightarrow B_j)\rangle_j \tag{6}$$

where $\Delta := \Gamma \cup \{z_1 : A_1, \cdots, z_n : A_n\}$ and $\Delta' := \Gamma' \cup \{z_1 : A_1, \cdots, z_n : A_n\}$.

From $\Gamma \leq \Gamma'$ we get $\Delta \leq \Delta'$, hence

$$(\mathscr{S}(\Delta' \Rightarrow B_j), [\Delta'/\Delta]\mathscr{S}(\Delta \Rightarrow B_j)) \in R \ .$$

To conclude the proof, it suffices to show that (i) each head-variable z that is a "capability" of the summation in (5) is matched by a head-variable w that is a "capability" of the summation in (6); and (ii) vice-versa.

(i) Let $z \in dom(\Delta')$. We have to exhibit $y \in dom(\Delta)$ such that $(z : \Delta(y)) \in \Delta'$. First case: $z \in dom(\Delta)$. By $\Delta \leq \Delta'$, $(z : \Delta(z)) \in \Delta'$. So we may take $y = z$. Second and last case: $z \in \Gamma' \backslash \Gamma$. By $\Gamma \leq \Gamma'$, there is $y \in \Gamma$ such that $(z : \Gamma(y)) \in \Gamma'$. But then $(z : \Delta(y)) \in \Delta'$.

(ii) We have to show that, for all $y \in dom(\Delta)$, and all $(w : \Delta(y)) \in \Delta'$, $w \in dom(\Delta')$. But this is immediate. □

Notice that we cannot expect that the summands appear in the same order in (5) and (6). Therefore, we have to be more careful with the notion of equality of Böhm forests. It is not just bisimilarity, but we assume that the sums of elimination alternatives are treated as if they were sets of alternatives, i. e., we further assume that $+$ is symmetric and idempotent. It has been shown by Picard and the second author [10] that bisimulation up to permutations in unbounded lists of children can be managed in a coinductive type even with the interactive proof assistant Coq. In analogy, this coarser notion of equality (even abstracting away from the number of occurrences of an alternative) should not present a major obstacle for a fully formal presentation.

In the rest of the paper – in particular in Theorem 24 – we assume that sums of alternatives are treated as if they were sets.

Example 21 (Example 10 continued) *Thanks to the preceding lemma, N_9 is obtained by co-contraction from N_5:*

$$N_9 = [x:\cdot, y:(p\supset q)\supset p, z:p, y_1:(p\supset q)\supset p, z_1:p \,/\, x:\cdot, y:(p\supset q)\supset p, z:p]N_5 \ ,$$

where the type of x has been omitted. Hence, N_6, N_7, N_8 and N_9 can be eliminated, and N_5 can be expressed as the (meta-level) fixed point:

$$N_5 = \nu N.x\langle\lambda y_1^{(p\supset q)\supset p}.y\langle\lambda z_1^p.[x,y,z,y_1,z_1/x,y,z]N\rangle + z + y_1\langle\lambda z_1^p.[x,y,z,y_1,z_1/x,y,z]N\rangle\rangle \ ,$$

now missing out all types in the context substitution. Finally, we obtain the closed Böhm forest

$$\mathscr{S}(\Rightarrow A) = \lambda x^{(((p\supset q)\supset p)\supset p)\supset q}.x\langle\lambda y^{(p\supset q)\supset p}.y\langle\lambda z^p.N_5\rangle\rangle$$

The question is now how to give a finitary meaning to terms like N_5 in the example above, which are defined by fixed points over variables subject to context substitution. We might expect to use the equation defining N_5 to obtain a finitary representation in $\overline{\lambda}^{\mathsf{gfp}}_{\Sigma}$, provided context substitution is defined on this system. But how to do that? Applying say $[x,y,z,y_1,z_1/x,y,z]$ to a plain fixed-point variable cannot make much sense.

The desired finitary representation in the full implicational case is obtained by adjusting the terms of $\overline{\lambda}^{\mathsf{gfp}}_{\Sigma}$ used in the Horn case as follows:

$$\text{(terms)} \qquad N \ ::= \ (\cdots) \,|\, \mathsf{gfp}\, X^{\sigma}.E_1 + \cdots + E_n \,|\, X^{\sigma}$$

Hence fixpoint variables are "typed" with *sequents* σ.

