Cathcart Burn, T., Ong, L., Ramsay, S., & Wagner, D. (2021). Initial Limit Datalog: a new extensible class of decidable constrained Horn clauses. In *Proceedings of the 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2021)* Institute of Electrical and Electronics Engineers (IEEE). https://doi.org/10.1109/LICS52264.2021.9470527

Initial limit Datalog: a new extensible class of decidable constrained Horn clauses

***Abstract*—We present *initial limit Datalog*, a new extensible class of constrained Horn clauses for which the satisfiability problem is decidable. The class may be viewed as a generalisation to higher-order logic (with a simple restriction on types) of the first-order language *limit Datalog*$_\mathbb{Z}$ (a fragment of Datalog modulo linear integer arithmetic), but can be instantiated with any suitable background theory. For example, the fragment is decidable over any countable well-quasi-order with a decidable first-order theory, such as natural number vectors under componentwise linear arithmetic, and words of a bounded, context-free language ordered by the subword relation. Formulas of initial limit Datalog have the property that, under some assumptions on the background theory, their satisfiability can be witnessed by a new kind of term model which we call *entwined structures*. Whilst the set of all models is typically uncountable, the set of all entwined structures is recursively enumerable, and model checking is decidable.**

I. Introduction

Constrained Horn Clauses (CHCs) are a class of formulas that have been found to be especially suitable for tasks in automated reasoning. They are the language of constrained logic programming [1]. More recently, there has been a concerted effort to exploit the class as a programming-language independent basis for automatic program verification [2, 3].

CHCs are a liberalisation of the class of Horn formulas in which, additionally, clauses may contain *constraints* drawn from a specified first-order *background theory*[1]. This extension preserves many of the good properties of the Horn format, such as the existence of canonical models and the sufficiency of SLD-style derivations, whilst allowing for the expression of domain-specific knowledge in the form of assertions from the background theory.

Unfortunately, this pleasing combination of expressivity and semantic characterisation comes with an algorithmic cost. In general, decidability of the satisfiability problem for a class of CHC depends on the choice of background theory, and for many theories that are typical in automated reasoning (e.g. because they are decidable), the class of CHC is undecidable. For example, [4] shows that not only is CHC over linear integer arithmetic undecidable [5], but so too CHC over complex, real or rational linear arithmetic. On the other hand, it is easy to see that CHC over the theory of equality on a finite set has decidable satisfiability.

Since the most promising applications concern theories of infinite structures, it becomes important to identify restrictions on the format that both preserve its essential character and yet guarantee decidability. In [4], a catalogue of (sub-recursive) complexity results are derived concerning limitations placed on the use of variables within clauses and the nature of parameter passing.

An alternative approach, and the starting point for the work in this paper, is the *limit* restriction of the language *limit Datalog*$_\mathbb{Z}$, which was proposed in [6] as a foundation for declarative data analysis. Limit Datalog$_\mathbb{Z}$ can be viewed as a language of first-order CHCs over the theory of linear integer arithmetic, but with the following proviso: predicates in limit Datalog$_\mathbb{Z}$ (called *limit predicates*) are restricted so as to capture only the minimum (or maximum) numeric values in their unique integer parameter. This restriction ensures that the satisfiability problem is decidable for this class of CHC, whilst remaining expressive enough to describe important problems in data analysis (in particular, one may still describe certain kinds of recursively defined predicates over the integers).

One way to implement the limit predicate restriction is to require that all predicates with an integer parameter are either upwards (for maximum) or downwards (for minimum) closed with respect to that parameter. We enforce this by the highlighted clauses in the examples below, taken from [6].

The background theory of these examples is the combination of linear integer arithmetic and the theory of equality over a finite set. We assume that the elements of this set can be arranged into a linear order $y_1, y_2, \ldots, y_k$ (which will differ from example to example), described by two constraint formulas (i.e. of the background theory) that we will abbreviate $\mathsf{FIRST}(y_1)$ and $\mathsf{NEXT}(y_n, y_{n+1})$.

Example I.1 (Social networking)**.** In this first example, the finite set describes people who tweet and follow each other's tweets. Let us suppose we have a constraint formula[2] (i.e. of the background theory) abbreviated by $\mathsf{TH}(x, m)$, indicating the *retweet threshold*. That is, asserting that a person x will tweet a (hypothetical) message if at least m of those they follow also tweet it. Suppose we have a constraint formula $\mathsf{FOLLOWS}(x, y)$, describing when one individual x follows the tweets of another y. The following clauses constrain a proposition $\mathrm{Tw}\,x$ so that it holds if x tweeted. The proposition $\mathrm{Nt}\,x\,y\,m$ holds if, out of the people at or before y (according to the ordering on the set of people), at least m people that x follows tweeted.

[1]Note: in this work we will assume the background theory has a fixed interpretation, as is common in the satisfiability-modulo-theories literature.

[2]One can also think more specifically of an intensional database predicate.

$$\mathrm{Nt}\, x\, y\, 0 \leftarrow \mathsf{FIRST}(y)$$
$$\mathrm{Nt}\, x\, y\, 1 \leftarrow \mathsf{FOLLOWS}(x,y) \wedge \mathsf{FIRST}(y) \wedge \mathrm{Tw}\, y$$
$$\mathrm{Nt}\, x\, y\, m \leftarrow \mathrm{Nt}\, x\, y'\, m \wedge \mathsf{NEXT}(y',y)$$
$$\mathrm{Nt}\, x\, y\, (m+1) \leftarrow \mathrm{Nt}\, x\, y'\, m \wedge \mathsf{FOLLOWS}(x,y) \wedge \mathsf{NEXT}(y',y) \wedge \mathrm{Tw}\, y$$
$$\mathrm{Nt}\ x\ y\ m \leftarrow m \leq n \wedge \mathrm{Nt}\ x\ y\ n$$
$$\mathrm{Tw}\, x \leftarrow \mathsf{TH}(x,m) \wedge \mathrm{Nt}\, x\, y\, n \wedge m \leq n$$

Example I.2 (Path counting)**.** In this second example, the finite set describes the vertices of a directed acyclic graph and the clauses can be used to reason about the number of paths between two nodes. We assume a constraint formula $\mathsf{EDGE}(x,y)$ indicating that there is an edge from x to y.

$$\mathrm{Np}'\, x\, y\, z\, 0 \leftarrow \mathsf{FIRST}(z)$$
$$\mathrm{Np}'\, x\, y\, z\, m \leftarrow \mathsf{FIRST}(z) \wedge \mathrm{Np}\, z\, y\, m \wedge \mathsf{EDGE}(x,z)$$
$$\mathrm{Np}'\, x\, y\, z\, m \leftarrow \mathsf{NEXT}(z',z) \wedge \mathrm{Np}'\, x\, y\, z'\, m$$
$$\mathrm{Np}'\, x\, y\, z\, (m+n) \leftarrow \mathsf{NEXT}(z',z) \wedge \mathrm{Np}'\, x\, y\, z'\, m \wedge \mathsf{EDGE}(x,z) \wedge \mathrm{Np}\, z\, y\, n$$
$$\mathrm{Np}'\, x\, y\, z\, m \leftarrow m \leq n \wedge \mathrm{Np}'\, x\, y\, z\, n$$
$$\mathrm{Np}\, x\, y\, m \leftarrow \mathrm{Np}'\, x\, y\, z\, m$$
$$\mathrm{Np}\, x\, x\, 1 \leftarrow \mathsf{true}$$
$$\mathrm{Np}\, x\, y\, m \leftarrow m \leq n \wedge \mathrm{Np}\, x\, y\, n$$

Here, $\mathrm{Np}'\, x\, y\, z\, m$ holds if there are at least m paths of the form $x, w, \cdots, y$ where w occurs at or before z according to the linear ordering of nodes. Finally, $\mathrm{Np}\, x\, y\, n$ holds if there are at least n paths from x to y.

Contributions

In this paper, we introduce a significant yet decidable extension of limit Datalog$_{\mathbf{Z}}$ which we call *initial limit Datalog*. Our language encompasses generalisations of the original work [6] along two dimensions and we define a new class of models:

(i) Parametrisation with respect to a wide range of background theories. We give a number of abstract conditions on the character of the background theory which, if satisfied, guarantee decidability of the language (Thm. III.4). Instances of particular note include all countable well-quasi orders (WQOs) with a decidable first-order theory. This contains, for example, the theory of tuples of naturals under componentwise ordering, allowing the use of predicates with more than one natural number argument.

(ii) (Un)decidability at higher type. We show that the most natural extension of limit Datalog$_{\mathbf{Z}}$ to higher-order logic, in which clauses can define predicates of arbitrary higher type, already has undecidable satisfiability (Thm. III.7). Through a careful analysis of the interaction between the typing discipline and model construction, we design a restriction on the types of predicates (automatically satisfied by all first-order predicates) that we call *initial*. We show that the resulting language, *initial limit Datalog*, regains decidable satisfiability (Thm. IV.3).

(iii) A recursively enumerable set of candidate models. The solution space for a given set of clauses is typically uncountable, because predicates are interpreted as subsets of the domain. A key step in proving our decidability results is to show that, remarkably, one can restrict attention to a recursively enumerable class of candidate models. To handle the higher-order case, we introduce a new representation which we call *entwined structures*, in which the interpretation of a higher *type* may depend on the interpretation of particular *terms* of lower types. They have many useful properties, and their conception is sufficiently general that we believe they may be of use for obtaining similar results beyond the scope of this paper.

Initial limit Datalog

The setting for our language is the fragment of higher-order logic known as *higher-order constrained Horn clauses* (HoCHC) [3, 7]. *Higher-order* constrained Horn clauses allow for the description of predicates of higher-types (i.e., whose subjects may themselves be predicates). Such predicates can be described by clauses built from terms of the simply typed λ-calculus when equipped with the appropriate logical constants. As in [3, 7], we forgo the use of explicit abstraction to simplify the Horn clause format.

As a first example, we demonstrate in Ex. I.3 and Ex. I.4 how the first-order limit Datalog$_{\mathbf{Z}}$ examples above share a common structure which can be factored out into a higher-order recursion combinator Iter of the following type:

$$\mathrm{Iter} : S \to \mathbb{Z} \to (S \to \mathbb{Z} \to o) \to o$$

Throughout (the examples of) this paper, we will use S to denote the type of a fixed finite set, o as the type of propositions and, by some abuse, $\mathbb{Z}$ as the type of the integers. This combinator can be defined as follows:

$$\mathrm{Iter}\, y\, n\, p \leftarrow \mathsf{FIRST}(y) \wedge p\, y\, n$$
$$\mathrm{Iter}\, y\, n\, p \leftarrow \mathsf{NEXT}(y',y) \wedge p\, y\, k \wedge \mathrm{Iter}\, y'\, m\, p \wedge n = k + m$$
$$\mathrm{Iter}\, y\, n\, p \leftarrow n \leq m \wedge \mathrm{Iter}\, y\, m\, p$$

The proposition $\mathrm{Iter}\, y\, n\, p$ describes iteration over a generic sequence of data items in S from the first item until item y, evaluating the predicate $p : S \to \mathbb{Z} \to o$ on each item and summing the associated integers to n. As in the first-order case, we must implement the limit predicate restriction, so we include the shaded clause to guarantee the (in this case) downwards closure of its integer argument.

Example I.3 (Refactoring social networking)**.** Using Iter, the whole of the social network example Ex. I.1, in which the data items are users, can be encoded more concisely as:

$$\mathrm{Inc?}\, x\, y\, n \leftarrow n = 0 \vee \big(\mathsf{FOLLOWS}(x,y) \wedge \mathrm{Tw}\, y \wedge n = 1\big)$$
$$\mathrm{Inc?}\, x\, y\, n \leftarrow n \leq m \wedge \mathrm{Inc?}\, x\, y\, m$$
$$\mathrm{Tw}\, x \leftarrow \mathsf{TH}(x,m) \wedge \mathrm{Iter}\, y\, n\, (\mathrm{Inc?}\, x) \wedge m \leq n$$

The predicate Inc?, which satisfies the limit restriction, expresses the domain specific reasoning that happens on each iteration, namely that the number of tweeters will either be increased by 0, or by 1 in case x follows some y who tweets the message.

Example I.4 (Refactoring path counting)**.** The path counting example Ex. I.2 uses a similar iterative structure. The whole example can be rewritten as:

$$\begin{aligned} \mathrm{NpExt}\, x\, y\, z\, m &\leftarrow \big(\mathrm{Np}\, z\, y\, m \land \mathsf{EDGE}(x, z)\big) \lor m = 0 \\ \mathrm{NpExt}\, x\, y\, z\, m &\leftarrow m \leq n \land \mathrm{NpExt}\, x\, y\, z\, n \\ \mathrm{Np}\, x\, y\, m &\leftarrow \mathrm{Iter}\, z\, m\, (\mathrm{NpExt}\, x\, y) \lor (x = y \land m = 1) \\ \mathrm{Np}\, x\, y\, m &\leftarrow m \leq n \land \mathrm{Np}\, x\, y\, n \end{aligned}$$

In this case, the second and fourth clauses ensure that the respective predicates adhere to the limit restriction.

Example I.5 (Generic query)**.** An orthogonal benefit of higher-type predicates is to allow the expression of higher-order properties (e.g. properties of the form *for all relations* r...). Returning to Ex. I.1, the follows relation was fixed by some first-order constraint formula (or intensional database predicate) $\mathsf{FOLLOWS}(x, y)$. Using predicates of higher type, we can instead parametrise the mutually recursive predicates Nt and Tw by an *arbitrary* follows relation f of type $S \to S \to o$:

$$\begin{aligned} \mathrm{Nt}\, x\, y\, 0\, f &\leftarrow \mathsf{FIRST}(y) \\ \mathrm{Nt}\, x\, y\, 1\, f &\leftarrow f\, x\, y \land \mathsf{FIRST}(y) \land \mathrm{Tw}\, y\, f \\ \mathrm{Nt}\, x\, y\, m\, f &\leftarrow \mathrm{Nt}\, x\, y'\, m \land \mathsf{NEXT}(y', y) \\ \mathrm{Nt}\, x\, y\, (m+1)\, f &\leftarrow \mathrm{Nt}\, x\, y'\, m \land f\, x\, y \\ &\qquad \land \mathsf{NEXT}(y', y) \land \mathrm{Tw}\, y \\ \mathrm{Nt}\, x\, y\, m\, f &\leftarrow m \leq n \land \mathrm{Nt}\, x\, y\, n\, f \\ \mathrm{Tw}\, x\, f &\leftarrow \mathsf{TH}(x, m) \land \mathrm{Nt}\, x\, y\, n\, f \land m \leq n \end{aligned}$$

This allows us to check that a property of the system holds *independently* of who follows whom. For example, according to Kaminski et al.'s formulation, nobody will tweet the message if we fix all retweet thresholds at 1. To verify this, we set $\mathsf{TH}(x, m)$ to the constraint formula $m = 1$ and decide satisfiability of the clauses extended with the following goal:

$$\mathit{false} \leftarrow \mathrm{Tw}\, x\, f$$

From the satisfiability of the clauses, we can deduce that there does not exist a choice of an individual x and a followers relation f for which the message would be tweeted.

Examples I.3 to I.5 are not limit $\mathrm{Datalog}_{\mathbf{Z}}$ problems, but they are problems of our generalisation: *initial limit Datalog*. As well as admitting the definition of higher-order relations, in place of the theory of integer linear arithmetic we allow for the theory of any preordered set W satisfying certain conditions.

Initial limit Datalog problem and satisfiability: Henceforth let W be a preordered set with a decidable first-order theory, such that every upwards closed subset of W is definable in the theory. We consider relational types generated from W and any finite set S (abusing notation by naming the types after their interpretations).

An ***initial limit Datalog problem*** is a (finite) set Γ of HoCHC clauses over W and S such that for every predicate $X : \rho$ in the signature, with $\rho = \sigma_1 \to \cdots \to \sigma_k \to o$ of order n (say):

(i) ρ is ***initial***, meaning $\sigma_j = W$ for at most one j, and if there is such a j then for all $i < j$, order$(\sigma_i) <$ order$(\sigma_j \to \cdots \to \sigma_k \to o)$; moreover each σ_i is S, or W, or initial.

