Decidability of several concepts of finiteness for simple types

José Espírito Santo, Ralph Matthes, Luís Pinto

Decidability of several concepts of finiteness for simple types

José Espírito Santo[*†]
Centro de Matemática
Universidade do Minho, Portugal
jes@math.uminho.pt

Luís Pinto[*†]
Centro de Matemática
Universidade do Minho, Portugal
luis@math.uminho.pt

Ralph Matthes[‡†]
Institut de Recherche en Informatique de Toulouse
(IRIT), CNRS and University of Toulouse, France
matthes@irit.fr

Abstract. If we consider as "member" of a simple type the outcome of any successful (possibly infinite) run of bottom-up proof search that starts from the type, then several concepts of "finiteness" for simple types are possible: the finiteness of the search space, the finiteness of any member, or the finiteness of the number of finite members (in other words, the inhabitants). In this paper we show that these three concepts are instances of the same parameterized notion of finiteness, and that a single, parameterized proof shows the decidability of all of them. One instance of this result means that termination of proof search is decidable. A separate result is that emptiness is also decidable (where emptiness is absence of "members" as above, not just absence of inhabitants). This fact is an ingredient of the main decidability result, but it also has a different application, the definition of the pruned search space - the one where branches leading to failure are chopped off. We conclude with our version of König's lemma for simple types: a simple type has an infinite member exactly when the pruned search space is infinite.

Keywords: lambda-calculus, proof search, coinduction, decision procedure

[*]The first and third authors were partially financed by Portuguese Funds through FCT (Fundação para a Ciência e a Tecnologia) within the Project UID/MAT/00013/2013.
[†]The three authors were partially financed by COST action CA15123 EUTYPES.
[‡]An early phase of the reported work was partially financed by the project *Climt*, ANR-11-BS02-016, of the French Agence Nationale de la Recherche.

1. Introduction

Like sets, simple types have an "extension", the collection of its inhabitants. But, more generally, a simple type determines a search space, a tree amalgamating all possible runs of bottom-up proof search. The collection of its inhabitants just corresponds to the successful traversals of this space which stop after finite time. A sensible proof search procedure will detect failure caused by a branch leading to a situation where no rule applies. But the procedure may fail to terminate because it may extend forever a branch where a further application of rule is always possible. The outcome of such a run is an infinite object which we tend to see positively, and call a "solution" of the search problem (after all, the procedure never faces the impossibility of applying a rule, which is the sign of failure). Thus, the inhabitants are just the finite solutions, and the collection of all solutions is another, more general, concept of extension of the simple type.

Given this richness, several concepts of *finiteness* for a simple type are possible: either the finiteness of the collection of inhabitants; or the property of all solutions being finite; or the finiteness of the search space *tout court*. In this paper we investigate these possible concepts of finiteness and show: they are all instances of a same, parameterized concept of finiteness; and they are all decidable. The parametrization allows to put the just mentioned concepts in decreasing order of generality; it permits as well a single proof of decidability, under mild conditions imposed on the parameter. The parameter is another, auxiliary predicate. It follows, as an instance, the decidability of the property of having a finite set of inhabitants, which is known [1, 2]; but the other instances are new decidability results.

One example of the parameter, determining an instance of finiteness capturing the property of all solutions being finite, is the predicate stating the existence of a solution. We have to study separately this property and establish that "solvability" of simple types (like inhabitation) is decidable. The property of all solutions being finite corresponds to termination of proof search, which is thus decidable. Additionally, the property of all solutions being finite is strictly stronger than the property of having a finite number of inhabitants. It is a kind of finiteness with an "operational" flavor: the members of the type can be found by a naive, depth-first proof search procedure; it is also a kind of finiteness we describe as "weakly extensional": the property cannot distinguish two types with the same set of solutions (and thus is "extensional"), but it can tell apart types with the same sets of inhabitants, like p and $(p \supset p) \supset p$ (which both have none).

On the other hand, the finiteness of the search space is strictly stronger than the property of all solutions being finite. However, this observation can be refined. We can redefine the search space as being the amalgamation of all the runs of a refined proof search procedure, one that applies a rule only if such a step does not create a branch where finite failure can be observed. The redefined search space is one where useless, finite branches have been pruned, i. e., where useless, finite runs are avoided. The procedure underlying this space is effective only because the extra test for applicability of a rule is decidable—and this is another application of "solvability" of simple types. Then, we show that the finiteness of the pruned search space is equivalent to all solutions being finite. In other words, the existence of an infinite run is equivalent to the infiniteness of this space—a result having the flavor of König's lemma.

The decidability results in this paper are established by means of a methodology previously developed by the authors [3, 4, 5]. All concepts pertaining to proof search are first given a coinductive

definition, integrated with the Curry-Howard style of representation of proofs. This means the search space is represented by a single λ-term in a suitable coinductive λ-calculus. In parallel, an alternative, finitary (inductive) syntax is developed in the form of a λ-calculus enriched with formal fixed points, where the search space has an equivalent representation, again as a single λ-term. Both calculi employ formal sums to represent choice points. Predicates of interest, say the (in)existence of inhabitants, are first given (co)inductive characterizations in the coinductive syntax, and later inductive, syntax directed characterizations in the finitary syntax. Decidability follows from syntax-directedness and the effectiveness of the functions that calculates the finitary representation of the search space of a given sequent.

This methodology is recalled in Section 2 (dedicated to background material), and in the beginning of Section 4, which is dedicated to decidability. There one finds a further application of decidability, leading to a refinement in the so-called coherence theorem. The parameterized concept of finiteness is developed in Section 3. Section 5 ends the paper.

2. Background

In this section we recall the presentation of the simply-typed λ-calculus we will be working with, and our coinductive representation of proof search [3, 4, 5]. Additionally, in the third subsection, we introduce new predicates to reason about our coinductive representation of proof search, and notions of extensionality over predicates.

2.1. Simply-typed λ-calculus, reduced to normal forms

It is well-known that η-long β-normal terms are complete for simply-typed λ-calculus in the sense that any typable term normalises to a β-normal form, which in turn can be expanded to an η-long β-normal form (see, e. g., 2D5 and 8A8 of the book [6]). We lay out a presentation of the η-long β-normal fragment of simply-typed λ-calculus, a system we often refer to by λ.

Simple types (or simply, types) are given by the grammar:

$$\text{(types)} \qquad A, B, C \quad := \quad p \mid A \supset B$$

where p, q, r range over *atoms*. We thus do not distinguish types from propositional implicational formulas. We will write $A_1 \supset A_2 \supset \cdots \supset A_k \supset p$, with $k \geq 0$, in vectorial notation as $\vec{A} \supset p$. For example, if the vector $\vec{A}$ is empty the notation means simply p.

Normal (*i.e.*, β-normal) λ-terms are given by:

$$\text{(terms)} \qquad t, u \quad ::= \quad \lambda x^A.t \mid x \langle t_1, \ldots, t_k \rangle$$

where a countably infinite set of variables, ranged over by letters x, y, w, z, is assumed. As is common-place with lambda-calculi, we will throughout identify terms up to α-equivalence. The term constructor $x \langle t_1, \ldots, t_k \rangle$ is called *application* (traditionally, this would be expressed as a multiple application $xt_1 \ldots t_k$ of λ-calculus). Often, we simply write the variable x when $k = 0$, and we will use the notation $\langle t_i \rangle_i$ for finite tuples.

Figure 1. Typing rules of λ

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x^A.t : A \supset B}\ \textit{RIntro} \qquad \frac{(x : \vec{B} \supset p) \in \Gamma \quad \forall i,\, \Gamma \vdash t_i : B_i}{\Gamma \vdash x\langle t_i\rangle_i : p}\ \textit{LVecIntro}$$

We view contexts Γ as finite sets of declarations $x : A$, where no variable x occurs twice. The letters Γ, Δ, Θ are used to range over contexts, and the notation $dom(\Gamma)$ stands for the set of variables declared in Γ. The context $\Gamma, x : A$ is obtained from Γ by adding the declaration $x : A$, and is only written if x is not declared in Γ. Context union is written as concatenation Γ, Δ for contexts Γ and Δ if $dom(\Gamma) \cap dom(\Delta) = \emptyset$. We will write $\Gamma(x)$ for the type associated with x for $x \in dom(\Gamma)$. Context inclusion $\Gamma \subseteq \Delta$ is just set inclusion.

The typing rules are in Fig. 1 and derive sequents $\Gamma \vdash t : A$. Note that, in the particular case of *LVecIntro* where $\vec{B}$ is empty, $(x : p) \in \Gamma$ is the only hypothesis, and we type variables with atoms.

2.2. Search for inhabitants, coinductively

We are concerned with a specific kind of search problems: given Γ and A, to find a λ-term t such that $\Gamma \vdash t : A$, that is, to find an *inhabitant* of type A in context Γ. Under the Curry-Howard correspondence, a pair (Γ, A) may be seen as a *logical sequent* $\Gamma \Rightarrow A$, and searching for an inhabitant of A in context Γ is the same as searching for a proof of that sequent. Our search process will receive a formal description in Definition 2.1.

Following [3, 4, 5], we model this search process through the coinductive λ-calculus, denoted λ^{co}. The terms of λ^{co}, also called *coterms*, are given by

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

This is exactly the previous grammar for λ-terms, but read coinductively, as indicated by the index *co* (still with finite tuples $\langle N_i\rangle_i$). Of course, since coterms are not built in finitary ways from finitary syntax, the notion of equality is not just syntactic equality, but rather *bisimilarity modulo α-equivalence*. Bisimilarity is given by a coinductive binary relation that considers as equal coterms M and N that finitely decompose in the same way, in other words, the successive deconstruction of M and N according to the grammar must proceed the same way (up to α-equivalence), and this to arbitrary depth. Following mathematical practice, bisimilarity is still written as plain equality.

In λ^{co}, also the typing rules of Fig. 1 have to be interpreted coinductively—but the types stay inductive and the contexts finite. Following common practice in the presentation of coinductive syntax, we will symbolize the coinductive reading of an inference (rule) by the double horizontal line, but we refrain from displaying Fig. 1 again with double lines—a figure where the two inference rules would be called *RIntro*$_{co}$ and *LVecIntro*$_{co}$. Such a system defines when $\Gamma \vdash N : A$ holds for a *finite* context Γ, a coterm N and a type A.

Suppose $\Gamma \vdash N : A$ holds. Then this sequent has a derivation, which is a (possibly infinite) tree of sequents, generated by applying the inference rules bottom-up; and N is a (possibly infinite) coterm,

Figure 2. Membership relations

$$\frac{\mathsf{mem}(M,N)}{\mathsf{mem}(\lambda x^A.M, \lambda x^A.N)} \qquad \frac{\mathsf{mem}(M,E_j)}{\mathsf{mem}(M, \sum_i E_i)} \qquad \frac{\forall i,\ \mathsf{mem}(M_i,N_i)}{\mathsf{mem}(x\langle M_i\rangle_i, x\langle N_i\rangle_i)}$$

Figure 3. Extra typing rule of λ^{co}_{Σ} w. r. t. λ^{co}

$$\frac{\forall i,\ \Gamma \vdash E_i : p}{\Gamma \vdash \sum_i E_i : p}\ \textit{Alts}$$

which we call a *solution* of σ, with $\sigma = (\Gamma \Rightarrow A)$. Therefore, such derivations are the structures generated by the search process when it does not fail—even if it runs forever—and so they subsume proofs; likewise solutions subsume typable terms, so we may refer to the latter as *finite* solutions. The next step is to extend even further the paradigm, representing also the choice points of the search process. To this end, we extend λ^{co} to λ^{co}_{Σ}, whose syntax is this:

(terms)	M, N	$::=_{co}$	$\lambda x^A.N \mid E_1 + \cdots + E_n$
(elimination alternatives)	E	$::=_{co}$	$x\langle N_1, \ldots, N_k\rangle$

where both $n, k \geq 0$ are arbitrary (thus including the empty sum of elimination alternatives). T ranges over both terms and elimination alternatives. We will often use $\sum_i E_i$ instead of $E_1 + \cdots + E_n$ and $\mathbb{O}$ instead the empty sum.

The notion of equality of terms in λ^{co}_{Σ} is again bisimilarity modulo α-equivalence, but we further assume that $+$ is associative, commutative and idempotent. So the sums of elimination alternatives can plainly be treated as if they were finite sets of elimination alternatives.