Different free occurrences of the same X may be "typed" with different σ's, as long as a lower bound of these σ's can be found w.r.t. $\leq$ (Definition 18).

Relatively to Definition 14, an environment ξ now assigns (co)terms N of $\overline{\lambda}^{co}_{\Sigma}$ to "typed" fixpoint variables X^{σ}, provided X does not occur with two different "types" in the domain of ξ, for all X; we also change the following clauses:

$$\begin{array}{rcll} [\![X^{\sigma'}]\!]_{\xi} & = & [\sigma'/\sigma]\xi(X^{\sigma}) & \text{if } \sigma \leq \sigma' \\ [\![\mathsf{gfp}\, X^{\sigma}.\sum_i E_i]\!]_{\xi} & = & \nu N.\sum_i [\![E_i]\!]_{\xi\cup[X^{\sigma}\mapsto N]} & \end{array}$$

We will have to assign some default value to $X^{\sigma'}$ in case there is no such σ, but this will not play a role in the main result below.

Map $N_p(\overrightarrow{X:q})$ used in the proof of Theorem 15 is replaced by the following:

Definition 22 *Let $\Xi := \overrightarrow{X : \Theta \Rightarrow q}$ be a vector of $m \geq 0$ declarations $(X_i : \Theta_i \Rightarrow q_i)$ where no fixpoint variable and no sequent occurs twice. $N_{\Gamma \Rightarrow \vec{A} \supset p}(\Xi)$ is defined as follows:*

If, for some $1 \leq i \leq m$, $p = q_i$ and $\Theta_i \subseteq \Gamma$ and $|\Theta_i| = |\Delta|$, then

$$N_{\Gamma \Rightarrow \vec{A} \supset p}(\Xi) = \lambda z_1^{A_1} \cdots z_n^{A_n}.X_i^{\sigma}$$

otherwise,

$$N_{\Gamma \Rightarrow \vec{A} \supset p}(\Xi) = \lambda z_1^{A_1} \cdots z_n^{A_n}.\mathsf{gfp}\, Y^{\sigma}. \sum_{(y : \vec{B} \supset p) \in \Delta} y\langle N_{\Delta \Rightarrow B_j}(\Xi, Y : \sigma)\rangle_j$$

where, in both cases, $\Delta := \Gamma \cup \{z_1 : A_1, \cdots, z_n : A_n\}$ and $\sigma := \Delta \Rightarrow p$.

The definition of $N_p(\overrightarrow{X : q})$ in the proof of Theorem 15 was by recursion on a certain number of atoms. The following lemma spells out the measure that is recursively decreasing in the definition of $N_{\Gamma \Rightarrow C}(\Xi)$.

Lemma 23 *For all $\Gamma \Rightarrow C$, $N_{\Gamma \Rightarrow C}(\cdot)$ is well-defined, where $\cdot$ denotes the empty vector.*

Proof Let us call *recursive call* a "reduction"

$$N_{\Gamma \Rightarrow \vec{A} \supset p}(\overrightarrow{X : \Theta \Rightarrow q}) \rightsquigarrow N_{\Delta \Rightarrow B_j}(\overrightarrow{X : \Theta \Rightarrow q}, Y : \sigma) \tag{7}$$

where the if-guard in Def. 22 fails; Δ and σ are defined as in the same definition; and, for some y, $(y : \vec{B} \supset p) \in \Delta$. We want to prove that every sequence of recursive calls from $N_{\Gamma \Rightarrow C}(\cdot)$ is finite.

First we introduce some definitions. $\mathscr{A}^{sub} := \{B \mid \text{there is } A \in \mathscr{A} \text{ such that } B \text{ is subformula of } A\}$, for $\mathscr{A}$ a finite set of formulas. We say $\mathscr{A}$ is *subformula-closed* if $\mathscr{A}^{sub} = \mathscr{A}$. A *stripped sequent* is a pair $(\mathscr{B}, p)$, where $\mathscr{B}$ is a finite set of formulas. If $\sigma = \Gamma \Rightarrow p$, then $|\sigma|$ denotes the stripped sequent $(|\Gamma|, p)$. We say $(\mathscr{B}, p)$ *is over* $\mathscr{A}$ if $\mathscr{B} \subseteq \mathscr{A}$ and $p \in \mathscr{A}$. There are $size(\mathscr{A}) := a \cdot 2^k$ stripped sequents over $\mathscr{A}$, if a (resp. k) is the number of atoms (resp. formulas) in $\mathscr{A}$.