(ii) if $\sigma_j = W$ for some j, then Γ contains the ***limit clause***

$$X\, \overline{z}\, x\, \overline{z'} \;\leftarrow\; x \leq y \land X\, \overline{z}\, y\, \overline{z'}$$

(and ρ is called an ***active type***).

The ***satisfiability problem*** for *initial limit Datalog* asks: given an initial limit Datalog problem Γ, is it satisfiable (modulo the theory of W and S)?

We show in Sec. III-B that a naïve extension to higher order leads to undecidability, but the forgoing examples and those we will present in the sequel all obey a certain discipline in the way that the background type W and higher types interact. This is captured by the *initial* restriction, (i), which requires that the types of terms that may be captured by a partial application are of strictly lower order than the partial application itself. It is easy to verify that this condition holds for the type of Iter and one can also see it in the types of our higher-order generalisation of Nt and (trivially) Tw:

$$\begin{aligned} \mathrm{Tw} &: S \to (S \to S \to o) \to o \\ \mathrm{Nt} &: S \to S \to \mathbb{Z} \to (S \to S \to o) \to o \end{aligned}$$

Note that all formulas of limit $\mathrm{Datalog}_{\mathbf{Z}}$ already satisfy requirements (i) and (ii); and $\mathbb{Z}$, under the theory of linear integer arithmetic, is an appropriate instantiation of W.

Parametrisation of initial limit Datalog by the type W allows for a variety of interesting background structures beyond integer linear arithmetic. For example, any countable well-quasi-ordering with a decidable background theory (which must include constants for each element of the structure) satisfies the requirements on W, such as:

a. Tuples of natural numbers, under componentwise ordering with the theory of linear arithmetic on components.

b. Words of a bounded, context-free language, under the subword order [8].

c. Basic process algebra under the subword order. BPA is an automatic structure, and so, has a decidable first-order theory. There are other examples in the same vein, e.g., communicating finite-state machines [9].

The following example is a higher-order instance of initial limit datalog where the preorder W is the WQO of tuples of natural numbers, with the theory of linear arithmetic on components. Notice that in this case, there may be multiple consecutive parameters of type $\mathbb{N}$ in a predicate.

Example I.6 (Integration)**.** Monotone decreasing functions $\mathbb{N} \to \mathbb{N}$ can be represented by downwards closed subsets of $\mathbb{N} \times \mathbb{N}$: such a function f is uniquely identified by $\{(x, y) : y < f(x)\}$. Higher-order initial limit Datalog allows us to define a predicate which computes integrals[3] over such functions.

$$\mathrm{Integral} : \mathbb{N} \to \mathbb{N} \to (\mathbb{N} \to \mathbb{N} \to o) \to o$$

$$\mathrm{Integral}\, \mathit{tot}\, \mathit{bd}\, f \leftarrow \mathit{tot} = 0$$

$$\mathrm{Integral}\, \mathit{tot}\, \mathit{bd}\, f \leftarrow \mathit{tot} = x + y + 1 \wedge \mathrm{Integral}\, x\, (\mathit{bd} + 1)\, f \wedge f\, \mathit{bd}\, y$$

$$\mathrm{Integral}\, \mathit{tot}\, \mathit{bd}\, f \leftarrow \mathit{tot} \leq s \wedge \mathit{bd} \leq c \wedge \mathrm{Integral}\, s\, c\, f$$

$$\mathrm{Exp} : \mathbb{N} \to \mathbb{N} \to o$$

$$\mathrm{Exp}\, m\, n \leftarrow m = 0 \wedge n < 128$$

$$\mathrm{Exp}\, m\, n \leftarrow \mathrm{Exp}\, x\, y \wedge m = x - 1 \wedge n + n < y$$

$$\mathrm{Exp}\, m\, n \leftarrow m \leq x \wedge n \leq y \wedge \mathrm{Exp}\, x\, y$$

$$\mathit{false} \leftarrow \mathrm{Integral}\, 255\, 0\, \mathrm{Exp}$$

In the canonical interpretation, Exp represents the function defined by $f(m) = \lfloor 2^{7-m} \rfloor$ and $\mathrm{Integral}\, \mathit{tot}\, \mathit{bd}\, f$ is true if tot is less than or equal to the integral (infinite sum) of the monotone function represented by f from bd to ∞. (Thus $\max\{\mathit{tot} \mid \text{`}\mathrm{Integral}\, \mathit{tot}\, 0\, \mathrm{Exp}\text{' holds}\} = 255$.)

This example is unsatisfiable (there is no consistent interpretation of $\mathrm{Integral}$ and Exp where $\mathrm{Integral}\, 255\, 0\, \mathrm{Exp}$ is false), but if the constant 255 is changed to 256, it becomes satisfiable.

Entwined structures

The key innovation of our decidability proof is the construction (given Γ) of a set of candidate models, called *entwined structures*, which satisfy a number of pleasing properties:

(P1) The set of entwined structures is r.e.
(P2) In each order-n entwined structure, the denotation of each (initial) relational type (that occurs in Γ) of order less than n is finite.
(P3) There is an algorithm that checks if a given entwined structure models Γ.
(P4) There is an entwined structure that models Γ if and only if Γ is satisfiable.

Entwined structures are built up by induction on order, via a bootstrapping process. Their name reflects the interplay between the interpretation of terms and types during this process: the interpretation of a type of order-n (the set from which the interpretations of order-n predicate symbols are chosen) can only be given once the interpretation of the relevant predicate symbols of lower-order types has already been fixed. A family of structures $\{\mathcal{B}_n\}_{n \in \omega}$, indexed by (order) n, is *entwined*, if $\mathcal{B}_0$ is the structure on the empty signature; and in each $\mathcal{B}_n$:

- Predicate symbols in $\mathcal{B}_{n-1}$ (those of the foreground signature of order $< n$) are interpreted as per $\mathcal{B}_{n-1}$.
- Each predicate of an order-n active type $\rho = \sigma_1 \to \cdots \to \sigma_k \to o$ is interpreted as a function (in the set-theoretic $[\mathcal{B}_n[\![\sigma_1]\!] \to \cdots \to \mathcal{B}_n[\![\sigma_k]\!] \to \mathbb{B}]$) monotone in the W-typed argument.

For types $\rho = \tau \to \sigma$ of order less than n, $\mathcal{B}_n[\![\rho]\!]$ is the full function space $[\mathcal{B}_n[\![\tau]\!] \to \mathcal{B}_n[\![\sigma]\!]]$ if that is finite, otherwise it is the least collection of relational functions allowing it to support the interpretations of predicates assigned by $\mathcal{B}_{n-1}$. This results in something similar to a term model. We cannot use the term model because there can be infinitely many terms and therefore uncountably many interpretations of higher-order predicates, but our decidability proof rests on enumeration.

In an unrestricted setting, it would not make sense to interpret all the order-$(n-1)$ active predicates (i.e. predicates of active type) before interpreting the order-n predicates, because an order-$(n-1)$ active predicate may be passed an argument involving a predicate of order-n.

However, thanks to the initial type restriction, if an order-n term N of an active type has an order-m subterm M with $m > n$, then M is a subterm of some L (another subterm of N) of type σ (say) whose order is less than n. Since $\mathcal{B}_n[\![\sigma]\!]$ is finite (P2), we don't need to know all possible values of M to know all possible values of N.

We show decidability (Thm. IV.3) by exhibiting two semi-decision procedures—one for proving the existence of a model, and the other for non-existence—and running them in parallel. The former semi-decision procedure is an immediate consequence of (P1), (P3) and (P4). The latter is an application of the semi-decidability of HoCHC unsatisfiability, via a refutationally complete resolution proof system (Γ is unsatisfiable if, and only if, there is a resolution proof of $\perp$ from Γ) [7].

Outline: We begin with some technical preliminaries in Sec. II before introducing (higher-order) limit Datalog in Sec. III. We give a proof that the first-order fragment has a decidable satisfiability problem and show that satisfiability in general is undecidable. In Sec. IV we present initial restriction on types, and prove that the initial limit Datalog satisfiability problem is decidable. In Sec. V, we give examples of how first-order limit Datalog problems can be used with the background theory of tuples of naturals, and other well-quasi orderings (WQOs) with a decidable first-order theory. After a review of related work (Sec. VI), we conclude and briefly discuss some further directions.

[3]Integral can equivalently be typed as $\mathbb{N} \times \mathbb{N} \to (\mathbb{N} \times \mathbb{N} \to o) \to o$.

II. Technical preliminaries

This section introduces a restricted form of higher-order logic (Sec. II-A), higher-order constrained Horn clauses (HoCHCs) (Sec. II-B) and their proof system (Sec. II-C).

A. Relational higher-order logic

1) Syntax: For a fixed set $\mathfrak{I}$ (intuitively the types of individuals), the set of *argument types*, *relational types*, *1st-order types* and *types* (generated by $\mathfrak{I}$) are defined by mutual recursion as follows

$$\begin{array}{llcl} \textit{Argument type} & \tau & ::= & \iota \mid \rho \\ \textit{Relational type} & \rho & ::= & o \mid \tau \to \rho \\ \textit{1st-order type} & \sigma_{\mathrm{FO}} & ::= & \iota \mid o \mid \iota \to \sigma_{\mathrm{FO}} \\ \textit{Type} & \sigma & ::= & \rho \mid \sigma_{\mathrm{FO}}, \end{array}$$

where $\iota \in \mathfrak{I}$. We sometimes abbreviate function types $\tau_1 \to \cdots \to \tau_n \to \sigma$ to $\overline{\tau} \to \sigma$. Intuitively, o (where $o \notin \mathfrak{I}$) is the type of the truth values (or Booleans). The types σ_{FO} are exactly those of the form $\overline{\iota} \to \iota$ or $\overline{\iota} \to o$, i.e. each argument is of some type $\iota_i \in \mathfrak{I}$. Moreover, each relational type has the form $\overline{\tau} \to o$.

A *type environment* (typically Δ) is a function mapping *variables* (typically x, y, z) to argument types; for $x \in \mathrm{dom}(\Delta)$, we write $x : \tau \in \Delta$ to mean $\Delta(x) = \tau$. A *signature* (typically Σ, Ξ) is a set of distinct typed *symbols* $c : \sigma$, where $c \notin \mathrm{dom}(\Delta)$. A signature Σ is *1st-order* if σ is 1st-order for all $c : \sigma \in \Sigma$. We often write $c \in \Sigma$ if $c : \sigma \in \Sigma$ for some σ.

The set of Σ*-pre-terms* is given by $M ::= x \mid c \mid M\,M$ where $c \in \Sigma$. We assume that application associates to the left, and write $M\,\overline{N}$ for $M\,N_1 \cdots N_n$, assuming implicitly that M is not an application.

The typing judgement $\Delta \vdash M : \sigma$ is defined by

$$\frac{x \in \mathrm{dom}(\Delta)}{\Delta \vdash x : \Delta(x)} \quad \frac{c : \sigma \in \Sigma}{\Delta \vdash c : \sigma} \quad \frac{\Delta \vdash M_1 : \sigma_1 \to \sigma_2 \quad \Delta \vdash M_2 : \sigma_1}{\Delta \vdash M_1 M_2 : \sigma_2}$$

We say that M is a Σ*-term of type* σ if $\Delta \vdash M : \sigma$. A Σ-term is a *1st-order* Σ*-term* if the symbols in its construction are restricted to symbols $c : \sigma_{\mathrm{FO}} \in \Sigma$ and variables $x : \iota \in \Delta$.

Remark II.1. It follows from the definitions that each term $\Delta \vdash M : \overline{\iota} \to \iota$ can only contain variables of type ι_i and constants of non-relational 1st-order type (and contains no logical symbols, a similar approach is adopted in [10]).

We define a Σ*-formula* F by

$$F ::= M \mid F \vee F \mid F \wedge F \mid \neg F$$

where M is any Σ-term of type o. For a Σ-term or Σ-formula M and Σ-terms $N_1, \ldots, N_n$ and variables $x_1, \ldots, x_n$ that satisfy $\Delta \vdash N_i : \Delta(x_i)$, the *substitution* $M[N_1/x_1, \ldots, N_n/x_n]$ is defined in the standard way.

2) Semantics: There are two classic semantics for higher-order logic: *standard* and *Henkin semantics* [11]. In this paper, we will not be concerned with the latter, but the notion of frame is useful. Assume, for each $\iota \in \mathfrak{I}$, an associated set D_ι. Formally, a ***frame*** $\mathcal{F}$ assigns to each type σ a nonempty set $\mathcal{F}[\![\sigma]\!]$ such that

(i) $\mathcal{F}[\![\iota]\!] := D_\iota$ for each $\iota \in \mathfrak{I}$
(ii) $\mathcal{F}[\![o]\!] := \mathbb{B} := \{0, 1\}$
(iii) For each $\sigma_1 \to \sigma_2$, $\mathcal{F}[\![\sigma_1 \to \sigma_2]\!] \subseteq [\mathcal{F}[\![\sigma_1]\!] \to \mathcal{F}[\![\sigma_2]\!]]$

where $[U \to V]$ is the set of functions from (sets) U to V.

Remark II.2. Unlike [7], we do not distinguish pre-frame and frame. Because λ-abstractions are not part of the HoCHC syntax here, the (weak) comprehension axiom in [7, p. 3] does not apply.

Example II.3 (Standard frame)**.** We define the *standard frame* $\mathcal{S}$ recursively by $\mathcal{S}[\![o]\!] := \mathbb{B}$; $\mathcal{S}[\![\iota]\!] := D_\iota$ for $\iota \in \mathfrak{I}$; and

$$\mathcal{S}[\![\tau \to \sigma]\!] := [\mathcal{S}[\![\tau]\!] \to \mathcal{S}[\![\sigma]\!]]$$

Let Σ be a signature, and $\mathcal{F}$ be a frame. A $(\Sigma, \mathcal{F})$***-structure*** $\mathcal{A}$ assigns to each $c : \sigma \in \Sigma$ an element $c^{\mathcal{A}} \in \mathcal{F}[\![\sigma]\!]$ and for convenience we set $\mathcal{A}[\![\sigma]\!] := \mathcal{F}[\![\sigma]\!]$ for types σ. A $(\Delta, \mathcal{F})$-***valuation*** α is a function such that for every $x : \tau \in \Delta$, $\alpha(x) \in \mathcal{F}[\![\tau]\!]$. For a $(\Delta, \mathcal{F})$-valuation α, variable x and $r \in \mathcal{F}[\![\Delta(x)]\!]$, $\alpha[x \mapsto r]$ is defined in the usual way.

Let $\mathcal{A}$ be a $(\Sigma, \mathcal{F})$-structure and let α be a $(\Delta, \mathcal{F})$-valuation. The *denotation* $\mathcal{A}[\![M]\!](\alpha)$ of a Σ-term M with respect to $\mathcal{A}$ and α is defined recursively by

$$\mathcal{A}[\![x]\!](\alpha) := \alpha(x) \quad \mathcal{A}[\![c]\!](\alpha) := c^{\mathcal{A}}$$
$$\mathcal{A}[\![M_1\, M_2]\!](\alpha) := \mathcal{A}[\![M_1]\!](\alpha)\big(\mathcal{A}[\![M_2]\!](\alpha)\big)$$

For each term $\Delta \vdash M : \sigma$, we have $\mathcal{A}[\![M]\!](\alpha) \in \mathcal{A}[\![\sigma]\!]$. (We will write $\mathcal{A}[\![M]\!](\alpha)$ as $\mathcal{A}^{\Sigma}[\![M]\!](\alpha)$ when we need to be explicit about the signature of the Σ-terms M.)

Example II.4 (LIA)**.** In this paper, many examples will use the signature of ***linear integer arithmetic***[4] (LIA) (aka Presburger arithmetic) $\Sigma_{\mathrm{LIA}} := \{0, 1, +, -, <, \leq, =, \neq, \geq, >\}$ and its standard model $\mathcal{A}_{\mathrm{LIA}}$.

