Take Command of Your Constraints! (Technical Report)

Sung-Shik T.Q. Jongmans and Farhad Arbab

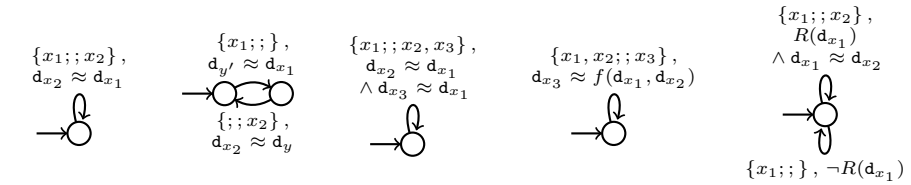
Centrum Wiskunde & Informatica, Amsterdam, Netherlands
[jongmans,farhad]@cwi.nl

Abstract. Constraint automata (CA) are a coordination model based on finite automata on infinite words. Although originally introduced for compositional *modeling* of coordinators, an interesting new application of CA is actually *implementing* coordinators (i.e., compiling CA to executable code). Such an approach guarantees correctness-by-construction and can even yield code that outperforms hand-crafted code. The extent to which these two potential advantages arise depends on the smartness of CA-compilers and the existence of proofs of their correctness.
We present and prove the correctness of a critical optimization for CA-compilers: a sound and complete translation from declarative constraints in transition labels to imperative commands in a sequential language. This optimization avoids expensive calls to a constraint solver at runtime, otherwise performed each time a transition fires, and thereby significantly improves the performance of generated coordination code.

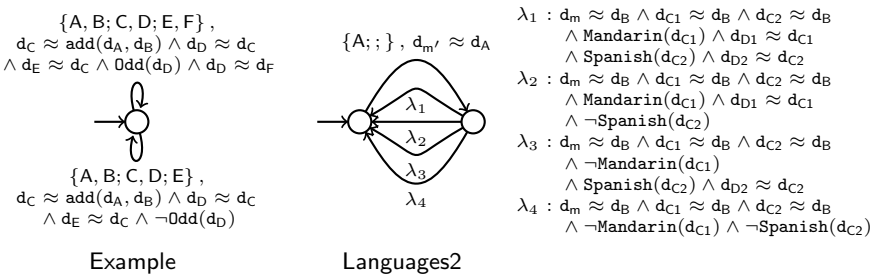
1 Introduction

Context. A promising application domain for coordination languages is programming protocols among threads in multicore applications. One reason for this is a classical software engineering advantage: coordination languages typically provide high-level constructs and abstractions that more easily compose into correct—with respect to programmers' intentions—protocol specifications than do conventional lower-level synchronization mechanisms (e.g., locks or semaphores). However, not only do coordination languages simplify programming protocols, but their high-level constructs and abstractions also leave more room for compilers to perform optimizations that conventional language compilers cannot apply. Eventually, sufficiently smart compilers for coordination languages should be capable of generating code (e.g., in Java or in C) that can compete with carefully hand-crafted code. Preliminary evidence for feasibility of this goal appears elsewhere [1,2]. A crucial step toward adoption of coordination languages for multicore programming, then, is the development of such compilers.

To study the performance advantages of using coordination languages for multicore programming, in ongoing work, we are developing compilation technology for *constraint automata* (CA) [3]. Constraint automata are a general coordination model based on finite automata on infinite words. Every CA models



(a) Abstract/parametric primitives

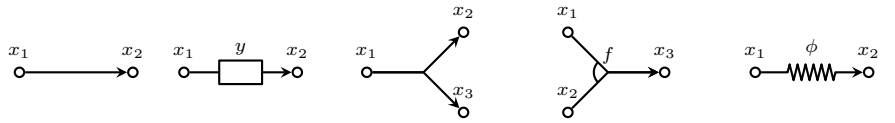


(b) Concrete/instantiated composites

Fig. 1: Example CA. Semicolons separate input/internal/output ports.

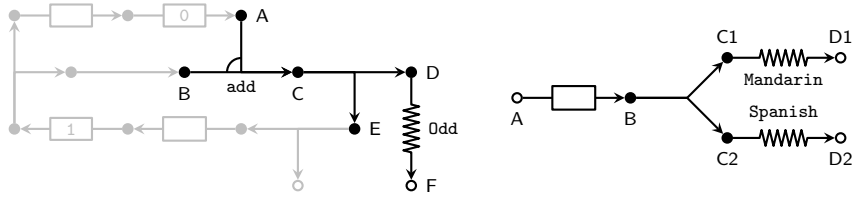
the behavior of a single coordinator; a product operator on CA models the synchronous composition of such coordinators (useful to construct complex coordinators out of simpler ones). Structurally, a CA consists of a finite set of states, a finite set of transitions, a set of directed *ports*, and a set of local *memory cells*. Ports represent the boundary/interface between a coordinator and its coordinated agents (e.g., computation threads). Such agents can perform blocking I/O-operations on ports: a coordinator's input ports admit `put` operations, while its output ports admit `get` operations. Memory cells represent internal buffers in which a coordinator can temporarily store data items. Different from classical automata, transition labels of CA consist of two elements: a set of ports, called a *synchronization constraint*, and a logical formula over ports and memory cells, called a *data constraint*. A synchronization constraint specifies which ports need an I/O-operation for its transition to fire (i.e., those ports synchronize in that transition and their pending I/O-operations complete), while a data constraint specifies which particular data items those I/O-operations may involve. Figure 1 already shows some examples; details follow shortly. Essentially, a CA constrains *when* I/O-operations may complete on *which* ports. As such, CA quite literally materialize Wegner's definition of coordination as "constrained interaction" [4].

Given a library of "small" CA, each of which models a primitive coordinator with its own local behavior, programmers can compositionally construct "big" CA, each of which models a composite coordinator with arbitrarily complex global behavior, fully tailored to the needs of these programmers and their programs. Our current CA-compilers can subsequently generate Java/C code.



$\mathsf{Sync}(x_1;;x_2)$ $\mathsf{Fifo}[y](x_1;x_2)$ $\mathsf{Repl}(x_1;;x_2,x_3)$ $\mathsf{BinOp}_f(x_1,x_2;;x_3)$ $\mathsf{Filter}_R(x_1;;x_2)$

(a) Abstract/parametric primitives (cf. Figure 1a)



Example (in black) and Fibonacci (in black+gray) Languages2

(b) Concrete/instantiated composites (cf. Figure 1b)

Fig. 2: Reo syntax for the CA in Figure 1. White vertices represent input/output ports; black vertices represent internal ports.

Afterward, these compilers either automatically blend their generated code into programs' computation code or provide programmers the opportunity to do this manually. At run-time, the code generated for a big CA (i.e., a composite coordinator) executes a state machine that simulates that CA, repeatedly firing transitions as computation threads perform I/O-operations. Straightforward as this may seem, one needs to overcome a number of serious issues before this approach can yield practically useful code. Most significantly, these issues include exponential explosion of the number of states or transitions of CA, and oversequentialization or overparallelization of generated code. We have already reported our work on these issues along with promising results elsewhere [5,6,1,7].

Instead of programming with CA directly, one can adopt a more programmer-friendly syntax for which CA serve as semantics. In our work, for instance, we adopted the syntax of Reo [8,9], a graphical calculus of channels. Figure 2 already shows some examples; details follow shortly. (Other CA syntaxes beside Reo exist though [10,11,12,13], which may be at least as programmer-friendly.)

Problem. To fire a transition at run-time, code generated for a CA must evaluate the data constraint of that transition: it must ensure that the data involved in blocking I/O-operations pending on the transition's ports satisfy that constraint.

A straightforward evaluation of data constraints requires expensive calls to a constraint solver. Such calls cause high run-time overhead. In particular, because transitions fire sequentially, avoiding constraint solving to reduce this sequential bottleneck is crucial in getting good performance for the whole program.

Contribution & organization. In this paper, we introduce a technique for statically translating a data constraint, off-line at compile-time, into a *data command*: an imperative implementation (in a sequential language with assignment and guarded failure) of a data constraint that avoids expensive calls to a constraint solver at run-time. As with our previous optimization techniques [5,1,7], we prove that the translation in this paper is sound and complete. Such correctness proofs are important, because they ensure that our compilation approach guarantees *correctness-by-construction* (e.g., model-checking results obtained for pre-optimized CA also hold for their generated, optimized implementations). We also give preliminary performance results to show our optimization's potential.

In Section 2, we discuss data constraints and CA. In Sections 3 and 4, we discuss our translation algorithm. In Section 5, we give preliminary performance results. Section 6 concludes this paper. Formal definitions and detailed proofs of Theorems 1 and 2 appear in Appendices A and B.

2 Preliminaries: Data Constraints, Constraint Automata

Data constraints. Let $\mathbb{D}$ denote the finite set of all *data items*, typically ranged over by d. Let $\mathsf{nil} \notin \mathbb{D}$ denote a special object for the *empty data item*. Let $\mathbb{P}$ denote the finite set of all *places* where data items can reside, typically ranged over by x or y; every place models either a port or a memory cells. We model atomic coordination steps—the letters in the alphabet of CA—with elements from the partial function space $\mathbb{D}\text{ISTR} = \mathbb{P} \rightharpoonup \mathbb{D} \cup \{\mathsf{nil}\}$, called *distributions*, typically ranged over by δ. Informally, a distribution δ associates every place x involved in the step modeled by δ with the data item $\delta(x)$ observable in x.

Let $\mathbb{F} = \bigcup\{\mathbb{D}^k \to \mathbb{D} \mid k > 0\}$ and $\mathbb{R} = \bigcup\{\wp(\mathbb{D}^k) \mid k > 0\}$ denote the sets of all *data functions* and *data relations* of finite arity. Let $\mathbb{D}\text{ATA}$, $\mathbb{F}\text{UN}$, and $\mathbb{R}\text{EL}$ denote the sets of all *data item symbols*, *data function symbols* and *data relation symbols*, typically ranged over by d, f, and R. Let $\mathsf{arity} : \mathbb{F}\text{UN} \cup \mathbb{R}\text{EL} \to \mathbb{N}_+$ denote a function that associates every data function/relation symbol with its positive arity. Let $\mathcal{I} : \mathbb{D}\text{ATA} \cup \mathbb{F}\text{UN} \cup \mathbb{R}\text{EL} \to \mathbb{D} \cup \mathbb{F} \cup \mathbb{R}$ denote a bijection that associates every data item/function/relation symbol with its interpretation. A *data term* is a word t generated by the following grammar:

$$t \quad ::= \quad \mathtt{d}_x \mid \mathtt{nil} \mid d \mid f(t_1, \ldots, t_k) \ \textbf{if} \ \mathsf{arity}(f) = k$$

Let $\mathbb{T}\text{ERM}$ denote the set of all data terms. Let $\mathsf{eval} : \mathbb{D}\text{ISTR} \times \mathbb{T}\text{ERM} \to \mathbb{D} \cup \{\mathsf{nil}\}$ denote a function that evaluates every data term t to a data item $\mathsf{eval}_\delta(t)$ under distribution δ. For instance, $\mathsf{eval}_\delta(\mathtt{d}_x) = \delta(x)$—if δ is defined for x—and $\mathsf{eval}_\delta(d) = \mathcal{I}(d)$. If a data term t contains $\mathtt{nil}$ or $\mathtt{d}_x$ for some $x \notin \text{Dom}(\delta)$, we have $\mathsf{eval}_\delta(t) = \mathsf{nil}$. This ensures that eval is a total function, even though the deltas in $\mathbb{D}\text{ISTR}$ are partial functions. See also Definition 7 in Appendix A. We call a term of the form $\mathtt{d}_x$ a *free variable*. Intuitively, $\mathtt{d}_x$ represents the data item residing in place x. Let $\mathsf{Free} : \mathbb{T}\text{ERM} \to \wp(\mathbb{T}\text{ERM})$ denote a function that maps every data term t to its set of free variables.

A *data constraint* is a word ϕ generated by the following grammar:

$$\begin{array}{lcl} a & ::= & \bot \mid \top \mid t \approx t \mid t \not\approx \texttt{nil} \mid R(t_1\,,\,\ldots\,,\,t_k)\ \textbf{if}\ \mathsf{arity}(R) = k \\ \phi & ::= & a \mid \neg\phi \mid \phi \vee \phi \mid \phi \wedge \phi \end{array}$$

Let $\mathbb{DC}$ denote the set of all data constraints. We often call $t_1 \approx t_2$ atoms *equalities*. We define the semantics of data constraints over distributions. Let $\models^{\text{dc}} \subseteq \mathbb{D}\text{ISTR} \times \mathbb{DC}$ denote the satisfaction relation on data constraints. Its definition is standard for $\bot$ (contradiction), $\top$ (tautology), $\neg$ (negation), $\vee$ (disjunction), and $\wedge$ (conjunction). For other atoms, we have the following:

$$\begin{array}{lll} \delta \models^{\text{dc}} t_1 \approx t_2 & \textbf{iff}\ \mathsf{eval}_\delta(t_1) = \mathsf{eval}_\delta(t_2) \neq \mathsf{nil} & \\ \delta \models^{\text{dc}} t \not\approx \texttt{nil} & \textbf{iff}\ \mathsf{eval}_\delta(t) \neq \mathsf{nil} & \textit{(i.e., notation for } \delta \models^{\text{dc}} t \approx t\textit{)} \\ \delta \models^{\text{dc}} R(t_1\,,\,\ldots\,,\,t_k) & \textbf{iff}\ \mathcal{I}(R)(\mathsf{eval}_\delta(t_1)\,,\,\ldots\,,\,\mathsf{eval}_\delta(t_k)) & \end{array}$$

In the second rule, if a t_i evaluates to nil, the right-hand side is undefined—hence false—because the domain of data relation $\mathcal{I}(R)$ excludes nil. If $\delta \models^{\text{dc}} \phi$, we call δ a *solution* for ϕ. Let $[\![\cdot]\!] : \mathbb{DC} \to \wp(\mathbb{D}\text{ISTR})$ denote a function that associates every data constraint ϕ with its meaning $[\![\phi]\!] = \{\delta \mid \delta \models^{\text{dc}} \phi\}$. We write $\phi \Rightarrow \phi'$ iff $[\![\phi]\!] \subseteq [\![\phi']\!]$. We also extend function Free from data terms to data constraints.

Constraint automata. A constraint automaton (CA) is a tuple $(Q\,,\,\mathcal{X}\,,\,\mathcal{Y}\,,\,\longrightarrow\,,\,\imath)$ with Q a set of states, $\mathcal{X} \subseteq \mathbb{P}$ a set of ports, $\mathcal{Y} \subseteq \mathbb{P}$ a set of memory cells, $\longrightarrow \subseteq Q \times (\wp(\mathcal{X}) \times \mathbb{DC}) \times Q$ a transition relation labeled with pairs $(X\,,\,\phi)$, and $\imath \in Q$ an initial state. For every label $(X\,,\,\phi)$, no ports outside X may occur in ϕ. Set $\mathcal{X}$ consists of three disjoint subsets of input ports $\mathcal{X}_{\text{in}}$, *internal ports* $\mathcal{X}_{\text{int}}$, and output ports $\mathcal{X}_{\text{out}}$. We call a CA for which $\mathcal{X}_{\text{int}} = \emptyset$ a *primitive*; otherwise, we call it a *composite*.

Although generally important, we skip the definition of the product operator on CA, because it does not matter in this paper. Every CA accepts infinite sequences of distributions [3]: $(Q\,,\,\mathcal{X}\,,\,\mathcal{Y}\,,\,\longrightarrow\,,\,\imath)$ accepts $\delta_0\delta_1\cdots$ if an infinite sequence of states $q_0q_1\cdots$ exists such that $q_0 = \imath$ and for all $i \geq 0$, a transition $(q_i\,,\,(X\,,\,\phi)\,,\,q_{i+1})$ exists such that $\text{Dom}(\delta_i) = X$ and $\delta_i \models^{\text{dc}} \phi$.

Without loss of generality, we assume that all data constraints occur in disjunctive normal form. Moreover, because replacing a transition $(q\,,\,(X\,,\,\phi_1 \vee \phi_2)\,,\,q')$ with two transitions $(q\,,\,(X\,,\,\phi_1)\,,\,q')$ and $(q\,,\,(X\,,\,\phi_2)\,,\,q')$ preserves behavioral congruence on CA [3], without loss of generality, we assume that the data constraint in every label is a conjunction of *literals*, typically ranged over by ℓ.

Figure 1a shows example primitives; Figure 2a shows their Reo syntax. Sync models a synchronous channel from an input x_1 to an output x_2. Fifo models an asynchronous channel with a 1-capacity buffer y from x_1 to x_2. Repl models a coordinator that, in each of its atomic coordination steps, replicates the data item on x_1 to both x_2 and x_3. BinOp models a coordinator that, in each of its atomic coordination steps, applies operation f to the data items on x_1 and x_2 and passes the result to x_3. Filter models a lossy synchronous channel from x_1 to x_2; data items pass this channel only if they satisfy predicate R.

Figure 1b shows example composites; Figure 2b shows their Reo syntax.