Let $\mathscr{A}$ be subformula-closed. We say $\Gamma \Rightarrow C$ and $\Xi := \overrightarrow{X : \Theta \Rightarrow q}$ satisfy the *$\mathscr{A}$-invariant* if:

(i) $|\Gamma| \cup \{C\} \subseteq \mathscr{A}$;

(ii) $\Theta_1 \subseteq \Theta_2 \subseteq \cdots \subseteq \Theta_m = \Gamma$ (if $m = 0$ then this is meant to be vacuously true);

(iii) For $1 \leq j \leq m$, $q_j \in |\Gamma|^{sub}$,

where $m \geq 0$ is the length of vector Ξ (if $m = 0$, also item (iii) is vacuously true). In particular, $|\sigma|$ is over $\mathscr{A}$, for all $\sigma \in \Xi$. We prove that, if $\Gamma \Rightarrow C$ and Ξ satisfy the $\mathscr{A}$-invariant for some $\mathscr{A}$, then every sequence of recursive calls from $N_{\Gamma \Rightarrow C}(\Xi)$ is finite. The proof is by induction on $size(\mathscr{A}) - size(\Xi)$, where $size(\Xi)$ is the number of elements of $|\Xi|$ and $|\Xi| := \{|\sigma| : \sigma \in \Xi\}$.

Let $C = \vec{A} \supset p$. We analyze an arbitrary recursive call (7) and prove that every sequence of recursive calls from $N_{\Delta \Rightarrow B_j}(\Xi, Y : \sigma)$ is finite. This is achieved by proving:

(I) $\Delta \Rightarrow B_j$ and $\Xi, Y : \sigma$ satisfy the $\mathscr{A}$-invariant;

(II) $size(\Xi, Y : \sigma) > size(\Xi)$.

Proof of (I). By assumption, (i), (ii), and (iii) above hold. We want to prove:

(i') $|\Delta| \cup \{B_j\} \subseteq \mathscr{A}$;

(ii') $\Theta_1 \subseteq \Theta_2 \subseteq \cdots \subseteq \Theta_m \subseteq \Delta = \Delta$;

(iii') For $1 \leq j \leq m+1$, $q_j \in |\Delta|^{sub}$.

Proof of (i'). $|\Delta| = |\Gamma| \cup \{A_1, \cdots, A_n\} \subseteq \mathscr{A}$ by (i) and $\mathscr{A}$ subformula-closed. B_j is a subformula of $\vec{B} \supset p$ and $\vec{B} \supset p \in |\Delta|$ because $(y : \vec{B} \supset p) \in \Delta$, for some y.

Proof of (ii'). Immediate by (ii) and $\Gamma \subseteq \Delta$.

Proof of (iii'). For $1 \leq j \leq m$, $q_j \in |\Gamma|^{sub} \subseteq |\Delta|^{sub}$, by (iii) and $\Gamma \subseteq \Delta$. On the other hand, $q_{j+1} = p \in |\Delta|^{sub}$ because $(y : \vec{B} \supset p) \in \Delta$, for some y.

Proof of (II). Given that the if-guard of Def. 22 fails, and that $\Theta_i \subseteq \Gamma$ due to (ii), we conclude: for all $1 \leq i \leq m$, $p \neq q_i$ or $|\Theta_i| \neq |\Delta|$. But this means that $|\Delta \Rightarrow p| \notin |\Xi|$, hence $size(\Xi, Y : \sigma) > size(\Xi)$.

Now, by I.H., every sequence of recursive calls from $N_{\Delta \Rightarrow B_j}(\Xi, Y : \sigma)$ is finite. This concludes the proof by induction.