B. Higher-order constrained Horn clauses (HoCHC)

We explicitly distinguish symbols of the *background* (bg) theory from those—in the *foreground* (fg)—which are constrained by clauses. This distinction enables a certain semantic separation required by a model construction (Def. IV.5), which is crucial to our decidability result (Thm. IV.3). [5]

Assumption 1. *Henceforth we fix a 1st-order signature* Σ_{bg}*, and a* $(\Sigma_{\mathrm{bg}}, \mathcal{S})$*-structure* $\mathcal{A}$*, and a finite signature* Σ_{fg} disjoint

[4] with the usual types $0, 1 : \iota$; $+, - : \iota \to \iota \to \iota$ and $\lhd : \iota \to \iota \to o$ for $\lhd \in \{<, \leq, =, \neq, \geq, >\}$; and we use the common abbreviation n for $\underbrace{1 + \cdots + 1}_{n}$, where $1 \leq n \in \mathbb{N}$

[5] Using notations in Def. IV.5 and Lem. IV.9, take $\leq\, \in \Sigma_{\mathrm{bg}}$. If $\leq^{\mathcal{A}} \in \mathcal{F}[\![\mathbb{Z} \to \mathbb{Z} \to o]\!]$ then $\mathcal{F}[\![\mathbb{Z} \to o]\!]$ must be infinite, contradicting Lem. IV.9.

from Σ_{bg} with only predicate symbols (of a relational type), typically X, Y, P and R and their variants. We will write such a pair of signatures as $\overline{\Sigma} = (\Sigma_{\mathrm{bg}}, \Sigma_{\mathrm{fg}})$.

Intuitively, Σ_{bg} and $\mathcal{A}$ correspond to the language and interpretation of the background theory, e.g. Σ_{LIA} together with its standard model $\mathcal{A}_{\mathrm{LIA}}$. In particular, *we (only) consider background theories with a single model*.

Next, we introduce higher-order constrained Horn clauses and their satisfiability problem [3].

Definition II.5. By *atom*, we mean background atom or foreground atom.

(i) A *background atom* is a 1st-order Σ_{bg}-term of type o.
(ii) A *foreground atom* is a Σ_{fg}-term of type o.

Note that a foreground atom has one of the following forms: (i) $R\,\overline{M}$ where $R \in \Sigma_{\mathrm{fg}}$, or (ii) $x\,\overline{M}$.

We use φ and A (and variants thereof) to refer to background atoms and (general) atoms, respectively.

Definition II.6 (HoCHC)**.** (i) A ***goal clause*** (typically G) is a disjunction $\neg A_1 \vee \cdots \vee \neg A_n$, where each A_i is an atom. We write $\bot$ to mean the empty (goal) clause.
(ii) If G is a goal clause, $R \in \Sigma_{\mathrm{fg}}$ and the variables in $\overline{x}$ are distinct, then $G \vee R\,\overline{x}$ is a ***definite clause***.
(iii) A ***higher-order constrained Horn clause (HoCHC)*** is a goal or definite clause.

Throughout the document, we will often write a clause $\neg A_1 \vee \cdots \vee \neg A_n \vee R\,\overline{x}$ as $R\,\overline{x} \leftarrow A_1 \wedge \cdots \wedge A_n$.

Next we give an example of HoCHCs from Sec. I, explicitly listing the types involved and illustrating the structures.

Example II.7 (A system of HoCHCs)**.** Let $\Sigma_{\mathrm{bg}} = \Sigma_{\mathrm{LIA}} \cup \{=_S\}$ and $\Sigma_{\mathrm{fg}} = \{\mathrm{Iter} : S \to \mathbb{Z} \to (S \to \mathbb{Z} \to o) \to o, \mathrm{Inc?} : S \to S \to \mathbb{Z} \to o, \mathrm{Tw} : S \to o\}$ and let Δ be a type environment satisfying $\Delta(m) = \Delta(n) = \Delta(k) = \mathbb{Z}$ and $\Delta(x) = \Delta(y) = \Delta(y') = S$ and $\Delta(p) = S \to \mathbb{Z} \to o$. The system consists of the HoCHCs in Ex. I.3, and the preceding three that define Iter.

A $\overline{\Sigma}$*-formula* is a formula where each term is either a Σ_{fg}-term or a 1st-order Σ_{bg}-term. Let $\mathcal{F}$ be a frame that agrees with the standard frame $\mathcal{S}$ on the base types $\mathfrak{I}$. Let $\mathcal{B}$ be a $(\Sigma_{\mathrm{fg}}, \mathcal{F})$-structure and let α be a $(\Delta, \mathcal{F})$-valuation. The definition of the *denotation* $\mathcal{B}[\![F]\!](\alpha)$ of a $\overline{\Sigma}$-formula F with respect to $\mathcal{B}$ and α is defined recursively by

$$\mathcal{B}[\![M]\!](\alpha) := \begin{cases} \mathcal{A}^{\Sigma_{\mathrm{bg}}}[\![M]\!](\alpha_1) & \text{if } M \text{ a 1st-order } \Sigma_{\mathrm{bg}}\text{-term} \\ \mathcal{B}^{\Sigma_{\mathrm{fg}}}[\![M]\!](\alpha) & \text{if } M \text{ a } \Sigma_{\mathrm{fg}}\text{-term} \end{cases}$$

$$\mathcal{B}[\![F \wedge G]\!](\alpha) := \min(\mathcal{B}[\![F]\!](\alpha), \mathcal{B}[\![G]\!](\alpha))$$

$$\mathcal{B}[\![F \vee G]\!](\alpha) := \max(\mathcal{B}[\![F]\!](\alpha), \mathcal{B}[\![G]\!](\alpha))$$

$$\mathcal{B}[\![\neg F]\!](\alpha) := 1 - \mathcal{B}[\![F]\!](\alpha)$$

where α_1 is taken to be some $(\Delta, \mathcal{S})$-valuation that agrees with α on the elements of Δ of type ι. The choice of such α_1 does not matter because it is only used to interpret 1st-order Σ_{bg}-formulas, which contain no variables from Δ that do not have type ι for some $\iota \in \mathfrak{I}$.

For $\overline{\Sigma}$-formulas F, we write $\mathcal{B}, \alpha \models F$ if $\mathcal{B}[\![F]\!](\alpha) = 1$, and $\mathcal{B} \models F$ if $\mathcal{B}, \alpha' \models F$ for all α'. We extend $\models$ in the usual way to sets of formulas.

Definition II.8. Let Γ be a set of HoCHCs, and suppose $\mathcal{F}$ is a frame which agrees with $\mathcal{S}$ on $\mathfrak{I}$.

(i) Γ is $(\mathcal{A}, \mathcal{F})$-***satisfiable*** if there exists a $(\Sigma_{\mathrm{fg}}, \mathcal{F})$-structure $\mathcal{B}$ such that $\mathcal{B} \models \Gamma$.
(ii) Γ is $\mathcal{A}$-***satisfiable*** (also called $\mathcal{A}$-*standard-satisfiable*) if it is $(\mathcal{A}, \mathcal{S})$-satisfiable.

Whilst the notion of $(\mathcal{A}, \mathcal{F})$-satisfiability may seem obscure, it is sometimes easier to construct $(\Sigma_{\mathrm{fg}}, \mathcal{F})$-structures (cf. Lem. IV.12); and for certain $\mathcal{F}$, $\mathcal{A}$-satisfiability implies $(\mathcal{A}, \mathcal{F})$-satisfiability (cf. Lem. IV.13).

Definition II.9. A ***program*** is a set of definite clauses.

Remark II.10. (i) Under the definition of satisfiability, a program can be seen as a conjunction of clauses, universally quantified over variables in Δ.

(ii) It is often convenient to write logically equivalent formulas such as $X\,x \leftarrow \exists y. Y\,x\,y \vee Z\,x$ instead of $\{X\,x \vee \neg Y\,x\,y,\; X\,z \vee \neg Z\,z\}$. We may even write the bodies of such formulas as existentially quantified formulas, over variables that do not appear in the head.

(iii) If Σ_{bg} contains a predicate interpreted by $\mathcal{A}$ as equality (as with Σ_{LIA} and $\mathcal{A}_{\mathrm{LIA}}$), then we may write terms of integer type inside foreground atoms. For example $X\,(x + y + 5)$ is equivalent to $X\,z \wedge (z = x + y + 5)$

(iv) Every satisfiable set of clauses Γ has, in each frame, a ***canonical model*** $\mathcal{B}$ which arises by saturating under all immediate consequences [7, Thm. 23]. In the higher-order setting, this model may not be least wrt inclusion, but for any goal clause G we have: $\Gamma \cup \{G\}$ satisfiable iff $\mathcal{B} \models G$.

C. Resolution proof system

We use a simple resolution proof system [7] consisting of only two rules: (i) a higher-order version of the usual resolution rule [12] between a goal clause and a definite clause (thus yielding a goal clause) and (ii) a rule to refute certain goal clauses which are not satisfied by the model of the background theory (similar to [13]).

$$\textbf{Resolution} \quad \frac{\neg R\,\overline{M} \vee G \qquad G' \vee R\,\overline{x}}{G \vee \big(G'[\overline{M}/\overline{x}]\big)}$$

$$\textbf{Refutation} \quad \frac{\neg x_1\,\overline{M}_1 \vee \cdots \vee \neg x_m\,\overline{M}_m \vee \neg\varphi_1 \vee \cdots \vee \neg\varphi_n}{\bot}$$

With the latter rule applicable only there exists a valuation α such that $\mathcal{A}, \alpha \models \varphi_1 \wedge \cdots \wedge \varphi_n$. (Recall that each x_i is assumed to be a variable and each φ_j is assumed to be a background atom.) Since variables are implicitly universally quantified, the rules must be applied modulo renaming of (free) variables; we write $\Gamma' \vdash_{\mathcal{A}} \Gamma' \cup \{G\}$ if G can be thus derived from the clauses in Γ' using the above rules and $\vdash^*_{\mathcal{A}}$ for the reflexive, transitive closure of $\vdash_{\mathcal{A}}$.

Theorem II.11 (Soundness and Completeness [7])**.** *Let Γ be a set of HoCHCs. Then Γ is $\mathcal{A}$-unsatisfiable if, and only if, $\Gamma \vdash^*_{\mathcal{A}} \{\bot\} \cup \Gamma'$ for some Γ'.*

It follows that a set of HoCHCs is $\mathcal{A}$-satisfiable if, and only if, it cannot be refuted by the proof system.

Consequently, the resolution proof system gives rise to a semi-decision procedure for the (standard) $\mathcal{A}$-unsatisfiability problem provided the consistency[6] of conjunctions of atoms in the background theory is semi-decidable.

III. Higher-order limit Datalog

In this section, we describe the limit restriction on HoCHC programs. We discuss the first-order fragment with this restriction, showing in Sec. III-A that its satisfiability problem is decidable. Then, in Sec. III-B, we show that this does not hold for higher-order problems, motivating the restrictions described in the rest of this paper.

To begin, we need some properties of the background theory, so we extend Assumption 1 by

Assumption 2. *Henceforth fix some set W, a 1st-order signature Σ_W and a $(\Sigma_W, \mathcal{S})$-structure $\mathcal{A}_W$ such that the first-order theory of $(\Sigma_W, \mathcal{A}_W)$ is decidable, $\leq \in \Sigma_W$, $\leq^{\mathcal{A}_W}$ is a preorder on W, and for each upset X (i.e. a subset of W such that if $x \in X$ and $x \leq^{\mathcal{A}_W} y$ then $y \in X$), there is a Σ_W-formula $\varphi(x)$ which expresses membership of X.*

Moreover fix some finite set S. We strengthen Assumption 1 by asserting that:

- $\Sigma_{\text{bg}} := \Sigma_W \cup \{=_S : S \to S \to o\} \cup \{s : S \mid s \in S\}$
- $c^{\mathcal{A}} := c^{\mathcal{A}_W}$ *if $c : \sigma \in \Sigma_W$; $s^{\mathcal{A}} := s$ if $s \in S$; and $(=_S)^{\mathcal{A}}$ is the standard equality between elements of S.* □

Note that the constraints on W imply that there are countably many upsets. Examples of such structures include the integers with $\mathcal{A}_{\text{LIA}}$ (the upsets are either $\mathbb{Z}$, $\emptyset$ or $\{x : x \geq k\}$ for some $k \in \mathbb{Z}$, each of which can easily be described by a formula) and any countable well-quasi-ordering (WQO) with a decidable background theory (for example, tuples of naturals under component-wise ordering also with the theory of linear integer arithmetic). Also note that any predicate on S can be expressed in terms of $=_S$, so our examples may freely make use of other predicates.

Recall that a *well-quasi-ordering* (WQO) [14] is a quasi-order $(W, \leq)$ such that every infinite sequence $w_1, w_2, \cdots$ contains an increasing pair: $w_i \leq w_j$ for some $i < j$. To see that all upsets of a countable WQO $(W, \leq)$ are expressible, note that any upset $X \subseteq W$ has a finite number of minimal elements $m_1, \cdots, m_n$ (say), hence X can be described as $\{w \in W : \bigvee_{i=1}^n w \geq m_i\}$. Unfortunately, this does mean that all elements of W must be constants in background theory, which makes it harder to obtain decidability (for example, the subword order is a WQO, but with constants, even the $\exists$-theory becomes undecidable [8, 15]).

Remark III.1. Note that the converse of a preorder is another preorder, and upsets under one are downsets (the complements of upsets) under the other. This means that if Assumption 2 holds for a relation, it also holds for its converse. We will make use of this by using upwards closed predicates in the definition below and in the proofs for technical convenience, despite the examples in Sec. I using downwards closed predicates.

Definition III.2. (i) An ***(upwards) limit Datalog problem*** Γ is a set of HoCHC clauses over a signature $\overline{\Sigma} = (\Sigma_{\text{bg}}, \Sigma_{\text{fg}})$, compatible with Assumptions 1 and 2, such that for each $X : \rho \in \Sigma_{\text{fg}}$, ρ contains at most one argument of type W, and if $\rho = \tau_1 \to \ldots \to \tau_n \to W \to \tau_{n+1} \to \ldots \tau_{n+m} \to o$ then Γ contains a *limit clause*:

$$X\,\overline{z}\,x\,\overline{z'} \;\leftarrow\; y \leq x \wedge X\,\overline{z}\,y\,\overline{z'}$$

writing $\overline{z}$ and $\overline{z'}$ for $z_1 \ldots z_n$ and $z_{n+1} \ldots z_{n+m}$ respectively.

(ii) A ***first-order limit Datalog problem*** Γ is a limit Datalog problem where for each $X : \rho \in \Sigma_{\text{fg}}$, it is the case that $\rho \in \sigma_{\text{FO}}$, and no atom that occurs in Γ is headed by a variable.

(iii) The ***satisfiability problem*** for limit Datalog asks: given a limit Datalog problem Γ, is it $\mathcal{A}$-satisfiable?.

A key idea of limit Datalog is that predicates with W-typed arguments must be interpreted as sets that are closed upward with respect to that argument. Consequently, a proposition $X\,\overline{z}\,y\,\overline{z'}$ asserts only that X holds of "at least y" (i.e., X is a min-predicate in the sense of [6]).

First-order limit Datalog$_{\mathbf{Z}}$: [6] describe *first-order limit Datalog$_{\mathbf{Z}}$* which is first-order limit Datalog over linear integer arithmetic. Examples of this (Examples I.1 and I.2) are given in Sec. I.

Remark III.3. [6] also allow predicates defining finite sets of integers; and both min- and max-predicates; and multiplication by constants, and by integers from fixed finite sets. They show that a limit Datalog$_{\mathbf{Z}}$ problem with these features can be transformed into one without them.

First-order limit Datalog$_{\mathbf{Z}}$ is motivated by aggregation in declarative data analysis, which is typified by its requirements for recursion and linear integer arithmetic. In declarative data analysis, the emphasis is on giving a specification of the required output rather than instructions on how to achieve it. Such an analysis is enabled by a declarative language, and experience suggests that support for high-level programming over collection types (e.g. list comprehensions, map, reduce) is particularly beneficial [16]. Consequently, a higher-order foundation, such as HoCHC, may be particularly appropriate.