Example—our running example in this paper—consists of instantiated primitives $\mathsf{BinOp}_{\texttt{add}}(\mathsf{A},\mathsf{B};;\mathsf{C})$, $\mathsf{Repl}(\mathsf{C};;\mathsf{D},\mathsf{E})$, and $\mathsf{Filter}_{\texttt{Odd}}(\mathsf{D};;\mathsf{F})$, where `add` and `Odd` have the obvious interpretation. In each of its atomic coordination steps, if the sum of the data items (supposedly integers) on its inputs A and B is odd, Example passes this sum to its outputs E and F. Otherwise, if the sum is even, Example passes this value only to E. Figure 2b shows that Example constitutes Fibonacci. Fibonacci coordinates two consumers by generating the Fibonacci sequence. Whenever Fibonacci generates an even number, it passes that number to *only one* consumer; whenever it generates an odd number, it passes that number to *both* consumers. Finally, Languages2 consists of instantiated primitives $\mathsf{Fifo}[\mathsf{m}](\mathsf{A};;\mathsf{B})$, $\mathsf{Repl}(\mathsf{B};;\mathsf{C1},\mathsf{C2})$, $\mathsf{Filter}_{\texttt{Mandarin}}(\mathsf{C1};;\mathsf{D1})$, and $\mathsf{Filter}_{\texttt{English}}(\mathsf{C2};;\mathsf{D2})$. Languages2 coordinates a producer and two consumers. If the producer `puts` a Mandarin (resp. English) data item on input A, Languages2 asynchronously passes this data item only to the consumer on output D1 (resp. D2). Languages2 easily generalizes to Languages$i$, for i different languages; we do so in Section 5.

3 From Data Constraints to Data Commands

At run-time, compiler-generated code executes in one or more CA-*threads*, each of which runs a state machine that simulates a CA. (We addressed the challenge of deciding the number of CA-threads elsewhere [5,6,7].) The *context* of a CA-thread is the collection of `put`/`get` operations on implementations of its input/output ports, performed by computation threads. Every time the context of a CA-thread changes, that CA-thread examines whether this change enables a transition in its current state q: for each transition $(q\,,\,(X\,,\,\phi)\,,\,q')$, it checks whether every port $x \in X$ has a pending I/O-operation and if so, whether the data items involved in the pending `put` operations and the current content of memory cells can constitute a solution for ϕ. For the latter, the CA-thread calls a constraint solver, which searches for a distribution δ such that $\delta \models^{\text{dc}} \phi$ and $\delta_{\text{init}} \subseteq \delta$, where:

$$\begin{aligned}\delta_{\text{init}} = \{&x \mapsto d \mid \text{the } \texttt{put} \text{ pending on input port } x \text{ involves data item } d\} \\ \cup\, \{&y \mapsto d \mid \text{memory cell } y \text{ contains data item } d\}\end{aligned} \tag{1}$$

Constraint solving over a finite discrete domain (e.g., $\mathbb{D}$) is NP-complete [14]. Despite carefully and cleverly optimized backtracking searches, using general-purpose constraint solving techniques for solving a data constraint ϕ inflicts not only overhead proportional to ϕ's size but also a constant overhead for preparing, making, and processing the result of the call itself. Although we generally cannot escape using conventional constraint solving techniques, a practically relevant class of data constraints exists for which we can: the data constraints of many CA in practice are in fact declarative specifications of sequences of imperative instructions (including those in Figure 1). In this section, we therefore develop a technique for statically translating such a data constraint ϕ, off-line at compile-time, into a *data command*: a little imperative program that computes a distribution δ such that $\delta \models^{\text{dc}} \phi$ and $\delta_{\text{init}} \subseteq \delta$, without conventional constraint solving hassle. Essentially, we formalize and automate what a programmer would

do if he/she were to write an imperative implementation of a declarative specification expressed as a data constraint. By the end of Section 4, we make the class of data constraints supported by our translation precise.

3.1 Data Commands

A data command is a word P generated by the following grammar:

$$P \quad ::= \quad \texttt{skip} \mid x \texttt{ := } t \mid \texttt{if } \phi \texttt{ -> } P \texttt{ fi} \mid P \texttt{ ; } P$$

(We often write "value of x" instead of "the data item assigned to x".)

We adopt the following operational semantics of Apt et al. [15]. True to the idea that data commands compute solutions for data constraints, the *state* that a data command executes in is either a function from places to data items—a distribution!—or the distinguished symbol fail, which represents abnormal termination. A *configuration* is a pair of a data command and a state to execute that data command in. Let ε denote the *empty data command*, and equate $\varepsilon \texttt{ ; } P$ with P. Let $\delta[x := \mathsf{eval}_\delta(t)]$ denote an update of δ as usual. The following rules define the transition relation on configurations, denoted by $\Longrightarrow$.

$$\frac{}{(\texttt{skip}\,,\,\delta) \Longrightarrow (\varepsilon\,,\,\delta)} \qquad \frac{}{(x \texttt{ := } t\,,\,\delta) \Longrightarrow (\varepsilon\,,\,\delta[x := \mathsf{eval}_\delta(t)])}$$

$$\frac{\delta \models^{\mathrm{dc}} \phi}{(\texttt{if } \phi \texttt{ -> } P \texttt{ fi}\,,\,\delta) \Longrightarrow (P\,,\,\delta)} \qquad \frac{\delta \not\models^{\mathrm{dc}} \phi}{(\texttt{if } \phi \texttt{ -> } P \texttt{ fi}\,,\,\delta) \Longrightarrow (\varepsilon\,,\,\mathsf{fail})}$$

$$\frac{(P\,,\,\delta) \Longrightarrow (P'\,,\,\delta')}{(P \texttt{ ; } P''\,,\,\delta) \Longrightarrow (P' \texttt{ ; } P''\,,\,\delta')}$$

Note that $\texttt{if } \phi \texttt{ -> } P \texttt{ fi}$ commands are *failure* statements rather than *conditional* statements: if the current state violates the *guard* ϕ, execution abnormally terminates. The *partial correctness semantics*, which ignores abnormal termination, of a data command P in a state δ is the set of final states $\mathcal{M}(P\,,\,\{\delta\}) = \{\,\delta' \mid (P\,,\,\delta) \Longrightarrow^* (\varepsilon\,,\,\delta')\}$; its *total correctness semantics* is the set consisting of fail or its final states $\mathcal{M}_{\mathrm{tot}}(P\,,\,\{\delta\}) = \{\mathsf{fail} \mid (P\,,\,\{\delta\}) \Longrightarrow^* (\varepsilon\,,\,\mathsf{fail})\} \cup \mathcal{M}(P\,,\,\{\delta\})$. Because data commands are deterministic, $|\mathcal{M}(P\,,\,\{\delta\})| \leq 1$ and $|\mathcal{M}_{\mathrm{tot}}(P\,,\,\{\delta\})| = 1$.

Shortly, to prove the correctness of our translation from data constraints to data commands, we use Hoare logic [16], where *triples* $\{\phi\}\ P\ \{\phi'\}$ play a central role. In such triples, ϕ characterizes the set of input states, P denotes the data command to execute in those states, and ϕ' characterizes the set of output states. A triple $\{\phi\}\ P\ \{\phi'\}$ holds in the sense of partial (resp. total) correctness, if $\mathcal{M}(P\,,\,[\![\phi]\!]) \subseteq [\![\phi']\!]$ (resp. $\mathcal{M}_{\mathrm{tot}}(P\,,\,[\![\phi]\!]) \subseteq [\![\phi']\!]$). To prove properties of data commands, we use the following sound proof systems for partial (resp. total) correctness, represented by $\vdash$ (resp. $\vdash_{\mathrm{tot}}$) and adopted with slight adaptation—a cosmetic

change in the rule for assignment—from Apt et al. [15].

$$\frac{}{\vdash \{\phi\}\ \texttt{skip}\ \{\phi\}} \qquad \frac{\vdash \{\phi'\}\ P\ \{\phi''\}\ \textbf{and}\ \phi \Rightarrow \phi'\ \textbf{and}\ \phi'' \Rightarrow \phi'''}{\vdash \{\phi\}\ P\ \{\phi'''\}} \qquad \frac{\vdash \{\phi\}\ P\ \{\phi'\}\ \textbf{and}\ \vdash \{\phi'\}\ P'\ \{\phi''\}}{\vdash \{\phi\}\ P\ ;\ P'\ \{\phi''\}}$$

$$\frac{}{\vdash \{\phi[\mathtt{d}_x := t]\}\ x\ \texttt{:=}\ t\ \{\phi\}}$$

$$\frac{\vdash \{\phi \wedge \phi_\mathrm{g}\}\ P\ \{\phi'\}}{\vdash \{\phi\}\ \texttt{if}\ \phi_\mathrm{g}\ \texttt{->}\ P\ \texttt{fi}\ \{\phi'\}} \qquad \frac{\phi \Rightarrow \phi_\mathrm{g}\ \textbf{and}\ \vdash_\mathrm{tot} \{\phi\}\ P\ \{\phi'\}}{\vdash_\mathrm{tot} \{\phi\}\ \texttt{if}\ \phi_\mathrm{g}\ \texttt{->}\ P\ \texttt{fi}\ \{\phi'\}}$$

The first four rules apply not only to $\vdash$ but also to $\vdash_\mathrm{tot}$. We use $\vdash$ to prove the soundness of our upcoming translation; we use $\vdash_\mathrm{tot}$ to prove its completeness.

3.2 Precedence

Recall the following typical data constraint over ports A, B, C, D, and E, where A and B are inputs, from Example in Figure 1b (its lower transition):

$$\phi = \mathtt{d}_\mathsf{C} \approx \mathtt{add}(\mathtt{d}_\mathsf{A}, \mathtt{d}_\mathsf{B}) \wedge \mathtt{d}_\mathsf{D} \approx \mathtt{d}_\mathsf{C} \wedge \mathtt{d}_\mathsf{E} \approx \mathtt{d}_\mathsf{C} \wedge \neg \mathtt{Odd}(\mathtt{d}_\mathsf{D}) \tag{2}$$

To translate data constraints to data commands, the idea is to enforce equalities, many of which occur in practice, with assignments and to check all remaining literals with failure statements. In the case of ϕ, for instance, we first assign the data items involved in their pending `put` operations to A and B, whose symbols are denoted by $\mathcal{I}^{-1}(\delta_\mathrm{init}(\mathsf{A}))$ and $\mathcal{I}^{-1}(\delta_\mathrm{init}(\mathsf{B}))$, with δ_init as defined in (1), page 6. Next, we assign the evaluation of $\mathtt{add}(\mathtt{d}_\mathsf{A}, \mathtt{d}_\mathsf{B})$ to C. The order in which we subsequently assign the value of C to D and E does not matter. After the assignment to D, we check $\neg\mathtt{Odd}(\mathtt{d}_\mathsf{D})$ with a failure statement. The following data command corresponds to one possible order of the last three steps.

$$\begin{aligned} P = \mathsf{A}\ \texttt{:=}\ \mathcal{I}^{-1}(\delta_\mathrm{init}(\mathsf{A}))\ \texttt{;}\ \mathsf{B}\ \texttt{:=}\ \mathcal{I}^{-1}(\delta_\mathrm{init}(\mathsf{B}))\ \texttt{;}\ \mathsf{C}\ \texttt{:=}\ \mathtt{add}(\mathtt{d}_\mathsf{A}, \mathtt{d}_\mathsf{B})\ \texttt{;} \\ \mathsf{D}\ \texttt{:=}\ \mathtt{d}_\mathsf{C}\ \texttt{;}\ \texttt{if}\ \neg\mathtt{Odd}(\mathtt{d}_\mathsf{D})\ \texttt{->}\ \texttt{skip}\ \texttt{fi}\ \texttt{;}\ \mathsf{E}\ \texttt{:=}\ \mathtt{d}_\mathsf{C} \end{aligned}$$

If execution of P in an empty initial state successfully terminates, the resulting final state δ should satisfy ϕ (soundness). Moreover, if a δ' exists such that $\delta' \overset{\mathrm{dc}}{\models} \phi$ and $\delta_\mathrm{init} \subseteq \delta'$, execution of P should successfully terminate (completeness).

Soundness and completeness crucially depend on the order in which assignments and failure statements occur in P. For instance, changing the order of $\mathsf{D}\ \texttt{:=}\ \mathtt{d}_\mathsf{C}$ and $\texttt{if}\ \neg\mathtt{Odd}(\mathtt{d}_\mathsf{D})\ \texttt{->}\ \texttt{skip}\ \texttt{fi}$ yields a data command whose execution always fails (because D does not have a value yet on evaluating the guard of the failure statement). Such a data command is trivially sound but incomplete. Another complication is that not every equality can become an assignment. In a first class of cases, no operand matches $\mathtt{d}_x$. An example is $\mathtt{add}(\mathtt{d}_\mathsf{A}, \mathtt{d}_\mathsf{B}) \approx \mathtt{mult}(\mathtt{d}_\mathsf{A}, \mathtt{d}_\mathsf{B})$: this equality should become a failure statement, because neither of its two operands can be assigned to the other. In a second class of cases, multiple equality literals have an operand that matches $\mathtt{d}_x$. An example is $\mathsf{C} \approx \mathtt{add}(\mathtt{d}_\mathsf{A}, \mathtt{d}_\mathsf{B}) \wedge \mathsf{C} \approx \mathtt{mult}(\mathtt{d}_\mathsf{A}, \mathtt{d}_\mathsf{B})$: only one of these equalities should become an assignment, while the other should become a failure statement, to avoid

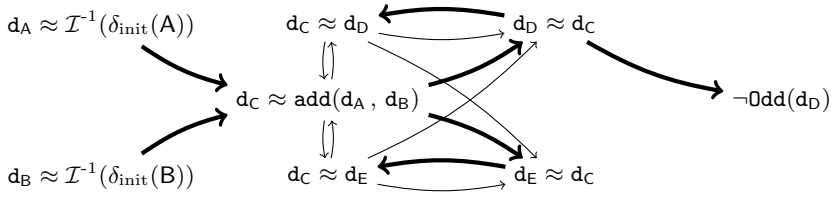


Fig. 3: Fragment of a digraph for an example precedence relation $\prec_L$ (e.g., without loops and without $\mathtt{add}(\mathtt{d_A}, \mathtt{d_B}) \approx \mathtt{d_C}$, for simplicity). An arc (ℓ, ℓ') corresponds to $\ell \prec_L \ell'$. Bold arcs represent a strict partial order extracted from $\prec_L$.

conflicting assignments to C.

To deal with these complications, we define a *precedence relation* on literals that formalizes their dependencies. Recall that the data constraint in every transition label (X, ϕ) is a conjunction of literals. Let L_ϕ denote the set of literals in ϕ, and let $X_{\text{in}} \subseteq X$ denote the set of *input places* (i.e., input ports and memory cells) involved in the transition. From L_ϕ and X_{in}, we construct a set of literals L to account for (i) symmetry of $\approx$ and (ii) the initial values of input places.

$$L = L_\phi \cup \{t_2 \approx t_1 \mid t_1 \approx t_2 \in L_\phi\} \cup \{\mathtt{d}_x \approx \mathcal{I}^{-1}(\delta_{\text{init}}(x)) \mid x \in X_{\text{in}}\} \tag{3}$$

Obviously, $\delta \models^{\text{dc}} \bigwedge L$ implies $\delta \models^{\text{dc}} \phi$ for all δ (i.e., extending L_ϕ to L is sound). Now, let $\prec_L$ denote the precedence relation on L defined by the following rules:

$$\frac{\mathtt{d}_x \approx t\,,\ \ell \in L \ \textbf{ and } \ \mathtt{d}_x \in \mathsf{Free}(\ell)}{\mathtt{d}_x \approx t \prec_L \ell} \qquad \frac{\ell_1 \prec_L \ell_2 \prec_L \ell_3 \ \textbf{ and } \ \ell_2 \notin \{\ell_1\,, \ell_3\}}{\ell_1 \prec_L \ell_3} \tag{4}$$

Informally, $\mathtt{d}_x \approx t \prec_L \ell$ means that assignment $x := t$ must precede ℓ (i.e., ℓ depends on x). Note also that the first rule deals with the first class of equalities-that-cannot-become-assignments; shortly, we comment on the second class.