We call *forests* the expressions of λ^{co}_{Σ} —and a coterm M is a member of a forest N when the relation $\mathsf{mem}(M,N)$, defined coinductively in Fig. 2, holds. Then, we define the *extension* of a forest to be the collection of its members, in symbols,

$$\mathcal{E}(T) := \{M \in \lambda^{co} \mid \mathsf{mem}(M,T)\}.$$

As in [5], we define the *finite extension* of a forest to be the collection of its finite members, in symbols,

$$\mathcal{E}_{\mathrm{fin}}(T) := \{t \in \lambda \mid \mathsf{mem}(t,T)\}.$$

Simple types can be assigned to forests [3, 4]. In the typing system for λ^{co}_{Σ}, one derives sequents $\Gamma \vdash N : A$ and $\Gamma \vdash E : p$. The coinductive typing rules are the ones of λ^{co}, together with the rule given in Fig. 3 (the empty sum of elimination alternatives receives any atom as type).

A typing derivation of λ^{co}_{Σ} is a possibly infinite tree of sequents, generated by the bottom-up application of the inference rules, with "multiplicative" branching (logically: "and" branching) caused by the list of arguments in elimination alternatives, and "additive" branching (logically: "or" branching)

caused by sums—the latter being able to express the alternatives found in the search process when an atom p can be proved by picking different head variables with their appropriate arguments. So, it is no surprise that, with this infrastructure, we can express, as a (single) forest, the entire *solution space* generated by the search process when applied to given Γ and A. That forest can be corecursively defined as the function $\mathcal{S}$ of $\Gamma \Rightarrow A$:

Definition 2.1. (Solution spaces)

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

Example 2.2. We introduce now the sequents that will play some role in the paper, and calculate their solution spaces. (More standard examples, but also complicated examples that illustrate the dynamic nature of the contexts of the sequents—when implications are nested in the hypotheses—can be found in [3, 4, 5].)

- Let $\sigma_a := (\Rightarrow p)$. Then, $\mathcal{S}(\sigma_a) = \mathbb{O}$. Obviously, this sequent has no solution.

- Let $\sigma_b := (x : p \supset p \Rightarrow p)$. Then, $\mathcal{S}(\sigma_b) = x\langle \mathcal{S}(\sigma_b)\rangle$, i. e., $\mathcal{S}(\sigma_b)$ is the forest $x\langle x\langle \ldots\rangle\rangle$ with an infinitely repeated application of x. The only solution for this sequent is again $x\langle x\langle \ldots\rangle\rangle$ (now seen as a coterm).

- Let $\sigma_c := (x : p \supset q \supset p \Rightarrow p)$. Then, $\mathcal{S}(\sigma_c) = x\langle \mathcal{S}(\sigma_c), \mathbb{O}\rangle$. This sequent has no solution (since we cannot find a coterm from the empty sum in the second component of the tuple).

- Let $\sigma_d := (x : p \supset q \supset p, y : p, z : q \Rightarrow p)$. Then, $\mathcal{S}(\sigma_d) = x\langle \mathcal{S}(\sigma_d), z\rangle + y$. Infinitely many inhabitants can be obtained for this sequent, starting with the inhabitant $t_1 := x\langle y, z\rangle$ and building new inhabitants via $t_{n+1} := x\langle t_n, z\rangle$ (for each $n \in \mathbb{N}$). This sequent has one infinite solution given by the coterm $x\langle x\langle x\langle \ldots\rangle, z\rangle, z\rangle$.

- Let $\sigma_e := (x : p \supset p, y : p \Rightarrow p)$, corresponding to a simplification of type CHURCH of the Church numerals, analysed in [5, 3, 4]. Then, $\mathcal{S}(\sigma_e) = x\langle \mathcal{S}(\sigma_e)\rangle + y$. Again, this is an example of a sequent with infinitely many inhabitants and one infinite solution.

The following properties witnessing the robustness of the definition of $\mathcal{S}$ are shown in [3, 4] (and reproduced as [5, Proposition 1]).

Fact 2.3. (Properties of solution spaces)

1. Given Γ and A, the typing $\Gamma \vdash \mathcal{S}(\Gamma \Rightarrow A) : A$ holds in λ^{co}_{Σ}.

2. For $N \in \lambda^{co}$, $\mathsf{mem}(N, \mathcal{S}(\Gamma \Rightarrow A))$ iff $\Gamma \vdash N : A$ in λ^{co}.

3. For $t \in \lambda$, $\mathsf{mem}(t, \mathcal{S}(\Gamma \Rightarrow A))$ iff $\Gamma \vdash t : A$ in λ.

The second property says the members of $\mathcal{S}(\sigma)$ are exactly the solutions of σ, and this justifies calling $\mathcal{S}(\sigma)$ the solution space of σ. The last two properties together say that solutions subsume finite solutions conservatively, i. e., given a λ-term t, $\Gamma \vdash t : A$ in λ iff $\Gamma \vdash t : A$ in λ^{co}. The first property says $\mathcal{S}(\Gamma \Rightarrow A)$ is a λ-term representing, in the Curry-Howard style, a type derivation in λ^{co}_{Σ} that amounts to the entire search space for $\Gamma \Rightarrow A$.

From this, and the coinductive definition of membership, it follows that the solutions are the outcomes of the successful runs of an idealized proof search procedure that non-deterministically applies bottom-up an inference rule, and proceeds by searching in parallel proofs for the premisses. This procedure will find a finite failure, caused by the impossibility of applying an inference rule, whenever it exists; but it may run forever while building an infinite solution. Termination of the search procedure amounts to the inexistence of an infinite solution. One of the concepts of finiteness for simple types developed in Section 3 (namely allfin) corresponds exactly to this notion of termination of proof search.

Finally, we recall from [3, 4, 5] the important phenomenon of *decontraction* (going by the name of cocontraction in the earlier papers). Intuitively, this is just the observation that, when searching for solutions in a given context with two variables x and y that are declared with the same type A, then, whenever a coterm of a given type B can be constructed using assumption x, another coterm of that type B can be constructed with y in place of x, and this choice between x and y can be operated for each occurrence of x independently. (This same phenomenon already occurs in the search for inhabitants.) The *operation* of decontraction on forests, for contexts and for sequents, denoted by $[\Gamma'/\Gamma]T$ and $[\sigma'/\sigma]T$, respectively, describes this phenomenon in terms of λ^{co}_{Σ}: the outcome of applying the operation is the forest where all the extra choices available in Γ' w. r. t. Γ (or in the contexts of the respective sequents σ' and σ) are included. Therefore the decontraction operation (henceforth called just decontraction) is only defined when Γ' (resp. σ') is an *inessential extension* of Γ (resp. σ), that is when the context of Γ' (resp. σ') has more declarations than that of Γ (resp. σ), but not with new types. Formally:

Definition 2.4. (Inessential extension of contexts and sequents)

1. $\Gamma \leq \Gamma'$ iff $\Gamma \subseteq \Gamma'$ and $|\Gamma| = |\Gamma'|$, with the set $|\Delta| := \{A \mid \exists x,\ (x : A) \in \Delta\}$ of assumed types of Δ for an arbitrary context Δ.

2. $\sigma \leq \sigma'$ iff for some $\Gamma \leq \Gamma'$ and for some atom p, $\sigma = (\Gamma \Rightarrow p)$ and $\sigma' = (\Gamma' \Rightarrow p)$.

Decontraction precisely captures the extension of the solution space when going from σ to some σ' with $\sigma \leq \sigma'$, as will be expressed in Fact 2.6 below.

Definition 2.5. (Decontraction of contexts and sequents)

1. Let $\Gamma \leq \Gamma'$. For T an expression of λ^{co}_{Σ}, we define the expression $[\Gamma'/\Gamma]T$ of λ^{co}_{Σ} by corecursion as follows:

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

where, in the last clause, $A := \Gamma(z)$ and $\Delta_z := \{(z : A)\} \cup (\Gamma' \setminus \Gamma)$.

2. Let $\sigma \leq \sigma'$, and $\sigma = (\Gamma \Rightarrow p)$ and $\sigma' = (\Gamma' \Rightarrow p)$. For T an expression of λ^{co}_{Σ}, $[\sigma'/\sigma]T$ is defined to be $[\Gamma'/\Gamma]T$.

As announced above, (the) decontraction (operation) properly captures the decontraction phenomenon:

Fact 2.6. (Solution spaces and decontraction [4, Lemma 33] (reproduced as [5, Lemma 7]))
Let $\sigma \leq \sigma'$. Then $\mathcal{S}(\sigma') = [\sigma'/\sigma]\mathcal{S}(\sigma)$.

2.3. Predicates on forests and extensionality

We call *strongly extensional* a predicate on forests that only depends on the finite extension of the forest, in other words, if Π is a predicate on forests, then Π is strongly extensional iff $\mathcal{E}_{\text{fin}}(T) = \mathcal{E}_{\text{fin}}(T')$ and $\Pi(T)$ imply $\Pi(T')$. Put differently, a strongly extensional predicate on forests cannot distinguish between forests that have the same finite extension. This is in particular the case when $\Pi = R \circ \mathcal{E}_{\text{fin}}$ for any predicate R on sets of λ-terms. But the existence of such a predicate R is also a necessary condition for Π being strongly extensional (take $R(X) :\Leftrightarrow \exists T(\mathcal{E}_{\text{fin}}(T) = X \wedge \Pi(T))$).

A predicate P on sequents is called strongly extensional if there is a strongly extensional predicate Π on forests such that $P = \Pi \circ \mathcal{S}$. A fortiori, this is in particular the case when there is a predicate R on sets of λ-terms such that $P = R \circ \mathcal{E}_{\text{fin}} \circ \mathcal{S}$. By [5, Proposition 1.3], this is equivalent to $P = R \circ \mathcal{I}$, where $\mathcal{I}(\Gamma \Rightarrow A) := \{t \in \lambda \mid \Gamma \vdash t : A \text{ in } \lambda\}$ is the set of inhabitants of a sequent. The existence of such a predicate R is again also a necessary condition for P being strongly extensional. In other words, a strongly extensional predicate on sequents is characterized by not being able to distinguish between sequents that have the same set of inhabitants.

Previously [5, Section 3], we analyzed the following strongly extensional predicates on forests:

- $\mathsf{exfinext}(T) :\Leftrightarrow \mathcal{E}_{\text{fin}}(T)$ is nonempty.
  $\mathsf{nofinext}(T) :\Leftrightarrow \mathcal{E}_{\text{fin}}(T)$ is empty.

- $\mathsf{finfinext}(T) :\Leftrightarrow \mathcal{E}_{\text{fin}}(T)$ is finite.
  $\mathsf{inffinext}(T) :\Leftrightarrow \mathcal{E}_{\text{fin}}(T)$ is infinite.

They are strongly extensional since their definition is expressed in terms of the finite extension, i. e., their definition is of the form $R \circ \mathcal{E}_{\text{fin}}$ with suitable predicates R.

Extensional predicates concern no longer only the finite extension $\mathcal{E}_{\text{fin}}(T)$ of a forest but the whole extension. Extensionality of a predicate on forests is defined like strong extensionality, but with reference to $\mathcal{E}$ in place of $\mathcal{E}_{\text{fin}}$. Of course, strongly extensional predicates are extensional (since having the same extension implies having the same finite extension).

Weakly extensional predicates are then defined as extensional predicates that are not strongly extensional. In particular, Π is extensional if there is a predicate R on sets of coterms such that $\Pi = R \circ \mathcal{E}$. The definitions of extensional and weakly extensional predicates on sequents are obtained analogously. Again, in particular, a predicate P on sequents is extensional if there is a predicate R on sets of coterms such that $P = R \circ \mathcal{E} \circ \mathcal{S}$.

We will be mainly interested in the following predicates on forests concerning their extension:

- $\mathsf{nosolext}(T) :\Leftrightarrow \mathcal{E}(T)$ is empty.
  $\mathsf{exsolext}(T) :\Leftrightarrow \mathcal{E}(T)$ is nonempty.

- $\mathsf{allfinext}(T) :\Leftrightarrow \mathcal{E}(T)$ consists only of finite terms, i. e., $\mathcal{E}(T) \subseteq \mathcal{E}_{\mathrm{fin}}(T)$.
  $\mathsf{exinfext}(T) :\Leftrightarrow \mathcal{E}(T)$ contains an infinite term, i. e., there is $M \in \lambda^{co} \setminus \lambda$ with $\mathsf{mem}(M, T)$.

These predicates factor through the extension $\mathcal{E}$, hence are extensional. They are not strongly extensional, as can be exemplified by forests that arise as solution spaces of sequents:

Example 2.7. (nosolext **and** allfinext **are only weakly extensional)**
In Example 2.2, we already observed $\mathcal{E}(\mathcal{S}(\sigma_a)) = \emptyset$ and $\mathcal{E}(\mathcal{S}(\sigma_b))$ consists only of the coterm $x\langle x\langle \ldots\rangle\rangle$. Hence, $\mathsf{nosolext}(\mathcal{S}(\sigma_a))$, $\mathsf{allfinext}(\mathcal{S}(\sigma_a))$ and neither $\mathsf{nosolext}(\mathcal{S}(\sigma_b))$ nor $\mathsf{allfinext}(\mathcal{S}(\sigma_b))$.