Finally let $\mathscr{A} = (|\Gamma| \cup \{C\})^{sub}$ and observe that $\Gamma \Rightarrow C$ and $\Xi = \cdot$ satisfy the $\mathscr{A}$-invariant. □

Theorem 24 (Equivalence) *For any Γ and C, there exists $N_{\Gamma \Rightarrow C} \in \overline{\lambda}^{\mathsf{gfp}}_{\Sigma}$ with no free occurrences of fixpoint variables such that $[\![N_{\Gamma \Rightarrow C}]\!] = \mathscr{S}(\Gamma \Rightarrow C)$.*

Proof We prove: if, for all i, $\xi(X_i^{\Theta_i \Rightarrow q_i}) = \mathscr{S}(\Theta_i \Rightarrow q_i)$, then

$$[\![N_{\Gamma \Rightarrow \vec{A} \supset p}(\Xi)]\!]_{\xi} = \mathscr{S}(\Gamma \Rightarrow \vec{A} \supset p) \ , \tag{8}$$

where $\Xi := \overrightarrow{X : \Theta \Rightarrow q}$. In this proof we re-use the concepts introduced in the proof of Lemma 23. Let $\mathscr{A} := (|\Gamma| \cup \{\vec{A} \supset p\})^{sub}$. The proof is by induction on $size(\mathscr{A}) - size(\Xi)$.

Case $p = q_i$ and $\Theta_i' \subseteq \Gamma$ and $|\Theta_i'| = |\Delta|$, for some $1 \leq i \leq m$, with m the length of Ξ. Then,

$$\begin{array}{rcll} LHS & = & \lambda z_1^{A_1} \cdots z_n^{A_n}.[\![X_i^{\Delta \Rightarrow q_i}]\!]_{\xi} & \text{(by definition)} \\ & = & \lambda z_1^{A_1} \cdots z_n^{A_n}.[\Delta \Rightarrow q_i / \Theta_i \Rightarrow q_i]\xi(X_i^{\Theta_i \Rightarrow q_i}) & \text{(by definition and (*) below)} \\ & = & \lambda z_1^{A_1} \cdots z_n^{A_n}.[\Delta \Rightarrow q_i / \Theta_i \Rightarrow q_i]\mathscr{S}(\Theta_i \Rightarrow q_i) & \text{(by assumption)} \\ & = & \lambda z_1^{A_1} \cdots z_n^{A_n}.\mathscr{S}(\Delta \Rightarrow q_i) & \text{(by Lemma 20 and (*))} \\ & = & RHS & \text{(by definition)} \end{array}$$

where $\Delta := \Gamma \cup \{z_1 : A_1, \cdots, z_n : A_n\}$, which implies $(\Theta_i \Rightarrow q_i) \leq (\Delta \Rightarrow q_i)$. The latter fact is the justification (*) used above.

The inductive case is an easy extension of the inductive case in Theorem 15. Suppose the case above holds for no $1 \leq i \leq m$. Then $LHS = \lambda z_1^{A_1} \cdots z_n^{A_n}.N^{\infty}$, where N^{∞} is the unique solution of the following equation

$$N^{\infty} \quad = \quad \sum_{(y : \vec{B} \supset p) \in \Delta} y\langle [\![N_{\Delta \Rightarrow B_j}(\Xi, Y : \sigma)]\!]_{\xi \cup [Y^{\sigma} \mapsto N^{\infty}]} \rangle_j \tag{9}$$

and, again, $\Delta := \Gamma \cup \{z_1 : A_1, \cdots, z_n : A_n\}$. Now observe that, by I.H., the following equations (10) and (11) are equivalent.

$$\mathscr{S}(\Delta \Rightarrow p) \quad = \quad \sum_{(y : \vec{B} \supset p) \in \Delta} y\langle [\![N_{\Delta \Rightarrow B_j}(\Xi, Y : \sigma)]\!]_{\xi \cup [Y^{\sigma} \mapsto \mathscr{S}(\Delta \Rightarrow p)]} \rangle_j \tag{10}$$

$$\mathscr{S}(\Delta \Rightarrow p) \quad = \quad \sum_{(y : \vec{B} \supset p) \in \Delta} y\langle \mathscr{S}(\Delta \Rightarrow B_j) \rangle_j \tag{11}$$

By definition of $\mathscr{S}(\Delta \Rightarrow p)$, (11) holds; hence - because of (10) - $\mathscr{S}(\Delta \Rightarrow p)$ is the solution N^{∞} of (9). Therefore $LHS = \lambda z_1^{A_1} \cdots z_n^{A_n}.\mathscr{S}(\Delta \Rightarrow p)$, and the latter is RHS by definition of $\mathscr{S}(\Gamma \Rightarrow \vec{A} \supset p)$.