For the sake of argument—generally, this is *not* the case—suppose that $\prec_L$ is a strict partial order on L. In that case, we can linearize $\prec_L$ to a total order $<$ on L (i.e., embedding $\prec_L$ into $<$ such that $\prec_L \subseteq <$) with a topological sort on the digraph $(L, \prec_L)$ [17,18]. Intuitively, such a linearization gives us an order in which we can translate literals in L to data commands in a sound and complete way. In Section 3.3, we give an algorithm for doing so and indeed prove its correctness. Problematically, however, $\prec_L$ is generally not a strict partial order on L: it is generally neither asymmetric nor irreflexive (i.e., graph-theoretically, it contains cycles). For instance, Figure 3 shows a fragment of the digraph $(L, \prec_L)$ for ϕ in (2), page 8, which contains cycles. For now, we defer this issue to Section 4, because it forms a concern orthogonal to our translation algorithm and its correctness. Until then, we simply assume the existence of a procedure for extracting a strict partial order from $\prec_L$, represented by bold arcs in Figure 3.[1]

[1] Another minor issue is that if we run our translation algorithm on the strict partial order extracted from $\prec_L$, we end up with both the assignment $\mathsf{D} := \mathtt{d_C}$ and the

Henceforth, we assume that every $\mathtt{d}_{x_i} \approx t_i$ literal precedes all differently shaped literals in a linearization of $\prec_L$. Although this assumption is conceptually unnecessary, it simplifies some of our notation and proofs. Formally, we can enforce it by adding a third rule to the definition of $\prec_L$:

$$\frac{\mathtt{d}_x \approx t\,,\ \ell \in L \ \textbf{ and } \ \big[\ell \neq \mathtt{d}_{x'} \approx t' \ \textbf{ for all } \ x'\,,\ t'\big]}{\mathtt{d}_x \approx t \prec_L \ell} \tag{5}$$

Proposition 1. *The rule in (5) introduces no cycles.*

Proof (Sketch). Reasoning toward a contradiction, suppose the third rule introduces a cycle by inducing $\mathtt{d}_x \approx t \prec_L \ell'$. Then, a path exists from ℓ' to $\mathtt{d}_x \approx t$. Then, $\ell' \prec_L \ell''$ for some ℓ''. Then, by inspecting the premises of all three rules, ℓ' must match $\mathtt{d}_{x'} \approx t'$, but the premise of the third rule excludes this. □

3.3 Algorithm

We start by stating the precondition of our translation algorithm. Suppose that L as defined in (3), page 9, contains n $\mathtt{d}_x \approx t$ literals and m differently shaped literals. Let $\prec_L$ denote a strict partial order on L such that for every $\mathtt{d}_x \approx t \in L$ and for every $\mathtt{d}_y \in \mathsf{Free}(t)$, a $\mathtt{d}_y \approx t'$ literal precedes $\mathtt{d}_x \approx t$ according to $\prec_L$. Then, let $\ell_1 < \cdots < \ell_n < \ell_{n+1} < \cdots < \ell_{n+m}$ denote a linearization of $\prec_L$, where $\ell_i = \mathtt{d}_{x_i} \approx t_i$ for all $1 \leq i \leq n$. The three rules of $\prec_L$ in Section 3.2 induce precedence relations for which all previous conditions hold, *except* that $\prec_L$ does not necessarily denote a strict partial order; we address this issue in the next section. The previous conditions aside, we also assume $\{\mathtt{d}_{x_1}\,,\ \ldots\,,\ \mathtt{d}_{x_n}\} = \bigcup\{\mathsf{Free}(\ell_i) \mid 1 \leq i \leq n+m\}$. This extra condition means that for every free variable $\mathtt{d}_{x_i}$ in every literal in L, a $\mathtt{d}_{x_i} \approx t_i$ literal exists in the linearization. If this condition fails, some places can get a value only through search—exactly what we try to avoid—and not through assignment. In such cases, the data constraint is underspecified, and our translation algorithm is fundamentally inapplicable. Finally, we trivially assume that `nil` does not occur syntactically in any literal. A formal definition of this precondition appears in Figure 10 in Appendix A.

Figure 4 shows our algorithm. It first loops over the first n (according to $<$) $\mathtt{d}_x \approx t$ literals. If an assignment for x already exists in data command P, the algorithm translates $\mathtt{d}_x \approx t$ to a failure statement; if not, it translates $\mathtt{d}_x \approx t$ to an assignment. This approach resolves issues with the second class of equalities-that-cannot-become-assignments. After the first loop, the algorithm uses a second loop to translate the remaining m differently shaped literals to failure statements. The algorithm runs in time linear in $n+m$, and it clearly terminates.

unnecessary failure statement `if` $\mathtt{d}_C \approx \mathtt{d}_D$ `->` `skip` `fi`. After all, the digraph contains both $\mathtt{d}_D \approx \mathtt{d}_C$ and $\mathtt{d}_C \approx \mathtt{d}_D$, the second of which was added while computing $L^{\approx}_{\phi}$ to account for the symmetry of $\approx$. Generally, such symmetric literals result either in one assignment and one failure statement or in two failure statements. (One can easily prove that symmetric literals never result in two assignments.) In both cases, one can safely remove one of the failure statements, because successful termination of the remaining statement already accounts for the removed failure statement.

```
P ← skip
i ← 1
while i ≤ n do
    if d_{x_i} ∈ {d_{x_j} | 1 ≤ j < i} then
        P ← P ; if d_{x_i} ≈ t_i -> skip fi
    else
        P ← P ; x_i := t_i
    i ← i + 1
while i ≤ n + m do
    P ← P ; if ℓ_i -> skip fi
    i ← i + 1
```

Fig. 4: Algorithm to translate data constraints to data commands

The desired postcondition of the algorithm consists of its soundness and completeness. We define soundness as $\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{n+m}\}$: after running the algorithm, execution of data command P yields a state that satisfies all literals in L on successful termination. We define completeness as $\big[\big[\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m}$ **implies** $\vdash_{\text{tot}} \{\top\}\ P\ \{\top\}\big]$ **for all** $\delta'\big]$: after running the algorithm, if a distribution δ' exists that satisfies all literals in L, data command P successfully terminated. Although soundness subsequently guarantees that the final state δ satisfies all literals in L, generally, $\delta \neq \delta'$. We use a different proof system for soundness (partial correctness, $\vdash$) than for completeness (total correctness, $\vdash_{\text{tot}}$).

Theorem 1 (Appendix A, Theorem 3). *The algorithm is sound and complete.*

Proof (Sketch). We prove soundness and completeness separately.

Soundness We start by arguing that $\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_i\}$ holds after every iteration of the first loop. For $1 \leq i \leq n$, after doing an assignment $x_i := t_i$ in a state δ, literal $\ell_i = \mathtt{d}_{x_i} \approx t_i$ holds in δ if all free variables in t_i have a non-nil value. (Otherwise, t_i evaluates to nil, which the definition of $\models^{\text{dc}}$ forbids.)

Reasoning toward a contradiction, suppose that some free variable $\mathtt{d}_y$ in t_i has a nil value. Then, because no assignment assigns nil, no $y := t$ assignment has occurred previously. But because $\mathtt{d}_y \in \mathsf{Free}(t_i)$, a literal $\mathtt{d}_y \approx t \in L$ (by the precondition of the algorithm) exists that precedes $\mathtt{d}_{x_i} \approx t_i$. Consequently, $\mathtt{d}_y \approx t$ precedes $\mathtt{d}_{x_i} \approx t_i$ also in the linearization of $\prec_L$, and so, a $y := t$ assignment must have occurred previously. Hence, $\mathtt{d}_y$ in fact has a non-nil value (namely, the evaluation of t).

Thus, $\ell_i = \mathtt{d}_{x_i} \approx t_i$ holds in δ after its update with $x_i := t_i$. Suppose that the preceding literals $\mathtt{d}_{x_j} \approx t_j$ (for $1 \leq j < i$) held before updating δ. Each of those literals can have become false only if the update overwrote $\mathtt{d}_{x_j}$. In that case, $\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}$. But then, the algorithm did not translate $\mathtt{d}_{x_i} \approx t_i$ to an assignment in the first place but to a failure statement `if` $\mathtt{d}_{x_i} \approx t_i$ `-> skip fi`. If execution of this statement successfully terminates, obviously $\mathtt{d}_{x_i} \approx t_i$ holds, and because it leaves δ unchanged, all preceding literals remain true. Note that the $\vdash$ proof rule for failure statements allows us to *assume* that the guard holds; we do not need to *establish* this yet (cf. completeness below, where we use $\vdash_{\text{tot}}$).

We can inductively repeat the reasoning in the previous paragraphs for all $1 \leq i \leq n$ to conclude that $\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_n\}$ holds after the first loop. The failure statements added in the second loop leave state δ unchanged, meaning that literals that held before executing those statements in δ remain true. Thus, if those statements successfully terminate, $\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{n+m}\}$ holds. □

Completeness Assume that $\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m}$ for some δ'. We start by arguing that $\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\}$ holds after every iteration of the first loop. This means that state δ maps every x_j (for $1 \leq j \leq i$) to the same value as δ' (i.e., $\delta(x_j) = \delta'(x_j)$). For $1 \leq i \leq n$ and $\mathtt{d}_{x_i} \notin \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}$, we know that $\ell_i = \mathtt{d}_{x_i} \approx t_i$ holds in δ after its update with $x_i := t_i$ (see soundness above). By our initial assumption, we also know that $\ell_i = \mathtt{d}_{x_i} \approx t_i$ holds in δ'. Thus, by the definition of $\models^{\text{dc}}$, we conclude $\delta(x_i) = \mathsf{eval}_{\delta}(t_i)$ and $\delta'(x_i) = \mathsf{eval}_{\delta'}(t_i)$.

Because a $\mathtt{d}_y \approx t$ literal precedes $\mathtt{d}_{x_i} \approx t_i$ for all $\mathtt{d}_y \in \mathsf{Free}(t_i)$ (see soundness above), δ maps every such a y to the same value as δ' (i.e., $y = x_j$ for some $1 \leq j < i$). Consequently, and because the interpretation of every function symbol is always a function (including those occurring in t_i), $\mathsf{eval}_{\delta}(t_i) = \mathsf{eval}_{\delta'}(t_i)$. Combining this with the previous intermediate result, the following equation holds: $\delta(x_i) = \mathsf{eval}_{\delta}(t_i) = \mathsf{eval}_{\delta'}(t_i) = \delta'(x_i)$. As before (see soundness above), we can also establish that updating δ with $x_i := t_i$ does not make $\mathtt{d}_{x_j} \approx \mathcal{I}^{-1}(\delta'(x_j))$ literals (for $1 \leq j < i$) that held before the update false.

If $\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}$, we can immediately conclude that $\mathtt{d}_{x_j} \approx \mathcal{I}^{-1}(\delta'(x_j))$ holds in state δ for all $1 \leq j \leq i$ before executing the failure statement `if` $\mathtt{d}_{x_i} \approx t_i$ `-> skip fi` added by the algorithm. To prove that this failure statement also successfully terminates, the $\vdash_{\text{tot}}$ proof rule for failure statements dictates that we must establish—instead of assume (cf. soundness above)—that the guard $\mathtt{d}_{x_i} \approx t_i$ holds in δ. This follows from the fact that $\mathtt{d}_{x_i} \approx t_i$ holds in δ' by our initial assumption, and because δ and δ' map all free variables in $\ell_i = \mathtt{d}_{x_i} \approx t_i$ to the same values. To prove the latter, we can use a similar argument involving the precedence relation and its linearization as before (see soundness above).

We can inductively repeat the reasoning in the previous paragraphs for all $1 \leq i \leq n$ to conclude that $\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}$ holds after the first loop. The failure statements added in the second loop leave state δ unchanged, meaning that the $\mathtt{d}_{x_j} \approx \mathcal{I}^{-1}(\delta'(x_j))$ literals (for $1 \leq j \leq n$) that held before executing those statements in δ remain true. To prove the successful termination of those failure statements, we can use a similar argument as for the failure statements added in the first loop: by our initial assumption, $\delta' \models^{\text{dc}} \ell_i$ for all $n+1 \leq j \leq n+m$, and δ and δ' still map the same free variables to the same values. Thus, $\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}$ holds also after the second loop, and consequently, $\vdash_{\text{tot}} \{\top\}\ P\ \{\top\}$ holds. □

(A detailed proof in Appendix B, Theorem 3, formalizes the previous proof sketch in terms of Hoare logic. The detailed proof has a structure quite different from the previous sketch. The main arguments are, however, the same.)

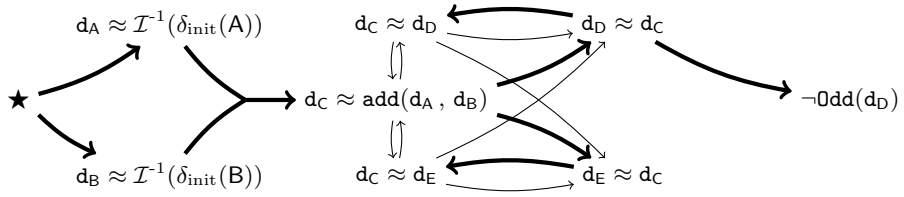


Fig. 5: Fragment of the B-graph corresponding to the digraph in Figure 3 (e.g., without looping B-arcs and without $\mathtt{add}(\mathtt{d_A}, \mathtt{d_B}) \approx \mathtt{d_C}$, for simplicity). Bold B-arcs represent an arborescence.

4 Handling Cycles

Our algorithm assumes that a precedence relation $\prec_L$ as defined in Section 3.2 is a strict partial order. However, this is generally not the case. In this section, we describe a procedure for extracting a strict partial order from $\prec_L$ without losing essential dependencies. We start by adding a distinguished symbol $\bigstar$ to the domain of $\prec_L$, and we extend the definition of $\prec_L$ with the following rules:

$$\frac{\ell \in L \;\textbf{ and }\; \mathsf{Free}(\ell) = \emptyset}{\bigstar \prec_L \ell} \qquad \frac{\mathtt{d}_x \approx t \in L \;\textbf{ and }\; \mathsf{Free}(t) = \emptyset}{\bigstar \prec_L \mathtt{d}_x \approx t} \tag{6}$$

These rules state that literals without free variables (e.g., $\mathtt{d}_x \approx \mathcal{I}^{-1}(\delta_{\text{init}}(x))$) do not depend on other literals. Now, $\prec_L$ is a strict partial order if the digraph $(L \cup \{\bigstar\}, \prec_L)$ is a $\bigstar$-*arborescence*: a digraph consisting of $n-1$ arcs such that each of its n vertices is reachable from $\bigstar$ [19]. Equivalently, in a $\bigstar$-arborescence, $\bigstar$ has no incoming arcs, every other vertex has exactly one incoming arc, and the arcs form no cycles [19]. The first formulation is perhaps most intuitive here: every path from $\bigstar$ to some literal ℓ represents an order in which our algorithm should translate the literals on that path to ensure the correctness of the translation of ℓ. The second formulation simplifies observing that arborescences correspond to strict partial orders (by their cycle-freeness).

A naive approach to extract a strict partial order from $\prec_L$ is to compute a $\bigstar$-arborescence of the digraph $(L \cup \{\bigstar\}, \prec_L)$. Unfortunately, however, this approach generally fails for $\mathtt{d}_x \approx t$ literals where t has more than one free variable. For instance, by definition, every arborescence of the digraph in Figure 3 has only one incoming arc for $\mathtt{d_C} \approx \mathtt{add}(\mathtt{d_A}, \mathtt{d_B})$, even though assignments to both A and B must precede an assignment to C. Because these dependencies exist as two separate arcs, no arborescence of a digraph can capture them. To solve this, we should somehow represent the dependencies of $\mathtt{d_C} \approx \mathtt{add}(\mathtt{d_A}, \mathtt{d_B})$ with a single incoming arc. We can do so by allowing arcs to have multiple tails (i.e., one for every free variable). In that case, we can replace the two separate incoming arcs of $\mathtt{d_C} \approx \mathtt{add}(\mathtt{d_A}, \mathtt{d_B})$ with a single two-tailed incoming arc as in Figure 5. The two tails make explicit that to evaluate an `add`-term, we need values for both its arguments: multiple tails represent a conjunction of dependencies of a literal.[2]

[2] Interestingly, when evaluating a term at run-time, it is not always necessary to

By replacing single-tail-single-head arcs with multiple-tails-single-head arcs, we effectively transform the digraphs considered so far into B-*graphs*, a special kind of hypergraph with only B-*arcs* (i.e., *b*ackward hyperarcs, i.e., hyperarcs with exactly one head) [20]. Deriving a B-graph over literals from a precedence relation as defined in Section 3.2 is generally impossible though: their richer structure makes B-graphs more expressive—they give more information—than digraphs. In contrast, one can easily transform a B-graph to a precedence relation by splitting B-arcs into single-tailed arcs in the obvious way. Deriving precedence relations from more expressive B-graphs is therefore a correct way of obtaining strict partial orders that satisfy the precondition of our algorithm. Doing so just eliminates information that this algorithm does not care about anyway.

Thus, we propose the following. Instead of formalizing dependencies among literals in a set $L \cup \{\bigstar\}$ directly as a precedence relation, we first formalize those dependencies as a B-graph. If the resulting B-graph is a $\bigstar$-arborescence, we can directly extract a precedence relation $\prec_L$. Otherwise, we compute a $\bigstar$-arborescence of the resulting B-graph and extract a precedence relation $\prec_L$ afterward. Either way, because $\prec_L$ is extracted from a $\bigstar$-arborescence, it is a strict partial order whose linearization satisfies the precondition of our algorithm.

Let $\blacktriangleleft_L$ denote a set of B-arcs on $L \cup \{\bigstar\}$ defined by the following rules, plus the straightforward B-arcs adaptation of the rules in (6), page 13:

$$\frac{\begin{array}{l}\ell \in L \\ \textbf{and } \mathsf{Free}(\ell) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\} \\ \textbf{and } \mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k \in L\end{array}}{\{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\} \blacktriangleleft_L \ell} \qquad \frac{\begin{array}{l}\mathtt{d}_x \approx t \in L \\ \textbf{and } \mathsf{Free}(t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\} \\ \textbf{and } \mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k \in L\end{array}}{\{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\} \blacktriangleleft_L \mathtt{d}_x \approx t} \tag{7}$$

The first rule generalizes the first rule in (4), page 9, by joining sets of dependencies of a literal in a single B-arc. The second rule states that $\mathtt{d}_x \approx t$ literals do not necessarily depend on $\mathtt{d}_x$ (as implied by the first rule) but only on the free variables in t: intuitively, a value for x can be derived from values of the free variables in t (cf. assignments). Note that literals can have multiple incoming B-arcs. Such multiple incoming B-arcs represent a disjunction of conjunctions of dependencies. Importantly, as long as all dependencies represented by *one* incoming B-arc are satisfied, the other incoming B-arcs do not matter. An arborescence, which contains one incoming B-arc for every literal, therefore preserves enough dependencies. Shortly, Theorem 2 makes this more precise. Figure 5 shows a fragment of the B-graph for data constraint ϕ in (2), page 8.