We are also interested in the property of forests of being "finite by definition", i. e., to be in the set λ_Σ of expressions that is inductively generated by the grammar for λ^{co}_Σ.

- $\mathsf{fin}(T) :\Leftrightarrow T \in \lambda_\Sigma$.
  $\mathsf{inf}(T) :\Leftrightarrow T \in \lambda^{co}_\Sigma \setminus \lambda_\Sigma$.

The elements of λ_Σ will also be called *finite forests*. Obviously, $\mathsf{fin} \subseteq \mathsf{allfinext} \cap \mathsf{finfinext}$. The predicate fin on forests is not even weakly extensional, as can again be exemplified by forests that arise as solution spaces of sequents:

Example 2.8. (fin **is not even weakly extensional)**
In Example 2.2, we observed $\mathcal{S}(\sigma_a) = \mathbb{O}$, $\mathcal{S}(\sigma_c) = x\langle \mathcal{S}(\sigma_c), \mathbb{O}\rangle$, and $\mathcal{E}(\mathcal{S}(\sigma_a)) = \mathcal{E}(\mathcal{S}(\sigma_c)) = \emptyset$. But, clearly, $\mathsf{fin}(\mathcal{S}(\sigma_a))$ and $\mathsf{inf}(\mathcal{S}(\sigma_c))$.

We remark that all properties P of sequents of interest to us in this work (with the sole exception being the condition on positivity in Theorem 4.21 in Section 4) factor through the function $\mathcal{S}$ that designates their solution space, thus they are always expressed as a property of the associated forests, which therefore gives a good sense to the question if P is strongly extensional, weakly extensional or not extensional.

3. Concepts of finiteness for simple types

In this section we develop the three decidable concepts of finiteness for a simple type A, referred to in Section 1: finiteness of the collection of inhabitants of A; all solutions of A are finite; the search space of A is itself finite. The first concept is given through the strongly extensional predicate $\mathsf{finfinext}(\mathcal{S}(\Rightarrow A))$, already studied in [5]. The second and third concepts are given, respectively, through the weakly extensional predicate $\mathsf{allfinext}(\mathcal{S}(\Rightarrow A))$ and the non-extensional predicate $\mathsf{fin}(\mathcal{S}(\Rightarrow A))$, both introduced in this paper. We will see in the second subsection that the three mentioned predicates can be obtained through instances of a parameterized predicate on forests that is inductively defined and has a complement enjoying a coinductive characterization. The third subsection will further explore this parameterized description of finiteness. But, for all this, we will need to study first the complementary weakly extensional predicates nosolext and exsolext, defined in Section 2.

Figure 4. nosol and exsol predicates

$$\frac{\mathsf{nosol}(N)}{\mathsf{nosol}(\lambda x^A.N)} \qquad \frac{\forall i,\ \mathsf{nosol}(E_i)}{\mathsf{nosol}(\sum_i E_i)} \qquad \frac{\mathsf{nosol}(N_j)}{\mathsf{nosol}(x\langle N_i\rangle_i)}$$

$$\frac{\mathsf{exsol}(N)}{\mathsf{exsol}(\lambda x^A.N)} \qquad \frac{\mathsf{exsol}(E_j)}{\mathsf{exsol}(\sum_i E_i)} \qquad \frac{\forall i,\ \mathsf{exsol}(N_i)}{\mathsf{exsol}(x\langle N_i\rangle_i)}$$

3.1. An auxiliary concept: absence of solutions

We introduce predicate $\mathsf{nosol}(T)$, for T an expression of λ^{co}_{Σ} (a forest), which holds iff $\mathsf{nosolext}(T)$, i. e., if the extension of T is empty, but it is defined inductively in Fig. 4, together (but independently) with the coinductive definition of the predicate $\mathsf{exsol}(T)$ that is supposed to mean the negation of $\mathsf{nosol}(T)$, but which is expressed positively as existence of a member (i. e., that the extension is non-empty—that $\mathsf{exsolext}(T)$ holds).

Lemma 3.1. Given a forest T, $\mathsf{nosol}(T)$ iff $\mathsf{exsol}(T)$ does not hold.

Proof:
This is an instance of the dualization principle recalled in our previous paper [5, proof of Lemma 20]. □

Lemma 3.2. (Coinductive characterization of existence of solutions)
Given a forest T. Then, $\mathsf{exsol}(T)$ iff $\mathsf{exsolext}(T)$, i. e., $\mathsf{exsol} = \mathsf{exsolext}$ as sets of forests.

Proof:
We have to show $\mathsf{exsol}(N) \iff (\exists M.\mathsf{mem}(M, N))$. The "if" direction is proved coinductively, by showing that $R := \{T : \exists M.\mathsf{mem}(M, T)\}$ is backwards closed for the defining rules of exsol. The "only if" direction is proved with the help of a corecursive procedure extracting an M s. t. $\mathsf{mem}(M, T)$ from a proof of $\mathsf{exsol}(T)$. □

In particular, we also get that $\mathsf{nosol} = \mathsf{nosolext}$. Notice that, although nosol is defined inductively, this does not mean that this predicate is decidable—the question does not even make sense in view of the semantic nature of the forests it receives as argument.

Definition 3.3. Let Π be a predicate on forests. Π is said to be closed under decontraction *both ways* if for all forests T and all sequents σ, σ' such that $\sigma \leq \sigma'$, we have $\Pi(T)$ iff $\Pi([\sigma'/\sigma]T)$.

It is obvious that Π is closed under decontraction both ways iff Π is closed under decontraction (i. e., for all forests T and all sequents σ, σ' such that $\sigma \leq \sigma'$, $\Pi(T)$ implies $\Pi([\sigma'/\sigma]T)$) and also the complement of Π is closed under decontraction. Therefore, Π is closed under decontraction both ways iff its complement is closed under decontraction both ways.

Figure 5. fin^{Π} predicate and inf^{Π} predicate

$$\frac{\neg\Pi(N)}{\mathsf{fin}^{\Pi}(\lambda x^{A}.N)} \qquad \frac{\mathsf{fin}^{\Pi}(N)}{\mathsf{fin}^{\Pi}(\lambda x^{A}.N)} \qquad \frac{\forall i,\ \mathsf{fin}^{\Pi}(E_i)}{\mathsf{fin}^{\Pi}(\sum_i E_i)} \qquad \frac{\neg\Pi(N_j)}{\mathsf{fin}^{\Pi}(x\langle N_i\rangle_i)} \qquad \frac{\forall i,\ \mathsf{fin}^{\Pi}(N_i)}{\mathsf{fin}^{\Pi}(x\langle N_i\rangle_i)}$$

$$\frac{\Pi(N) \quad \mathsf{inf}^{\Pi}(N)}{\mathsf{inf}^{\Pi}(\lambda x^{A}.N)} \qquad \frac{\mathsf{inf}^{\Pi}(E_j)}{\mathsf{inf}^{\Pi}(\sum_i E_i)} \qquad \frac{\forall i,\ \Pi(N_i) \quad \mathsf{inf}^{\Pi}(N_j)}{\mathsf{inf}^{\Pi}(x\langle N_i\rangle_i)}$$

Lemma 3.4. The predicates nosol and exsol are closed under decontraction both ways.

Proof:
This is already obvious in the semantics, i. e., for the extensional version nosolext, but also provable by induction on the inductive definition of nosol, and the same property for the complement follows, as mentioned above. □

3.2. A parameterized concept of finiteness

Throughout this section, Π will stand for a predicate on forests (in λ^{co}_{Σ}). The parameterized predicates over forests fin^{Π} and inf^{Π} are defined inductively and coinductively, respectively, in Fig. 5.

Lemma 3.5. Given a forest T, $\mathsf{fin}^{\Pi}(T)$ iff $\mathsf{inf}^{\Pi}(T)$ does not hold.

Proof:
This is another instance of the dualization principle recalled in our previous paper [5, proof of Lemma 20]. □

By inspecting the defining rules, it is immediate to see that inf^{Π} is monotone on Π and fin^{Π} is antitone on Π, i. e.:

Lemma 3.6. (Monotonicity properties of the parameterized predicates on forests)
For any predicates Π, Π', if $\Pi \subseteq \Pi'$, then $\mathsf{inf}^{\Pi} \subseteq \mathsf{inf}^{\Pi'}$ and $\mathsf{fin}^{\Pi'} \subseteq \mathsf{fin}^{\Pi}$.

We will consider three specific instantiations for predicate Π:

- $\Pi_1 :=$ exfin, where exfin is the predicate defined in [5, Fig. 5] that inductively characterizes exfinext;
- $\Pi_2 :=$ exsol (defined in Fig. 4);
- $\Pi_3 := \lambda^{co}_{\Sigma}$ (the always true predicate).

Then:

- $\mathsf{fin}^{\Pi_1} = \mathsf{finfin}$ and $\mathsf{inf}^{\Pi_1} = \mathsf{inffin}$ (for the predicates finfin and inffin defined in [5, Fig. 7]), with the latter equality following by definition of inffin, and the former following with the help of [5, Lemma 20] (that just says that Π_1 is the negation of a coinductively defined predicate nofin that is used in the definition of finfin instead of the negated Π_1)—in both cases, no (co-)induction is needed since even the defining rules are provably the same;[1]

- let us define $\mathsf{allfin} := \mathsf{fin}^{\Pi_2}$ and $\mathsf{exinf} := \mathsf{inf}^{\Pi_2}$; we show below in Lemma 3.9 that $\mathsf{exinf} = \mathsf{exinfext}$, from which, by duality, follows $\mathsf{allfin} = \mathsf{allfinext}$;

- $\mathsf{fin}^{\Pi_3} = \mathsf{fin}$; hence $\mathsf{inf}^{\Pi_3} = \mathsf{inf}$; these equalities are justified next.

Lemma 3.7. (Coinductive characterization of infinity of forests)
Let T be a forest T. Then, $\mathsf{inf}^{\Pi_3}(T)$ iff $\mathsf{inf}(T)$.

Proof:
Due to Lemma 3.5, this is equivalent to: $\mathsf{fin}^{\Pi_3}(T)$ iff $T \in \lambda_\Sigma$. When $\Pi = \Pi_3 = \lambda_\Sigma^{co}$, the first and fourth rules in the definition of fin^Π (Fig. 5) may be erased, the predicate Π plays no role in the remaining rules, and what remains is the inductive definition of λ_Σ.[2] □

Corollary 3.8. (Invariants of inf^Π and of fin^Π)
1. For any Π and T, if $\mathsf{inf}^\Pi(T)$, then $\mathsf{inf}(T)$.

2. For any Π and T, if $\mathsf{fin}(T)$, then $\mathsf{fin}^\Pi(T)$.

Proof:
For any Π, since $\Pi \subseteq \lambda_\Sigma^{co} = \Pi_3$, Lemma 3.6 gives $\mathsf{inf}^\Pi \subseteq \mathsf{inf}^{\Pi_3}$. Hence, the previous lemma gives $\mathsf{inf}^\Pi \subseteq \mathsf{inf}$. Part 2 follows from part 1 by contraposition, the obvious fact that fin and inf are complementary predicates and Lemma 3.5. □

So, T must be an infinite forest if $\mathsf{inf}^\Pi(T)$, for some Π, and, equivalentely, for any finite forest T, $\mathsf{fin}^\Pi(T)$ holds for any Π.

Lemma 3.9. (Coinductive characterization of existence of infinite members)
Given a forest T. Then, $\mathsf{exinf}(T)$ iff $\mathsf{exinfext}(T)$.

Proof:
The proof is analogous to the one of Lemma 3.2. The "if" direction follows by coinduction on exinf. It uses the fact $\mathsf{exinfext} \subseteq \mathsf{exsol}$ (an immediate consequence of $\mathsf{exinfext} \subseteq \mathsf{exsolext}$ and Lemma 3.2). A corecursive extraction procedure out of proofs of $\mathsf{exinf}(T)$ shows the "only if" direction. It requires the subsidiary corecursive extraction procedure of "members of T" from proofs of $\mathsf{exsol}(T)$ mentioned in Lemma 3.2. □

[1] In hindsight, we could just as well have defined finfin as being $\mathsf{fin}^{\mathsf{exfinext}}$, and the proof that $\mathsf{finfin}(\mathcal{S}(\sigma))$ characterizes the finiteness of $\mathcal{I}(\sigma)$ [5, Theorem 33.2] would still work.