Of course higher-order programming is most important for larger codebases where it can be reused many times,

[6] i.e. whether there exists a valuation α such that $\mathcal{A}, \alpha \models \varphi_1 \wedge \cdots \wedge \varphi_n$

but Examples I.3 and I.4 show that already the (first-order) examples given in [6] have a shared structure that can be factored out using a higher-order combinator.

A. Decidability at order 1

A key result of [6] is that the decision problem for the first-order language is decidable. We give an alternative proof of this theorem extended to first-order limit Datalog (allowing for structures other than linear integer arithmetic) which is helpful when understanding similar proofs in the sections that follow.

Theorem III.4. *The satisfiability problem for first-order (upwards) limit Datalog is decidable.*

Proof. Take a first-order limit Datalog problem Γ. It follows from Def. III.2 that for any $(\Sigma_{\text{fg}}, \mathcal{S})$-structure $\mathcal{B}$ such that $\mathcal{B} \models \Gamma$, predicate $X : S^n \to W \to S^m \to o \in \Sigma_{\text{fg}}$, and tuples of constants $\overline{s}, \overline{s'} \subseteq S$, the set

$$U = \{w \in W \mid \mathcal{B}[\![X\,\overline{s}\,x\,\overline{s'}]\!]([x \mapsto w]) = 1\}$$

is upwards closed. By Assumption 2, there exists a 1st-order formula $\varphi_{X,\overline{s},\overline{s'}}(x)$ such that

$$U = \{w \in W \mid \mathcal{A}_W[\![\varphi_{X,\overline{s},\overline{s'}}(x)]\!]([x \mapsto w])\}.$$

As there are finitely many predicates, finitely many tuples of elements of S and countably many such formulas φ, we have an r.e. set of candidate models. Given a $(\Sigma_{\text{fg}}, \mathcal{S})$-structure $\mathcal{B}$ of this form, because we may ground all instances of variables from S, then substitute formulas $\varphi_{X,\overline{s},\overline{s'}}$ as appropriate, removing all instances of predicate symbols. Since the 1st-order theory of $(W, \mathcal{A}_W)$ is decidable, we can decide if $\mathcal{B} \models \Gamma$.

If Γ is satisfiable, we can find such a structure by enumeration. If not, then resolution (Thm. II.11) can prove that. $\square$

Corollary III.5. *The satisfiability problem for first-order downwards limit Datalog is decidable.*

Proof. Although the above proof covers upwards limit Datalog, it only relies on the fact that upwards closed sets are expressible as 1st-order formulas. Since the complement of every downwards closed set D is an upwards closed set U, D is described by the negation of the formula describing U. Thus the proof also holds for downwards limit Datalog. $\square$

B. Undecidability in general

Unlike the first-order case, higher-order limit Datalog in general is undecidable[7], which can be proved by demonstrating that multiplication, hence Diophantine equations, is definable.

The idea is to use a pair of terms of type $\mathbb{Z} \to o$ to represent an integer. Fix a higher-order limit Datalog$_{\mathbf{Z}}$ program Γ and let $\mathcal{B}$ be its canonical model. For an integer k, we write $[\![(M,N)]\!] \equiv k$ just if $\mathcal{B}[\![M]\!] = \{x : x \geq k\}$ and $\mathcal{B}[\![N]\!] = \{x : x \geq -k\}$. This ensures that, for any $n \in \mathbb{Z}$, $\mathcal{B}[\![M\,n \wedge N\,(-n)]\!] = 1$ iff $n = k$. Then we say that a partial function $f : \mathbb{Z}^m \to \mathbb{Z}$ is *definable in* Γ just if there exist two closed terms M_1 and M_2 of type

$$\underbrace{(\mathbb{Z} \to o) \to \cdots \to (\mathbb{Z} \to o)}_{2m\text{-times}} \to (\mathbb{Z} \to o)$$

such that: if for each $i \in \{1, \ldots, m\}$, $[\![(P_i, P'_i)]\!] \equiv k_i$ then

$$\begin{aligned} & [\![(M_1\,P_1\,P'_1 \cdots P_m\,P'_m, M_2\,P_1\,P'_1 \cdots P_m\,P'_m)]\!] \\ \equiv \; & f(k_1, \ldots, k_m). \end{aligned}$$

Example III.6 (Addition)**.** Consider the following program which defines addition and the constant 5.

$$\begin{gathered} \mathrm{Add}_1, \mathrm{Add}_2 : \sigma \to \sigma \to \sigma \to \sigma \to \mathbb{Z} \to o \\ \mathrm{I}_{51}, \mathrm{I}_{52} : \sigma \quad \text{where } \sigma = \mathbb{Z} \to o \end{gathered}$$

$$\begin{aligned} & \mathrm{Add}_1\,f_1\,f_2\,g_1\,g_2\,x \\ & \qquad \leftarrow f_1\,y \wedge f_2\,(-y) \wedge g_1\,z \wedge g_2\,(-z) \wedge x \geq y + z \\ & \mathrm{Add}_2\,f_1\,f_2\,g_1\,g_2\,x \\ & \qquad \leftarrow f_1\,y \wedge f_2\,(-y) \wedge g_1\,z \wedge g_2\,(-z) \wedge x \geq -(y + z) \\ & \mathrm{I}_{51}\,x \leftarrow x \geq 5 \\ & \mathrm{I}_{52}\,x \leftarrow x \geq -5 \end{aligned}$$

In the canonical model of this program,

$$\mathrm{Add}_1\,\mathrm{I}_{51}\,\mathrm{I}_{52}\,\mathrm{I}_{51}\,\mathrm{I}_{52}\,x \wedge \mathrm{Add}_2\,\mathrm{I}_{51}\,\mathrm{I}_{52}\,\mathrm{I}_{51}\,\mathrm{I}_{52}\,(-x)$$

would hold exactly when $x = 10$. This means that the pair of partially applied functions $\mathrm{Add}_1\,\mathrm{I}_{51}\,\mathrm{I}_{52}\,\mathrm{I}_{51}\,\mathrm{I}_{52}$ and $\mathrm{Add}_2\,\mathrm{I}_{51}\,\mathrm{I}_{52}\,\mathrm{I}_{51}\,\mathrm{I}_{52}$ can be used as arguments to other functions; for example

$$\mathrm{Add}_1\,(\mathrm{Add}_1\,\mathrm{I}_{51}\,\mathrm{I}_{52}\,\mathrm{I}_{51}\,\mathrm{I}_{52})\,(\mathrm{Add}_2\,\mathrm{I}_{51}\,\mathrm{I}_{52}\,\mathrm{I}_{51}\,\mathrm{I}_{52})\,\mathrm{I}_{51}\,\mathrm{I}_{52}\,x$$

would hold for $x \geq 15$.

In Sec. B, we give another example of how functions may be composed, and recursion can work, by defining multiplication. With this we can define a goal clause corresponding to any Diophantine equation, in such a way that the program as a whole is satisfiable iff the equation has a solution. Consequently:

Theorem III.7 (Undecidability)**.** *The satisfiability problem for higher-order limit Datalog$_{\mathbf{Z}}$ is undecidable.*

Proof. Since solvability of Diophantine equations is undecidable, so is the problem of determining if a higher-order limit Datalog$_{\mathbf{Z}}$ problem is satisfiable. $\square$

IV. Initial limit Datalog

In this section, we prove Thm. IV.3, which says that a particular fragment of higher-order limit Datalog is decidable. The proof follows the same strategy as that of Thm. III.4. The key difference occurs when we enumerate candidate models; even though we can restrict the first-order predicates to an enumerable set, there are still uncountably many inhabitants of higher-order types under standard semantics.

[7]The proof given here covers integers with linear integer arithmetic. A variant works for naturals, but higher-order limit Datalog is not undecidable for all structures $(W, \mathcal{A}_W)$.

To work around this, first note that there are finitely many predicate symbols. If these were the only higher-order terms, we would be fine since Thm. II.11 can be seen as saying that for satisfiability, only concrete terms matter, so we don't need to worry about the behaviour of predicates on objects that don't correspond to terms. However, terms can contain arbitrarily deeply nested subterms, as seen in Ex. III.6 (and in Ex. B.1 which is used in the proof of undecidability). This means there can be a countable infinity of terms with distinct interpretations, leading to an uncountable infinity of interpretations for predicates over those terms. We can prevent this kind of nesting by restricting the types of predicates in the following way.

We insist that among the arguments to a predicate from Σ_{fg}, at most one is of type W, and every argument that occurs to the left of the W-typed argument (if there is one) must be of a smaller order than this function of W. For example, we would admit predicates of type $S \to W \to o$ and $(W \to o) \to W \to S \to (W \to o) \to o$, but not those of type $W \to W \to o$ nor $(W \to o) \to W \to o$.

Definition IV.1. (i) An ***initial*** type is a relational type $\sigma_1 \to \cdots \to \sigma_n \to o$ where $n \geq 0$ satisfying
- (O1) at most one of $\sigma_1, \cdots, \sigma_n$ is W, and
- (O2) if $\sigma_j = W$ then for all $i < j$, order(σ_i) < order$(\sigma_j \to \sigma_{j+1} \to \cdots \to \sigma_n \to o)$, and
- (O3) each σ_j is S, or W, or initial.

(ii) Let $\rho = \overline{\sigma} \to o$ be an initial type. We say that ρ is an ***active*** type (typically ξ) if some σ_i is W; otherwise it is an ***inactive*** type (typically ν).

(iii) An ***initial limit Datalog problem*** is a limit Datalog problem where for each $X : \rho \in \Sigma_{\mathrm{fg}}$, ρ is initial.

Example IV.2. All the types in Examples I.3 to I.6 are initial; but neither Add_1 nor Add_2 in Ex. III.6 have an initial type.

Theorem IV.3 (Decidability)**.** *Given Assumptions 1 and 2, there is an algorithm that decides whether a given initial limit Datalog problem is $\mathcal{A}$-satisfiable.*

The initial type restriction does not prevent nested terms, but it does prevent problematic ones by making the subterm relationship compatible with the type-theoretic order of the terms involved. If an order-n term N of an active type contains an order-m subterm M where $m > n$, then M is a subterm of some L (another subterm of N) of type σ (say) whose order is less than n. (This is because N must have the form $X\,\overline{L}$ where $X \in \Sigma_{\mathrm{fg}}$ is an active type, and each L_i has order less than n.) We will see later that we can take the interpretation of this type σ to be a finite set, and hence we don't need to know all possible values of M to know all possible ways in which we can interpret N.

This ensures that we can enumerate candidate models up to their behaviour on constructible elements. It allows (a) for interpretations to be defined inductively: the interpretation of all order-n predicates is given before any of order-$(n+1)$ and (b) the behaviour of a predicate on a non-W argument need only be specified on *finitely many* definable elements (Lem. IV.9).

Consider a predicate symbol $X : (W \to o) \to o \in \Sigma_{\mathrm{fg}}$. Without restriction, there may be an infinity of definable elements of type $W \to o$ and hence uncountably many choices of interpretation of X. However, a Σ_{fg}-term of type $W \to o$ can only be constructed by applying a predicate symbol Y to some arguments $N_1, \cdots, N_k$. It follows that Y has a type of shape $\sigma_1 \to \cdots \to \sigma_k \to W \to o$. By the initial type restriction, each σ_i is necessarily S and hence finite. If we have already fixed the interpretation of each such Y (each being of lower order than X), then there are only finitely many definable elements at type $W \to o$. Hence, there are only finitely many definable relations at type $(W \to o) \to o$.

Of course, when first fixing the interpretation of Y there can be infinitely many choices; but thanks to the limit Datalog restriction, only countably many can satisfy the limit clause which requires that any such interpretation is upward-closed in its W argument. It is straightforward to see that the choices are, moreover, r.e. (Lem. IV.11).

This leads to the notion of an interpretation that is built up inductively by order, in which the domains of the higher-order predicates (i.e. interpretation of types) are not determined until the interpretations of lower-order predicates have been fixed. The process of choosing interpretations for the types (i.e. the frame) and the process of choosing interpretations for the predicate symbols are entwined.

Assumptions: Recall disjoint signatures Σ_{bg} and Σ_{fg} and 1st-order structure $\mathcal{A}$ from Assumptions 1 and 2. Henceforth fix an initial limit Datalog problem Γ and take

$$l := \max\{\mathrm{order}(\rho) \mid X : \rho \in \Sigma_{\mathrm{fg}}\}.$$

Definition IV.4. Let Ξ_1 and Ξ_2 be (possibly higher-order) signatures such that $\Xi_1 \subseteq \Xi_2$; and $\mathcal{F}_1$ and $\mathcal{F}_2$ be frames.

Suppose $\mathcal{B}_1$ is a $(\Xi_1, \mathcal{F}_1)$-structure. We say that a $(\Xi_2, \mathcal{F}_2)$-structure $\mathcal{B}_2$ is a $(\Xi_2, \mathcal{F}_2)$-***expansion*** of $\mathcal{B}_1$ just if $c^{\mathcal{B}_2} = c^{\mathcal{B}_1}$ for all $c \in \Xi_1$.

We first define, given sets $U_1, \cdots, U_n$, a relation on relational functions, $\leq_{o,n} \subseteq [U_n \to \cdots \to U_1 \to \mathbb{B}]^2$, by

$$f \leq_{o,0} g := (f = 0 \text{ or } g = 1)$$
$$f \leq_{o,n+1} g := \forall x \in U_{n+1} \,.\, f(x) \leq_{o,n} g(x)$$

Define $\top_n \in [U_n \to \cdots \to U_1 \to \mathbb{B}]$ as $\forall \overline{x} \,.\, \top_n\, \overline{x} = 1$. Henceforth we elide the subscript n from $\leq_{o,n}$ and $\top_n$.

Definition IV.5. Let $n \geq 1$ and $\Xi \subseteq \Sigma_{\mathrm{fg}}$. Given a $(\Xi, \mathcal{F})$-structure $\mathcal{B}$, define the ***entwined order-n frame derived from*** $\mathcal{B}$, written $\langle \mathcal{B} \rangle_n$, by case analysis of σ as follows.

(i) σ is initial and order$(\sigma) \leq n-2$, or σ is (inactive, or S, or W, or o) and order$(\sigma) \leq n-1$:

$$\langle \mathcal{B} \rangle_n [\![\sigma]\!] := \mathcal{F}[\![\sigma]\!]$$

(ii) σ active and order$(\sigma) = n-1$:

$$\langle\mathcal{B}\rangle_n[\![W \to \nu]\!] := \{\top \in [W \to \langle\mathcal{B}\rangle_n[\![\nu]\!]]\} \cup \{X^{\mathcal{B}}\,\overline{s} \mid X : \overline{\tau} \to W \to \nu \in \Xi,\ s_i \in \mathcal{F}[\![\tau_i]\!]\}$$

$$\langle\mathcal{B}\rangle_n[\![\sigma_1 \to \xi]\!] := [\langle\mathcal{B}\rangle_n[\![\sigma_1]\!] \to \langle\mathcal{B}\rangle_n[\![\xi]\!]]$$

(iii) σ is initial and order$(\sigma) = n$:

$$\langle\mathcal{B}\rangle_n[\![W \to \nu]\!] := \{f \in [W \to \langle\mathcal{B}\rangle_n[\![\nu]\!]] \mid \forall z \leq^{\mathcal{A}} z'.f(z) \leq_o f(z')\}$$

$$\langle\mathcal{B}\rangle_n[\![\sigma_1 \to \sigma_2]\!] := [\langle\mathcal{B}\rangle_n[\![\sigma_1]\!] \to \langle\mathcal{B}\rangle_n[\![\sigma_2]\!]] \quad (\sigma_1 \neq W)$$

(iv) σ is not initial, or order$(\sigma) > n$:

$$\langle\mathcal{B}\rangle_n[\![\sigma_1 \to \sigma_2]\!] := [\langle\mathcal{B}\rangle_n[\![\sigma_1]\!] \to \langle\mathcal{B}\rangle_n[\![\sigma_2]\!]]$$

Remark IV.6. Case (iv) of Def. IV.5 is only there to ensure that $\langle\mathcal{B}\rangle_n$ is technically a frame. Such types are not used anywhere. Observe that $\top \in \langle\mathcal{B}\rangle_n[\![\rho]\!]$ for all relational ρ (provided $\top \in \mathcal{F}[\![\rho]\!]$ for all relational ρ).