One can straightforwardly compute an arborescence of a B-graph $(L \cup \{\bigstar\}, \blacktriangleleft_L)$ with a graph exploration algorithm reminiscent of breadth-first search. Let $\blacktriangleleft_L^{\text{arb}} \subseteq \blacktriangleleft_L$ denote the aborescence under computation, and let $L_{\text{done}} \subseteq L$ denote the set of vertices (i.e., literals in L) that have already been explored; initially,

resolve each of its dependencies (i.e., have a value for each of its free variables). For instance, if $\mathtt{d}_{x_1}$ evaluates to false, we do not need a value for $\mathtt{d}_{x_2}$ to evaluate $\mathtt{and}(\mathtt{d}_{x_1}, \mathtt{d}_{x_2})$. However, whether or not one can do such *short-circuiting* is known only at run-time. At compile-time, we must therefore conservatively assume that short-circuiting never happens by modeling all dependencies of a term.

$\blacktriangleleft_L^{\text{arb}} = \emptyset$ and $L_{\text{done}} = \{\bigstar\}$. Now, given some L_{done}, compute a set of vertices L_{next} that are connected only to vertices in L_{done} by a B-arc in $\blacktriangleleft_L$. Then, for every vertex in L_{next}, add an incoming B-arc to $\blacktriangleleft_L^{\text{arb}}$.[3] Afterward, add L_{next} to L_{done}. Repeat this process until L_{next} becomes empty. Once that happens, either $\blacktriangleleft_L^{\text{arb}}$ contains an arborescence (if $L_{\text{done}} = L$) or no arborescence exists. This computation runs in linear time, in the size of the B-graph. See also Footnote 3. Given $\blacktriangleleft_L^{\text{arb}}$, the following rules yield a cycle-free precedence relation on $L \cup \{\bigstar\}$:

$$\frac{\{\ell_1, \ldots, \ell_k\} \blacktriangleleft_L^{\text{arb}} \ell \text{ and } 1 \leq i \leq k}{\ell_i \prec_L \ell} \qquad \frac{\ell_1 \prec_L \ell_2 \prec_L \ell_3 \text{ and } \ell_2 \notin \{\ell_1, \ell_3\}}{\ell_1 \prec_L \ell_3} \tag{8}$$

Theorem 2 (Appendix A, Theorem 4). *$\prec_L$ as defined by the rules in (5)(8), pages 10 and 15, is a strict partial order and a large enough subset of $\prec_L$ as defined by the rules in (4)(5)(6), pages 9, 10, and 13, to satisfy the precondition of our translation algorithm in Section 3.3.*

Proof (Sketch). Let $\prec_L^{\text{Sect.4}}$ denote the precedence relation as defined by the rules in (5)(8), pages 10 and 15. Let $\prec_L^{\text{Sect.3}}$ denote the precedence relation as defined by the rules in (4)(5)(6), pages 9, 10, and 13.

First, the fact that $\prec_L^{\text{Sect.4}}$ is a strict partial order follows from $\blacktriangleleft_L^{\text{arb}}$ forming an arborescence (because arborescences contain no cycles by definition).

Second, to show $\prec_L^{\text{Sect.4}} \subseteq \prec_L^{\text{Sect.3}}$, take any pair (ℓ, ℓ') such that $\ell \prec_L^{\text{Sect.4}} \ell'$ by the first rule in (8), page 15. Then, by the premise of that rule, $\{\ell_1, \ldots, \ell_k\} \blacktriangleleft_L^{\text{arb}} \ell'$ such that $\ell = \ell_i$ for some $1 \leq i \leq k$. Because $\blacktriangleleft_L^{\text{arb}} \subseteq \blacktriangleleft_L$, the premises of the rules in (7), page 14, subsequently guarantee after some manipulation that $\ell = \ell_i = \mathtt{d}_x \approx t$ for some x and t. Moreover, $\mathtt{d}_x \in \mathsf{Free}(\ell')$. By the first rule in (4), page 9, we subsequently conclude that $\ell \prec_L^{\text{Sect.3}} \ell'$ holds. We can inductively show the same for pairs (ℓ, ℓ') such that $\ell \prec_L^{\text{Sect.4}} \ell'$ by the second rule in (8), page 15.

Finally, we must show that $\prec_L^{\text{Sect.4}}$ is "large enough" for it to satisfy the precondition of our translation algorithm. Informally, this means that the algorithm for computing arborescences does not exclude B-arcs from the arborescence that represent essential dependencies: for every free variable $\mathtt{d}_y$ that a literal $\ell \in L$ depends on, $\prec_L^{\text{Sect.4}}$ must contain at least one pair $(\mathtt{d}_y \approx t, \ell)$ (for some t). To see that this holds, note that every B-arc entering a literal ℓ represents a complete set of dependencies of ℓ. If ℓ has multiple incoming B-arcs, this simply means that several ways exist to resolve ℓ's dependencies. In principle, however, keeping one of those options suffices for our purpose. Therefore, the single incoming B-arc that ℓ has in an arborescence represents enough dependencies of ℓ. □

[3] If a vertex ℓ in L_{next} has multiple incoming B-arcs, the choice among them matters not: the choice is local, because every B-arc has only one head (i.e., adding an ℓ-headed B-arc to $\blacktriangleleft_L^{\text{arb}}$ cannot cause another vertex to get multiple incoming B-arcs, which would invalidate the arborescence). General hypergraphs, whose hyperarcs can have multiple heads, violate this property (i.e., the choice of which hyperarc to add is global instead of local). Computing arborescences of such hypergraphs is NP-complete [21], whereas one can compute aborescences of B-graphs in linear time.

(A detailed proof in Appendix B, Theorem 4, formalizes the previous proof sketch by formally establishing that (the formal definition of) $\prec_L^{\text{Sect.4}}$ satisfies (the formal definition of) the precondition of our translation algorithm.) For instance, the bold arcs in Figure 3 represent the precedence relation so derived from the arborescence in Figure 5.

If a ★-arborescence of $(L \cup \{\bigstar\}, \blacktriangleleft_L)$ does *not* exist, every $|L|$-cardinality subset of $\blacktriangleleft_L$ has at least one vertex ℓ that is unreachable from ★. In that case, by the rules in (6), page 13, ℓ depends on at least one free variable (otherwise, $\{\bigstar\} \blacktriangleleft_L \ell$). Because no B-graph equivalent of a path [22] exists from ★ to ℓ, the other literals in L fail to resolve at least one of ℓ's dependencies. This occurs, for instance, when ℓ depends on $\mathtt{d}_y$, while L contains no $\mathtt{d}_y \approx t$ literal. Another example is a recursive literal $\mathtt{d}_x \approx t$ with $\mathtt{d}_x \in \mathsf{Free}(t)$: unless another literal $\mathtt{d}_x \approx t'$ with $t \neq t'$ exists, all its incoming B-arcs contain loops to itself, meaning that no arborescence exists. In practice, such cases inherently require constraint solving techniques to find a value for $\mathtt{d}_x$. Nonexistence of a ★-arborescence thus signals a fundamental boundary to the applicability of our translation algorithm (although more advanced techniques of translating some parts of a data constraint to a data command and leaving other parts to a constraint solver are imaginable and left for future work). Thus, the set of data constraints to which our translation algorithm is applicable contains exactly those (i) whose B-graph has a ★-arborescence, which guarantees linearizability of the induced precedence, and (ii) that satisfy also the rest of the precondition of our algorithm in Section 3.3.

5 Preliminary Performance Results

In the work that we presented in this paper, we focused on the formal definition of our translation and its proof of correctness. A comprehensive quantitative evaluation remains future work. Indeed, constructing a set of representative examples, identifying independent variables that may influence the outcome (e.g., number of cores, memory architecture, etc.), setting up and performing the corresponding experiments, processing/analyzing the measurements, and eventually presenting the results is a whole other challenge. Still, presenting an optimization technique and not shedding any light on its performance may leave the reader with an unsatisfactory feeling. Therefore, in this section, we provide preliminary performance results to give a rough indication of our translation's merits.

We extended our most recent CA-to-Java compiler and used this compiler to generate both *constraint-based* coordination code (i.e., generated without our translation) and *command-based* coordination code (i.e., generated with our translation) for ten coordinators modeled as CA: three elementary primitives from Figures 1a and 2a (to see how our optimization affects such basic cases) and seven more complex composites, including those in Figures 1b and 2b. See Section 2 for a discussion of these coordinators' behavior. The constraint-based implementations use a custom constraint solver with constraint propagation [23], tailored to our setting of data constraints. The data commands in the generated command-based implementations are imperative Java code, very similar to what

	Constr.	*Comm.*	×
Sync	33119333	39800986	1.20
Fifo	33050122	41398084	1.25
Replicator	17961129	21803913	1.21
Example	10573857	12687767	1.20
Fibonacci	1818671	88947751	48.91

	Constr.	*Comm.*	×
Language2	17278247	24646838	1.43
Language4	4423326	11512506	2.60
Language6	1062306	5294838	4.98
Language8	194374	1746440	8.98
Language10	25649	362050	14.12

Fig. 6: Preliminary performance results for ten coordinators. Column "*Constr.*" shows results for constraint-based implementations (in number of coordination steps completed in four minutes); column "*Comm.*" shows restults for command-based implementations; column "×" shows the ratio of the second over the first.

programmers would hand-craft (modulo style).

In total, thus, we generated twenty coordinators in Java. We ran each of those implementations ten times on a quadcore machine at 2.4 GHz (no Hyper-Threading; no Turbo Boost) and averaged our measurements. In every run, we warmed up the JVM for thirty seconds before starting to measure the number of coordination steps that an implementation could finish in the subsequent four minutes. Figure 6 shows our results. The command-based implementations outperform their constraint-based versions in all cases. The Language*i* coordinators furthermore show that the speed-up achieved by their command-based implementations increases as i increases. This may suggest that our optimization becomes relatively more effective as the size/complexity of a coordinator increases, as also witnessed by Fibonacci. Figure 6 shows first evidence for the effectiveness of our translation in practice, although further study is necessary.

6 Discussion

In constraint programming, it is well-known that "if domain specific methods are available they should be applied *instead* [sic] of the general methods" [23, page 2]. The work presented in this paper takes this guideline to an extreme: essentially, every data command generated for a data constraint ϕ by our translation algorithm is a little constraint solver capable of solving only ϕ, with good performance. This good performance comes from the fact that the order of performing assignments and failure statements has already been determined at compile-time. Moreover, this precomputed order guarantees that backtracking is unnecessary: the data constraint for ϕ finds a solution if one exists without search (i.e., Theorem 1). In contrast, general constraint solvers need to do this work, which our approach does at compile-time, as part of the solving process at run-time.

Execution of data commands bears similarities with *constraint propagation* techniques [23], in particular with *forward checking* [24,25]. Generally, in constraint propagation, the idea is to reduce the search space of a given *constraint satisfaction problem* (CSP) by transforming it into an equivalent "simpler" CSP, where variables have smaller domains, or where constraints refer to fewer vari-

ables. With forward checking, whenever a variable x gets a value v, a constraint solver removes values from the domains of all subsequent variables that, together with v, violate a constraint. In the case of an equality $x = y$, for instance, forward checking reduces the domain of y to the singleton $\{v\}$ after an assignment of v to x. That same property of equality is implicitly used in executing our data commands (i.e., instead of representing the domain of a variable and the reduction of this domain to a singleton explicitly, we directly make an assignment).

Our translation from data constraints to data commands may also remind one of classical *Gaussian elimininationation* for solving systems of linear equations over the reals [23]: there too, variables are ordered and values/expressions for some variables are substituted into other expressions. The difference is that we have functions, relations, and our data domain may include other data types, which makes solving data constraints directly via Gaussian elimination at least not obvious. However, Gaussian elimination does seem useful as a preprocessing step for translating certain data constraints to data commands that our current algorithm does not support. Future work should clarify this possibility.

Clarke et al. worked on purely constraint-based implementations of coordinators [26]. Essentially, they specify not only the transition labels of a CA as boolean constraints but also its state space and transition relation. In recent work, Proença and Clarke developed a variant of compile-time *predicate abstraction* to improve performance [27]. They used this technique also to allow a form of interaction between the constraint solver and external components during constraint solving [28]. The work of Proença and Clarke resembles ours in the sense that we all try to "simplify" constraints at compile-time. Main differences are that (i) we fully avoid constraint solving and (ii) we consider a richer language of data constraints. For instance, Proença and Clarke have only unary functions in their language, which would have cleared our need for B-graphs.

References

1. Jongmans, S.S., Halle, S., Arbab, F.: Automata-based Optimization of Interaction Protocols for Scalable Multicore Platforms. In: COORDINATION 2014. Volume 8459 of LNCS. Springer (2014) 65–82
2. Jongmans, S.S., Halle, S., Arbab, F.: Reo: A Dataflow Inspired Language for Multicore. In: DFM 2013, IEEE (2014) 42–50
3. Baier, C., Sirjani, M., Arbab, F., Rutten, J.: Modeling component connectors in Reo by constraint automata. SCP **61**(2) (2006) 75–113
4. Wegner, P.: Coordination as Constrained Interaction (Extended Abstract). In: COORDINATION 1996. Volume 1061 of LNCS. Springer (1996) 28–33
5. Jongmans, S.S., Arbab, F.: Global Consensus through Local Synchronization. In: FOCLASA 2013. Volume 393 of CCIS. Springer (2013) 174–188
6. Jongmans, S.S., Arbab, F.: Toward Sequentializing Overparallelized Protocol Code. In: ICE 2014. Volume 166 of EPTCS. CoRR (2014) 38–44
7. Jongmans, S.S., Santini, F., Arbab, F.: Partially-Distributed Coordination with Reo. In: PDP 2014, IEEE (2014) 697–706
8. Arbab, F.: Reo: a channel-based coordination model for component composition. MSCS **14**(3) (2004) 329–366

9. Arbab, F.: Puff, The Magic Protocol. In: Talcott Festschrift. Volume 7000 of LNCS. Springer (2011) 169–206
10. Arbab, F., Kokash, N., Meng, S.: Towards Using Reo for Compliance-Aware Business Process Modeling. In: ISoLA 2008. Volume 17 of CCIS. Springer (2008) 108–123
11. Changizi, B., Kokash, N., Arbab, F.: A Unified Toolset for Business Process Model Formalization. In: FESCA 2010. (2010) 147–156
12. Meng, S., Arbab, F., Baier, C.: Synthesis of Reo circuits from scenario-based interaction specifications. SCP **76**(8) (2011) 651–680
13. Bliudze, S., Sifakis, J.: Causal semantics for the algebra of connectors. FMSD **36**(2) (2010) 167–194
14. Russell, S., Norvig, P.: Artificial Intelligence. 2nd edn. Prentice Hall (2003)
15. Apt, K., de Boer, F., Olderog, E.R.: Verification of Sequential and Concurrent Programs. 3rd edn. Springer (2009)
16. Hoare, T.: An Axiomatic Basis for Computer Programming. CACM **12**(10) (1969) 576–580
17. Kahn, A.: Topological Sorting in Large Networks. CACM **5**(11) (1962) 558–562
18. Knuth, D.: Fundamental Algorithms. 3rd edn. Volume 1 of The Art of Computer Programming. Addison-Wesley (1997)
19. Korte, B., Vygen, J.: Combinatorial Optimization: Theory and Algorithms. 4th edn. Volume 21 of Algorithms and Combinatorics. Springer (2008)
20. Gallo, G., Longo, G., Pallottino, S., Nguyen, S.: Directed hypergraphs and applications. DAM **42**(2–3) (1993) 177–201
21. Woeginger, G.: The complexity of finding arborescences in hypergraphs. IPL **44**(3) (1992) 161–164
22. Ausiello, G., Franciosa, P., Frigioni, D.: Directed Hypergraphs: Problems, Algorithmic Results, and a Novel Decremental Approach. In: ICTCS 2001. Volume 2202 of LNCS. Springer (2001) 312–328
23. Apt, K.: Principles of Constraint Programming. Cambridge University Press (2009)
24. Bessière, C., Meseguer, P., Freuder, E., Larrosa, J.: On forward checking for non-binary constraint satisfaction. Artificial Intelligence **141**(1–2) (2002) 205–224
25. McGregor, J.: Relational consistency algorithms and their application in finding subgraph and graph isomorphism. Information Science **19** (1979) 229–250
26. Clarke, D., Proença, J., Lazovik, A., Arbab, F.: Channel-based coordination via constraint satisfaction. SCP **76**(8) (2011) 681–710
27. Proença, J., Clarke, D.: Data Abstraction in Coordination Constraints. In: FOCLASA 2013. Volume 393 of CCIS. Springer (2013) 159–173
28. Proença, J., Clarke, D.: Interactive Interaction Constraints. In: COORDINATION 2013. Volume 7890 of LNCS. Springer (2013) 211–225

A Definitions and Properties

A.1 Execution Steps

Definition 1 (Places). $\mathbb{P}$ *denotes the set of all places.*

Definition 2 (Data). $\mathbb{D}$ *denotes the set of all data items.*

Definition 3 (Nil). nil *denotes an object such that* $\mathsf{nil} \notin \mathbb{D}$.

Definition 4 (Distributions).

1. *A distribution is a partial function* $\delta : \mathbb{P} \rightharpoonup \mathbb{D} \cup \{\mathsf{nil}\}$ *from places to data items.*
2. $\mathbb{D}\textsc{istr}$ *denotes the set of all distributions.*
3. $\cdot[\cdot := \cdot] : \mathbb{D}\textsc{istr} \times \mathbb{P} \times \mathbb{D} \to \mathbb{D}\textsc{istr}$ *denotes the function from* $\big[$*distribution, place, data item*$\big]$*-tuples to distributions defined by the following equation:*

$$\delta[x := d] = (\delta \setminus \{x \mapsto d' \mid d' \in \mathbb{D}\}) \cup \{x \mapsto d\}$$

A.2 Terms

Definition 5 (Functions and relations).