[2] In the proof of Lemma 21 of [5], a similar characterization is found of the infinite coterms (terms in $\lambda^{co} \setminus \lambda$). There is no risk of confusion with the symbol inf used there since, in the absence of sums in an expression, both predicates coincide.

It will be useful to know the following sufficient condition for closedness under decontraction of inf^{Π}:

Lemma 3.10. Let Π be closed under decontraction both ways. Then, fin^{Π} and inf^{Π} are closed under decontraction both ways.

Proof:
The proof can be done for fin^{Π} by induction on its inductive definition (for each direction separately), its complement inf^{Π} then has the same property. □

3.3. Exploring the parameterized concept of finiteness

Let us pause to relate the instances of the predicates fin^{Π} and inf^{Π} we just considered. First note the following strictly ascending chain:

$$\mathsf{exfin} \subset \mathsf{exsol} \subset \lambda^{co}_{\Sigma} \tag{1}$$

The first inclusion follows from the fact $\mathsf{exfin}(T)$ iff $\mathcal{E}_{\mathrm{fin}}(T)$ is nonempty (an easy consequence of [5, Lemma 21]—for $\mathsf{exfinext}$ in place of exfin, this would just be the definition) and $\mathsf{exsolext} = \mathsf{exsol}$ (Lemma 3.2). The second inclusion is trivial.

Now, by (1) and antitonicity of fin^{Π} (Lemma 3.6), we have $\mathsf{fin} \subseteq \mathsf{allfin} \subseteq \mathsf{finfin}$ and, equivalently, $\mathsf{inffin} \subseteq \mathsf{exinf} \subseteq \mathsf{inf}$.

Example 3.11. ($\mathsf{inffin} \neq \mathsf{exinf} \neq \mathsf{inf}$)
Take the sequents σ_b and σ_c from Example 2.2. Then, $\mathcal{S}(\sigma_b) \in \mathsf{exinf} \setminus \mathsf{inffin}$, and $\mathcal{S}(\sigma_c) \in \mathsf{inf} \setminus \mathsf{exinf}$.

Therefore, we even get the following strictly ascending chains:

$$\mathsf{fin} \subset \mathsf{allfin} \subset \mathsf{finfin} \tag{2}$$

$$\mathsf{inffin} \subset \mathsf{exinf} \subset \mathsf{inf} \tag{3}$$

The equivalent inclusions $\mathsf{allfin} \subseteq \mathsf{finfin}$ and $\mathsf{inffin} \subseteq \mathsf{exinf}$ (that we have thus got in particular) are not obvious, and neither of them looks amenable to a direct proof. We interpret the second one in terms of inhabitants and solutions and then strengthen the observation to the case of infinitely many solutions:

Proposition 3.12. (Infinite number of solutions vs infinity of solution)

1. Any sequent having infinitely many inhabitants has an infinite solution.

2. Any sequent having infinitely many solutions has an infinite solution.

Proof:
To prove 1, suppose σ has infinitely many inhabitants. Hence $\mathsf{inffin}(\mathcal{S}(\sigma))$ holds by [5, Theorem 33.2 and Lemma 29] (where the latter just identifies finfin and inffin as complements). So, because $\mathsf{inffin} \subseteq \mathsf{exinf}$, $\mathsf{exinf}(\mathcal{S}(\sigma))$, which, by Lemma 3.9, yields an infinite solution. The seemingly stronger part 2

is a direct consequence of part 1: if σ has infinitely many solutions, not all of them can be finite, since this would give infinitely many inhabitants, which by part 1 would imply an infinite solution.[3] □

We now turn to the equivalent inclusions $\mathsf{fin} \subset \mathsf{allfin}$ and $\mathsf{exinf} \subset \mathsf{inf}$. We will show a (sufficient) condition on forests relative to which these strong inclusions turn into identities (of sets of forests satisfying this condition).

For a forest T, we say that "T has an empty sum" whenever in the generation process of T appears $E_1 + \cdots + E_n$ with $n = 0$, in other words, the empty sum $\mathbb{O}$. The negation of this property is written as "T has no empty sum".

Lemma 3.13. If $\mathsf{nosol}(T)$ then T has an empty sum.

Proof:
By induction on $\mathsf{nosol}(T)$. There are 3 cases, according to the 3 rules in the inductive definition of nosol—recall the upper half of Fig. 4. The first and third cases follow routinely by induction hypothesis. Let us detail the second case. Suppose $\mathsf{nosol}(\sum_i E_i)$ with $\mathsf{nosol}(E_i)$ for all i. If $T = \sum_i E_i$ itself is not an empty sum, then the empty sums that exist in E_i by induction hypothesis are also empty sums of T. □

Lemma 3.14. If T has no empty sum, then $\mathsf{allfin}(T)$ iff $\mathsf{fin}(T)$.

Proof:
Given $\mathsf{fin} \subset \mathsf{allfin}$, it suffices to prove, for all T such that $\mathsf{allfin}(T)$: If T has no empty sum then $\mathsf{fin}(T)$. The proof is by induction on $\mathsf{allfin}(T)$. Recall $\mathsf{allfin} = \mathsf{fin}^{\Pi_2}$, with $\Pi_2 = \mathsf{exsol}$. There are 5 cases, according to the 5 rules in the inductive definition of fin^{Π}—recall the upper half of Fig. 5. The second, third and fifth cases follow routinely by induction hypothesis. The remaining cases rely on Lemma 3.13. We detail the fourth case (the remaining first case is similar). Suppose $\mathsf{allfin}(x\langle N_i\rangle_i)$ with $\mathsf{nosol}(N_j)$ for some j (thanks to Lemma 3.1). Then T has an empty sum since already N_j has an empty sum, by Lemma 3.13. □

This result will have an important application towards the end of Section 4.

4. Decidability of finiteness

In this section we establish decidability of the finiteness predicate $\mathsf{fin}^{\Pi}(\mathcal{S}(\sigma))$ (when $\Pi(\mathcal{S}(\sigma))$ itself is decidable), and of the auxiliary predicate $\mathsf{nosol}(\mathcal{S}(\sigma))$. This is done following the methodology of [5] to obtain decidability of the predicates $\mathsf{exfin}(\mathcal{S}(\sigma))$ and $\mathsf{finfin}(\mathcal{S}(\sigma))$. The methodology rests on an equivalent, effective representation of solution spaces of sequents as expressions of the *finitary* extension $\lambda_{\Sigma}^{\mathsf{gfp}}$ of λ, introduced in [3, 4]. It comprises the definition of syntax-directed predicates over the finitary expressions of $\lambda_{\Sigma}^{\mathsf{gfp}}$ that characterize the uneffective predicates over forests to be decided. Such syntax-directed predicates lead immediately to simple recursive decision procedures.

[3] Notice that this argument is not specific to solutions of sequents but could be stated in terms of forests. However, we did not introduce the corresponding notion of infinity on forests that would ask if $\mathcal{E}(T)$ is infinite, in order to avoid confusion with the inf predicate.

4.1. Background: search for inhabitants, inductively

The syntax of the *finitary calculus* $\lambda_\Sigma^{\mathsf{gfp}}$ is given by the following grammar (read inductively):

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

where X is assumed to range over a countably infinite set of *fixpoint variables* (also letters Y, Z will range over them), and where, as for λ_Σ^{co}, both $n, k \geq 0$ are arbitrary. (Generally, the conventions adopted for the expressions of λ_Σ^{co} will also be adopted for the expressions of $\lambda_\Sigma^{\mathsf{gfp}}$.) If $n = 0$, we write $\mathbb{O}^\sigma$ for $\mathsf{gfp}\ X^\sigma.E_1 + \cdots + E_n$.

In the term formation rules, σ in X^σ is required to be *atomic*, i. e., of the form $\Gamma \Rightarrow p$. We write $FPV(T)$ to denote the set of free occurrences of typed fixpoint variables in T. We say T is *closed* when $FPV(T) = \emptyset$. In $\mathsf{gfp}\ X^\sigma.\sum_i E_i$ the fixed-point construction gfp binds *all* free occurrences of $X^{\sigma'}$ in the elimination alternatives E_i, not just X^σ, when $\sigma \leq \sigma'$. This raises the need for a notion of well-bound expression: $T \in \lambda_\Sigma^{\mathsf{gfp}}$ is *well-bound* if, for any of its subterms $\mathsf{gfp}\ X^\sigma.\sum_i E_i$ and any (free) occurrence of $X^{\sigma'}$ in the E_i's, $\sigma \leq \sigma'$.

In the sequel, when we refer to *finitary forests* we have in mind the expressions of $\lambda_\Sigma^{\mathsf{gfp}}$. (Recall we use forests for the expressions of λ_Σ^{co}.)

We recall now the *simplified interpretation* of expressions of $\lambda_\Sigma^{\mathsf{gfp}}$ in terms of the coinductive syntax of λ_Σ^{co} [5, Subsec. 3.2]. This simplified interpretation turns out to coincide with the original interpretation of expressions of $\lambda_\Sigma^{\mathsf{gfp}}$ introduced in [3, 4] for the $\lambda_\Sigma^{\mathsf{gfp}}$-terms representing solution spaces [5, Corollary 18].

Definition 4.1. (Simplified interpretation of finitary forests as forests)
For an expression T of $\lambda_\Sigma^{\mathsf{gfp}}$, the simplified interpretation $[\![T]\!]^s$ is a forest given by structural recursion on T:

$$\begin{array}{rclcrcl} [\![X^\sigma]\!]^s & = & \mathcal{S}(\sigma) & \qquad & [\![\lambda x^A.N]\!]^s & = & \lambda x^A.[\![N]\!]^s \\ [\![\mathsf{gfp}\ X^\sigma.\sum_i E_i]\!]^s & = & \sum_i [\![E_i]\!]^s & & [\![x\langle N_i\rangle_i]\!]^s & = & x\langle [\![N_i]\!]^s\rangle_i \end{array}$$

Note that the base case profits from the sequent annotation at fixpoint variables, and the interpretation of the gfp-constructor has nothing to do with a greatest fixed point, contrary to what happens in the original interpretation of expressions of $\lambda_\Sigma^{\mathsf{gfp}}$ given in [3, 4] (explaining the name of the fixed-point construction gfp).

We will be specially interested in the finitary forests which guarantee that a $\mathsf{gfp}\,X^\sigma$ construction represents the solution space of σ:

Definition 4.2. (Proper expressions)
An expression $T \in \lambda_\Sigma^{\mathsf{gfp}}$ is *proper* if for any of its subterms T' of the form $\mathsf{gfp}\ X^\sigma.\sum_i E_i$ (which could be T itself), it holds that $[\![T']\!]^s = \mathcal{S}(\sigma)$.

Now we recall the alternative representation $\mathcal{F}(\sigma)$ of the search space generated by a sequent σ as a finitary forest, introduced in [3, 4].

Definition 4.3. (Finitary solution space)
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 name and no sequent occurs twice. The specification of $\mathcal{F}(\sigma; \Xi)$ is as follows, with $\sigma = (\Gamma \Rightarrow \vec{A} \supset p)$:

If, for some $1 \leq i \leq m$, $p = q_i$ and $\Theta_i \subseteq \Gamma$ and $|\Theta_i| = |\Gamma| \cup \{A_1, \ldots, A_n\}$, then

$$\mathcal{F}(\sigma; \Xi) = \lambda z_1^{A_1} \ldots z_n^{A_n}.X_i^{\sigma'},$$

where i is taken to be the biggest such index. Otherwise,

$$\mathcal{F}(\sigma; \Xi) = \lambda z_1^{A_1} \ldots z_n^{A_n}.\mathsf{gfp}\ Y^{\sigma'}.\ \sum_{(y:\vec{B}\supset p)\in\Delta} y\langle \mathcal{F}(\Delta \Rightarrow B_j; \Xi, Y : \sigma')\rangle_j$$

where, in both cases, $\Delta := \Gamma, z_1 : A_1, \ldots, z_n : A_n$ and $\sigma' := \Delta \Rightarrow p$.

Notice that, in the first case, X_i occurs with sequent σ' in the resulting finitary forest instead of with $\Theta_i \Rightarrow p$ in Ξ, but $\Theta_i \leq \Delta$, hence $(\Theta_i \Rightarrow p) \leq \sigma'$ makes it plausible that this process generates well-bound terms.

$\mathcal{F}(\sigma)$ denotes $\mathcal{F}(\sigma; \Xi)$ with empty Ξ.

Fact 4.4. [4, Lemma 52 and Lemma 53]

1. $\mathcal{F}(\sigma)$ is well-defined (the above recursive definition terminates).
2. $\mathcal{F}(\sigma)$ is a closed well-bound term.