Finally, the theorem follows as the particular case of (8) where $C = \vec{A} \supset p$ and the vector of fixpoint variable declarations is empty. □

5 Conclusion

We proposed a coinductive approach to proof search, which we illustrated in the case of the cut-free system LJT for intuitionistic implication (and its proof-annotated version $\overline{\lambda}$). As the fundamental tool, we introduced the coinductive calculus $\overline{\lambda}^{co}_{\Sigma}$, which besides the coinductive reading of $\overline{\lambda}$, introduces a construction for finite alternatives. The (co)terms of this calculus (also called Böhm forests) are used to represent the solution space of proof search for LJT-sequents, and this is achieved by means of a corecursive function, whose definition arises naturally by taking a reductive view of the inference rules and by using the finite alternatives construction to account for multiple alternatives in deriving a given sequent.

We offered also a finitary representation of proof search in LJT, based on the inductive calculus $\overline{\lambda}^{\mathsf{gfp}}_{\Sigma}$ with finite alternatives and a fixed point construction, and showed equivalence of the representations. The equivalence results turned out to be an easy task in the case of the Horn fragment, but demanded for co-contraction of contexts (contraction bottom-up) in the case of full implication.

With Pym and Ritter [11] we share the general goal of setting a framework for studying proof search, and the reductive view of inference rules, by which each inference rule is seen as a reduction operator (from a putative conclusion to a collection of sufficient premises), and reduction (the process of repeatedly applying reduction operators) may fail to yield a (finite) proof. However, the methods are very different. Instead of using a coinductive approach, Pym and Ritter introduce the $\lambda\mu\nu\varepsilon$-calculus for classical sequent calculus as the means for representing derivations and for studying intuitionistic proof search (a task that is carried out both in the context of the sequent calculus LJ and of intuitionistic resolution).

In the context of logic programming with classical first-order Horn clauses, and building on their previous work [6, 4], Komendantskaya and Power [5] establish a coalgebraic semantics uniform for both finite and infinite SLD-resolutions. In particular, a notion of coinductive (and-or) derivation tree of an atomic goal w. r. t. a (fixed) program is introduced. Soundness and completeness results of SLD-resolution relative to coinductive derivation trees and to the coalgebraic semantics are also proved. Logic programming is viewed as search for uniform proofs in sequent calculus by Miller *et al.* [8]. For intuitionistic implication, uniform proofs correspond to the class of (η-)expanded normal natural deductions (see Dyckoff and Pinto [2]), hence to the typed $\overline{\lambda}$-terms we considered in this paper (recall the restriction to atoms in rule *Der* of Fig. 1 for typing application). Under this view, our work relates to Komendantskaya and Power [5], as both works adopt a coinductive approach in the context of proof search. However, the two approaches are different in methods and in goals. As the basis of the coinductive representation of the search space, instead of and-or infinite trees, we follow the Curry-Howard view of proofs as terms, and propose the use of a typed calculus of coinductive lambda-terms. Whereas Komendantskaya and Power [5] are already capable of addressing first-order quantification, we only consider intuitionistic implication. Still, as we consider full intuitionistic implication, our study is not contained in classical Horn logic. The fact that we need to treat negative occurrences of implication, raises on the logic programming side the need for dealing with programs to which clauses can be added dynamically.

As a priority for future work, we plan to develop notions of normalisation for the calculi $\overline{\lambda}^{co}_{\Sigma}$ and $\overline{\lambda}^{\mathsf{gfp}}_{\Sigma}$ in connection with aspects of proof search like pruning search spaces and reading off (finite) proofs.

In order to test for the generality of our approach, we intend to extend it to treat the first-order case. Staying within intuitionistic implication, but changing the proofs searched for, another case study we intend to investigate is Dyckhoff's contraction-free system [1].