For $i \geq 1$, let $\Sigma_i \subseteq \Sigma_{\text{fg}}$ consist of the predicate symbols of Σ_{fg} with types of order at most i.

Definition IV.7. (i) A family of structures $\{\mathcal{B}_n\}_{n\in\omega}$, indexed by (order) n, is said to be ***entwined*** just if $\mathcal{B}_0$ is the unique $(\emptyset, \mathcal{S})$-structure, and each $\mathcal{B}_{n+1}$ is a $(\Sigma_{n+1}, \langle\mathcal{B}_n\rangle_{n+1})$-expansion of $\mathcal{B}_n$.
(ii) An ***entwined structure*** is a member of some entwined family. An *entwined model* of Γ is an entwined structure $\mathcal{B}_{l+1}$ such that $\mathcal{B}_{l+1} \models \Gamma$.

An entwined structure is built up by an inductive process; at the frontier of the process the interpretation of an order-n type σ is a potentially infinite set, thus allowing for the full spectrum of possible interpretations for the predicates of type σ. Once the interpretation of these predicates is fixed, however, and frontier moves onto order-$(n+1)$, these same types will be reinterpreted as *finite* sets, inhabited only by the definable elements in the sense of clause (ii).

Example IV.8. Using the background theory LIA (so $W = \mathbb{Z}$), take, for example, the term $X\,a\,(Y\,b)\,(Z\,5\,X)$ for some a, b such that $\Delta(a), \Delta(b) = S$ and

$$Y : S \to S \to o$$
$$X : \rho = S \to (S \to o) \to o \to W \to (W \to o) \to o$$
$$Z : W \to \rho \to o$$

Now Z has a complicated type, but $Z\,5\,X$ must be either true or false, so we can select behaviours for X ignorant of Z (and the choices for Z can depend on this without introducing a problematic cycle). This example is elaborated in Sec. C-A of the appendix.

In the following lemmas, let $\{\mathcal{B}_n\}_{n\in\omega}$ be an entwined family, and set $\mathcal{F}_n := \langle\mathcal{B}_{n-1}\rangle_n$. Note that $\Sigma_{l+1} = \Sigma_{\text{fg}}$.

Lemma IV.9. *Let σ be an initial type. If $n >$ order(σ), or $n =$ order(σ) and σ is an inactive type, then $\mathcal{F}_n[\![\sigma]\!]$ is finite.*

Lemma IV.10. *Let σ be an initial active type. If $n =$ order(σ) then $\mathcal{F}_n[\![\sigma]\!]$ is r.e.*

Lemma IV.11. *The set of entwined families of structures is r.e.*

For each resolution proof rule, if $\mathcal{B}_{l+1}$ entails the premises of the rule, then it entails the conclusion. Since $\mathcal{B}_{l+1}[\![\bot]\!] = 0$, there is no resolution proof of $\bot$.

Lemma IV.12. *If there is an entwined family such that $\mathcal{B}_{l+1}$ models Γ, then there is no resolution proof of $\bot$ from Γ.*

On the other hand, the inductive construction gives enough freedom to choose appropriate interpretations for the predicate symbols whenever the clauses are satisfiable. Any model can be reconstructed as an entwined structure that also satisfies the clauses, with the relationship between the two mediated by a logical relation.

Lemma IV.13. *If Γ is satisfiable then there is an entwined structure that models Γ.*

Observe that, for any entwined family, $\mathcal{F}_{l+1}[\![\rho]\!]$ is finite whenever $X : \rho \in \Sigma_{\text{fg}}$. We can ask whether a particular $\mathcal{B}_{l+1} \models \Gamma$ and this is decidable because it is equivalent to a formula in the first-order theory of Σ_W.

Lemma IV.14. *Given an entwined structure $\mathcal{B}_{l+1}$, determining if it satisfies a goal or definite clause G is decidable.*

Proof of Thm. IV.3: If there is a refutation of Γ by resolution, then we know Γ is $\mathcal{A}$-unsatisfiable. By Lem. IV.12, there is no entwined structure $\mathcal{B}_{l+1}$ such that $\mathcal{B}_{l+1} \models \Gamma$.

If there is no resolution proof of $\bot$, then there is some model for Γ in standard semantics. This model can be converted into an entwined model by Lem. IV.13. Hence enumerating entwined structures—possible because they are r.e. (Lem. IV.11) and determining if $\mathcal{B}_{l+1} \models \Gamma$ is decidable (Lem. IV.14)—will find a model.

Therefore, we may interleave a search for resolution proofs of $\bot$ with a search for entwined models resulting in a decision procedure for the initial limit Datalog decision problem. ☐

Example IV.15. For a concrete example of an entwined structure, see Appendix Sec. C-A.

Remark IV.16 (Higher-order initial limit Datalog$_{\mathbf{Z}}$). It can be shown that higher-order initial limit Datalog$_{\mathbf{Z}}$ is strictly more expressive than first-order limit Datalog$_{\mathbf{Z}}$. By this, we mean that there are queries about databases (aka structures on finites sets) that can be expressed with higher-order initial limit Datalog$_{\mathbf{Z}}$ but not first-order limit Datalog$_{\mathbf{Z}}$. This follows from a result in [17] which shows that k-order Datalog captures $(k-1)$-EXPTIME. Since this only uses finite sets, the programs involved are valid higher-order initial limit Datalog$_{\mathbf{Z}}$ programs. [6] shows that first-order limit Datalog$_{\mathbf{Z}}$ has some reasonable time bounds (polynomial in database size) hence it must be less expressive.

V. EXAMPLES

In this section, we give examples of how first-order limit Datalog problems can be used with the background theory of tuples of naturals (we use currying to avoid explicitly specifying projection functions), and other WQOs.

A. Theory of tuples of naturals

In the context of limit Datalog, the theory of tuples of naturals with componentwise ordering is much more powerful than that that of integers as demonstrated by the examples below, neither of which could be accomplished using limit Datalog$_{\mathbf{Z}}$.

The following set of clauses express multiplication:

$$\begin{aligned} &F : \mathbb{N} \to \mathbb{N} \to o \\ &F\,x\,y \leftarrow x \geq 0 \wedge y \geq \alpha \\ &F\,x\,y \leftarrow y + 1 \geq n \wedge F\,r\,n \wedge x \geq r + \beta \\ &G\,x \leftarrow F\,x\,0 \end{aligned}$$

Here α and β must be constants known when constructing the clauses (alternatively, they could be seen as being taken from some database). In the canonical model of this set of clauses, the interpretation of G is the set $\{x \in \mathbb{N} : x \geq \alpha * \beta\}$.

This does not lead to undecidability like Ex. B.1 because this only expresses multiplication of constants, not variables.

This may be extended to express exponentiation α^β (demonstrated below) and further to other hyperoperations.

$$\begin{aligned} &F : \mathbb{N} \to \mathbb{N} \to \mathbb{N} \to o \\ &F\,x\,y\,z \leftarrow x \geq 0 \wedge y \geq \alpha \wedge z \geq 0 \\ &F\,x\,y\,z \leftarrow x \geq 1 \wedge y \geq 0 \wedge z \geq \beta \\ &F\,x\,y\,z \leftarrow z + 1 \geq n \wedge F\,d\,0\,n \wedge y + 1 \geq m \wedge \\ &\qquad\qquad F\,r\,m\,z \wedge x \geq r + d \\ &G\,x \leftarrow F\,x\,0\,0 \end{aligned}$$

B. Lossy counter machines

A *(classic) lossy n-counter machine (n-LCM)*, due to [18], consists of: a finite set of states Q, an initial state $q_0 \in Q$, a final state $q_f \in Q$, n counters $c_1, \ldots, c_n$, and a finite set of instructions, each of one of the two shapes A or B:

A. $(q:\ c_i := c_i + 1;\ \mathsf{goto}\ q')$
B. $(q:\ \mathsf{if}\ c_i = 0\ \mathsf{then\ goto}\ q'\ \mathsf{else}\ c_i := c_i - 1;\ \mathsf{goto}\ q'')$

A *configuration* s of such a machine is an $(n+1)$-tuple of shape $(q, m_1, \ldots, m_n)$ where $q \in Q$ and each $m_i \in \mathbb{N}$ being the current value of counter c_i.

A *transition* of such a machine consists of spontaneous loss, followed by the execution of an instruction, followed by spontaneous loss:

$$s_1 \Rightarrow s_2 \quad \text{iff} \quad \exists s_1',\, s_2'.\ s_1 \xrightarrow{l} s_1' \to s_2' \xrightarrow{l} s_2$$

The execution of an instruction $(p, m_1, \ldots, m_i, \ldots, m_n) \to (p', m_1, \ldots, m_i', \ldots, m_n)$ is defined iff:

- there is an instruction of shape A and $p = q$, $m_i' = m_i + 1$ and $p' = q'$
- or, there is an instruction of shape B and $p = q$, $m_i = 0$, $m_i' = 0$ and $p' = q'$
- or, there is an instruction of shape B and $p = q$, $m_i > 0$, $m_i' = m_i - 1$ and $p' = q''$.

The spontaneous (classic) loss $(q, m_1, \ldots, m_n) \xrightarrow{l} (q, m_1', \ldots, m_n')$ is defined iff $\forall i \in [1, n].\ m_i' \leq m_i$. Let us write $\Rightarrow^*$ for the reflexive, transitive closure of the transition relation.

The *reachability problem* for n-LCM is to decide the following: given a configuration s, does $(q_0, 0, \ldots, 0) \Rightarrow^* s$? It is known that the reachability problem is decidable as a special case of [19]. Here we give an alternative approach using initial limit Datalog over tuples of natural numbers.

To decide the problem, it suffices to construct a set of definite clauses C over the foreground signature

$$\{R_q : \mathbb{N} \to \cdots \to \mathbb{N} \to o \mid q \in Q\}$$

(each R_q is of arity n) with canonical model $\mathcal{B}$, in such a way that, for each state q, we have $\mathcal{B} \models R_q\ m_1\ \cdots m_n$ iff $(q_0, 0, \ldots, 0) \Rightarrow^* (q, m_1, \ldots, m_n)$. We define C as follows, abbreviating $x_1 \cdots x_n$ and $y_1 \cdots y_n$ by $\vec{x}$ and $\vec{y}$ respectively.

- The clause $R_{q_0}\ \vec{x} \leftarrow \bigwedge_{i \in [1,n]} x_i = 0$ is in C.
- For each state $q \in Q$, the following limit clause is in C:

$$R_q\ \vec{x} \leftarrow R_q\ \vec{y} \wedge \bigwedge_{i \in [1,n]} x_i \leq y_i$$

- For each instruction of shape A, the following clause:

$$R_{q'}\ \vec{x} \leftarrow R_q\ \vec{y} \wedge x_i = y_i + 1 \wedge \bigwedge_{j \in [1,n] \setminus \{i\}} x_j = y_j$$

- For each instruction of shape B, the two clauses:

$$R_{q'}\ \vec{x} \leftarrow R_q\ \vec{y} \wedge y_i = 0 \wedge \bigwedge_{j \in [1,n]} x_j = y_j$$

$$R_{q''}\ \vec{x} \leftarrow R_q\ \vec{y} \wedge y_i > 0 \wedge x_i = y_i - 1 \bigwedge_{j \in [1,n] \setminus \{i\}} x_j = y_j$$

Lossy channel systems and other WSTSs: Lossy counter machines are an example of a well structured transition system (WSTS) [9]. Other examples of these, such as lossy channel systems (LCSs), also have decidable reachability problems, but these do not immediately fall under our theorem because the relevant first-order theories are not decidable (in the case of LCSs the relevant theory is that of strings with concatenation with constants and the subword ordering). In some cases these results can be proved by inspecting details of exactly where in our proof the decidability property is required.

Part of our result is subsumed by the decidability of the coverability problem form WSTSs - specifically the first-order fragment where clauses only have a single foreground atom in the body and the background theory is a WQO.

C. Languages ordered by the subword order

The subword relation is a simple and important example of a WQO. [8] study the decidability of first-order theories (and fragments thereof) of languages with the subword order. Recall that a language $L \subseteq \Delta^*$ is *bounded* if $L \subseteq w_1^* \cdots w_n^*$ for some $n \geq 0$, and $w_1, \cdots, w_n \in \Delta^*$. Consider structures of the form $(L, \sqsubseteq, (w)_{w \in L})$ for some $L \subseteq \Sigma^*$ where $\sqsubseteq$ is the subword relation, and we can use every word from L as a constant.

Theorem V.1 (Kuske and Zetzsche [8]). *Let $L \subseteq \Delta^*$ be bounded and context free. Then the first-order theory of $(L, \sqsubseteq, (w)_{w \in L})$ is decidable.*

The theorem in fact holds for (a larger signature, and) a more expressive logic, first-order logic extended by a modulo counting quantifier [8]. The proof is by interpreting the structure in Presburger arithmetic, $(\mathbb{N}, +)$, which is known to be decidable in this logic.

Since $(L, \sqsubseteq)$ is a countable WQO, it follows from Thm. V.1 and Thm. IV.3 that the associated initial limit Datalog problem is decidable.

D. Basic process algebras and pushdown systems

An important class of countable WQO are *context-free processes* (or *basic process algebra*) and the more general collection of *pushdown systems*, with respect to the subword ordering [9]; moreover they are *automatic structures* (folklore but see e.g. [20, 21]) and so have decidable first-order theories ([22, 23] and various others). It follows that they satisfy Assumption 2.

VI. Related work and further directions

a) Decidable classes of constrained Horn clauses: Cox, McAloon and Tretkoff [4] have shown a catalogue of sub-recursive complexity results for various fragments of CHC obtained by restricting the syntax (in particular, the placement of variables) and the mechanism by which parameters are passed. Our work, however, takes Kaminski, Cuenca Grau, Kostylev and Motik's limit restriction [6] as the starting point.

The limit restriction was introduced as a way of taming the undecidability of Datalog$_\mathbf{Z}$ [24] that was compatible with the desire to express problems in declarative data analysis. Moreover, it is shown in [6] that, under reasonable assumptions, the query complexity of the entailment in the logic is PTIME. Our work extends limit Datalog$_\mathbf{Z}$ to higher-orders. Higher-order extensions of Datalog are interesting in their own right: [17] have shown that, on ordered databases, order-k Datalog captures $(k-1)$-EXPTIME.

b) Decidability beyond first order: There is a lot of interest in the decidability of theories that go beyond first-order logic. A very well studied case is that of monadic second-order theories (see e.g. [25]). Of these, perhaps the best known is Rabin's celebrated result on the decidability of the theory of two successor functions [26], from which the decidability of several other monadic second-order theories can be deduced.

For applications in e.g. higher-order program verification, however, it is important to retain higher-type relations of all arities and to admit background theories. A recent work with similar requirements is that of [27] who, motivated by applications in program synthesis, have introduced the logic EQSMT. Formulas of this logic have a $\exists^*\forall^*$ prefix supporting second-order quantification at certain types. They show that satisfiability of EQSMT formulas is decidable whenever satisfiability for the relevant fragments of the background theories is decidable.

c) Higher-order constrained Horn clauses: Our work takes place in the setting of HoCHC [3]. Even when the background theory is decidable, satisfiability of HoCHC is typically undecidable (already, first-order constrained Horn is typically undecidable [24]). However [7, § VIII] identified the so-called Bernays-Schönfinkel-Ramsey fragment of HoCHC, modulo a restricted form of linear integer arithmetic, has a decidable satisfiability problem by showing equi-satisfiability to clauses w.r.t. a finite number of background theories with finite domains. (HoCHC satisfiability is decidable for trivial background theories (e.g. those of finite domains).)

An alternative higher-order logic supporting integer arithmetic is HoFL$_Z$ of [28]. Whilst we do not know of any work on decidable fragments of HoFL$_Z$, we expect that a version of our results on initial limit Datalog$_\mathbf{Z}$ could be transposed into that setting.