This result is central to our methodology of decision algorithms that conform to a coinductive specification. Its proof is by establishing an invariant on the recursive calls to $\mathcal{F}$. It is a manifestation of the "subformula property" of the fragment λ of lambda-calculus we are studying in this paper. The proof we gave in *op. cit.* does not try to bound recursion depth polynomially in the length of σ. However, we could profit from very recent work on the analysis of inhabitation problems through the pre-grammar of a given type A [7]. First of all, it exploits that, in the implicational fragment we are considering, the search for proofs of A only generates goals that are subformulas at positive positions of A, with assumptions that are negative subpremises of A (occur at specific negative positions of A, see already [8]). Inspired by [7, Proposition 33], we can bound the number of declarations in Ξ for any $\mathcal{F}(\sigma'; \Xi)$ appearing in the definition of $\mathcal{F}(\Rightarrow A)$ by the product of the number of atoms at positive positions in A and the number of negative subpremises of A.[4] Since this optimization of the analysis is orthogonal to the aims of the present paper, it will be detailed elsewhere.

Example 4.5. Let us see the finitary solution space of each of the sequents in Example 2.2:

- $\mathcal{F}(\sigma_a) = \mathbb{O}^{\sigma_a}$.
- $\mathcal{F}(\sigma_b) = \mathsf{gfp}\ X^{\sigma_b}.x\langle X^{\sigma_b}\rangle$.

[4] This quantitative result only changes the analysis, not the definition. It needs a modification of the invariant used in the proof of [4, Lemma 52].

Figure 6. NES and ES predicate

$$\frac{P(\sigma)}{\mathsf{NES}_P(X^\sigma)} \qquad \frac{\mathsf{NES}_P(N)}{\mathsf{NES}_P(\lambda x^A.N)} \qquad \frac{\forall i,\, \mathsf{NES}_P(E_i)}{\mathsf{NES}_P(\mathsf{gfp}\, X^\sigma.\, \sum_i E_i)} \qquad \frac{\mathsf{NES}_P(N_j)}{\mathsf{NES}_P(x\langle N_i\rangle_i)}$$

$$\frac{\neg P(\sigma)}{\mathsf{ES}_P(X^\sigma)} \qquad \frac{\mathsf{ES}_P(N)}{\mathsf{ES}_P(\lambda x^A.N)} \qquad \frac{\mathsf{ES}_P(E_j)}{\mathsf{ES}_P(\mathsf{gfp}\, X^\sigma.\, \sum_i E_i)} \qquad \frac{\forall i,\, \mathsf{ES}_P(N_i)}{\mathsf{ES}_P(x\langle N_i\rangle_i)}$$

- $\mathcal{F}(\sigma_c) = \mathsf{gfp}\ X^{\sigma_c}.x\langle X^{\sigma_c}, \mathbb{O}^{x:p\supset q\supset p\Rightarrow q}\rangle$.
- $\mathcal{F}(\sigma_d) = \mathsf{gfp}\ X^{\sigma_d}.(x\langle X^{\sigma_d}, z\rangle + y)$.
- $\mathcal{F}(\sigma_e) = \mathsf{gfp}\ X^{\sigma_e}.(x\langle X^{\sigma_e}\rangle + y)$.

Not surprisingly, the meta-level fixed points of the coinductive representations turn into formal fixed points of the finitary calculus. In these simple examples the occurrences of fixpoint variables always have the sequent of the respective binder. When the decontration phenomenon shows up, we observe occurrences of fixpoint variables with sequents which are different, but in any case inessential extensions of the sequents in the respective binders. For a more complicated example where decontraction is seen to play a role, we refer to [5, Example 11].

The semantics into λ^{co}_{Σ} of the finitary representation coincides with $\mathcal{S}(\sigma)$.

Fact 4.6. (Equivalence for simplified semantics [5, Theorem 19])
Let σ be a sequent.

1. $\mathcal{F}(\sigma)$ is proper.
2. $[\![\mathcal{F}(\sigma)]\!]^s = \mathcal{S}(\sigma)$.

4.2. Auxiliary result: decidability of absence of solutions

Analogously to the inductive definition of EF and its negation for the decision of $\mathsf{exfin} \circ \mathcal{S}$ in [5, Section 3.3], we introduce a parameterized inductive predicate NES and its negation for the decision of $\mathsf{nosol} \circ \mathcal{S}$.

We consider a predicate P on sequents and in general require that P is decidable and that, for all sequents σ, $P(\sigma)$ implies $\mathsf{nosol}(\mathcal{S}(\sigma))$, i. e., $P \subseteq \mathsf{nosol} \circ \mathcal{S}$.

The definition of this (parameterized) predicate NES_P is inductive and presented in the first line of Fig. 6, although, as in [5] for EF_P, it is clear that it could equivalently be given by a definition by recursion over the term structure. Thus, the predicate NES_P is decidable. Intuitively, $\mathsf{NES}_P(T)$ holds when the search for a member of T gets "stuck". For this question, all sequents σ for which $P(\sigma)$ holds, are considered as "hopeless", which is why those X^σ are then also considered as "stuck". As

in [5], our first instance of P is $P := \emptyset$, hence without "hopeless" sequents, which will be used in part 2 of Theorem 4.9 below.

Formally, the definition of NES_P only differs from EF_P in [5, Fig. 6] in having the universal quantifier over i in the case of fixed points over sums and the (implicit) existential quantifier over j in the case of tuples, while this is the other way around for EF_P. This is the reason why most proofs in this section are variants of those in [5, Section 3.3].

Lemma 4.7. For all $T \in \lambda_\Sigma^{\mathsf{gfp}}$, $\mathsf{ES}_P(T)$ iff $\mathsf{NES}_P(T)$ does not hold.

Proof:
As straightforward as the proof of [5, Lemma 22]. □

Proposition 4.8. (Finitary characterization)

1. If $\mathsf{NES}_P(T)$ then $\mathsf{nosol}([\![T]\!]^s)$.

2. Let $T \in \lambda_\Sigma^{\mathsf{gfp}}$ be well-bound and proper. If $\mathsf{ES}_P(T)$ and for all $X^\sigma \in FPV(T)$, $\mathsf{nosol}(\mathcal{S}(\sigma))$ implies $P(\sigma)$, then $\mathsf{exsol}([\![T]\!]^s)$.

The proof follows that of [5, Prop. 2] in lockstep (except for the fact that the clauses of NES_P for fixed points and tuples switched their logical connector in the premise, relative to EF_P, as mentioned above).

Proof:
1. is proved by induction on the predicate NES_P (or, equivalently, on T). The base case for fixpoint variables needs the proviso on P, and all other cases are immediate by the induction hypothesis.

2. is proved by induction on the predicate ES_P (which can also be seen as a proof by induction on T).

Case $T = X^\sigma$. Then $\neg P(\sigma)$, hence, since $X^\sigma \in FPV(T)$, by contraposition and Lemma 3.1, we get $\mathsf{exsol}(\mathcal{S}(\sigma))$.

Case $T = \mathsf{gfp}\, X^\sigma.\, \sum_i E_i$. Let $N := [\![T]\!]^s = \sum_i [\![E_i]\!]^s$. As T is proper, $N = \mathcal{S}(\sigma)$. We hence have to show $\mathsf{exsol}(\mathcal{S}(\sigma))$, which we do by an embedded coinduction for the coinductively defined predicate exsol. Coinduction is now used in a more sophisticated way than in the proof of Lemma 3.2 where backwards closure of a suitable relation suffices. We have to establish evidence for $\mathsf{exsol}(\mathcal{S}(\sigma))$, and we may use evidence for $\mathsf{exsol}(\mathcal{S}(\sigma))$ as coinductive hypothesis, provided that use is "guarded" by the construction process: if we set out to observe the generated proof of $\mathsf{exsol}(\mathcal{S}(\sigma))$ up to a certain depth (in a "thought process"), we have to make sure (in the "thought process") that the construction only accesses the coinductive hypothesis up to a smaller depth. The construction of evidence is as follows: We have $\mathsf{ES}_P(E_j)$ for some j and want to use the induction hypothesis, which would give us $\mathsf{exsol}([\![E_j]\!]^s)$ and thus $\mathsf{exsol}(\sum_i [\![E_i]\!]^s)$, which was our goal. Of course, E_j is also well-bound and proper. We have to consider all $Y^{\sigma'} \in FPV(E_j)$. Either $Y^{\sigma'} \in FPV(T)$, and we are fine by hypothesis, or $Y = X$ and, since T is well-bound, $\sigma \leq \sigma'$. We just show that $\mathsf{nosol}(\mathcal{S}(\sigma'))$ does not hold: from our coinductive hypothesis $\mathsf{exsol}(\mathcal{S}(\sigma))$, we get through Fact 2.6 and Lemma 3.4 even $\mathsf{exsol}(\mathcal{S}(\sigma'))$, and this is the negation of $\mathsf{nosol}(\mathcal{S}(\sigma'))$. (See the corresponding case in the proof of Proposition 4.14.2 that argues in an analogous situation that this is proper coinductive reasoning, in

other words, that the proof construction is guarded. An even closer analogy is with the proof of [5, Proposition 2].)

The other cases are simple applications of the induction hypothesis. □

Theorem 4.9. (Decidability of existence of solutions)

1. For any $T \in \lambda_{\Sigma}^{\mathsf{gfp}}$ well-bound, proper and closed, $\mathsf{NES}_P(T)$ iff $\mathsf{nosol}([\![T]\!]^s)$.

2. $\mathsf{nosol}(\mathcal{S}(\sigma))$ is decided, by deciding $\mathsf{NES}_\emptyset(\mathcal{F}(\sigma))$.

Proof:

1. Follows from both parts of Prop. 4.8, Lemmas 3.1 and 4.7, and the fact that, trivially, the extra condition in Prop. 4.8.2 is satisfied for closed terms.

2. Apply 1. with both parts of Fact 4.6. □

Definition 4.10. Let the predicates $\mathsf{NES}_\star$ and $\mathsf{ES}_\star$ on $\lambda_{\Sigma}^{\mathsf{gfp}}$ be defined by $\mathsf{NES}_\star := \mathsf{NES}_P$ and $\mathsf{ES}_\star := \mathsf{ES}_P$ for $P(\sigma) :\Leftrightarrow \mathsf{NES}_\emptyset(\mathcal{F}(\sigma))$, which satisfies the proviso by Theorem 4.9.2. In particular, $\mathsf{NES}_\star$ and $\mathsf{ES}_\star$ are decidable.

As for $\mathsf{EF}_\star$ in [5], we get an improvement (no conditions on T are needed) over the theorem for this special situation.

Lemma 4.11. (Sharp finitary characterization)
For all $T \in \lambda_{\Sigma}^{\mathsf{gfp}}$, $\mathsf{NES}_\star(T)$ iff $\mathsf{nosol}([\![T]\!]^s)$.

Proof:
In lockstep with the proof of [5, Lemma 27] (again modulo the interchange of quantifiers between the rules for sums and tuples). □

In particular, $\mathsf{nosol}([\![T]\!]^s)$ is decidable, by deciding $\mathsf{NES}_\star(T)$.

We close this discussion by a necessary condition on finitary expressions T to satisfy $\mathsf{NES}_\emptyset(T)$. For $T \in \lambda_{\Sigma}^{\mathsf{gfp}}$, we say that "$T$ has an empty sum" whenever T has a (not necessarily proper) subterm of the form $\mathsf{gfp}\ X^\sigma.E_1 + \cdots + E_n$ with $n = 0$. The negation of this property is written as "T has no empty sum".

Lemma 4.12. If $\mathsf{NES}_\emptyset(T)$ then T has an empty sum.