1. $\mathbb{F} = \bigcup\{\mathbb{D}^k \to \mathbb{D} \mid k > 0\}$ *and* $\mathbb{R} = \bigcup\{\wp(\mathbb{D}^k) \mid k > 0\}$ *denote the sets of all data functions and all data relations.*
2. $\mathbb{F}\textsc{un}$ *and* $\mathbb{R}\textsc{el}$ *denote the sets of all data function symbols and all data relation symbols.*
3. $\mathsf{arity} : \mathbb{F}\textsc{un} \cup \mathbb{R}\textsc{el} \to \mathbb{N}_+$ *denotes a function from data function symbols and data relation symbols to positive natural numbers.*

Definition 6 (Interpretation of symbols).

1. $\mathcal{I} : \mathbb{D}\textsc{ata} \cup \mathbb{F}\textsc{un} \cup \mathbb{R}\textsc{el} \to \mathbb{D} \cup \mathbb{F} \cup \mathbb{R}$ *denotes a bijection from data symbols, data function symbols, and data relation symbols to data, functions, and relations such that:*
   - $\big[d \in \mathbb{D}\textsc{ata}$ **iff** $\mathcal{I}(d) \in \mathbb{D}\big]$ **for all** d
   - $\big[f \in \mathbb{F}\textsc{un}$ **iff** $\mathcal{I}(f) : \mathbb{D}^{\mathsf{arity}(f)} \to \mathbb{D}\big]$ **for all** f
   - $\big[R \in \mathbb{R}\textsc{el}$ **iff** $\mathcal{I}(R) \in \wp(\mathbb{D}^{\mathsf{arity}(R)})\big]$ **for all** R
2. $\mathcal{I}^{-1} : \mathbb{D} \cup \mathbb{F} \cup \mathbb{R} \to \mathbb{D}\textsc{ata} \cup \mathbb{F}\textsc{un} \cup \mathbb{R}\textsc{el}$ *denotes the inverse of* $\mathcal{I}$.

Definition 7 (Terms).

1. *A data term is a word* t *generated by the following grammar:*

$$\begin{array}{l} x ::= \textit{any element from } \mathbb{P}\textsc{ort} \\ d ::= \textit{any element from } \mathbb{D}\textsc{ata} \\ f ::= \textit{any element from } \mathbb{F}\textsc{un} \\ t ::= \mathtt{d}_x \mid \mathtt{nil} \mid d \mid f(t_1, \ldots, t_k) \textbf{ if } \mathsf{arity}(f) = k \end{array}$$

2. $\mathbb{T}\textsc{erm}$ *denotes the set of all terms.*
3. $\mathsf{eval} : \mathbb{D}\textsc{istr} \times \mathbb{T}\textsc{erm} \to \mathbb{D} \cup \{\mathsf{nil}\}$ *denotes the function from* $\big[$*distribution, term*$\big]$*-pairs to data or* nil *defined by the following equations:*

$$\begin{array}{ll} \mathsf{eval}_\delta(\mathtt{d}_x) & = \begin{cases} \delta(x) & \textbf{if } x \in \mathrm{Dom}(\delta) \\ \mathsf{nil} & \textbf{otherwise} \end{cases} \\ \mathsf{eval}_\delta(\mathtt{nil}) & = \mathsf{nil} \\ \mathsf{eval}_\delta(d) & = \mathcal{I}(d) \\ \mathsf{eval}_\delta(f(t_1, \ldots, t_k)) & = \begin{cases} \mathcal{I}(f)(\mathsf{eval}_\delta(t_1), \ldots, \mathsf{eval}_\delta(t_k)) & \textbf{if } \big[\mathsf{eval}_\delta(t_1) \neq \mathsf{nil}, \ldots, \mathsf{eval}_\delta(t_k) \neq \mathsf{nil}\big] \\ \mathsf{nil} & \textbf{otherwise} \end{cases} \end{array}$$

4. $\mathsf{Free} : \mathbb{T}\text{ERM} \to \wp(\mathbb{T}\text{ERM})$ *denotes the function from terms to sets of terms defined by the following equations:*

$$\begin{array}{ll} \mathsf{Free}(\mathtt{d}_x) & = \{\mathtt{d}_x\} \\ \mathsf{Free}(\mathtt{nil})\,,\, \mathsf{Free}(d) & = \emptyset \\ \mathsf{Free}(f(t_1\,,\,\ldots\,,\,t_k)) & = \mathsf{Free}(t_1) \cup \cdots \cup \mathsf{Free}(t_k) \end{array}$$

5. $\mathsf{Arg} : \mathbb{T}\text{ERM} \to \wp(\mathbb{T}\text{ERM})$ *denotes the function from terms to sets of terms defined by the following equations:*

$$\begin{array}{ll} \mathsf{Arg}(\mathtt{d}_x)\,,\, \mathsf{Arg}(\mathtt{nil})\,,\, \mathsf{Arg}(d) & = \emptyset \\ \mathsf{Arg}(f(t_1\,,\,\ldots\,,\,t_k)) & = \{t_1\,,\,\ldots\,,\,t_k\} \end{array}$$

6. $\mathtt{nil}\text{-}\mathsf{Free} \subseteq \mathbb{T}\text{ERM}$ *denotes the relation on terms defined as follows:*

$$\begin{array}{ll} \mathtt{nil}\text{-}\mathsf{Free}(\mathtt{d}_x) & \textbf{iff } \textbf{true} \\ \mathtt{nil}\text{-}\mathsf{Free}(\mathtt{nil}) & \textbf{iff } \textbf{false} \\ \mathtt{nil}\text{-}\mathsf{Free}(d) & \textbf{iff } \textbf{true} \\ \mathtt{nil}\text{-}\mathsf{Free}(f(t_1\,,\,\ldots\,,\,t_k)) & \textbf{iff } \big[\mathtt{nil}\text{-}\mathsf{Free}(t_1) \textbf{ and } \cdots \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(t_k)\big] \end{array}$$

A.3 Data Constraints

Definition 8 (Data constraints).

1. *A data constraint is a word ϕ generated by the following grammar:*

$$\begin{array}{ll} t & ::= \textit{ any element from } \mathbb{T}\text{ERM} \\ R & ::= \textit{ any element from } \mathbb{R}\text{EL} \\ a & ::= \bot \mid \top \mid t \approx t \mid t \not\approx \mathtt{nil} \mid R(t_1\,,\,\ldots\,,\,t_k) \textbf{ if } \mathsf{arity}(R) = k \\ \phi & ::= a \mid \neg\phi \mid \phi \vee \phi \mid \phi \wedge \phi \end{array}$$

2. $\mathbb{DC}$ *denotes the set of all data constraints.*
3. $\models^{\mathrm{dc}} \subseteq \mathbb{D}\text{ISTR} \times \mathbb{DC}$ *denotes the relation on* $\big[$*distribution, data constraint*$\big]$*-pairs defined as follows:*

$$\begin{array}{ll} \delta \models^{\mathrm{dc}} \bot & \textbf{iff } \textbf{false} \\ \delta \models^{\mathrm{dc}} \top & \textbf{iff } \textbf{true} \\ \delta \models^{\mathrm{dc}} t_1 \approx t_2 & \textbf{iff } \mathsf{eval}_\delta(t_1) = \mathsf{eval}_\delta(t_2) \neq \mathsf{nil} \\ \delta \models^{\mathrm{dc}} t \not\approx \mathtt{nil} & \textbf{iff } \mathsf{eval}_\delta(t) \neq \mathsf{nil} \\ \delta \models^{\mathrm{dc}} R(t_1\,,\,\ldots\,,\,t_k) & \textbf{iff } \mathcal{I}(R)(\mathsf{eval}_\delta(t_1)\,,\,\ldots\,,\,\mathsf{eval}_\delta(t_k)) \\ \delta \models^{\mathrm{dc}} \neg\phi & \textbf{iff } \delta \not\models^{\mathrm{dc}} \phi \\ \delta \models^{\mathrm{dc}} \phi_1 \vee \phi_2 & \textbf{iff } \big[\delta \models^{\mathrm{dc}} \phi_1 \textbf{ or } \delta \models^{\mathrm{dc}} \phi_2\big] \\ \delta \models^{\mathrm{dc}} \phi_1 \wedge \phi_2 & \textbf{iff } \big[\delta \models^{\mathrm{dc}} \phi_1 \textbf{ and } \delta \models^{\mathrm{dc}} \phi_2\big] \end{array}$$

4. $\Rightarrow \subseteq \mathbb{DC} \times \mathbb{DC}$ *denotes the relation on pairs of data constraints defined as follows:*

$$\phi \Rightarrow \phi' \textbf{ iff } \big[\big[\delta \models^{\mathrm{dc}} \phi \textbf{ implies } \delta \models^{\mathrm{dc}} \phi'\big] \textbf{ for all } \delta\big]$$

5. $[\![\cdot]\!] : \mathbb{DC} \to \wp(\mathbb{D}\text{ISTR})$ *denotes the function from data constraints to sets of distributions defined by the following equation:*

$$[\![\phi]\!] = \{\delta \mid \delta \models^{\mathrm{dc}} \phi\}$$

6. $\mathsf{Free} : \mathbb{DC} \to \wp(\mathbb{T}\text{ERM})$ *denotes the function from data constraints to sets of terms defined by the following equations:*

$$\begin{array}{ll} \mathsf{Free}(\bot)\,,\, \mathsf{Free}(\top) & = \emptyset \\ \mathsf{Free}(t_1 \approx t_2) & = \mathsf{Free}(t_1) \cup \mathsf{Free}(t_2) \\ \mathsf{Free}(t \not\approx \mathtt{nil}) & = \mathsf{Free}(t) \\ \mathsf{Free}(R(t_1\,,\,\ldots\,,\,t_k)) & = \mathsf{Free}(t_1) \cup \cdots \cup \mathsf{Free}(t_k) \\ \mathsf{Free}(\neg\phi) & = \mathsf{Free}(\phi) \\ \mathsf{Free}(\phi_1 \vee \phi_2)\,,\, \mathsf{Free}(\phi_1 \wedge \phi_2) & = \mathsf{Free}(\phi_1) \cup \mathsf{Free}(\phi_2) \end{array}$$

Proposition 2 (Monotonicity of $\models^{\mathrm{dc}}$).

$$\big[\mathsf{Free}(\phi) \subseteq \{\mathtt{d}_x \mid x \in \mathrm{Dom}(\delta')\} \textbf{ and } \delta' \subseteq \delta \textbf{ and } \delta' \models^{\mathrm{dc}} \phi\big] \textbf{ implies } \delta \models^{\mathrm{dc}} \phi$$

Proposition 3 ([15, Lemma 2.1, page 42]).

1. $[\![\neg\phi]\!] = \mathbb{D}\textsc{istr} \setminus [\![\phi]\!]$
2. $[\![\phi_1 \vee \phi_2]\!] = [\![\phi_1]\!] \cup [\![\phi_2]\!]$
3. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
4. $\phi_1 \Rightarrow \phi_2$ **iff** $[\![\phi_1]\!] \subseteq [\![\phi_2]\!]$

A.4 Data Commands

Definition 9 (Data commands).

1. *A data command is a word P generated by the following grammar:*

$$\begin{array}{rl} x & ::= \textit{ any element from } \mathbb{P} \\ t & ::= \textit{ any element from } \mathbb{T}\textsc{erm} \\ P & ::= \mathtt{skip} \mid x := t \mid \mathtt{if}\ \phi\ \mathtt{->}\ P\ \mathtt{fi} \mid P\ ;\ P \end{array}$$

2. *$\mathbb{C}\textsc{md}$ denotes the set of all data commands.*
3. *ε denotes the empty data command such that $\varepsilon\ ;\ P = P\ ;\ \varepsilon = P$.*

Definition 10 (Configurations).

1. *$\mathbb{C}\textsc{onf} = \mathbb{C}\textsc{md} \times \mathbb{D}\textsc{istr}$ denotes the set of all configurations.*
2. *$\Longrightarrow \subseteq \mathbb{C}\textsc{onf} \times \mathbb{C}\textsc{onf}$ denotes the relation on pairs of configurations defined by the following rules:*

$$\frac{}{(\mathtt{skip}\,,\,\delta) \Longrightarrow (\varepsilon\,,\,\delta)} \qquad \frac{}{(x := t\,,\,\delta) \Longrightarrow (\varepsilon\,,\,\delta[x := \mathsf{eval}_\delta(t)])} \qquad \frac{(P\,,\,\delta) \Longrightarrow (P'\,,\,\delta')}{(P\ ;\ P''\,,\,\delta) \Longrightarrow (P'\ ;\ P''\,,\,\delta')}$$

$$\frac{\delta \models^{\mathrm{dc}} \phi}{(\mathtt{if}\ \phi\ \mathtt{->}\ P\ \mathtt{fi}\,,\,\delta) \Longrightarrow (P\,,\,\delta)} \qquad \frac{\delta \not\models^{\mathrm{dc}} \phi}{(\mathtt{if}\ \phi\ \mathtt{->}\ P\ \mathtt{fi}\,,\,\delta) \Longrightarrow (\varepsilon\,,\,\mathsf{fail})}$$

3. *fail denotes an object such that $\mathsf{fail} \notin \mathbb{D}\textsc{istr}$.*
4. *$\mathcal{M} : \mathbb{C}\textsc{md} \times \wp(\mathbb{D}\textsc{istr}) \to \wp(\mathbb{D}\textsc{istr})$ denotes a function from $\big[$data command, set of distributions$\big]$-pairs to sets of distributions defined by the following equation:*

$$\mathcal{M}(P\,,\,\Delta) = \bigcup\{\Delta' \mid \delta \in \Delta \textbf{ and } \Delta' = \{\delta' \mid (P\,,\,\delta) \Longrightarrow^* (\varepsilon\,,\,\delta')\}\}$$

5. *$\mathcal{M}_{tot} : \mathbb{C}\textsc{md} \times \wp(\mathbb{D}\textsc{istr}) \to \wp(\mathbb{D}\textsc{istr} \cup \{\mathsf{fail}\})$ denotes a function from $\big[$data command, set of distributions$\big]$-pairs to sets of distributions defined by the following equation:*

$$\mathcal{M}_{tot}(P\,,\,\Delta) = \mathcal{M}(P\,,\,\Delta) \cup \bigcup\{\Delta' \mid \delta \in \Delta \textbf{ and } \Delta' = \{\mathsf{fail} \mid (P\,,\,\{\delta\}) \Longrightarrow^* (\varepsilon\,,\,\mathsf{fail})\}\}$$

Definition 11 (Triples).

1. *$\mathbb{T}\textsc{riple} = \mathbb{DC} \times \mathbb{C}\textsc{md} \times \mathbb{DC}$ denotes the set of all triples.*
2. *$\models \subseteq \mathbb{T}\textsc{riple}$ denotes the relation on triples defined as follows:*

$$\models \{\phi\}\ P\ \{\phi'\} \textbf{ iff } \mathcal{M}(P\,,\,[\![\phi]\!]) \subseteq [\![\phi']\!]$$

3. *$\models_{\mathrm{tot}} \subseteq \mathbb{T}\textsc{riple}$ denotes the relation on triples defined as follows:*

$$\models_{\mathrm{tot}} \{\phi\}\ P\ \{\phi'\} \textbf{ iff } \mathcal{M}_{tot}(P\,,\,[\![\phi]\!]) \subseteq [\![\phi']\!]$$

Proposition 4 ([15, Theorems 3.1 and 3.6, pages 74 and 97]).

1. $\vdash \{\phi\}\ P\ \{\phi'\}$ **implies** $\models \{\phi\}\ P\ \{\phi'\}$
2. $\vdash_{\mathrm{tot}} \{\phi\}\ P\ \{\phi'\}$ **implies** $\models_{\mathrm{tot}} \{\phi\}\ P\ \{\phi'\}$

	Partial correctness	*Total correctness*
Axiom–Skip	$\vdash \{\phi\}\ \texttt{skip}\ \{\phi\}$	$\vdash_{\text{tot}} \{\phi\}\ \texttt{skip}\ \{\phi\}$
Axiom–Assignment	$\vdash \{\phi[\mathtt{d}_x := t]\}\ x\ \texttt{:=}\ t\ \{\phi\}$	$\vdash_{\text{tot}} \{\phi[\mathtt{d}_x := t]\}\ x\ \texttt{:=}\ t\ \{\phi\}$
Rule–Composition	$\dfrac{\vdash \{\phi\}\ P\ \{\phi'\}\ \textbf{and}\ \vdash \{\phi'\}\ P'\ \{\phi''\}}{\vdash \{\phi\}\ P\ ;\ P'\ \{\phi''\}}$	$\dfrac{\vdash_{\text{tot}} \{\phi\}\ P\ \{\phi'\}\ \textbf{and}\ \vdash_{\text{tot}} \{\phi'\}\ P'\ \{\phi''\}}{\vdash_{\text{tot}} \{\phi\}\ P\ ;\ P'\ \{\phi''\}}$
Rule–Consequence	$\dfrac{\phi \Rightarrow \phi'\ \textbf{and}\ \vdash \{\phi'\}\ P\ \{\phi''\}\ \textbf{and}\ \phi'' \Rightarrow \phi'''}{\vdash \{\phi\}\ P\ \{\phi'''\}}$	$\dfrac{\phi \Rightarrow \phi'\ \textbf{and}\ \vdash_{\text{tot}} \{\phi'\}\ P\ \{\phi''\}\ \textbf{and}\ \phi'' \Rightarrow \phi'''}{\vdash_{\text{tot}} \{\phi\}\ P\ \{\phi'''\}}$
Rule–Failure	$\dfrac{\vdash \{\phi \wedge \ell\}\ P\ \{\phi'\}}{\vdash \{\phi\}\ \texttt{if}\ \ell\ \texttt{->}\ P\ \texttt{fi}\ \{\phi'\}}$	
Rule–Failure II		$\dfrac{\phi \Rightarrow \ell\ \textbf{and}\ \vdash_{\text{tot}} \{\phi\}\ P\ \{\phi'\}}{\vdash_{\text{tot}} \{\phi\}\ \texttt{if}\ \ell\ \texttt{->}\ P\ \texttt{fi}\ \{\phi'\}}$
Rule–Decomposition	$\dfrac{\vdash \{\phi\}\ P\ \{\phi'\}\ \textbf{and}\ \vdash_{\text{tot}} \{\phi\}\ P\ \{\phi''\}}{\vdash_{\text{tot}} \{\phi\}\ P\ \{\phi' \wedge \phi''\}}$	

Fig. 7: Proof rules [15, Sections 3.3 and 3.7]

A.5 Algorithm

Lemma 1 (Auxiliary I).

$$\begin{bmatrix} \mathsf{Pre}\ \textbf{and}\ \mathsf{Inv}_1\ \textbf{and} \\ i \leq n\ \textbf{and}\ \mathtt{d}_y \in \mathsf{Free}(t_i) \end{bmatrix}\ \textbf{implies}\ \big[\big[\mathtt{d}_y = \mathtt{d}_{x_j}\ \textbf{and}\ 1 \leq j \leq i-1\big]\ \textbf{for some}\ j\big]$$

Proof. See page 29.