Future directions: One question that remains open is: for which sets of types is the higher-order limit Datalog$_\mathbf{Z}$ problem decidable when predicates are restricted to those types? There are alternatives, broadly similar to Def. IV.1, which are neither a superset nor a subset of the set of initial types, for which the same proof strategy works (and we conjecture that such results can be proved as corollaries to Thm. IV.3, by inserting dummy variables).

Except for the lower bounds due to being a superset of higher-order Datalog, we have not considered runtime complexity of this problem. If the algorithm derived from the decidability proof were used, calculating its runtime would be an exercise in the construction of large numbers. Since many practical uses would have shapes that could be converted to 1st-order programs, there is some hope for tractable performance on useful subsets of initial limit Datalog$_\mathbf{Z}$.

Conclusion: We have presented *initial limit Datalog*, the first higher-order extension of constrained Horn clauses (over a non-trivial background theory) for which the satisfiability problem is decidable. Moreover the decision procedure extends to a variety of background theories, including linear integer arithmetic, and any countable well-quasi-order with a decidable first-order theory. Our decidability proof uses a new kind of term model, called *entwined structures*, which are recursively enumerable, and model checking is decidable.

References

[1] J. Jaffar and M. J. Maher, "Constraint logic programming: a survey," *The Journal of Logic Programming*, vol. 19-20, pp. 503 – 581, 1994, special Issue: Ten Years of Logic Programming.

[2] N. Bjørner, A. Gurfinkel, K. L. McMillan, and A. Rybalchenko, "Horn clause solvers for program verification," in *Fields of Logic and Computation II - Essays Dedicated to Yuri Gurevich on the Occasion of His 75th Birthday*, 2015, pp. 24–51.

[3] T. Cathcart Burn, C.-H. L. Ong, and S. J. Ramsay, "Higher-order constrained horn clauses for verification," *Proc. ACM Program. Lang.*, vol. 2, no. POPL, pp. 11:1–11:28, Dec. 2017. [Online]. Available: http://doi.acm.org/10.1145/3158099

[4] J. Cox, K. McAloon, and C. Tretkoff, "Computational complexity and constraint logic programming languages," *Ann. Math. Artif. Intell.*, vol. 5, no. 2-4, pp. 163–189, 1992. [Online]. Available: https://doi.org/10.1007/BF01543475

[5] P. J. Downey, "Undecidability of presburger arithmetic with a single monadic predicate letter," Center for Research in Computer Technology, Harvard University, Technical Report TR-18-72, 1972.

[6] M. Kaminski, B. Cuenca Grau, E. V. Kostylev, B. Motik, and I. Horrocks, "Foundations of declarative data analysis using limit datalog programs," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, 2017, pp. 1123–1130. [Online]. Available: https://doi.org/10.24963/ijcai.2017/156

[7] C.-H. L. Ong and D. Wagner, "HoCHC: A refutationally complete and semantically invariant system of higher-order logic modulo theories," in *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, 2019, pp. 1–14. [Online]. Available: https://doi.org/10.1109/LICS.2019.8785784

[8] D. Kuske and G. Zetzsche, "Languages ordered by the subword order," in *Foundations of Software Science and Computation Structures - 22nd International Conference, FOSSACS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings*, ser. Lecture Notes in Computer Science, M. Bojanczyk and A. Simpson, Eds., vol. 11425. Springer, 2019, pp. 348–364. [Online]. Available: https://doi.org/10.1007/978-3-030-17127-8_20

[9] A. Finkel and P. Schnoebelen, "Well-structured transition systems everywhere!" *Theor. Comput. Sci.*, vol. 256, no. 1-2, pp. 63–92, 2001. [Online]. Available: https://doi.org/10.1016/S0304-3975(00)00102-X

[10] A. Charalambidis, K. Handjopoulos, P. Rondogiannis, and W. W. Wadge, "Extensional higher-order logic programming," *ACM Trans. Comput. Log.*, vol. 14, no. 3, pp. 21:1–21:40, 2013.

[11] L. Henkin, "Completeness in the theory of types," *J. Symb. Log.*, vol. 15, no. 2, pp. 81–91, 1950.

[12] J. A. Robinson, "A machine-oriented logic based on the resolution principle," *J. ACM*, vol. 12, no. 1, pp. 23–41, 1965.

[13] L. Bachmair, H. Ganzinger, and U. Waldmann, "Refutational theorem proving for hierarchic first-order theories," *Appl. Algebra Eng. Commun. Comput.*, vol. 5, pp. 193–212, 1994.

[14] S. Schmitz and P. Schnoebelen, "Algorithmic aspects of WQO theories," Tech. Rep., 2017, cMI Lecture Notes. [Online]. Available: http://www.lsv.fr/~phs/algorithmic_aspects_of_wqos.pdf

[15] S. Halfon, P. Schnoebelen, and G. Zetzsche, "Decidability, complexity, and expressiveness of first-order logic over the subword ordering," in *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*. IEEE Computer Society, 2017, pp. 1–12. [Online]. Available: https://doi.org/10.1109/LICS.2017.8005141

[16] P. Alvaro, T. Condie, N. Conway, K. Elmeleegy, J. M. Hellerstein, and R. Sears, "Boom analytics: Exploring data-centric, declarative programming for the cloud," in *Proceedings of the 5th European Conference on Computer Systems*, ser. EuroSys '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 223–236. [Online]. Available: https://doi.org/10.1145/1755913.1755937

[17] A. Charalambidis, C. Nomikos, and P. Rondogiannis, "The expressive power of higher-order datalog," *TPLP*, vol. 19, no. 5-6, pp. 925–940, 2019. [Online]. Available: https://doi.org/10.1017/S1471068419000279

[18] R. Mayr, "Undecidable problems in unreliable computations," *Theoretical Computer Science*, vol. 297, no. 1, pp. 337 – 354, 2003, latin American Theoretical Informatics.

[19] A. Bouajjani and R. Mayr, "Model checking lossy vector addition systems," in *STACS 99*, C. Meinel and S. Tison, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 323–333.

[20] A. W. Lin, "Model checking infinite-state systems: Generic and specific approaches," Ph.D. dissertation, University of Edinburgh, 2010.

[21] V. Barany, "Automatic presentations of infinite structures," Ph.D. dissertation, RWTH Aachen University, 2007.

[22] B. R. Hodgson, "On direct products of automaton decidable theories," *Theor. Comput. Sci.*, vol. 19, pp. 331–335, 1982. [Online]. Available: https://doi.org/10.1016/0304-3975(82)90042-1

[23] B. Khoussainov and A. Nerode, "Automatic presentations of structures," in *Logical and Computational Complexity. Selected Papers. Logic and Computational Complexity, International Workshop LCC '94, Indianapolis, Indiana, USA, 13-16 October 1994*, ser. Lecture Notes in Computer Science, D. Leivant, Ed., vol. 960. Springer, 1994, pp. 367–392. [Online]. Available: https://doi.org/10.1007/3-540-60178-3_93

[24] E. Dantsin, T. Eiter, G. Gottlob, and A. Voronkov, "Complexity and expressive power of logic programming," *ACM Comput. Surv.*, vol. 33, no. 3, pp. 374–425, 2001.

[25] Y. Gurevich, "Monadic second-order theories," in *Model-Theoretical Logics*, J. Barwise and S. Feferman, Eds. Springer-Verlag, 1985, ch. XIII, pp. 479–506.

[26] M. O. Rabin, "Decidability of second-order theories and automata on infinite trees," *Transactions of the American Mathematical Society*, vol. 141, pp. 1–35, 1969.

[27] P. Madhusudan, U. Mathur, S. Saha, and M. Viswanathan, "A decidable fragment of second order logic with applications to synthesis," in *Proceedings of CSL'18*, 2018, pp. 31:1–31:19.

[28] N. Kobayashi, T. Tsukada, and K. Watanabe, "Higher-order program verification via HFL model checking," in *Programming Languages and Systems - 27th European Symposium on Programming, ESOP 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings*, 2018, pp. 711–738. [Online]. Available: https://doi.org/10.1007/978-3-319-89884-1_25

APPENDIX A
SUPPLEMENTARY MATERIALS FOR SEC. II

A. Logical relations

Definition A.1. Let $\mathcal{F}$ and $\mathcal{F}'$ be frames. A family of relations $\precsim_\sigma \subseteq \mathcal{F}[\![\sigma]\!] \times \mathcal{F}'[\![\sigma]\!]$ are *logical* if for all types $\tau \to \sigma$, $r \in \mathcal{F}[\![\tau \to \sigma]\!]$ and $r' \in \mathcal{F}'[\![\tau \to \sigma]\!]$, $r \precsim_{\tau \to \sigma} r'$ iff for all $s \in \mathcal{F}[\![\tau]\!], s' \in \mathcal{F}'[\![\tau]\!]$, if $s \precsim_\tau s'$ then $r(s) \precsim_\sigma r'(s')$.

Extending the $\precsim_\tau$ in the usual pointwise fashion to frames and valuations we obtain:

Lemma A.2. *Let $\mathcal{B}$ be a $(\Sigma_{\mathrm{fg}}, \mathcal{F})$-structure and $\mathcal{B}'$ be a $(\Sigma_{\mathrm{fg}}, \mathcal{F}')$-structure, α be a $(\Delta, \mathcal{F})$-valuation, α' be a $(\Delta, \mathcal{F}')$-valuation and M be a term.*

If $\mathcal{B} \precsim \mathcal{B}'$ and $\alpha \precsim \alpha'$ then $\mathcal{B}[\![M]\!](\alpha) \precsim \mathcal{B}'[\![M]\!](\alpha')$.

Proof. We prove the claim by induction on the structure of M.

The cases for variables and symbols from the foreground signature follow immediately from the assumptions.

If M is an application $M_1\, M_2$ then by the inductive hypothesis $\mathcal{B}[\![M_1]\!](\alpha) \precsim \mathcal{B}'[\![M_1]\!](\alpha')$ and $\mathcal{B}[\![M_2]\!](\alpha) \precsim \mathcal{B}'[\![M_2]\!](\alpha')$. Therefore, by definition of $\precsim$,

$$\begin{aligned}\mathcal{B}[\![M]\!](\alpha) &= \mathcal{B}[\![M_1]\!](\alpha)(\mathcal{B}[\![M_2]\!](\alpha)) \\ &\precsim \mathcal{B}'[\![M_1]\!](\alpha')(\mathcal{B}'[\![M_2]\!](\alpha')) = \mathcal{B}'[\![M]\!](\alpha')\end{aligned}$$

□

Corollary A.3. *Let $\mathcal{B}$, $\mathcal{B}'$, α and α' be as before, and let M be a $\overline{\Sigma}$-formula which does not contain a subterm of the form $\neg F$. If $\precsim_o$ is $\leq$ and $\precsim_\iota$ is $=$ for each $\iota \in \mathfrak{I}$ and $\mathcal{B} \precsim \mathcal{B}'$ and $\alpha \precsim \alpha'$ then $\mathcal{B}[\![M]\!](\alpha) \precsim \mathcal{B}'[\![M]\!](\alpha')$.*

Proof. Since $\lesssim_\iota$ is equality, α' must agree with α on variables of type ι hence $\mathcal{B}[\![M]\!](\alpha) = \mathcal{B}'[\![M]\!]$ when M is a 1st order Σ_{bg} term, hence $\mathcal{B}[\![M]\!](\alpha) \lesssim_o \mathcal{B}'[\![M]\!]$. By Lem. A.2, this also holds when M is a Σ_{fg}-term. Otherwise, $M = M_1 \wedge M_2$ or $M = M_1 \wedge M_2$ and the property holds by induction on the structure of M. □

APPENDIX B
SUPPLEMENTARY MATERIALS FOR SEC. III

Example B.1 (Multiplication, Diophantine equations)**.** In Fig. 1, Inc_i and Dec_i increment or decrement an integer represented by a pair of functions. Define the following families of formulas:

$$x : \iota \vdash \mathrm{GT}(\mathrm{x}) : \mathbb{Z} \to o$$

where $\mathrm{GT}(x) := \mathrm{Gt}\ (\mathrm{Fn}\ x)\ (\mathrm{Fn}\ (-x))$, and

$$x, y, z : \iota \vdash \mathrm{MUL}(x, y, z)$$

where $\mathrm{MUL}(x, y, z)$ is defined to be

$$\begin{aligned}&\mathrm{Mul}_1\,(\mathrm{GT}(x))\,(\mathrm{GT}(-x))\,(\mathrm{GT}(y))\,(\mathrm{GT}(-y))\,z \\ \wedge\quad &\mathrm{Mul}_2\,(\mathrm{GT}(x))\,(\mathrm{GT}(-x))\,(\mathrm{GT}(y))\,(\mathrm{GT}(-y))\,(-z)\end{aligned}$$

$$\begin{aligned}
\mathrm{Mul}_1, \mathrm{Mul}_2 &: (\mathbb{Z} \to o) \to (\mathbb{Z} \to o) \to (\mathbb{Z} \to o) \to (\mathbb{Z} \to o) \to \mathbb{Z} \to o \\
\mathrm{Dec}_1, \mathrm{Dec}_2 &: (\mathbb{Z} \to o) \to (\mathbb{Z} \to o) \to \mathbb{Z} \to o \\
\mathrm{Inc}_1, \mathrm{Inc}_2 &: (\mathbb{Z} \to o) \to (\mathbb{Z} \to o) \to \mathbb{Z} \to o \\
\mathrm{Fn} &: \mathbb{Z} \to (\mathbb{Z} \to o) \to o \\
\mathrm{Gt} &: (\mathsf{Nat} \to o) \to (\mathsf{Nat} \to o) \to \mathbb{Z} \to o \\
\mathrm{Gt}' &: (\mathsf{Nat} \to o) \to (\mathsf{Nat} \to o) \to \mathsf{Nat} \to \mathsf{Nat} \to \mathbb{Z} \to o
\end{aligned}$$