Proof:
By induction on $\mathsf{NES}_\emptyset(T)$. There are 4 cases, according to the 4 rules in the inductive definition of $\mathsf{NES}_\emptyset$—recall the upper half of Fig. 6. The first rule is never applicable, while the second and fourth cases follow routinely by induction hypothesis. Let us detail the third case. Suppose $\mathsf{NES}_\emptyset(\mathsf{gfp}\ X^\sigma.\sum_i E_i)$ with $\mathsf{NES}_\emptyset(E_i)$ for all i. If $\sum_i E_i$ itself is not an empty sum, then the empty sums that exist in E_i by induction hypothesis are also empty sums of the whole expression. □

Figure 7. F^{Π}_P and NF^{Π}_P predicates, for P satisfying the *proviso*: $P \subseteq \mathsf{fin}^{\Pi} \circ \mathcal{S}$ and P decidable.

$$\frac{P(\sigma)}{\mathsf{F}^{\Pi}_P(X^\sigma)} \qquad \frac{\mathsf{F}^{\Pi}_P(N)}{\mathsf{F}^{\Pi}_P(\lambda x^A.N)} \qquad \frac{\forall i,\ \mathsf{F}^{\Pi}_P(E_i)}{\mathsf{F}^{\Pi}_P(\mathsf{gfp}\, X^\sigma.\sum_i E_i)} \qquad \frac{\forall i,\ \mathsf{F}^{\Pi}_P(N_i)}{\mathsf{F}^{\Pi}_P(x\langle N_i\rangle_i)} \qquad \frac{\neg\Pi([\![N_j]\!]^s)}{\mathsf{F}^{\Pi}_P(x\langle N_i\rangle_i)}$$

$$\frac{\neg P(\sigma)}{\mathsf{NF}^{\Pi}_P(X^\sigma)} \qquad \frac{\mathsf{NF}^{\Pi}_P(N)}{\mathsf{NF}^{\Pi}_P(\lambda x^A.N)} \qquad \frac{\mathsf{NF}^{\Pi}_P(E_j)}{\mathsf{NF}^{\Pi}_P(\mathsf{gfp}\, X^\sigma.\sum_i E_i)} \qquad \frac{\mathsf{NF}^{\Pi}_P(N_j) \quad \forall i,\ \Pi([\![N_i]\!]^s)}{\mathsf{NF}^{\Pi}_P(x\langle N_i\rangle_i)}$$

4.3. Decidability of parameterized finiteness

Analogously to the previous section, we now develop the companion finitary predicates that will lead to a decidability result on fin^{Π} and its instances.

Given a decidable predicate P over sequents such that $P \subseteq \mathsf{fin}^{\Pi} \circ \mathcal{S}$, the doubly parametric predicate F^{Π}_P over $\lambda^{\mathsf{gfp}}_{\Sigma}$ is defined in the first line of Fig. 7. Again, it is clear that when $\Pi \circ [\![\cdot]\!]^s$ is decidable so is F^{Π}_P. As before, we also define inductively the negation of F^{Π}_P. This is through the also doubly parametric predicate NF^{Π}_P in the second line of Fig. 7. Below, when F^{Π}_P or NF^{Π}_P is written, it is implicitly assumed that P satisfies the proviso of Fig. 7.

Lemma 4.13. For all $T \in \lambda^{\mathsf{gfp}}_{\Sigma}$, $\mathsf{NF}^{\Pi}_P(T)$ iff $\mathsf{F}^{\Pi}_P(T)$ does not hold.

Proof:
Routine induction on T. □

Before proving the main results of the section, linking the coinductive predicates and the inductive ones, we consider the example $\mathsf{F}^{\Pi_1}_P$. We will argue that indeed it coincides with the predicate FF_P in [5, Fig. 8], used to capture finiteness of the finite extension of sequents. First observe that the definitions of $\mathsf{F}^{\Pi_1}_P$ and FF_P impose the same proviso on P, since $\mathsf{fin}^{\Pi_1} = \mathsf{finfin}$. It is immediate to see that the rules defining the two predicates coincide except for one of the tuple rules. For this case, we need to argue: $\neg\mathsf{exfin}([\![N]\!]^s)$ iff $\mathsf{NEF}_\star(N)$. This is immediate from Lemmas 22 and 27 of [5]. Obviously, the negations of $\mathsf{F}^{\Pi_1}_P$ and FF_P must also coincide, i. e., $\mathsf{NF}^{\Pi_1}_P = \mathsf{NFF}_P$ (the latter defined in [5, Fig. 8]).

Proposition 4.14. (Finitary characterization)

1. If $\mathsf{F}^{\Pi}_P(T)$ then $\mathsf{fin}^{\Pi}([\![T]\!]^s)$.
2. Let $T \in \lambda^{\mathsf{gfp}}_{\Sigma}$ be well-bound and proper, and assume inf^{Π} is a subset of Π, and Π is closed under decontraction both ways. If $\mathsf{NF}^{\Pi}_P(T)$ and for all $X^\sigma \in FPV(T)$, $\mathsf{fin}^{\Pi}(\mathcal{S}(\sigma))$ implies $P(\sigma)$, then $\mathsf{inf}^{\Pi}([\![T]\!]^s)$.

Proof:
A generalization of the proof of [5, Prop. 3].

1. By induction on F^{Π}_P (or equivalently by structural induction on T).

Case $T = X^\sigma$. From the assumption $\mathsf{F}^\Pi_P(X^\sigma)$ follows $P(\sigma)$. The proviso on P gives $\mathsf{fin}^\Pi(\mathcal{S}(\sigma))$. Hence $\mathsf{fin}^\Pi([\![X^\sigma]\!]^s)$ by definition of $[\![\cdot]\!]^s$.

Case $T = x\langle N_i\rangle_i$. From the assumption, one of two sub-cases holds.

Sub-case for some j, $\neg\Pi([\![N_j]\!]^s)$. By the first rule for tuples, $\mathsf{fin}^\Pi(x\langle [\![N_i]\!]^s\rangle_i)$, hence we obtain $\mathsf{fin}^\Pi([\![x\langle N_i\rangle_i]\!]^s)$.

Sub-case for all i, $\mathsf{F}^\Pi_P(N_i)$. By induction hypothesis, $\mathsf{fin}^\Pi([\![N_i]\!]^s)$ for all i, hence $\mathsf{fin}^\Pi([\![x\langle N_i\rangle_i]\!]^s)$.

The other inductive cases are equally simple.

2. By induction on NF^Π_P (or equivalently by structural induction on T).

Case $T = X^\sigma$. Then $\neg P(\sigma)$. So, since $X^\sigma \in FPV(T)$, by contraposition, $\neg\mathsf{fin}^\Pi(\mathcal{S}(\sigma))$. Hence, by Lemma 3.5, $\mathsf{inf}^\Pi(\mathcal{S}(\sigma))$, which is the same as $\mathsf{inf}^\Pi([\![X^\sigma]\!]^s)$, by definition of $[\![\cdot]\!]^s$.

Case $T = x\langle N_i\rangle_i$. Then, for some j, $\mathsf{NF}^\Pi_P(N_j)$ and, for all i, $\Pi([\![N_i]\!]^s)$. The induction hypothesis is applicable for N_j since $FPV(N_j) \subseteq FPV(T)$. Therefore, we have $\mathsf{inf}^\Pi([\![N_j]\!]^s)$. So, the tuple rule can be applied to give $\mathsf{inf}^\Pi(x\langle [\![N_i]\!]^s\rangle_i)$, hence $\mathsf{inf}^\Pi([\![x\langle N_i\rangle_i]\!]^s)$.

Case $T = \lambda x^A.N$. Then, we have $\mathsf{NF}^\Pi_P(N)$. The induction hypothesis is applicable for N, since $FPV(N) \subseteq FPV(T)$, therefore we obtain $\mathsf{inf}^\Pi([\![N]\!]^s)$. Using the assumption $\mathsf{inf}^\Pi \subseteq \Pi$, we get $\mathsf{inf}^\Pi(\lambda x^A.[\![N]\!]^s)$, hence $\mathsf{inf}^\Pi([\![\lambda x^A.N]\!]^s)$. (This is the only place in the proof where the assumption $\mathsf{inf}^\Pi \subseteq \Pi$ is used.)

Case $T = \mathsf{gfp}\, X^\sigma.\sum_i E_i$. Then, for some j, $\mathsf{NF}^\Pi_P(E_j)$. Let $N := [\![T]\!]^s = \sum_i [\![E_i]\!]^s$. As T is proper, $N = \mathcal{S}(\sigma)$. We have to show $\mathsf{inf}^\Pi(\mathcal{S}(\sigma))$. This is done by an embedded coinduction on the coinductively defined predicate inf^Π, using the concept of guardedness as in the proof of Proposition 4.8.2, showing $\mathsf{inf}^\Pi(\mathcal{S}(\sigma))$ with limited access to the same $\mathsf{inf}^\Pi(\mathcal{S}(\sigma))$ as coinductive hypothesis. We want to use the induction hypothesis for E_j, which would give us $\mathsf{inf}^\Pi([\![E_j]\!]^s)$ and thus $\mathsf{inf}^\Pi(\sum_i [\![E_i]\!]^s)$, our goal. Since T is well-bound and proper, so is E_j. We have to consider all $Y^{\sigma'} \in FPV(E_j)$. Either $Y^{\sigma'} \in FPV(T)$, and we are fine by hypothesis, or $Y = X$ and, since T is well-bound, $\sigma \leq \sigma'$. We just show that $\mathsf{fin}^\Pi(\mathcal{S}(\sigma'))$ does not hold. From Fact 2.6, we know $\mathcal{S}(\sigma') = [\sigma'/\sigma]\mathcal{S}(\sigma)$. The assumption that Π is closed under decontraction both ways gives that inf^Π is also closed under decontraction (Lemma 3.10). Hence, applying our coinductive hypothesis $\mathsf{inf}^\Pi(\mathcal{S}(\sigma))$, we get $\mathsf{inf}^\Pi(\mathcal{S}(\sigma'))$, which is the negation of $\mathsf{fin}^\Pi(\mathcal{S}(\sigma'))$ (Lemma 3.5). The application of the coinductive hypothesis is guarded (and hence the whole proof construction a proper use of the coinduction principle) since it enters a lemma on inf^Π that does not change needed observation depths and then goes into an elimination alternative, where the occurrences of free fixpoint variables are at least "guarded" by an ordinary variable of a tuple, which creates extra depth of the outcome of the proof construction. (We invite the reader to check that the whole argument just abstracts away from the specific situation of [5, Proposition 3] but is structurally the same, using in particular the same coinductive reasoning.)

□

Theorem 4.15. (Deciding generalized finiteness)
Let Π be closed under decontraction both ways and such that $\mathsf{inf}^\Pi \subseteq \Pi$.

1. For any $T \in \lambda^{\mathsf{gfp}}_\Sigma$ well-bound, proper and closed, $\mathsf{F}^\Pi_P(T)$ iff $\mathsf{fin}^\Pi([\![T]\!]^s)$.

2. If $\Pi \circ [\![\cdot]\!]^s$ is decidable, $\mathsf{fin}^\Pi(\mathcal{S}(\sigma))$ is decided by deciding $\mathsf{F}^\Pi_\emptyset(\mathcal{F}(\sigma))$.

Proof:
1. Follows from both parts of Prop. 4.14, Lemmas 3.5 and 4.13, and the fact that, trivially, the extra condition in Prop. 4.14.2 is satisfied for closed terms.

2. By 1 and both parts of Fact 4.6. $\mathsf{fin}^{\Pi}(\mathcal{S}(\sigma))$ iff $\mathsf{F}^{\Pi}_{\emptyset}(\mathcal{F}(\sigma))$. Then use computability of $\mathcal{F}$ and the equivalence of the inductively defined $\mathsf{F}^{\Pi}_{\emptyset}$ with a recursive procedure over the term structure of its argument, where, corresponding to the last rule of Fig. 7, the decisions for predicate $\Pi \circ [\![\cdot]\!]^s$ are invoked. □

Corollary 4.16. 1. $\mathsf{finfin}(\mathcal{S}(\sigma))$ is decided by deciding $\mathsf{F}^{\Pi_1}_{\emptyset}(\mathcal{F}(\sigma))$.

2. $\mathsf{exinf}(\mathcal{S}(\sigma))$ is decided by deciding $\mathsf{F}^{\Pi_2}_{\emptyset}(\mathcal{F}(\sigma))$.

3. $\mathsf{fin}(\mathcal{S}(\sigma))$ is decided by deciding $\mathsf{F}^{\Pi_3}_{\emptyset}(\mathcal{F}(\sigma))$.

Proof:
By the previous theorem, it suffices to argue (a) $\mathsf{inf}^{\Pi_i} \subseteq \Pi_i$, (b) Π_i is closed under decontraction both ways and (c) $\Pi_i \circ [\![\cdot]\!]^s$ is decidable, for $i \in \{1, 2, 3\}$.

Case Π_1. Regarding (a), we need $\mathsf{inffin} \subseteq \mathsf{exfin}$, which is an easy consequence of the obvious $\mathsf{inffinext} \subseteq \mathsf{exfinext}$ and [5, Lemmas 20, 21, 28, 29] (that allow to replace the extensional versions of the predicates by their characterizations). Regarding (b), we need exfin to be closed under decontraction both ways, which is established in [5, Lemma 23]. Regarding (c), we note that decidability of $\mathsf{exfin} \circ [\![\cdot]\!]^s$ is a consequence of the sharp finitary characterization of exfin in [5, Lemma 27].