Lemma 2 (Auxiliary II). $\big[\big[\mathtt{d}_y \in \mathsf{Free}(t)\ \textbf{implies}\ \delta(y) = \delta'(y)\big]\ \textbf{for all}\ y\big]\ \textbf{implies}\ \mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)$

Proof. See page 32.

Lemma 3 (Auxiliary III).

$$\big[\big[\big[\mathtt{d}_y \in \mathsf{Free}(t)\ \textbf{implies}\ \delta \models^{\text{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\big]\ \textbf{for all}\ y\big]\ \textbf{and}\ \mathtt{nil}\text{-}\mathsf{Free}(t)\big]\ \textbf{implies}\ \mathsf{eval}_\delta(t) \in \mathbb{D}$$

Proof. See page 35.

Lemma 4 (Correctness I). Pre **implies** $\big[\mathsf{Pre}\ \textbf{and}\ \mathsf{Inv}_1[i \leftarrow 1][P \leftarrow \mathtt{skip}]\big]$

Proof. See page 38.

Lemma 5 (Correctness II).

$$\begin{bmatrix} \mathsf{Pre}\ \textbf{and}\ \mathsf{Inv}_1\ \textbf{and} \\ i \leq n\ \textbf{and}\ n - i = z\ \textbf{and} \\ \mathtt{d}_{x_i} \in \{\mathtt{d}_{x_j} \mid 1 \leq j < i\} \end{bmatrix}\ \textbf{implies}\ \begin{bmatrix} \mathsf{Pre}\ \textbf{and}\ \mathsf{Inv}_1[i \leftarrow i+1][P \leftarrow P\ ;\ \texttt{if}\ \mathtt{d}_{x_i} \approx t_i\ \texttt{->}\ \texttt{skip}\ \texttt{fi}] \\ \textbf{and}\ n - (i+1) < z \end{bmatrix}$$

Proof. See page 40.

$P \leftarrow$ `skip`
$i \leftarrow 1$
while $i \leq n$ **do**
  if $\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}$ **then**
    $P \leftarrow P$; `if` $\mathtt{d}_{x_i} \approx t_i$ `->` `skip` `fi`
  else
    $P \leftarrow P$; x_i `:=` t_i
  fi
  $i \leftarrow i + 1$
od
while $i \leq n + m$ **do**
  $P \leftarrow P$; `if` ℓ_i `->` `skip` `fi`
  $i \leftarrow i + 1$
od

Fig. 8: Algorithm for translating constraints to commands

$\{\mathsf{Pre}\}$
$\{\mathsf{Pre}$ **and** $\mathsf{Inv}_1[i \leftarrow 1][P \leftarrow \mathtt{skip}]\}$
$P \leftarrow \mathtt{skip}$
$i \leftarrow 1$
$\{\mathsf{Pre}$ **and** $\mathsf{Inv}_1\}$
while $i \leq n$ **do**
  $\{\mathsf{Pre}$ **and** Inv_1 **and** $i \leq n$ **and** $n - i = z\}$
  if $\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}$ **then**
    $\{\mathsf{Pre}$ **and** Inv_1 **and** $i \leq n$ **and** $n - i = z$ **and** $\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}\}$
    $\{\mathsf{Pre}$ **and** $\mathsf{Inv}_1[i \leftarrow i + 1][P \leftarrow P$; `if` $\mathtt{d}_{x_i} \approx t_i$ `->` `skip` `fi`$]$ **and** $n - (i + 1) < z\}$
    $P \leftarrow P$; `if` $\mathtt{d}_{x_i} \approx t_i$ `->` `skip` `fi`
    $\{\mathsf{Pre}$ **and** $\mathsf{Inv}_1[i \leftarrow i + 1]$ **and** $n - (i + 1) < z\}$
  else
    $\{\mathsf{Pre}$ **and** Inv_1 **and** $i \leq n$ **and** $n - i = z$ **and** $\mathtt{d}_{x_i} \notin \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}\}$
    $\{\mathsf{Pre}$ **and** $\mathsf{Inv}_1[i \leftarrow i + 1][P \leftarrow P$; x_i `:=` $t_i]$ **and** $n - (i + 1) < z\}$
    $P \leftarrow P$; x_i `:=` t_i
    $\{\mathsf{Pre}$ **and** $\mathsf{Inv}_1[i \leftarrow i + 1]$ **and** $n - (i + 1) < z\}$
  fi
  $\{\mathsf{Pre}$ **and** $\mathsf{Inv}_1[i \leftarrow i + 1]$ **and** $n - (i + 1) < z\}$
  $i \leftarrow i + 1$
  $\{\mathsf{Pre}$ **and** Inv_1 **and** $n - i < z\}$
od
$\{\mathsf{Pre}$ **and** Inv_1 **and** $i > n\}$
$\{\mathsf{Pre}$ **and** $\mathsf{Inv}_2\}$
while $i \leq n + m$ **do**
  $\{\mathsf{Pre}$ **and** Inv_2 **and** $i \leq n + m$ **and** $n + m - i = z\}$
  $\{\mathsf{Pre}$ **and** $\mathsf{Inv}_2[i \leftarrow i + 1][P \leftarrow P$; `if` ℓ_i `->` `skip` `fi`$]$ **and** $n + m - (i + 1) < z\}$
  $P \leftarrow P$; `if` ℓ_i `->` `skip` `fi`
  $i \leftarrow i + 1$
  $\{\mathsf{Pre}$ **and** Inv_2 **and** $i \leq n + m$ **and** $n + m - i < z\}$
od
$\{\mathsf{Pre}$ **and** Inv_2 **and** $i > n + m\}$
$\{\mathsf{Post}\}$

Fig. 9: Algorithm for translating constraints to commands, annotated with assertions for (total) correctness

$$
\begin{array}{ll}
\mathsf{Pre} : & \left[\begin{array}{l}
\prec_L \text{ is a strict partial order} \\
\textbf{and } \left[\left[\left[\mathtt{d}_x \approx t \in L \textbf{ and } \mathtt{d}_y \in \mathsf{Free}(t)\right] \textbf{ implies } \left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_x \approx t \textbf{ for some } u\right]\right] \textbf{ for all } x, y, t\right] \\
\textbf{and } < \text{ is a linear extension of } \prec_L \\
\textbf{and } L = \{\ell_j \mid 1 \le j \le n+m\} \\
\textbf{and } \underbrace{\mathtt{d}_{x_1} \approx t_1}_{\ell_1} < \cdots < \underbrace{\mathtt{d}_{x_n} \approx t_n}_{\ell_n} < \ell_{n+1} < \ldots < \ell_{n+m} \\
\textbf{and } \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} = \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \le j \le n+m\} \\
\textbf{and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_1) \textbf{ and } \cdots \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_{n+m})
\end{array}\right] \\
\mathsf{Inv}_1 : & \left[\begin{array}{l}
i \ge 1 \\
\textbf{and } \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\
\textbf{and } \left[\left[\begin{array}{c}[\delta' \overset{\mathrm{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}] \textbf{ implies} \\ \vdash_{\mathrm{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\}\end{array}\right] \textbf{ for all } \delta'\right]
\end{array}\right] \\
\mathsf{Inv}_2 : & \left[\begin{array}{l}
i \ge 1 + n \\
\textbf{and } \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \\
\textbf{and } \left[\left[\begin{array}{c}[\delta' \overset{\mathrm{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}] \textbf{ implies} \\ \vdash_{\mathrm{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}\end{array}\right] \textbf{ for all } \delta'\right]
\end{array}\right] \\
\mathsf{Post} : & \left[\begin{array}{l}
\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{n+m}\} \\
\textbf{and } \left[\left[\begin{array}{c}[\delta' \overset{\mathrm{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}] \textbf{ implies} \\ \vdash_{\mathrm{tot}} \{\top\}\ P\ \{\top\}\end{array}\right] \textbf{ for all } \delta'\right]
\end{array}\right]
\end{array}
$$

Fig. 10: Macros used in Figure 9

Lemma 6 (Correctness III).

$$
\left[\begin{array}{c}\mathsf{Pre} \textbf{ and } \mathsf{Inv}_1 \textbf{ and} \\ i \le n \textbf{ and } n - i = z \textbf{ and} \\ \mathtt{d}_{x_i} \notin \{\mathtt{d}_{x_j} \mid 1 \le j < i\}\end{array}\right] \textbf{ implies } \left[\mathsf{Pre} \textbf{ and } \mathsf{Inv}_1[i \leftarrow i+1][P \leftarrow P\ ;\ x_i := t_i] \textbf{ and } n - (i+1) < z\right]
$$

Proof. See page 47.

Lemma 7 (Correctness IV). $\left[\mathsf{Pre} \textbf{ and } \mathsf{Inv}_1 \textbf{ and } i > n\right]$ **implies** $\left[\mathsf{Pre} \textbf{ and } \mathsf{Inv}_2\right]$

Proof. See page 58.

Lemma 8 (Correctness V).

$$
\left[\begin{array}{c}\mathsf{Pre} \textbf{ and } \mathsf{Inv}_2 \textbf{ and} \\ i \le n + m \textbf{ and } n + m - i = z\end{array}\right] \textbf{ implies } \left[\begin{array}{c}\mathsf{Pre} \textbf{ and } \mathsf{Inv}_2[i \leftarrow i+1][P \leftarrow P\ ;\ \mathtt{if}\ \ell_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}] \\ \textbf{and } n + m - (i+1) < z\end{array}\right]
$$

Proof. See page 60.

Lemma 9 (Correctness VI). $\left[\mathsf{Pre} \textbf{ and } \mathsf{Inv}_2 \textbf{ and } i > n + m\right]$ **implies** Post

Proof. See page 66.

Theorem 3 (Correctness). *Figure 9 is correct.*

Proof. Figure 9 is correct if for all six pair of consecutive assertions, the upper assertion implies the lower one. Lemmas 4, 5, 6, 7, 8, and 9 establish these implications. □

A.6 Precondition

Definition 12. *L denotes a set of literals.*

Definition 13 (Hyperprecedence). *$\blacktriangleleft_L \subseteq \wp(L \cup \{\bigstar\}) \times L$ denotes a relation on $\big[$set of literals or $\bigstar$, literal$\big]$-pairs defined by the following rules:*

$$\frac{\ell \in L \text{ and } \mathsf{Free}(\ell) = \emptyset}{\{\bigstar\} \blacktriangleleft_L \ell} \qquad \frac{\mathtt{d}_x \approx t \in L \text{ and } \mathsf{Free}(t) = \emptyset}{\{\bigstar\} \blacktriangleleft_L \mathtt{d}_x \approx t}$$

$$\frac{\ell \in L \text{ and } \mathsf{Free}(\ell) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\} \text{ and } \mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k \in L}{\{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\} \blacktriangleleft_L \ell} \qquad \frac{\begin{array}{c}\mathtt{d}_x \approx t \in L \text{ and} \\ \mathsf{Free}(t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\} \text{ and} \\ \mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k \in L\end{array}}{\{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\} \blacktriangleleft_L \mathtt{d}_x \approx t}$$

Definition 14 (Arborescences). *$\blacktriangleleft_L^{arb}$ denotes a $\bigstar$-arborescence of $\blacktriangleleft_L$.*

Definition 15 (Precedence I). *$\prec_L^{arb} \subseteq L \times L$ denotes a relation on pairs of literals defined by the following rules:*

$$\frac{\{\ell_1, \ldots, \ell_k\} \blacktriangleleft_L^{arb} \ell \text{ and } 1 \leq i \leq k}{\ell_i \prec_L^{arb} \ell} \qquad \frac{\ell_1 \prec_L^{arb} \ell_2 \prec_L^{arb} \ell_3 \text{ and } \ell_2 \notin \{\ell_1, \ell_3\}}{\ell_1 \prec_L^{arb} \ell_3}$$

Definition 16 (Precedence II). *$\prec_L \subseteq L \times L$ denotes a relation on pairs of literals defined by the following rules:*

$$\frac{\ell \prec_L^{arb} \ell'}{\ell \prec_L \ell'} \qquad \frac{\mathtt{d}_x \approx t, \ell \in L \text{ and } \big[\ell \neq \mathtt{d}_{x'} \approx t \text{ for all } x', t'\big]}{\mathtt{d}_x \approx t \prec_L \ell} \qquad \frac{\ell_1 \prec_L \ell_2 \prec_L \ell_3 \text{ and } \ell_2 \notin \{\ell_1, \ell_3\}}{\ell_1 \prec_L \ell_3}$$

Proposition 5 (Arborescences).

1. $\ell \in L$ **implies** $\big[L' \blacktriangleleft_L^{arb} \ell$ **for some** $L'\big]$
2. **not** $\Big[\begin{bmatrix} L^1 \blacktriangleleft_L^{arb} \ell^2 \text{ and } \cdots \text{ and } L^{k-1} \blacktriangleleft_L^{arb} \ell^k \text{ and} \\ \ell^1 \in L^1 \text{ and } \cdots \text{ and } \ell^{k-1} \in L^{k-1} \text{ and } \ell^1 = \ell^k \end{bmatrix}$ **for some** $L^1, \ldots, L^k, \ell^1, \ldots \ell^k, k\Big]$

Lemma 10 (Irreflexivity I).

1. $\ell \prec_L^{arb} \ell''$ **implies** $\Big[\begin{bmatrix} L^1 \blacktriangleleft_L^{arb} \ell^2 \text{ and } \cdots \text{ and } L^{k-1} \blacktriangleleft_L^{arb} \ell^k \\ \text{and } \ell^1 \in L^1 \text{ and } \cdots \text{ and } \ell^{k-1} \in L^{k-1} \\ \text{and } \ell^1 = \ell \text{ and } \ell^k = \ell'' \end{bmatrix}$ **for some** $L^1, \ldots, L^k, \ell^1, \ldots \ell^k, k\Big]$
2. $\prec_L^{arb}$ *is irreflexive*

Proof. See page 68.

Lemma 11 (Irreflexivity II).

1. $\ell \prec_L^{arb} \ell''$ **implies** $\big[\ell = \mathtt{d}_x \approx t$ **for some** $x, t\big]$
2. $\ell \prec_L \ell''$ **implies** $\big[\ell = \mathtt{d}_x \approx t$ **for some** $x, t\big]$
3. $\ell \prec_L \ell''$ **implies** $\big[\ell \prec_L^{arb} \ell''$ **or** $\big[\ell'' \neq \mathtt{d}_x \approx t$ **for all** $x, t\big]\big]$
4. $\prec_L$ *is irreflexive*

Proof. See page 73.

Lemma 12 (Precondition I). $\prec_L$ *is a strict partial order*

Proof. See page 78.

Lemma 13 (Precondition II).

$$\left[\mathtt{d}_x \approx t \in L \textbf{ and } \mathtt{d}_y \in \mathsf{Free}(t)\right] \textbf{ implies } \left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_x \approx t \textbf{ for some } u\right]$$

Proof. See page 79.

Lemma 14 (Precondition III).

$$\left[\begin{array}{l}\left[< \textit{ is a linear extension of } \prec_L\right] \\ \textbf{and } L = \{\ell_j \mid 1 \leq j \leq n+m\} \\ \textbf{and } \ell_1 < \cdots < \ell_n < \ell_{n+1} < \cdots < \ell_{n+m} \\ \textbf{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \textbf{ and } \cdots \textbf{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \textbf{and } \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} = \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\}\end{array}\right]$$
$$\textbf{for some } <, \ell_1, \ldots, \ell_{n+m}, n, m, x_1, \ldots, x_n, t_1, \ldots, t_n$$

Proof. See page 82.