$$\begin{aligned}
\mathrm{Dec}_1\ f_1\ f_2\ x &\leftarrow f_1\ y \wedge f_2\ (-y) \wedge x \geq y - 1 \\
\mathrm{Dec}_2\ f_1\ f_2\ x &\leftarrow f_1\ y \wedge f_2\ (-y) \wedge x \geq -(y - 1) \\
\mathrm{Inc}_1\ f_1\ f_2\ x &\leftarrow f_1\ y \wedge f_2\ (-y) \wedge x \geq y + 1 \\
\mathrm{Inc}_2\ f_1\ f_2\ x &\leftarrow f_1\ y \wedge f_2\ (-y) \wedge x \geq -(y + 1) \\
\mathrm{Mul}_1\ f_1\ f_2\ g_1\ g_2\ x &\leftarrow y = 0 \wedge f_1\ y \wedge f_2\ (-y) x \geq 0 \\
\mathrm{Mul}_1\ f_1\ f_2\ g_1\ g_2\ x &\leftarrow g_1\ z \wedge g_2\ (-z) \wedge \\
&\quad \mathrm{Mul}_1\ (\mathrm{Dec}_1\ f_1\ f_2)\ (\mathrm{Dec}_2\ f_1\ f_2)\ g_1\ g_2\ w \wedge \\
&\quad \mathrm{Mul}_2\ (\mathrm{Dec}_1\ f_1\ f_2)\ (\mathrm{Dec}_2\ f_1\ f_2)\ g_1\ g_2\ (-w) \wedge \\
&\quad x \geq w + z \\
\mathrm{Mul}_1\ f_1\ f_2\ g_1\ g_2\ x &\leftarrow g_1\ z \wedge g_2\ (-z) \wedge \\
&\quad \mathrm{Mul}_1\ (\mathrm{Inc}_1\ f_1\ f_2)\ (\mathrm{Inc}_2\ f_1\ f_2)\ g_1\ g_2\ w \wedge \\
&\quad \mathrm{Mul}_2\ (\mathrm{Inc}_1\ f_1\ f_2)\ (\mathrm{Inc}_2\ f_1\ f_2)\ g_1\ g_2\ (-w) \wedge \\
&\quad x \geq w - z \\
\mathrm{Mul}_2\ f_1\ f_2\ g_1\ g_2\ x &\leftarrow y = 0 \wedge f_1\ y \wedge f_2\ (-y) \wedge x \geq -0 \\
\mathrm{Mul}_2\ f_1\ f_2\ g_1\ g_2\ x &\leftarrow g_1\ z \wedge g_2\ (-z) \wedge \\
&\quad \mathrm{Mul}_1\ (\mathrm{Dec}_1\ f_1\ f_2)\ (\mathrm{Dec}_2\ f_1\ f_2)\ g_1\ g_2\ w \wedge \\
&\quad \mathrm{Mul}_2\ (\mathrm{Dec}_1\ f_1\ f_2)\ (\mathrm{Dec}_2\ f_1\ f_2)\ g_1\ g_2\ (-w) \wedge \\
&\quad x \geq -(w + z) \\
\mathrm{Mul}_2\ f_1\ f_2\ g_1\ g_2\ x &\leftarrow g_1\ z \wedge g_2\ (-z) \wedge \\
&\quad \mathrm{Mul}_1\ (\mathrm{Inc}_1\ f_1\ f_2)\ (\mathrm{Inc}_2\ f_1\ f_2)\ g_1\ g_2\ w \wedge \\
&\quad \mathrm{Mul}_2\ (\mathrm{Inc}_1\ f_1\ f_2)\ (\mathrm{Inc}_2\ f_1\ f_2)\ g_1\ g_2\ (-w) \wedge \\
&\quad x \geq -(w - z) \\
\mathrm{Fn}\ x\ f &\leftarrow f\ y \wedge x \geq y \\
\mathrm{Gt}\ h_1\ h_2\ x &\leftarrow \mathrm{Gt}'\ h_1\ h_2\ \mathrm{C}_{01}\ \mathrm{C}_{01}\ y \wedge x \geq y \\
\mathrm{Gt}'\ h_1\ h_2\ f_1\ f_2\ x &\leftarrow \mathrm{Gt}'\ h_1\ h_2\ (\mathrm{Inc}_1\ f_1\ f_2)\ (\mathrm{Inc}_2\ f_1\ f_2)\ y \wedge x \geq y \\
\mathrm{Gt}'\ h_1\ h_2\ f_1\ f_2\ x &\leftarrow \mathrm{Gt}'\ h_1\ h_2\ (\mathrm{Dec}_1\ f_1\ f_2)\ (\mathrm{Dec}_2\ f_1\ f_2)\ y \wedge x \geq y \\
\mathrm{Gt}'\ h_1\ h_2\ f_1\ f_2\ x &\leftarrow h_1\ f_1 \wedge h_2\ f_2 \wedge f_1\ y \wedge f_2\ (-y) \wedge x \geq y
\end{aligned}$$

Figure 1. Coding Multiplication

The family GT can be used to turn an integer x of type $\mathbb{Z}$ into either function from the pair representing x. This allows us to obtain MUL such that in the canonical model, MUL $x\ y\ z$ holds iff $x \times y = z$.

With this we can define a goal clause corresponding to any Diophantine equation. For example the equation $x^3 = y + z$ corresponds to the goal clause

$$\mathrm{MUL}(x,\, x,\, w) \wedge \mathrm{MUL}(x,\, w,\, y + z).$$

The limit $\mathrm{Datalog}_{\mathbf{Z}}$ problem consisting of this clause together with the set of clauses above are satisfiable if, and only if, the

Diophantine equation has a solution.

Appendix C
Supplementary materials for Sec. IV

Lemma C.1. *Given a $(\Sigma_{\mathrm{fg}}, \mathcal{F})$-structure $\mathcal{B}$, $\langle\mathcal{B}\rangle_n$ is a frame.*

Proof. To see that $\langle\mathcal{B}\rangle_n$ meets the requirements on o, W and S, note that $\mathcal{F}$ is a frame, and agrees with it on those types.

The other condition is that for types σ of the form $\sigma_1 \to \sigma_2$, we need to show that $\langle\mathcal{B}\rangle_n[\![\sigma]\!]$ is a subset of $[\langle\mathcal{B}\rangle_n[\![\sigma_1]\!] \to \langle\mathcal{B}\rangle_n[\![\sigma_2]\!]]$:

If order$(\sigma) \leq n-2$, or order$(\sigma) \leq n-1$ and σ is not an active type, then order$(\sigma_1) \leq n-2$ and order$(\sigma_2) \leq n-2$, or order$(\sigma_2) \leq n-1$ and σ_2 is not an active type, so $\langle\mathcal{B}\rangle_n[\![\tau]\!] = \mathcal{F}[\![\tau]\!]$ where τ is σ, σ_1 or σ_2, hence the property holds because $\mathcal{F}$ is a frame.

If order$(\sigma) = n-1$ (and σ is an active type), then the non-trivial case is when $\sigma = W \to \nu$. Here, we rely on the fact that order$(\nu) \leq n-1$ and ν is not an active type, hence $\langle\mathcal{B}\rangle_n[\![\nu]\!] = \mathcal{F}[\![\nu]\!]$. For any $X : \overline{\tau} \to W \to \nu \in \Xi \setminus \Sigma$, $X^{\mathcal{B}}$ is an element of $\mathcal{F}[\![\overline{\tau} \to W \to \nu]\!]$ hence $X^{\mathcal{B}}\,\overline{s}$ is in $\mathcal{F}[\![W \to \nu]\!] \subseteq [W \to \mathcal{F}[\![\nu]\!]]$. Therefore $\langle\mathcal{B}\rangle_n[\![\sigma]\!]$ is a subset of $[\langle\mathcal{B}\rangle_n[\![\sigma_1]\!] \to \langle\mathcal{B}\rangle_n[\![\sigma_2]\!]]$.

Otherwise, $\langle\mathcal{B}\rangle_n[\![\sigma]\!]$ is explicitly constructed as a subset of $[\langle\mathcal{B}\rangle_n[\![\sigma_1]\!] \to \langle\mathcal{B}\rangle_n[\![\sigma_2]\!]]$ □

Lemma IV.9. *Let σ be an initial type. If $n >$ order(σ), or $n =$ order(σ) and σ is an inactive type, then $\mathcal{F}_n[\![\sigma]\!]$ is finite.*

Proof. We prove by lexicographical induction on n followed by the structure of σ.

If order$(\sigma) = 0$ then $\mathcal{F}_n[\![\sigma]\!] = \{0,1\}$ or S, hence finite.

Suppose order$(\sigma) > 0$. We may assume that either order$(\sigma) = n-1$ and σ is an active type, or order$(\sigma) = n$ and σ is an inactive type, for otherwise $\mathcal{F}_n[\![\sigma]\!] := \mathcal{F}_{n-1}[\![\sigma]\!]$ which is finite by the induction hypothesis.

If $\sigma = W \to \nu$ then

$$\mathcal{F}_n[\![\sigma]\!] := \left\{X^{\mathcal{B}_{n-1}}\,\overline{s} \mid X : \overline{\tau} \to W \to \nu \in \Sigma_{\mathrm{fg}}, s_i \in \mathcal{F}_{n-1}[\![\tau_i]\!]\right\}$$

which is finite because there are only finitely many $X \in \Sigma_{\mathrm{fg}}$, and $\mathcal{F}_{n-1}[\![\tau_i]\!]$ is finite (because order$(\tau_i) <$ order$(W \to \nu) = n-1$).

Otherwise, $\sigma = \sigma_1 \to \sigma_2$ with $\sigma_1 \neq W$.

If σ is an active type of order $n-1$, then order$(\sigma_1) < n-1$, and so $\mathcal{F}_n[\![\sigma_1]\!] = \mathcal{F}_{n-1}[\![\sigma_1]\!]$ which is finite by IH. Now σ_2 is an active type of order at most $n-1$. By the IH (σ_2 is smaller than σ), $\mathcal{F}_n[\![\sigma_2]\!]$ is finite.

If σ is an inactive type of order n, then order$(\sigma_1) \leq n-1$. The type σ_1 could be an active type, but it is smaller than σ, and so, by IH we have $\mathcal{F}_n[\![\sigma_1]\!]$ is finite. The type σ_2 is inactive and has order at most n, but as it is smaller than σ, $\mathcal{F}_n[\![\sigma_2]\!]$ is finite by IH.

Therefore, in both cases, $\mathcal{F}_n[\![\sigma]\!] = [\mathcal{F}_n[\![\sigma_1]\!] \to \mathcal{F}_n[\![\sigma_2]\!]]$ is finite.

□

Lemma IV.10. *Let σ be an initial active type. If $n =$ order(σ) then $\mathcal{F}_n[\![\sigma]\!]$ is r.e.*

Proof. By induction on arity(σ). If $\sigma = W \to \nu$ and order$(\sigma) = n$, then $\nu = \sigma_1 \to \cdots \to \sigma_k \to o$ and order$(\sigma_i) \leq n-1$. It follows from Lemma IV.9 that each $\mathcal{F}_n[\![\sigma_i]\!]$ is finite. Thanks to Assumption 2 each monotone function $f : [W \to \mathcal{F}_n[\![\nu]\!]]$ can be described by a map

$$g : [\mathcal{F}_n[\![\sigma_1]\!] \to \cdots \to [\mathcal{F}_n[\![\sigma_k]\!] \to \mathrm{Form}] \cdots]$$

where Form is the set of formulas over Σ_W with one free variable x and $f(z, \overline{s}) = \mathcal{A}[\![g(\overline{s})]\!]([x \mapsto z])$. The set of such g is recursively enumerable (Form is r.e. and $\mathcal{F}[\![\sigma_i]\!]$ are finite).

Otherwise, $\sigma = \sigma_1 \to \xi$ with order$(\sigma_1) < n$, so $\mathcal{F}_n[\![\sigma_1]\!]$ is finite, hence $[\mathcal{F}_n[\![\sigma_1]\!] \to \mathcal{F}_n[\![\xi]\!]]$ is r.e. if $\mathcal{F}_n[\![\xi]\!]$ is, which holds by induction on arity. □

Lemma IV.11. *The set of entwined families of structures is r.e.*

Proof. For $i > l+1$, $\mathcal{B}_i$ is a structure extending $\mathcal{B}_{i-1}$ over the same signature, therefore $\mathcal{B}_i = \mathcal{B}_{l+1}$ by induction. Hence we only need to show that there are enumerably many possible $\mathcal{B}_{l+1}$.

There is exactly one possible $\mathcal{B}_0$.

If we fix $\mathcal{B}_{n-1}$, then to construct the structure $\mathcal{B}_n$ we need to choose interpretations for predicates in Σ_{fg} which have order n type, and predicates that are inactive type with order $n-1$. Since these choices must come from r.e. or finite sets (by Lemmas IV.10 and IV.9), there are enumerably many options for $\mathcal{B}_n$. Since an r.e. collection of r.e. sets is r.e., the number of families (up to order $l+1$) is r.e. by induction. □

Lemma IV.12. *If there is an entwined family such that $\mathcal{B}_{l+1}$ models Γ, then there is no resolution proof of $\bot$ from Γ.*

Proof. For each resolution proof rule, if $\mathcal{B}_{l+1}$ entails the premises of the rule, then it entails the conclusion. Since $\mathcal{B}_{l+1}[\![\bot]\!] = 0$, there is no resolution proof of $\bot$.

Resolution:

If $\mathcal{B}_{l+1} \models \neg R\,\overline{M} \vee G$ and $\mathcal{B}_{l+1} \models G' \vee R\,\overline{x}$, then for each α either $\mathcal{B}_{l+1}[\![\ R\ \overline{M}]\!](\alpha) = 1$ (and $\mathcal{B}_{l+1}[\![G]\!](\alpha) = 1$) or $\mathcal{B}_{l+1}[\![R\ \overline{M}]\!](\alpha) = 0$ (and we have $\mathcal{B}_{l+1}[\![(G'[\overline{M}/\overline{x}])]\!](\alpha) = \mathcal{B}_{l+1}[\![G']\!](\alpha[\overline{M}/\overline{x}]) = 1$). Therefore $\mathcal{B}_{l+1}[\![G \vee \left(G'[\overline{M}/\overline{x}]\right)]\!](\alpha) = 1$

Constraint refutation:

If there exists a valuation α such that $\mathcal{A}, \alpha \models \varphi_1 \wedge \cdots \wedge \varphi_n$ then, setting $\alpha(x_i) = \top$ for each i, $\mathcal{B}_{l+1}[\![\neg(x_1\,\overline{M}_1) \vee \cdots \vee \neg(x_m\,\overline{M}_m) \vee \neg\varphi_1 \vee \cdots \vee \neg\varphi_n]\!](\alpha) = 0$, so if $\mathcal{B}_{l+1} \models \neg(x_1\,\overline{M}_1) \vee \cdots \vee \neg(x_m\,\overline{M}_m) \vee \neg\varphi_1 \vee \cdots \vee \neg\varphi_n$, we get $\mathcal{B}_{l+1} \models \bot$ vacuously. □

Lemma IV.13. *If Γ is satisfiable then there is an entwined structure that models Γ.*

Proof. The proof is by logical relations and Cor. A.3.

Given a standard model $\mathcal{B}$, we construct a family of entwined structures by induction on n. As we construct an entwined family $\{\mathcal{B}_n\}_{n\in\omega}$, we define logical relations $\precsim^n$ between $\mathcal{F}_n := \langle \mathcal{B}_{n-1}\rangle_n$ and $\mathcal{S}$ recursively as follows:

$$i \precsim^n_\iota i' := i = i' \qquad \iota \in \mathfrak{I}$$
$$b \precsim^n_o b' := b \leq b'$$
$$r \precsim^n_{\tau\to\sigma} r' := \forall s \in \mathcal{F}_n[\![\tau]\!], s' \in \mathcal{S}[\![\tau]\!].\, s \precsim^n_\tau s' \to r(s) \precsim^n_\sigma r'(s')$$

Thus $\precsim^n$ is, by construction, the unique logical relation which is $\leq$ on o and is $=$ on other base types.

Let us suppose the $(\Sigma_{n-1}, \mathcal{F}_{n-1})$-structure $\mathcal{B}_{n-1}$ is defined. We construct $\mathcal{B}_n$ as a $(\Sigma_n, \mathcal{F}_n)$-expansion of $\mathcal{B}_{n-1}$ by:

$$Y^{\mathcal{B}_n} f_1 \cdots f_r := \bigwedge_{\overline{f} \precsim \overline{g}} Y^{\mathcal{B}} g_1 \cdots g_r$$

where $Y : \sigma_1 \to \cdots \to \sigma_r \to o$ is an order-n initial type, and each $f_i \in \mathcal{F}_n[\![\sigma_i]\!]$.

Showing that $\mathcal{B}_n$ is well defined: We need to show that $Y^{\mathcal{B}_n} \in \mathcal{F}_n[\![\overline{\sigma} \to o]\!]$. Suppose Y is of an order-n active type $Y : \overline{\tau} \to W \to \overline{\alpha} \to o$ where:

- each τ_i (and each α_j) is initial or S
- for each i, order(τ_i) $<$ order($W \to \overline{\alpha} \to o$).

Since order($W \to (\overline{\alpha} \to o)$) $= n$ and:

$$\begin{aligned} &\mathcal{F}_n[\![\overline{\tau} \to W \to \overline{\alpha} \to o]\!] \\ =\ &\big[\mathcal{F}_n[\![\tau_1]\!] \to \cdots \to [\mathcal{F}_n[\![\tau_k]\!] \to \mathcal{F}_n[\![W \to (\overline{\alpha} \to o)]\!]]\big] \end{aligned}$$

we need to show that, for each $s_i \in \mathcal{F}_n[\![\tau_i]\!]$, $Y^{\mathcal{B}_n}\ \overline{s}$ is monotone in the first argument. I.e., assume $z \leq w$, we want to show $Y^{\mathcal{B}_n}\ \overline{s}\ z \leq_o Y^{\mathcal{B}_n}\ \overline{s}\ w$; or equivalently

$$\forall f_1, \cdots, f_k \,.\, Y^{\mathcal{B}_n}\ \overline{s}\ z\ f_1 \cdots f_k \leq Y^{\mathcal{B}_n}\ \overline{s}\ w\ f_1 \cdots f_k. \quad (1)$$

Now, since $\mathcal{B}$ is a model of Γ which is assumed to be a limit Datalog problem, $Y^{\mathcal{B}}\ \overline{t} \in \mathcal{S}[\![W \to \overline{\alpha} \to o]\!]$ is upward closed in the first argument for all $\overline{t}$ (and in particular for those satisfying $\overline{s} \precsim \overline{t}$), i.e., for all $\overline{g}$ (and in particular for those satisfying $\overline{f} \precsim \overline{g}$)

$$Y^{\mathcal{B}}\ \overline{t}\ z\ \overline{g} \leq Y^{\mathcal{B}}\ \overline{t}\ w\ \overline{g}$$

which implies (1), by considering the definition of $Y^{\mathcal{B}_n}$.