Acknowledgments We thank our anonymous referees for their helpful comments. José Espírito Santo and Luís Pinto have been financed by FEDER funds through "Programa Operacional Factores de Competitividade – COMPETE" and by Portuguese funds through FCT – "Fundação para a Ciência e a Tecnologia", within the project PEst-C/MAT/UI0013/2011. Ralph Matthes thanks the Centro de Matemática of Universidade do Minho for funding research visits to José Espírito Santo and Luís Pinto to start this research (2011/2012). Subsequently, he has been funded by the *Climt* project (ANR-11-BS02-016 of the French Agence Nationale de la Recherche).

References

[1] Roy Dyckhoff (1992): *Contraction-Free Sequent Calculi for Intuitionistic Logic*. *J. Symb. Log.* 57(3), pp. 795–807, doi:10.2307/2275431.

[2] Roy Dyckhoff & Luís Pinto (1994): *Uniform Proofs and Natural Deductions*. In Didier Galmiche & Lincoln Wallen, editors: *Proceedings of CADE–12 Workshop on Proof Search in Type-Theoretic Languages*, INRIA Lorraine – CRIN, pp. 717–23. Available at http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.9659.

[3] H. Herbelin (1995): *A λ-calculus structure isomorphic to a Gentzen-style sequent calculus structure*. In L. Pacholski & J. Tiuryn, editors: *Proceedings of CSL'94*, *Lecture Notes in Computer Science* 933, Springer-Verlag, pp. 61–75, doi:10.1007/BFb0022247.

[4] Ekaterina Komendantskaya, Guy McCusker & John Power (2010): *Coalgebraic Semantics for Parallel Derivation Strategies in Logic Programming*. In Michael Johnson & Dusko Pavlovic, editors: *AMAST*, *Lecture Notes in Computer Science* 6486, Springer, pp. 111–127, doi:10.1007/978-3-642-17796-5_7.

[5] Ekaterina Komendantskaya & John Power (2011): *Coalgebraic Derivations in Logic Programming*. In Marc Bezem, editor: *CSL*, *LIPIcs* 12, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 352–366, doi:10.4230/LIPIcs.CSL.2011.352.

[6] Ekaterina Komendantskaya & John Power (2011): *Coalgebraic Semantics for Derivations in Logic Programming*. In Andrea Corradini, Bartek Klin & Corina Cîrstea, editors: *CALCO*, *Lecture Notes in Computer Science* 6859, Springer, pp. 268–282, doi:10.1007/978-3-642-22944-2_19.

[7] Chuck Liang & Dale Miller (2009): *Focusing and Polarization in Linear, Intuitionistic, and Classical Logic*. *Theoretical Computer Science* 410, pp. 4747–4768, doi:10.1016/j.tcs.2009.07.041.

[8] Dale Miller, Gopalan Nadathur, Frank Pfenning & Andre Scedrov (1991): *Uniform Proofs as a Foundation for Logic Programming*. *Annals of Pure and Applied Logic* 51(1-2), pp. 125–157, doi:10.1016/0168-0072(91)90068-W.

[9] Keiko Nakata, Tarmo Uustalu & Marc Bezem (2011): *A Proof Pearl with the Fan Theorem and Bar Induction - Walking through Infinite Trees with Mixed Induction and Coinduction*. In Hongseok Yang, editor: *APLAS*, *LNCS* 7078, Springer, pp. 353–368, doi:10.1007/978-3-642-25318-8_26.

[10] Celia Picard & Ralph Matthes (2012): *Permutations in Coinductive Graph Representation*. In Dirk Pattinson & Lutz Schröder, editors: *Coalgebraic Methods in Computer Science (CMCS 2012)*, *Lecture Notes in Computer Science, IFIP subseries* 7399, Springer, pp. 218–237, doi:10.1007/978-3-642-32784-1_12.

[11] D.J. Pym & E. Ritter (2004): *Reductive Logic and Proof-search: Proof Theory, Semantics, and Control*. Oxford Logic Guides, Oxford University Press, Incorporated, doi:10.1093/acprof:oso/9780198526339.001.0001.