Theorem 4 (Precondition).

$$\left[\left[\ell \in L \textbf{ implies } \mathtt{nil}\text{-}\mathsf{Free}(\ell)\right] \textbf{ for all } \ell\right] \textbf{ implies } \left[\prec_L \textit{ in Definition 16 satisfies } \mathsf{Pre} \textit{ in Figure 10}\right]$$

Proof. The conjunction of the results in Lemmas 12, 13, and 14 established the required result. □

B Proofs

B.1 Lemma 1

Proof (of Lemma 1). First, assume:

(A1) Pre

(A2) Inv_1

(A3) $i \leq n$

(A4) $\mathtt{d}_y \in \mathsf{Free}(t_i)$

Next, observe:

(Z1) Recall Pre from (A1). Then, by applying the definition of Pre in Figure 10, conclude:

$$\begin{array}{l}
\prec_L \text{ is a strict partial order} \\
\textbf{and } \big[\big[\big[\mathtt{d}_x \approx t \in L \textbf{ and } \mathtt{d}_y \in \mathsf{Free}(t)\big] \textbf{ implies } \big[\mathtt{d}_y \approx u \prec_L \mathtt{d}_x \approx t \textbf{ for some } u\big]\big] \textbf{ for all } x, y, t\big] \\
\textbf{and } < \text{ is a linear extension of } \prec_L \\
\textbf{and } L = \{\ell_j \mid 1 \leq j \leq n+m\} \\
\textbf{and } \underbrace{\mathtt{d}_{x_1} \approx t_1}_{\ell_1} < \cdots < \underbrace{\mathtt{d}_{x_n} \approx t_n}_{\ell_n} < \ell_{n+1} < \ldots < \ell_{n+m} \\
\textbf{and } \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} = \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\} \\
\textbf{and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_1) \textbf{ and } \cdots \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_{n+m})
\end{array}$$

(Z2) Recall Inv_1 from (A2). Then, by applying the definition of Inv_1 in Figure 10, conclude:

$$\begin{array}{l}
i \geq 1 \\
\textbf{and } \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\
\textbf{and } \left[\left[\begin{array}{l}[\delta' \overset{\text{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}] \textbf{ implies} \\ \vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\}\end{array}\right] \textbf{ for all } \delta'\right]
\end{array}$$

(Z3) Recall $i \leq n$ from (A3). Then, by introducing (Z2), conclude, $1 \leq i \leq n$.

(Z4) Recall $1 \leq i \leq n$ from (Z3). Then, by applying arithmetic, conclude $1 \leq i \leq n+m$.

(Z5) Recall $1 \leq i \leq n$ from (Z3). Then, by applying (Z1), conclude $\ell_i = \mathtt{d}_{x_i} \approx t_i$.

(Z6) Recall $1 \leq i \leq n+m$ from (Z4). Then, by applying set theory, conclude $\ell_i \in \{\ell_j \mid 1 \leq j \leq n+m\}$. Then, by applying (Z1), conclude $\ell_i \in L$. Then, by applying (Z5), conclude $\mathtt{d}_{x_i} \approx t_i \in L$.

(Z7) Suppose:

$$\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \textbf{ for some } u$$

Then, by applying (Z1), conclude $\mathtt{d}_y \approx u \in L$. Then, by applying (Z1), conclude:

$$\mathtt{d}_y \approx u \in \{\ell_j \mid 1 \leq j \leq n+m\}$$

Then, by applying set theory, conclude $\big[\big[\mathtt{d}_y \approx u = \ell_j \textbf{ and } 1 \leq j \leq n+m\big] \textbf{ for some } j\big]$.

(Z8) Suppose:

$$1 \leq j \leq n+m \textbf{ for some } j$$

Then, by introducing (Z4), conclude $\big[1 \leq j \leq n+m \textbf{ and } 1 \leq i \leq n+m\big]$. Then, by applying arithmetic, conclude $\big[1 \leq j \leq i-1 \textbf{ or } i = j \textbf{ or } i+1 \leq j \leq n+m\big]$.

(Z9) Recall $\big[\prec_L \text{ is a strict partial order}\big]$ from (Z1). Then, by applying order theory, conclude:

$$\prec_L \text{ is transitive, asymmetric, and irreflexive}$$

Z0 Suppose:

$$\left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } \mathtt{d}_y \approx u = \ell_j \text{ and } i = j\right] \text{ for some } u, j$$

Then, by applying substitution, conclude $\left[\ell_j \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } i = j\right]$. Then, by applying Z5, conclude $\left[\ell_j \prec_L \ell_i \text{ and } i = j\right]$. Then, by applying substitution, conclude $\ell_i \prec_L \ell_i$. Then, by applying Z9, conclude **false**.

Y1 Recall $\left[\left[\prec_L \text{ is a strict partial order}\right] \text{ and } \left[< \text{ is a linear extension of } \prec_L\right]\right]$ from Z1. Then, by applying order theory, conclude $\left[\left[< \text{ is a strict total order}\right] \text{ and } \prec_L \subseteq <\right]$. Then, by applying order theory, conclude $\left[\left[< \text{ is transitive, asymmetric, and irreflexive}\right] \text{ and } \prec_L \subseteq <\right]$.

Y2 Suppose:

$$i + 1 \leq j \leq n + m \text{ for some } j$$

Then, by introducing Z1, conclude $\ell_i < \cdots < \ell_j$. Then, by applying Y1, conclude $\ell_i < \ell_j$.

Y3 Suppose:

$$\left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } \mathtt{d}_y \approx u = \ell_j \text{ and } i + 1 \leq j \leq n + m\right] \text{ for some } u, j$$

Then, by applying substitution, conclude $\left[\ell_j \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } i + 1 \leq j \leq n + m\right]$. Then, by applying Z5, conclude $\left[\ell_j \prec_L \ell_i \text{ and } i + 1 \leq j \leq n + m\right]$. Then, by applying Y1, conclude:

$$\ell_j < \ell_i \text{ and } i + 1 \leq j \leq n + m$$

Then, by applying Y2, conclude $\left[\ell_j < \ell_i \text{ and } \ell_i < \ell_j\right]$. Then, by applying Y1, conclude **false**.

Y4 Suppose:

$$1 \leq j \leq i - 1 \text{ for some } j$$

Then, by applying Z1, conclude $\ell_j = \mathtt{d}_{x_j} \approx t_j$.

Finally, conclude the lemma by the following reduction. Recall $\mathtt{d}_y \in \mathsf{Free}(t_i)$ from A4. Then, by introducing Z6, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(t_i) \text{ and } \mathtt{d}_{x_i} \approx t_1 \in L\right]$. Then, by applying Z1, conclude:

$$\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \text{ for some } u$$

Then, by applying Z7, conclude:

$$\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } \left[\left[\mathtt{d}_y \approx u = \ell_j \text{ and } 1 \leq j \leq n + m\right] \text{ for some } j\right]$$

Then, by applying standard inference rules, conclude:

$$\left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } \mathtt{d}_y \approx u = \ell_j \text{ and } 1 \leq j \leq n + m\right] \text{ for some } j$$

Then, by applying Z8, conclude $\left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } \mathtt{d}_y \approx u = \ell_j \text{ and } \left[1 \leq j \leq i - 1 \text{ or } i = j \text{ or } i + 1 \leq j \leq n + m\right]\right]$. Then, by applying standard inference rules, conclude:

$$\begin{array}{l}\left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } \mathtt{d}_y \approx u = \ell_j \text{ and } 1 \leq j \leq i - 1\right] \text{ or} \\ \left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } \mathtt{d}_y \approx u = \ell_j \text{ and } i = j\right] \text{ or} \\ \left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } \mathtt{d}_y \approx u = \ell_j \text{ and } i + 1 \leq j \leq n + m\right]\end{array}$$

Then, by applying Z0, conclude:

$$\begin{array}{l}\left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } \mathtt{d}_y \approx u = \ell_j \text{ and } 1 \leq j \leq i - 1\right] \text{ or} \\ \textbf{false or} \\ \left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \text{ and } \mathtt{d}_y \approx u = \ell_j \text{ and } i + 1 \leq j \leq n + m\right]\end{array}$$

Then, by applying (Y3), conclude:

$$\left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \textbf{ and } \mathtt{d}_y \approx u = \ell_j \textbf{ and } 1 \leq j \leq i-1\right] \textbf{ or false or false}$$

Then, by applying standard inference rules, conclude:

$$\mathtt{d}_y \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \textbf{ and } \mathtt{d}_y \approx u = \ell_j \textbf{ and } 1 \leq j \leq i-1$$

Then, by applying standard inference rules, conclude $\left[\mathtt{d}_y \approx u = \ell_j \textbf{ and } 1 \leq j \leq i-1\right]$. Then, by applying (Y4), conclude $\left[\mathtt{d}_y \approx u = \mathtt{d}_{x_j} \approx t_j \textbf{ and } 1 \leq j \leq i-1\right]$. Then, by applying standard inference rules, conclude $\left[\mathtt{d}_y = \mathtt{d}_{x_j} \textbf{ and } 1 \leq j \leq i-1\right]$.

((QED.))

B.2 Lemma 2

Proof (of Lemma 2). First, assume:

(A1) $\big[\mathtt{d}_y \in \mathsf{Free}(t)$ **implies** $\delta(y) = \delta'(y)\big]$ **for all** y

Now, prove the lemma by induction on the structure of t.

– **Base:** $\big[t = \mathtt{d}_y$ **for some** $y\big]$ **or** $t = \mathtt{nil}$ **or** $\big[t = d$ **for some** $d\big]$
First, observe:

(Z1) Recall $\big[\mathsf{Free}(\mathtt{d}_y) = \{\mathtt{d}_y\}$ **for all** $y\big]$ from Definition 7 of Free. Then, by applying set theory, conclude $\big[\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_y)$ **for all** $y\big]$.

(Z2) Suppose:

$$t = \mathtt{d}_y \text{ for some } y$$

Then, by applying (Z1), conclude $\big[t = \mathtt{d}_y$ **and** $\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_y)\big]$. Then, by applying substitution, conclude $\mathtt{d}_y \in \mathsf{Free}(t)$. Then, by applying (A1), conclude $\delta(y) = \delta'(y)$. Then, by applying Definition 7 of eval, conclude $\mathsf{eval}_\delta(\mathtt{d}_y) = \mathsf{eval}_{\delta'}(\mathtt{d}_y)$.

(Z3) Suppose:

$$t = \mathtt{d}_y \text{ for some } y$$

Then, by introducing (Z2), conclude $\big[t = \mathtt{d}_y$ **and** $\mathsf{eval}_\delta(\mathtt{d}_y) = \mathsf{eval}_{\delta'}(\mathtt{d}_y)\big]$. Then, by applying substitution, conclude $\mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)$.

(Z4) Recall $\big[\mathsf{eval}_\delta(\mathtt{nil}) = \mathsf{nil}$ **and** $\mathsf{eval}_{\delta'}(\mathtt{nil}) = \mathsf{nil}\big]$ from Definition 7 of eval. Then, by applying substitution, conclude $\mathsf{eval}_\delta(\mathtt{nil}) = \mathsf{eval}_{\delta'}(\mathtt{nil})$.

(Z5) Suppose $t = \mathtt{nil}$. Then, by introducing (Z4), conclude $\big[t = \mathtt{nil}$ **and** $\mathsf{eval}_\delta(\mathtt{nil}) = \mathsf{eval}_{\delta'}(\mathtt{nil})\big]$. Then, by applying substitution, conclude $\mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)$.

(Z6) Recall $\big[\mathsf{eval}_\delta(d) = \mathcal{I}(d)$ **and** $\mathsf{eval}_{\delta'}(d) = \mathcal{I}(d)\big]$ from Definition 7 of eval. Then, by applying substitution, conclude $\mathsf{eval}_\delta(d) = \mathsf{eval}_{\delta'}(d)$.

(Z7) Suppose:

$$t = d \text{ for some } d$$

Then, by applying (Z6), conclude $\big[t = d$ **and** $\mathsf{eval}_\delta(d) = \mathsf{eval}_{\delta'}(d)\big]$. Then, by applying substitution, conclude $\mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)$.

Now, prove the base case of the induction by the following reduction. Recall from **Base**:

$$\big[t = \mathtt{d}_y \text{ for some } y\big] \text{ or } t = \mathtt{nil} \text{ or } \big[t = d \text{ for some } d\big]$$

Then, by applying (Z3), conclude $\big[\mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)$ **or** $t = \mathtt{nil}$ **or** $\big[t = d$ **for some** $d\big]\big]$. Then, by applying (Z5), conclude $\big[\mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)$ **or** $\mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)$ **or** $\big[t = d$ **for some** $d\big]\big]$. Then, by applying (Z7), conclude $\big[\mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)$ **or** $\mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)$ **or** $\mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)\big]$. Then, by applying standard inference rules, conclude $\mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)$.

– **IH:**

$$\Big[\big[\hat{t} \in \mathsf{Arg}(t) \text{ and } \big[\big[\mathtt{d}_y \in \mathsf{Free}(\hat{t}) \text{ implies } \delta(y) = \delta'(y)\big] \text{ for all } y\big]\big] \text{ implies } \mathsf{eval}_\delta(\hat{t}) = \mathsf{eval}_{\delta'}(\hat{t})\Big] \text{ for all } \hat{t}$$

– **Step:** $t = f(t_1, \ldots, t_k)$ **for some** $f, t_1, \ldots, t_k, k$
First, observe:

(Y1) Suppose:

$$t = f(t_1, \ldots, t_k) \text{ for some } f, t_1, \ldots, t_k, k$$

Then, by introducing (A1), conclude:

$$t = f(t_1 , \ldots , t_k) \textbf{ and } \big[\big[\mathtt{d}_y \in \mathsf{Free}(t) \textbf{ implies } \delta(y) = \delta'(y)\big] \textbf{ for all } y\big]$$

Then, by applying substitution, conclude:

$$\big[\mathtt{d}_y \in \mathsf{Free}(f(t_1 , \ldots , t_k)) \textbf{ implies } \delta(y) = \delta'(y)\big] \textbf{ for all } y$$

Then, by applying Definition 7 of Free, conclude:

$$\big[\mathtt{d}_y \in \mathsf{Free}(t_1) \cup \cdots \cup \mathsf{Free}(t_k) \textbf{ implies } \delta(y) = \delta'(y)\big] \textbf{ for all } y$$

Then, by applying set theory, conclude:

$$\big[\big[\big[\mathtt{d}_y \in \mathsf{Free}(t') \textbf{ and } t' \in \{t_1 , \ldots , t_k\}\big] \textbf{ for some } t'\big] \textbf{ implies } \delta(y) = \delta'(y)\big] \textbf{ for all } y$$

Then, by applying standard inference rules, conclude:

$$\big[\big[\mathtt{d}_y \in \mathsf{Free}(t') \textbf{ and } t' \in \{t_1 , \ldots , t_k\}\big] \textbf{ implies } \delta(y) = \delta'(y)\big] \textbf{ for all } y , t'$$

Then, by applying standard inference rules, conclude:

$$\big[t' \in \{t_1 , \ldots , t_k\} \textbf{ and } \big[\big[\mathtt{d}_y \in \mathsf{Free}(t') \textbf{ implies } \delta(y) = \delta'(y)\big] \textbf{ for all } y\big]\big] \textbf{ for all } t'$$

(Y2) Suppose:

$$t = f(t_1 , \ldots , t_k) \textbf{ for some } f , t_1 , \ldots , t_k , k$$

Then, by introducing Definition 7 of Arg, conclude:

$$t = f(t_1 , \ldots , t_k) \textbf{ and } \mathsf{Arg}(f(t_1 , \ldots , t_k)) = \{t_1 , \ldots , t_k\}$$

Then, by applying substitution, conclude $\mathsf{Arg}(t) = \{t_1 , \ldots , t_k\}$.

(Y3) Suppose:

$$t = f(t_1 , \ldots , t_k) \textbf{ for some } f , t_1 , \ldots , t_k , k$$

Then, by applying (Y1), conclude:

$$\begin{gathered} t = f(t_1 , \ldots , t_k) \textbf{ and} \\ \big[\big[t' \in \{t_1 , \ldots , t_k\} \textbf{ and } \big[\big[\mathtt{d}_y \in \mathsf{Free}(t') \textbf{ implies } \delta(y) = \delta'(y)\big] \textbf{ for all } y\big]\big] \textbf{ for all } t'\big] \end{gathered}$$

Then, by applying (Y2), conclude:

$$\begin{gathered} \mathsf{Arg}(t) = \{t_1 , \ldots , t_k\} \textbf{ and} \\ \big[\big[t' \in \{t_1 , \ldots , t_k\} \textbf{ and } \big[\big[\mathtt{d}_y \in \mathsf{Free}(t') \textbf{ implies } \delta(y) = \delta'(y)\big] \textbf{ for all } y\big]\big] \textbf{ for all } t'\big] \end{gathered}$$

Then, by applying substitution, conclude:

$$\big[t' \in \mathsf{Arg}(t) \textbf{ and } \big[\big[\mathtt{d}_y \in \mathsf{Free}(t') \textbf{ implies } \delta(y) = \delta'(y)\big] \textbf{ for all } y\big]\big] \textbf{ for all } t'$$

Then, by applying **IH**, conclude $\big[\big[t' \in \mathsf{Arg}(t) \textbf{ implies } \mathsf{eval}_\delta(t') = \mathsf{eval}_{\delta'}(t')\big] \textbf{ for all } t'\big]$.

(Y4) Suppose:

$$t = f(t_1 , \ldots , t_k) \textbf{ for some } f , t_1 , \ldots , t_k , k$$

Then, by applying Definition 7 of Term, conclude $\mathsf{arity}(f) = k$. Then, by introducing Definition 6 of $\mathcal{I}$, conclude $\big[\mathsf{arity}(f) = k \textbf{ and } \mathcal{I}(f) : \mathbb{D}^{\mathsf{arity}(f)} \to \mathbb{D}\big]$. Then, by applying substitution, conclude $\mathcal{I}(f) : \mathbb{D}^k \to \mathbb{D}$.