Next suppose $Y : \sigma_1 \to \cdots \to \sigma_k \to o$ is an inactive type. Then

$$\mathcal{F}_n[\![\overline{\sigma} \to o]\!] = \big[\mathcal{F}_n[\![\sigma_1]\!] \to \cdots \to [\mathcal{F}_n[\![\sigma_k]\!] \to \mathcal{F}_n[\![o]\!]]\big]$$

(and each $\mathcal{F}_n[\![\sigma_i]\!]$ is finite, by Lem. IV.9). It follows from the definition that $Y^{\mathcal{B}_n} \in \mathcal{F}_n[\![\overline{\sigma} \to o]\!]$, as desired.

Note that $\mathcal{B}_{l+1} \precsim \mathcal{B}$. The construction of $Y^{\mathcal{B}_n}$ as a greatest lower bound matches the definition of logical relations and ensures that $\mathcal{B}_{l+1}$ is the greatest entwined model such that this holds.

We now show that $\mathcal{B}_{l+1}$ models Γ: Consider a goal clause $G \in \Gamma$. For any $(\Delta, \mathcal{F}_{l+1})$-valuation α, there is a standard valuation α' such that $\alpha' \gtrsim \alpha$ (just take $\alpha'(x) = \alpha(x)$ on variables of type W or S and take $\alpha'(x) = \top$ otherwise). Since $\mathcal{B}$ is a model, $\mathcal{B}[\![G]\!](\alpha') = 1$, and by Lem. A.2, $\mathcal{B}_{l+1}[\![G]\!](\alpha) = 1$. Hence G is satisfied by $\mathcal{B}_{l+1}$.

Now consider a definite clause $X\,\overline{x} \vee G \in \Gamma$ and some $(\Delta, \mathcal{F}_{l+1})$-valuation α. If $\mathcal{B}_{l+1}[\![X\ \overline{x}]\!](\alpha) = 0$ then we have $X^{\mathcal{B}_{l+1}}\alpha(x_1)\cdots\alpha(x_k) = 0$. By construction of $\mathcal{B}_{l+1}$, there exist $g_1, \cdots, g_k$ such that $X^{\mathcal{B}}\ \overline{g} = 0$ and each $g_i \gtrsim \alpha(x_i)$. This allows us to construct $\alpha' \gtrsim \alpha\ \mathcal{B}[\![X\ \overline{x}]\!](\alpha') = 0$. Again, since $\mathcal{B}$ is a model, $\mathcal{B}[\![G]\!](\alpha) = 1$, and by Cor. A.3, $\mathcal{B}_{l+1}[\![G]\!](\alpha) = 1$. Therefore $\mathcal{B}_{l+1}$ satisfies $X\,\overline{x} \vee G$. ☐

Lemma IV.14. *Given an entwined structure $\mathcal{B}_{l+1}$, determining if it satisfies a goal or definite clause G is decidable.*

Proof. We prove this by converting the clause into a first-order formula over Σ_W. We obtain the formula by a 2 step transformation. These steps involve formulas of higher-order logic as defined in Sec. II, not just those with the structure of Horn clauses.

a) Preprocessing - replacing variables (not of type W) by constants: We begin by assuming that for each relational type ρ, for each element of $f \in \mathcal{B}_{l+1}[\![\rho]\!]$, Σ_{fg} contains a predicate symbol X_f. We also assume that $\mathcal{B}^{X_f} = f$. If this did not hold, it is trivial to construct a new Σ_{fg} and the corresponding $\mathcal{B}_{l+1}$. This obviously does not affect satisfiability of a set of clauses which do not mention the newly added predicate symbols.

For each variable x in G that is not of type W, $\mathcal{B}_{l+1}[\![\Delta(x)]\!]$ is finite. Therefore we may replace G by a conjunction of clauses – one for each possible interpretation of x (we introduce constants corresponding to each possibility).

Every foreground atom in the clause now has the shape $X\,\overline{M}$ where at most one M_i is a variable (of type W), thanks to the initiality restriction on the type of X. All other M_i are either constants or have this same shape.

b) Eliminating variables in foreground terms: We will now eliminate variables of type W in foreground atoms. We do so by induction on the number of free variables in a foreground atom - at each step, we replace an atom by a disjunction of guarded atoms, each with one fewer free variable.

A foreground atom has the form $X\,\overline{M}$ and not more than one of M_i are variables (because all remaining variables are of type W and X has initial type). If exactly one M_j is a variable x and no other M_i contain variables, then we may replace the atom $X\,\overline{M}$ by the background formula $\varphi(x)$ corresponding to the upset $\{w \in W \mid \mathcal{B}[\![X\,\overline{M}]\!]([x \mapsto w])\}$. Such a formula exists by Assumption 2 (see also the proof of Lem. IV.10, which shows we can enumerate entwined structures while having access to the formulas corresponding to these upsets).

For foreground atoms involving a more deeply nested variable (including those with more than one variable), any atom M with at least one variable must contain some subterm N of the shape $X\,\overline{M}\,x\,\overline{N}$ where x is a variable and neither $\overline{M}$

nor $\overline{N}$ contain any variables. We take $M[-]$ to be the one-holed context such that $M[N] = M$. Take the type of X to be $\overline{\sigma} \to W \to \overline{\sigma'} \to \overline{\tau} \to o$ where $\overline{\sigma'}$ are the types of $\overline{N}$. Note that the type of N is $\overline{\tau} \to o$.

Since X has an initial type, we know that each $\mathcal{B}[\![\tau_i]\!]$ is finite. Therefore the set of tuples $T = \Pi_{i=1}^{n} \mathcal{B}[\![\tau_i]\!]$ is finite. For each tuple $\overline{t} \in T$, there is a formula $\varphi_{\overline{t}}(x)$ (like φ above) defining the values z such that $\mathcal{B}[\![N\,\overline{y}]\!]([x \mapsto z, \overline{y} \mapsto \overline{t}]) = 1$. If we now consider $S \subseteq T$, the function $f(z) = \mathcal{B}[\![N]\!]([x \mapsto z])$ (which returns a function in $[T \to \mathbb{B}]$) is constant on the regions where $\bigwedge_{\overline{t} \in S} \varphi_{\overline{t}}(x) \wedge \bigwedge_{\overline{t} \in T \setminus S} \neg\varphi_{\overline{t}}(x)$ holds (if the formula holds at $x = z$ then $f(z)(\overline{t})$ can be determined by whether or not $\overline{t} \in S$, hence does not depend on the precise value of z). Denote this constant by N_S.

Since W can be partitioned by according to the powerset of such $\varphi_{\overline{t}}$, we may replace M by the formula $\bigvee_{S \subseteq T}(\bigwedge_{\overline{t} \in S} \varphi_{\overline{t}}(x) \wedge \bigwedge_{\overline{t} \in T \setminus S} \neg\varphi_{\overline{t}}(x) \wedge M[N_S])$. Since N_S contains no variables, each $M[N_S]$ contains one fewer variable than M, allowing us to inductively remove all variables from atoms until we are left with constant expressions that reduce to booleans and background atoms. Since the first-order theory of Σ_W is decidable, we are done. $\square$

A. Example of an Entwined Structure

Let the background theory be the theory of equality on the finite set $S = \{\clubsuit, \diamondsuit, \spadesuit, \heartsuit\}$ in combination with the theory of linear integer arithmetic, in which $\mathbb{Z}$ is ordered by $\leq$. We give an entwined structure that interprets the three predicate symbols X, Y and Z from Example IV.8. The signature of the foreground, Σ_{fg}, in this case consists of:

$$\begin{aligned} &X : \rho \\ &Y : S \to S \to o \\ &Z : W \to \rho \to o \end{aligned}$$

where ρ is shorthand for $S \to (S \to o) \to o \to \xi$ and ξ for $W \to (W \to o) \to o$. So the type of Y is order 1, the type of X order 2 and the type of Z order 3. Correspondingly, we have $\Sigma_1 = \{Y : S \to S \to o\}$, $\Sigma_2 = \{X : \rho, Y : S \to S \to o\}$ and then $\Sigma_3 = \Sigma_{\text{fg}}$.

We can build an entwined structure to interpret Σ_{fg} in stages, according to the definition.

- Define $\mathcal{B}_0$ as the unique $(\emptyset, \mathcal{S})$-structure, i.e. that interprets the background theory standardly and interprets the empty signature vacuously. In particular, we have the following interpretations of the base types:

$$\begin{aligned} \mathcal{S}[\![o]\!] &= \mathbb{B} \quad (= \{0,1\}) \\ \mathcal{S}[\![S]\!] &= \{\clubsuit, \diamondsuit, \spadesuit, \heartsuit\} \\ \mathcal{S}[\![W]\!] &= \mathbb{Z} \end{aligned}$$

- Then $\langle\mathcal{B}_0\rangle_1$ the order-1 entwined frame derived from $\mathcal{B}_0$ is determined by the definition. According to clause (i), the base types are interpreted as in $\mathcal{S}$ and by clause (iii) we have, in particular, the following interpretations of first-order types:

$$\begin{aligned} \langle\mathcal{B}_0\rangle_1[\![S \to o]\!] &= [S \to \mathbb{B}] \\ \langle\mathcal{B}_0\rangle_1[\![S \to (S \to o)]\!] &= [S \to [S \to \mathbb{B}]] \end{aligned}$$

and $\langle\mathcal{B}_0\rangle_1[\![W \to o]\!]$ as the set of monotone functions:

$$\{f \in [\mathbb{Z} \to \mathbb{B}] \mid \forall z \leq z'.\, f(z) \leq f(z')\}$$

Of course, all other types are interpreted too, but these are the order-1 types that will be important in assigning a meaning to X, Y and Z.

- We can use $\langle\mathcal{B}_0\rangle_1$ to frame the entwined interpretation of the order-1 foreground symbols in Σ_1, via a $(\Sigma_1, \langle\mathcal{B}_0\rangle_1)$-expansion of $\mathcal{B}_0$. The definition of expansion forces $\mathcal{B}_1$ to interpret the background theory in the same way as $\mathcal{B}_0$ (i.e. standardly) but we have free choice of interpretation of $Y \in \Sigma_1$ as any element of $[S \to [S \to \mathbb{B}]]$. Let us pick (it is not important here since we are not interested in satisfying a particular set of clauses):

$$\mathcal{B}_0[\![Y]\!] = \begin{cases} 1 & \text{if } \{s_1, s_2\} \subseteq \{\diamondsuit, \spadesuit\} \\ 0 & \text{otherwise} \end{cases}$$

- Then $\langle\mathcal{B}_1\rangle_2$, the order-2 entwined frame derived from $\mathcal{B}_1$ is determined according to the definition. From clause (i) we have that the base types are interpreted as in $\langle\mathcal{B}_0\rangle_1$ and also the inactive initial types:

$$\begin{aligned} \langle\mathcal{B}_1\rangle_2[\![S \to o]\!] &= \langle\mathcal{B}_0\rangle_1[\![S \to o]\!] \\ \langle\mathcal{B}_1\rangle_2[\![S \to S \to o]\!] &= \langle\mathcal{B}_0\rangle_1[\![S \to S \to o]\!] \end{aligned}$$

In this way, the interpretation $\mathcal{B}_1[\![Y]\!]$ still makes sense within this frame. According to clause (ii), the active initial type $W \to o$ is reinterpreted as follows:

$$\langle\mathcal{B}_1\rangle_2[\![W \to o]\!] = \{\top\}$$

since there are no terms of type $W \to o$ that can be obtained as partial applications of X, Y or Z. By clause (iii), we have:

$$\langle\mathcal{B}_1\rangle_2[\![\rho]\!] = [S \to [[S \to \mathbb{B}] \to [\mathbb{B} \to \langle\mathcal{B}_1\rangle_2[\![\xi]\!]]]]$$

and $\langle\mathcal{B}_1\rangle_2[\![\xi]\!]$ is the set:

$$\{f \in [\mathbb{Z} \to [\{\top\} \to \mathbb{B}]] \mid \forall z \leq z'.\, f(z) \leq f(z')\}$$

- When we define $\mathcal{B}_2$ as a $(\Sigma_2, \langle\mathcal{B}_1\rangle_2)$-expansion of $\mathcal{B}_1$ we are forced by the notion of expansion to take the interpretation of the background as in $\mathcal{B}_1$ and also:

$$\mathcal{B}_2[\![Y]\!] = \mathcal{B}_1[\![Y]\!]$$

The definition of entwined frame ensures that the interpretation also makes sense in $\langle\mathcal{B}_1\rangle_2[\![S \to S \to o]\!]$. On the other hand, we are free to choose any element of the infinite set $\langle \mathcal{B}_1 \rangle_2 [\![\rho]\!]$ with which to interpret the other element of Σ_2, namely X. We pick:

$$\mathcal{B}_2[\![X]\!](s)(f)(b)(w)(g) = \begin{cases} 1 & \text{if } b \text{ and } f(s) \text{ and } w > 5 \\ 0 & \text{otherwise} \end{cases}$$

Here we can see concretely the intuition explained in Example IV.8: even though $Z\,5\,X$ is a potential (third) argument to X and so must be accounted for when describing how to interpret the type of X (so that the application is defined), we can understand the (finitely many) values that are possible for $Z\,5\,X$ without knowing how to interpret Z.

- Then $\langle \mathcal{B}_2 \rangle_3$ is determined as follows. By (i) all base types, the inactive initial types of order-1 and, now, also the active initial type $W \to o$ are interpreted as in the previous entwined frame. Next, by (ii) we have:

  $$\langle \mathcal{B}_2 \rangle_3 [\![\rho]\!] = [S \to [[S \to \mathbb{B}] \to [\mathbb{B} \to \langle \mathcal{B}_2 \rangle_3 [\![\xi]\!]]]]$$

  and $\langle \mathcal{B}_2 \rangle_3 [\![\xi]\!]$ is the set:

  $$\{\top\} \cup \{\mathcal{B}_2[\![X]\!](s)(f)(b) \mid s \in S \wedge f \in [S \to o] \wedge b \in \mathbb{B}\}$$

  which is equal to $\{\top, \bot, (w \mapsto (g \mapsto w > 5))\}$.
  By clause (iii) we interpret the order-3 type $W \to \rho \to o$ as:

  $$\{f \in [\mathbb{Z} \to [\langle \mathcal{B}_2 \rangle_3 [\![\rho]\!] \to \mathbb{B}]] \mid \forall z \leq z'.\, f(z) \leq f(z')\}$$

- Now we can define a $(\Sigma_3, \langle \mathcal{B}_2 \rangle_3)$-expansion of $\mathcal{B}_2$. We are forced to take $\mathcal{B}_3[\![Y]\!] = \mathcal{B}_2[\![Y]\!]$ and $\mathcal{B}_3[\![X]\!] = \mathcal{B}_3[\![X]\!]$, but this is possible because the definition of $\langle \mathcal{B}_2 \rangle_3 [\![\rho]\!]$ ensures that $\mathcal{B}_2[\![X]\!]$ remains an element. We are free to choose an appropriate way to interpret the third-order symbol Z, let us take:

  $$\mathcal{B}_3[\![Z]\!](w)(f) = \begin{cases} 1 & \text{if } f(\heartsuit)(\mathcal{B}_3[\![Y]\!](\spadesuit))(1)(w)(\top) \\ 0 & \text{otherwise} \end{cases}$$