Case Π_2. Regarding (a), we need $\mathsf{exinf} \subseteq \mathsf{exsol}$, which follows by coinduction on exsol (one can also prove the equivalent $\mathsf{nosol} \subseteq \mathsf{allfin}$ by induction on nosol). (b) follows from Lemma 3.4. (c) is a consequence of the sharp finitary characterization of exsol in Lemma 4.11.

Case Π_3. As $\Pi_3 = \lambda^{co}_{\Sigma}$, the three conditions hold trivially. □

Notice that part 1 of the corollary is only another form of stating the result in [5, Theorem 33.3], while the other parts are original contributions of the present paper. Notice also part 2 implies that $\mathsf{allfin}(\sigma)$ is decidable, hence, termination of proof search is decidable. (Recall in Section 1 we argued that the property of all solutions being finite corresponds to termination of proof search.)

The analysis we did with the sharp finitary characterizations $\mathsf{NES}_\star$ and $\mathsf{ES}_\star$ can be replayed.

Definition 4.17. Let Π be closed under decontraction both ways and such that $\mathsf{inf}^{\Pi} \subseteq \Pi$ and $\Pi \circ [\![\cdot]\!]^s$ is decidable. The predicates $\mathsf{F}^{\Pi}_{\star}$ and $\mathsf{NF}^{\Pi}_{\star}$ on $\lambda^{\mathsf{gfp}}_{\Sigma}$ are defined by $\mathsf{F}^{\Pi}_{\star} := \mathsf{F}^{\Pi}_{P}$ and $\mathsf{NF}^{\Pi}_{\star} := \mathsf{NF}^{\Pi}_{P}$ for $P := \mathsf{fin}^{\Pi} \circ \mathcal{S}$, which satisfies the proviso for P thanks to Theorem 4.15.2 and the assumptions on Π. In particular, $\mathsf{F}^{\Pi}_{\star}$ and $\mathsf{NF}^{\Pi}_{\star}$ are decidable.

Lemma 4.18. (Sharp finitary characterization)
Let Π be closed under decontraction both ways and such that $\mathsf{inf}^{\Pi} \subseteq \Pi$ and $\Pi \circ [\![\cdot]\!]^s$ is decidable. For all $T \in \lambda^{\mathsf{gfp}}_{\Sigma}$, $\mathsf{F}^{\Pi}_{\star}(T)$ iff $\mathsf{fin}^{\Pi}([\![T]\!]^s)$.

Proof:
The "only if" direction follows by part 1 of Prop. 4.14. For the "if" direction, one proves the contrapositive that $\mathsf{NF}^{\Pi}_{\star}(T)$ implies $\mathsf{inf}^{\Pi}([\![T]\!]^s)$ (thanks to Lemma 3.5 and Lemma 4.13), by an easy induction on $\mathsf{NF}^{\Pi}_{\star}$. □

In particular, note that for the three instances of Π considered above, $\mathsf{fin}^{\Pi}([\![T]\!]^s)$ is decidable, by deciding $\mathsf{F}^{\Pi}_{\star}(T)$. (Recall the proof of Cor. 4.16, where it is already argued why the three conditions on Π hold for $\Pi \in \{\lambda^{co}_{\Sigma}, \mathsf{exfin}, \mathsf{exsol}\}$.)

Other instances of Π can be considered for which Theorem 4.15.2 and Lemma 4.18 produce decidability results.

Example 4.19. Let $\Pi_4 := \mathsf{inffin} = \mathsf{inf}^{\mathsf{exfin}}$. We have Π_4 closed under decontraction both ways (by Lemma 3.10 and the fact that exfin also has this property, as observed in the proof of Corollary 4.16), $\mathsf{inf}^{\Pi_4} \subseteq \Pi_4$ (which follows by $\Pi_4 = \mathsf{inf}^{\mathsf{exfin}}$, monotonicity of inf^{Π} in Π, and $\mathsf{inffin} \subseteq \mathsf{exfin}$), and $\Pi_4 \circ [\![\cdot]\!]^s$ is decidable (Lemma 4.18 and the remark following it). So, $\mathsf{fin}^{\Pi_4}([\![T]\!]^s)$ (and in particular $\mathsf{fin}^{\Pi_4}(\mathcal{S}(\sigma))$) is decidable. Additionally, note that $\mathsf{inf}^{\Pi_4} \subset \mathsf{inffin}$. The weak inclusion is the already stated $\mathsf{inf}^{\Pi_4} \subseteq \Pi_4$, and for the sequent σ_d of Example 2.2, we get $\mathsf{inffin}(\mathcal{S}(\sigma_d))$ and $\mathsf{fin}^{\Pi_4}(\mathcal{S}(\sigma_d))$ (for the rather trivial reason that $\mathsf{inffin}(z)$ does not hold), hence $\mathsf{inf}^{\Pi_4}(\mathcal{S}(\sigma_d))$ cannot hold. We also note that the predicate "$\mathsf{inf}^{\Pi_4}(\mathcal{S}(\sigma))$" does not trivialise to the empty set, as, for example, $\mathsf{inf}^{\Pi_4}(\mathcal{S}(\sigma_e))$ holds (since $\mathsf{inf}^{\Pi_4}(x\langle\mathcal{S}(\sigma_e)\rangle)$ follows coinductively thanks to the fact $\mathsf{inffin}(\mathcal{S}(\sigma_e))$). So, inf^{Π_4} gives rise to yet a new decidable notion of (in)finiteness for simple types.

We are not yet aware of uses of these other possible notions of "finiteness" (as the example showed, this can be a rather wide extension of usual notions of finiteness).

4.4. Applications of decidability results

We close the section with three applications of the decidability results. The first is a sharpening of a coherence theorem, the second is the definition of the pruned solution space of a sequent, the third is a kind of König's lemma for simple types.

Part 3 of the corollary allows us to sharpen (see Theorem 4.21 below) a result by Broda and Damas [8] that we reproved with our method as [5, Theorem 46.1]: If no atom occurs positively more than once in A, then A has only finitely many inhabitants. (Positive and negative occurrences in a formula are defined as usual, with change of *polarity* when moving to the left argument of $\supset$). In [5] we proved this in two steps. Let a finitary expression T be called *strongly acyclic* if T has no occurrence, free or bound, of fixpoint variables (other than the binding occurrences after gfp) [5, Definition 43.1]. In [5, Lemma 44.1], we showed the very simple fact that if T is strongly acyclic, then a predicate equivalent to $\mathsf{F}^{\Pi_1}_{\emptyset}$ holds of T. A more profound analysis [5, Lemma 45.1] showed that if no atom occurs positively more than once in A, then $\mathcal{F}(\Rightarrow A)$ is strongly acyclic. Then, a result similar to Corollary 4.16.1 was invoked. Now we prove the sharpened theorem.

Lemma 4.20. Let $T \in \lambda^{\mathsf{gfp}}_{\Sigma}$. Then T is strongly acyclic iff $\mathsf{F}^{\Pi_3}_{\emptyset}(T)$.

Proof:
For the "only if" direction, we can do a straightforward induction on T (as was done for [5, Lemma 44.1]). The parameters of the predicate play no role at all, which is why they are set here to the values that yield the smallest possible predicate. For the "if" direction, we just reason by induction on $\mathsf{F}_{\emptyset}^{\Pi_3}$ and observe that the first and fifth rule in Fig. 7 cannot have been used thanks to this specific choice of parameters. □

We now obtain that the syntactic criterion of no atom occurring positively more than once in the type A guarantees finiteness of the solution space of A in the strongest sense of finiteness we consider in this paper.

Theorem 4.21. (Generalizing the positive part of generalized coherence)
If no atom occurs positively more than once in A, then $\mathsf{fin}(\mathcal{S}(\Rightarrow A))$, which in particular implies that A has only finitely many inhabitants and only finite solutions.

Proof:
If no atom occurs positively more than once in A, then $\mathcal{F}(\Rightarrow A)$ is strongly acyclic by [5, Lemma 45.1] (as mentioned above), hence by the previous lemma $\mathsf{F}_{\emptyset}^{\Pi_3}(\mathcal{F}(\Rightarrow A))$, but this equivalent to $\mathsf{fin}(\mathcal{S}(\sigma))$ by Corollary 4.16.3. □

Decidability of the predicate $\mathsf{ES}_\star(T)$ allows the definition of a refined solution space for a given sequent.

Definition 4.22. (Pruned solution space of a sequent)

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

where

- $\underline{\lambda}x : A.T := \lambda x : A.T$, if $T \neq \mathbb{O}$; and $\underline{\lambda}x : A.T := \mathbb{O}$, otherwise.
- $(y : \vec{B} \supset p)\underline{\in}\Delta :\Leftrightarrow (y : \vec{B} \supset p) \in \Delta$ and, for all j, $\mathsf{ES}_\star(\mathcal{F}(\Delta \Rightarrow B_j))$.

Comparing with Def. 2.1: if the displayed sum above is empty, it annihilates the enclosing λ's; and when forming the displayed sum above, we filter out those summands $y\langle N_j\rangle_j$ with $\mathsf{ES}_\star(N_j)$ failing for some j, because such summands contribute no solution.

Lemma 4.23. (Properties of the pruned solution space)

1. $\mathcal{E}(\underline{\mathcal{S}}(\sigma)) = \mathcal{E}(\mathcal{S}(\sigma))$.
2. If σ has a solution, then $\underline{\mathcal{S}}(\sigma)$ has no empty sum.
3. If σ has no solution, then $\underline{\mathcal{S}}(\sigma) = \mathbb{O}$.

Proof:
We prove part 2 first. Let us write $\mathsf{no}\mathbb{O}(T)$ to mean that T has no empty sum. So we want to prove: if $\mathsf{exsol}(\mathcal{S}(\sigma))$ then $\mathsf{no}\mathbb{O}(\underline{\mathcal{S}}(\sigma))$. A coinductive characterization of $\mathsf{no}\mathbb{O}(\underline{\mathcal{S}}(\sigma))$ is as follows:

$$\frac{\mathsf{no}\mathbb{O}(\underline{\mathcal{S}}(\Gamma, x : A \Rightarrow B))}{\mathsf{no}\mathbb{O}(\underline{\mathcal{S}}(\Gamma \Rightarrow A \supset B))}\ (a) \qquad \frac{\exists(y : \vec{B} \supset p)\underline{\in}\Gamma \quad \forall(y : \vec{B} \supset p)\underline{\in}\Gamma\ \forall j,\ \mathsf{no}\mathbb{O}(\underline{\mathcal{S}}(\Gamma \Rightarrow B_j))}{\mathsf{no}\mathbb{O}(\underline{\mathcal{S}}(\Gamma \Rightarrow p))}\ (b)$$

The proof is by coinduction on $\mathsf{no}\mathbb{O}(\underline{\mathcal{S}}(\sigma))$, i. e., we prove $\mathsf{exsol}(\mathcal{S}(\sigma))$ is backward closed w. r. t. (a) and (b). Recall the lower half of Fig. 4.

(a) Suppose $\mathsf{exsol}(\mathcal{S}(\Gamma \Rightarrow A \supset B))$. We want $\mathsf{exsol}(\mathcal{S}(\Gamma, x : A \Rightarrow B))$. If $\mathcal{S}(\Gamma, x : A \Rightarrow B) = N$, then $\mathcal{S}(\Gamma \Rightarrow A \supset B) = \lambda x : A.N$. We are done by the first rule for exsol in that figure.

(b) Suppose $\mathsf{exsol}(\mathcal{S}(\Gamma \Rightarrow p))$. We need to prove: (b1) $\exists(y : \vec{B} \supset p)\underline{\in}\Gamma$; (b2) $\forall(y : \vec{B} \supset p)\underline{\in}\Gamma\ \forall j$, $\mathsf{exsol}(\mathcal{S}(\Gamma \Rightarrow B_j))$. From the definition of $\mathcal{S}$ and the second and third rules in the mentioned figure, we see that $\mathsf{exsol}(\mathcal{S}(\Gamma \Rightarrow p))$ implies that there is $(y : \vec{B} \supset p) \in \Gamma$ such that, for all j, $\mathsf{exsol}(\mathcal{S}(\Gamma \Rightarrow B_j))$. Having in mind the definition of $(y : \vec{B} \supset p)\underline{\in}\Gamma$, we obtained precisely (b1), because $\mathsf{exsol}(\mathcal{S}(\Gamma \Rightarrow B_j))$ is equivalent to $\mathsf{ES}_\star(\mathcal{F}(\Gamma \Rightarrow B_j))$. (b2) also follows by definition of $(y : \vec{B} \supset p)\underline{\in}\Gamma$ and the equivalence just mentioned.

1. We want to prove: $\mathsf{mem}(N, \underline{\mathcal{S}}(\sigma))$ iff $\mathsf{mem}(N, \mathcal{S}(\sigma))$. Both implications will be proved by coinduction, so we give immediately the coinductive characterizations of the two members of the sought equivalence:

$$\frac{\mathsf{mem}(M, \underline{\mathcal{S}}(\Gamma, x : A \Rightarrow B)) \quad \underline{\mathcal{S}}(\Gamma, x : A \Rightarrow B) \neq \mathbb{O}}{\mathsf{mem}(\lambda x.M, \underline{\mathcal{S}}(\Gamma \Rightarrow A \supset B))}\ (\underline{a})$$

$$\frac{\exists(y : \vec{B} \supset p)\underline{\in}\Gamma\ \forall i,\ \mathsf{mem}(M_i, \underline{\mathcal{S}}(\Gamma \Rightarrow B_i))}{\mathsf{mem}(y\langle M_i\rangle_i, \underline{\mathcal{S}}(\Gamma \Rightarrow p))}\ (\underline{b})$$

$$\frac{\mathsf{mem}(M, \mathcal{S}(\Gamma, x : A \Rightarrow B))}{\mathsf{mem}(\lambda x.M, \mathcal{S}(\Gamma \Rightarrow A \supset B))}\ (a) \qquad \frac{\exists(y : \vec{B} \supset p) \in \Gamma\ \forall i,\ \mathsf{mem}(M_i, \mathcal{S}(\Gamma \Rightarrow B_i))}{\mathsf{mem}(y\langle M_i\rangle_i, \mathcal{S}(\Gamma \Rightarrow p))}\ (b)$$

First we prove $\mathsf{mem}(N, \underline{\mathcal{S}}(\sigma))$ is backward closed w. r. t. (a) and (b). As expected, this is immediate, and establishes the easy implication "only if". Next we prove that $\mathsf{mem}(N, \mathcal{S}(\sigma))$ is backward closed w. r. t. (a) and (b).

(a) Suppose $\mathsf{mem}(\lambda x.M, \mathcal{S}(\Gamma \Rightarrow A \supset B))$. We have $\mathsf{mem}(M, \mathcal{S}(\Gamma, x : A \Rightarrow B))$ by inversion of (a); additionally, this means $\Gamma, x : A \Rightarrow B$ has a solution, so part 2 (already proved) guarantees $\underline{\mathcal{S}}(\Gamma, x : A \Rightarrow B)$ has no empty sum, in particular $\underline{\mathcal{S}}(\Gamma, x : A \Rightarrow B) \neq \mathbb{O}$.

(b) Suppose $\mathsf{mem}(y\langle M_i\rangle_i, \mathcal{S}(\Gamma \Rightarrow p))$. We want

$$\exists(y : \vec{B} \supset p)\underline{\in}\Gamma\ \forall i,\ \mathsf{mem}(M_i, \mathcal{S}(\Gamma \Rightarrow B_i)) \qquad (*)$$

By inversion of (b) we obtain that there is $(y : \vec{B} \supset p) \in \Gamma$ such that $\mathsf{mem}(M_i, \mathcal{S}(\Gamma \Rightarrow B_i))$, for all i. To obtain $(*)$, it remains to see that, for all i, $\mathsf{ES}_\star(\mathcal{F}(\Gamma \Rightarrow B_i))$. But this is equivalent to $\mathsf{exsol}(\mathcal{S}(\Gamma \Rightarrow B_i))$, and the latter follows from $\mathsf{mem}(M_i, \mathcal{S}(\Gamma \Rightarrow B_i))$.

3. Suppose $\mathsf{nosol}(\mathcal{S}(\sigma))$. By part 1, $\mathsf{nosol}(\underline{\mathcal{S}}(\sigma))$. By Lemma 3.13, $\underline{\mathcal{S}}(\sigma)$ has an empty sum. Looking at the definition of $\underline{\mathcal{S}}$, such empty sum could live in $\underline{\mathcal{S}}(\Delta \Rightarrow B_j)$ of some summand of the outer sum, or be the outer sum itself. But the first option is impossible: given the criterion for a y to contribute a summand, we have $\mathsf{ES}_\star(\mathcal{F}(\Delta \Rightarrow B_j))$, which implies $\mathsf{exsol}(\mathcal{S}(\Delta \Rightarrow B_j))$, which guarantees $\underline{\mathcal{S}}(\Delta \Rightarrow B_j)$ has no empty sum, thanks to part 2. So, the outer sum is $\mathbb{O}$, hence $\underline{\mathcal{S}}(\sigma) = \mathbb{O}$. □

We close the exploration of concepts of finiteness with a discussion of König's lemma that we recall for the present situation.

Lemma 4.24. (König's lemma for forests)
$\mathsf{inf}(T)$ iff T has an infinite branch.

Proof:
Our definition of forests only allows finite branching (finitely many summands and finitely many arguments in the tuple, respectively), hence König's lemma applies. □

The following result is in the same spirit, extracting for sequents a consequence of Lemma 3.14, with the help of the pruned solution space and its properties.

Theorem 4.25. (König's lemma for simple types)
For all sequents σ, the pruned solution space of σ is infinite iff σ has an infinite solution.

Proof:
We want to prove: $\mathsf{inf}(\underline{\mathcal{S}}(\sigma))$ iff $\mathsf{exinf}(\mathcal{S}(\sigma))$. If σ has no solution (hence $\mathsf{exinf}(\mathcal{S}(\sigma))$ does not hold), then, by Lemma 4.23.3, $\underline{\mathcal{S}}(\sigma) = \mathbb{O}$, hence $\mathsf{fin}(\underline{\mathcal{S}}(\sigma))$. If σ has a solution, then $\underline{\mathcal{S}}(\sigma)$ has no empty sum, by Lemma 4.23.2. Then, by Lemma 3.14, $\mathsf{inf}(\underline{\mathcal{S}}(\sigma))$ iff $\mathsf{exinf}(\underline{\mathcal{S}}(\sigma))$. But $\mathsf{exinf}(\underline{\mathcal{S}}(\sigma))$ iff $\mathsf{exinf}(\mathcal{S}(\sigma))$, thanks to Lemma 4.23.1. □

Notice that the previous theorem would not be valid when formulated with the solution space $\mathcal{S}(\sigma)$ in place of the pruned solution space $\underline{\mathcal{S}}(\sigma)$. For instance, the sequent $\sigma_c = (x : p \supset q \supset p \Rightarrow p)$ considered in Example 2.2 has no solution, in particular no infinite one, but $\mathcal{S}(\sigma_c)$ is infinite. The pruned solution space $\underline{\mathcal{S}}(\sigma_c)$ is just $\mathbb{O}$, confirming Lemma 4.23.3 in this specific case.

5. Final remarks

In this paper we introduced several concepts of finiteness for simple types, involving not only finite objects (type inhabitants), but also infinite objects (type solutions and forests determined by types). One of these concepts corresponds to the question of whether a simple type has finitely many inhabitants, and is given through the predicate allfin already introduced in [5]. (In *op. cit.* we have done an

extensive literature review, and related our approach in particular to the study of this question done in [9].) The other three concepts of finiteness considered in this paper (given through the predicates fin, allfin and $\mathsf{fin}^{\mathsf{inffin}}$) are new. The following strict chain of inclusions holds, and determines weaker and weaker conditions for a simple type to be "finite":

$$\mathsf{fin} \subset \mathsf{allfin} \subset \mathsf{finfin} \subset \mathsf{fin}^{\mathsf{inffin}}$$

The main results of this paper are the realization that all these concepts of finiteness are instances of a single, parameterized concept of finiteness; and the proof of their decidability. Our study of finiteness is rounded up with a result in the spirit of König's lemma. Additionally, some other results were obtained that can be seen as an increment on our previous work [5]: the decidability of the problem "Does simple type A have no solution?" (i. e., not even an infinite one), and a generalization of a coherence theorem. But, while the latter is just an immediate corollary of the results obtained here, the former is actually instrumental for the development of the present paper.

One obvious question to ask is about possible uses of these concepts of finiteness. In this paper we characterized fin-finiteness of the pruned solution space of a simple type by allfin-finiteness of its solution space, identified allfin-finiteness as a weakly extensional predicate and related it to termination of proof search and to the previously studied strongly extensional finfin-finiteness. As mentioned before, we obtained decidability results for all these predicates, but pursued no uses of $\mathsf{fin}^{\mathsf{inffin}}$-finiteness, and it is not even clear in which sense this is still a concept of finiteness (other than another instance of our results). Another natural question is whether this chain of inclusions can be prolonged, even infinitely, and originate yet new finiteness concepts for simple types.

One of the conceptual tools that emerges from this work is the notion of pruned solution space of a sequent given by $\underline{\mathcal{S}}(\sigma)$. By chopping off elimination alternatives with certain occurrences of empty sums, $\underline{\mathcal{S}}(\sigma)$ may produce much smaller forests than $\mathcal{S}(\sigma)$, while still preserving the (full) extension of sequents. It would be natural to investigate an analogous pruning function for the finitary solution space, call it $\underline{\mathcal{F}}(\sigma)$. In particular, we would expect such a function to allow for more efficient decidability of the extensional predicates on forests addressed in this paper and in [5], such as a decision of $\mathsf{allfin}(\mathcal{S}(\sigma))$ via a decision of $\mathsf{F}^{\Pi_2}_{\emptyset}(\underline{\mathcal{F}}(\sigma))$.

Another line of research could be inspired by the very recent work on the analysis of inhabitation problems through the pre-grammar of a given type A [7]. We already mentioned after the presentation of Fact 4.4 in Section 4.1 that their work made us find a bound on the recursion depth of the finitary representation function. Based on this, we plan to describe variants of our decision algorithms that fall into PSPACE, as did [7] in their framework of pre-grammars for several classical problems related to inhabitation. However, the problem of principal inhabitation (shown to be PSPACE-complete in [10] and also covered by the method in [7]) does not seem to fit with our approach since we rely on closure under decontraction as justification for adequacy of our finitary representation function (which is not available for principal inhabitation, see the discussion in [10]). The decontraction phenomenon is also the root cause why the algorithms in their present form are not guaranteed to be in PSPACE, but this seems to be the price worth paying for a direct finitary representation of the whole search space, as a "first-class citizen" of a dedicated data structure.

References

[1] Ben-Yelles CB. Type assignment in the lambda-calculus: syntax & semantics. Ph.D. thesis, University of College of Swansea, 1979.

[2] Hirokawa S. Infiniteness of proof(α) is polynomial-space complete. *Theor. Comput. Sci.*, 1998. **206**(1-2):331–339. doi:10.1016/S0304-3975(97)00168-0.

[3] Espírito Santo J, Matthes R, Pinto L. A Coinductive Approach to Proof Search. In: Baelde D, Carayol A (eds.), Proc. of FICS 2013, volume 126 of *EPTCS*. 2013 pp. 28–43. doi:10.4204/EPTCS.126.3.

[4] Espírito Santo J, Matthes R, Pinto L. A Coinductive Approach to Proof Search through Typed Lambda-Calculi, 2016. Only published online at `http://arxiv.org/abs/1602.04382v2`.

[5] Espírito Santo J, Matthes R, Pinto L. Inhabitation in Simply-Typed Lambda-Calculus through a Lambda-Calculus for Proof Search. *Mathematical Structures in Computer Science*, 2018. pp. 1–33. doi:10.1017/S0960129518000099. First View - volume not yet known.

[6] Hindley JR. Basic Simple Type Theory, volume 42 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1997.

[7] Alves S, Broda S. A Unifying Framework for Type Inhabitation. In: Kirchner H (ed.), 3rd International Conference on Formal Structures for Computation and Deduction, FSCD 2018, July 9-12, 2018, Oxford, UK, volume 108 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik. ISBN 978-3-95977-077-4, 2018 pp. 5:1–5:16. doi:10.4230/LIPIcs.FSCD.2018.5.

[8] Broda S, Damas L. On Long Normal Inhabitants of a Type. *J. Log. Comput.*, 2005. **15**(3):353–390. doi:10.1093/logcom/exi016.

[9] Takahashi M, Akama Y, Hirokawa S. Normal Proofs and Their Grammar. *Inf. Comput.*, 1996. **125**(2):144–153. doi:10.1006/inco.1996.0027.

[10] Dudenhefner A, Rehof J. The Complexity of Principal Inhabitation. In: Miller D (ed.), 2nd International Conference on Formal Structures for Computation and Deduction, FSCD 2017, September 3-9, 2017, Oxford, UK, volume 84 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik. ISBN 978-3-95977-047-7, 2017 pp. 15:1–15:14. doi:10.4230/LIPIcs.FSCD.2017.15.