(Y5) Suppose:

$$t = f(t_1\,,\,\ldots\,,\,t_k) \;\textbf{for some}\; f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$$

Then, by applying (Y3), conclude:

$$t = f(t_1\,,\,\ldots\,,\,t_k) \;\textbf{and}\; \big[\big[t' \in \mathsf{Arg}(t) \;\textbf{implies}\; \mathsf{eval}_\delta(t') = \mathsf{eval}_{\delta'}(t')\big] \;\textbf{for all}\; t'\big]$$

Then, by applying (Y2), conclude:

$$\mathsf{Arg}(t) = \{t_1\,,\,\ldots\,,\,t_k\} \;\textbf{and}\; \big[\big[t' \in \mathsf{Arg}(t) \;\textbf{implies}\; \mathsf{eval}_\delta(t') = \mathsf{eval}_{\delta'}(t')\big] \;\textbf{for all}\; t'\big]$$

Then, by applying substitution, conclude:

$$\big[t' \in \{t_1\,,\,\ldots\,,\,t_k\} \;\textbf{implies}\; \mathsf{eval}_\delta(t') = \mathsf{eval}_{\delta'}(t')\big] \;\textbf{for all}\; t'$$

Then, by applying set theory, conclude $\big[\mathsf{eval}_\delta(t_1) = \mathsf{eval}_{\delta'}(t_1) \;\textbf{and}\; \cdots \;\textbf{and}\; \mathsf{eval}_\delta(t_k) = \mathsf{eval}_{\delta'}(t_k)\big]$.

(Y6) Suppose:

$$t = f(t_1\,,\,\ldots\,,\,t_k) \;\textbf{for some}\; f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$$

Then, by applying (Y4), conclude $\big[t = f(t_1\,,\,\ldots\,,\,t_k) \;\textbf{and}\; \mathcal{I}(f) : \mathbb{D}^k \to \mathbb{D}\big]$. Then, by applying (Y5), conclude $\big[\mathsf{eval}_\delta(t_1) = \mathsf{eval}_{\delta'}(t_1) \;\textbf{and}\; \cdots \;\textbf{and}\; \mathsf{eval}_\delta(t_k) = \mathsf{eval}_{\delta'}(t_k) \;\textbf{and}\; \mathcal{I}(f) : \mathbb{D}^k \to \mathbb{D}\big]$. Then, by applying set theory, conclude $\mathcal{I}(f)(\mathsf{eval}_\delta(t_1)\,,\,\ldots\,,\,\mathsf{eval}_\delta(t_k)) = \mathcal{I}(f)(\mathsf{eval}_{\delta'}(t_1)\,,\,\ldots\,,\,\mathsf{eval}_{\delta'}(t_k))$. Then, by applying Definition 7 of eval, conclude $\mathsf{eval}_\delta(f(t_1\,,\,\ldots t_k)) = \mathsf{eval}_{\delta'}(f(t_1\,,\,\ldots t_k))$.

Now, prove the inductive step of the induction by the following reduction. Recall from **Step**:

$$t = f(t_1\,,\,\ldots\,,\,t_k) \;\textbf{for some}\; f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$$

Then, by applying (Y6), conclude $\big[t = f(t_1\,,\,\ldots\,,\,t_k) \;\textbf{and}\; \mathsf{eval}_\delta(f(t_1\,,\,\ldots t_k)) = \mathsf{eval}_{\delta'}(f(t_1\,,\,\ldots t_k))\big]$. Then, by applying substitution, conclude $\mathsf{eval}_\delta(t) = \mathsf{eval}_{\delta'}(t)$.

(‖ QED. ‖)

B.3 Lemma 3

Proof (of Lemma 3). First, assume:

(A1) $\left[\mathtt{d}_y \in \mathsf{Free}(t)\ \textbf{implies}\ \delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\right]\ \textbf{for all}\ y$

(A2) $\mathtt{nil}\text{-}\mathsf{Free}(t)$

Now, prove the lemma by induction on the structure of t.

– **Base:** $\left[t = \mathtt{d}_y\ \textbf{for some}\ y\right]\ \textbf{or}\ t = \mathtt{nil}\ \textbf{or}\ \left[t = d\ \textbf{for some}\ d\right]$
First, observe:

(Z1) Recall $\left[\mathsf{Free}(\mathtt{d}_y) = \{\mathtt{d}_y\}\ \textbf{for all}\ y\right]$ from Definition 7 of Free. Then, by applying set theory, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_y)\ \textbf{for all}\ y\right]$.

(Z2) Suppose:

$$t = \mathtt{d}_y\ \textbf{for some}\ y$$

Then, by applying (Z1), conclude $\left[t = \mathtt{d}_y\ \textbf{and}\ \mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_y)\right]$. Then, by applying substitution, conclude $\mathtt{d}_y \in \mathsf{Free}(t)$. Then, by applying (A1), conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}$. Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude $\mathsf{eval}_\delta(\mathtt{d}_y) \neq \mathsf{nil}$. Then, by applying set theory, conclude $\mathsf{eval}_\delta(\mathtt{d}_y) \notin \{\mathsf{nil}\}$. Then, by introducing Definition 7 of eval, conclude $\left[\mathsf{eval}_\delta(\mathtt{d}_y) \notin \{\mathsf{nil}\}\ \textbf{and}\ \mathsf{eval}_\delta(\mathtt{d}_y) \in \mathbb{D} \cup \{\mathsf{nil}\}\right]$. Then, by applying set theory, conclude $\mathsf{eval}_\delta(\mathtt{d}_y) \in \mathbb{D}$.

(Z3) Suppose:

$$t = \mathtt{d}_y\ \textbf{for some}\ y$$

Then, by introducing (Z2), conclude $\left[t = \mathtt{d}_y\ \textbf{and}\ \mathsf{eval}_\delta(\mathtt{d}_y) \in \mathbb{D}\right]$. Then, by applying substitution, conclude $\mathsf{eval}_\delta(t) \in \mathbb{D}$.

(Z4) Suppose $t = \mathtt{nil}$. Then, by introducing Definition 7 of nil-Free, conclude:

$$t = \mathtt{nil}\ \textbf{and}\ \left[\textbf{not}\ \mathtt{nil}\text{-}\mathsf{Free}(\mathtt{nil})\right]$$

Then, by applying substitution, conclude $\left[\textbf{not}\ \mathtt{nil}\text{-}\mathsf{Free}(t)\right]$. Then, by introducing (A2), conclude $\left[\left[\textbf{not}\ \mathtt{nil}\text{-}\mathsf{Free}(t)\right]\ \textbf{and}\ \mathtt{nil}\text{-}\mathsf{Free}(t)\right]$. Then, by applying standard inference rules, conclude **false**.

(Z5) Recall $\left[\mathsf{eval}_\delta(d) = \mathcal{I}(d)\ \textbf{for all}\ d\right]$ from Definition 7 of eval. Then, by applying Definition 6 of $\mathcal{I}$, conclude $\left[\mathsf{eval}_\delta(d) \in \mathbb{D}\ \textbf{for all}\ d\right]$.

(Z6) Suppose:

$$t = d\ \textbf{for some}\ d$$

Then, by applying (Z5), conclude $\left[t = d\ \textbf{and}\ \mathsf{eval}_\delta(d) \in \mathbb{D}\right]$. Then, by applying substitution, conclude $\mathsf{eval}_\delta(t) \in \mathbb{D}$.

Now, prove the base case of the induction by the following reduction. Recall from **Base**:

$$\left[t = \mathtt{d}_y\ \textbf{for some}\ y\right]\ \textbf{or}\ t = \mathtt{nil}\ \textbf{or}\ \left[t = d\ \textbf{for some}\ d\right]$$

Then, by applying (Z3), conclude $\left[\mathsf{eval}_\delta(t) \in \mathbb{D}\ \textbf{or}\ t = \mathtt{nil}\ \textbf{or}\ \left[t = d\ \textbf{for some}\ d\right]\right]$. Then, by applying (Z4), conclude $\left[\mathsf{eval}_\delta(t) \in \mathbb{D}\ \textbf{or false or}\ \left[t = d\ \textbf{for some}\ d\right]\right]$. Then, by applying standard inference rules, conclude $\left[\mathsf{eval}_\delta(t) \in \mathbb{D}\ \textbf{or}\ \left[t = d\ \textbf{for some}\ d\right]\right]$. Then, by applying (Z6), conclude:

$$\mathsf{eval}_\delta(t) \in \mathbb{D}\ \textbf{or}\ \mathsf{eval}_\delta(t) \in \mathbb{D}$$

Then, by applying standard inference rules, conclude $\mathsf{eval}_\delta(t) \in \mathbb{D}$.

– **IH:**

$$\left[\begin{array}{l}\big[\hat{t} \in \mathsf{Arg}(t) \textbf{ and } \big[\big[\mathtt{d}_y \in \mathsf{Free}(\hat{t}) \textbf{ implies } \delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\big] \textbf{ for all } y\big] \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(\hat{t})\big] \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \textbf{implies } \mathsf{eval}_\delta(\hat{t}) \in \mathbb{D}\end{array}\right] \textbf{ for all } \hat{t}$$

– **Step:** $t = f(t_1\,,\,\ldots\,,\,t_k)$ **for some** $f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$
First, observe:

(Y1) Suppose:

$$t = f(t_1\,,\,\ldots\,,\,t_k) \textbf{ for some } f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$$

Then, by introducing (A1), conclude:

$$t = f(t_1\,,\,\ldots\,,\,t_k) \textbf{ and } \big[\big[\mathtt{d}_y \in \mathsf{Free}(t) \textbf{ implies } \delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\big] \textbf{ for all } y\big]$$

Then, by applying substitution, conclude:

$$\big[\mathtt{d}_y \in \mathsf{Free}(f(t_1\,,\,\ldots\,,\,t_k)) \textbf{ implies } \delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\big] \textbf{ for all } y$$

Then, by applying Definition 7 of Free, conclude:

$$\big[\mathtt{d}_y \in \mathsf{Free}(t_1) \cup \cdots \cup \mathsf{Free}(t_k) \textbf{ implies } \delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\big] \textbf{ for all } y$$

Then, by applying set theory, conclude:

$$\big[\big[\big[\mathtt{d}_y \in \mathsf{Free}(t') \textbf{ and } t' \in \{t_1\,,\,\ldots\,,\,t_k\}\big] \textbf{ for some } t'\big] \textbf{ implies } \delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\big] \textbf{ for all } y$$

Then, by applying standard inference rules, conclude:

$$\big[\big[\mathtt{d}_y \in \mathsf{Free}(t') \textbf{ and } t' \in \{t_1\,,\,\ldots\,,\,t_k\}\big] \textbf{ implies } \delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\big] \textbf{ for all } y\,,\,t'$$

Then, by applying standard inference rules, conclude:

$$\big[t' \in \{t_1\,,\,\ldots\,,\,t_k\} \textbf{ and } \big[\big[\mathtt{d}_y \in \mathsf{Free}(t') \textbf{ implies } \delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\big] \textbf{ for all } y\big]\big] \textbf{ for all } t'$$

(Y2) Suppose:

$$t = f(t_1\,,\,\ldots\,,\,t_k) \textbf{ for some } f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$$

Then, by introducing (A2), conclude $\big[t = f(t_1\,,\,\ldots\,,\,t_k) \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(t)\big]$. Then, by applying substitution, conclude $\mathtt{nil}\text{-}\mathsf{Free}(f(t_1\,,\,\ldots\,,\,t_k))$. Then, by applying Definition 7 of nil-Free, conclude $\big[\mathtt{nil}\text{-}\mathsf{Free}(t_1) \textbf{ and } \cdots \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(t_k)\big]$. Then, by applying set theory, conclude:

$$\big[t' \in \{t_1\,,\,\ldots\,,\,t_k\} \textbf{ implies } \mathtt{nil}\text{-}\mathsf{Free}(t')\big] \textbf{ for all } t'$$

(Y3) Suppose:

$$t = f(t_1\,,\,\ldots\,,\,t_k) \textbf{ for some } f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$$

Then, by introducing Definition 7 of Arg, conclude:

$$t = f(t_1\,,\,\ldots\,,\,t_k) \textbf{ and } \mathsf{Arg}(f(t_1\,,\,\ldots\,,\,t_k)) = \{t_1\,,\,\ldots\,,\,t_k\}$$

Then, by applying substitution, conclude $\mathsf{Arg}(t) = \{t_1\,,\,\ldots\,,\,t_k\}$.

(Y4) Suppose:

$$t = f(t_1\,,\,\ldots\,,\,t_k) \textbf{ for some } f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$$

Then, by applying (Y1), conclude:

$$t = f(t_1\,,\,\ldots\,,\,t_k) \textbf{ and}$$
$$\big[\big[t' \in \{t_1\,,\,\ldots\,,\,t_k\} \textbf{ and } \big[\big[\mathtt{d}_y \in \mathsf{Free}(t') \textbf{ implies } \delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\big] \textbf{ for all } y\big]\big] \textbf{ for all } t'\big]$$

Then, by applying (Y2), conclude:

$$t = f(t_1\,,\,\ldots\,,\,t_k)\ \textbf{and}$$
$$\left[\begin{bmatrix} t' \in \{t_1\,,\,\ldots\,,\,t_k\}\ \textbf{and}\ \left[\left[\mathtt{d}_y \in \mathsf{Free}(t')\ \textbf{implies}\ \delta \models^{\mathsf{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\right]\ \textbf{for all}\ y\right] \\ \textbf{and}\ \left[\left[t'' \in \{t_1\,,\,\ldots\,,\,t_k\}\ \textbf{implies}\ \mathtt{nil}\text{-}\mathsf{Free}(t'')\right]\ \textbf{for all}\ t''\right] \end{bmatrix}\right]\ \textbf{for all}\ t'\right]$$

Then, by applying standard inference rules, conclude:

$$t = f(t_1\,,\,\ldots\,,\,t_k)\ \textbf{and}$$
$$\left[\left[t' \in \{t_1\,,\,\ldots\,,\,t_k\}\ \textbf{and}\ \left[\begin{bmatrix}\mathtt{d}_y \in \mathsf{Free}(t')\ \textbf{implies} \\ \delta \models^{\mathsf{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\end{bmatrix}\ \textbf{for all}\ y\right]\ \textbf{and}\ \mathtt{nil}\text{-}\mathsf{Free}(t')\right]\ \textbf{for all}\ t'\right]$$

Then, by applying (Y3), conclude:

$$\mathsf{Arg}(t) = \{t_1\,,\,\ldots\,,\,t_k\}\ \textbf{and}$$
$$\left[\left[t' \in \{t_1\,,\,\ldots\,,\,t_k\}\ \textbf{and}\ \left[\begin{bmatrix}\mathtt{d}_y \in \mathsf{Free}(t')\ \textbf{implies} \\ \delta \models^{\mathsf{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\end{bmatrix}\ \textbf{for all}\ y\right]\ \textbf{and}\ \mathtt{nil}\text{-}\mathsf{Free}(t')\right]\ \textbf{for all}\ t'\right]$$

Then, by applying substitution, conclude:

$$\left[t' \in \mathsf{Arg}(t)\ \textbf{and}\ \left[\begin{bmatrix}\mathtt{d}_y \in \mathsf{Free}(t')\ \textbf{implies} \\ \delta \models^{\mathsf{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\end{bmatrix}\ \textbf{for all}\ y\right]\ \textbf{and}\ \mathtt{nil}\text{-}\mathsf{Free}(t')\right]\ \textbf{for all}\ t'$$

Then, by applying **IH**, conclude $\left[\left[t' \in \mathsf{Arg}(t)\ \textbf{implies}\ \mathsf{eval}_\delta(t') \in \mathbb{D}\right]\ \textbf{for all}\ t'\right]$.

(Y5) Suppose:

$$t = f(t_1\,,\,\ldots\,,\,t_k)\ \textbf{for some}\ f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$$

Then, by applying Definition 7 of TERM, conclude $\mathsf{arity}(f) = k$. Then, by introducing Definition 6 of $\mathcal{I}$, conclude $\left[\mathsf{arity}(f) = k\ \textbf{and}\ \mathcal{I}(f) : \mathbb{D}^{\mathsf{arity}(f)} \to \mathbb{D}\right]$. Then, by applying substitution, conclude $\mathcal{I}(f) : \mathbb{D}^k \to \mathbb{D}$.

(Y6) Suppose:

$$t = f(t_1\,,\,\ldots\,,\,t_k)\ \textbf{for some}\ f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$$

Then, by applying (Y4), conclude:

$$t = f(t_1\,,\,\ldots\,,\,t_k)\ \textbf{and}\ \left[\left[t' \in \mathsf{Arg}(t)\ \textbf{implies}\ \mathsf{eval}_\delta(t') \in \mathbb{D}\right]\ \textbf{for all}\ t'\right]$$

Then, by applying (Y3), conclude:

$$\mathsf{Arg}(t) = \{t_1\,,\,\ldots\,,\,t_k\}\ \textbf{and}\ \left[\left[t' \in \mathsf{Arg}(t)\ \textbf{implies}\ \mathsf{eval}_\delta(t') \in \mathbb{D}\right]\ \textbf{for all}\ t'\right]$$

Then, by applying substitution, conclude $\left[\left[t' \in \{t_1\,,\,\ldots\,,\,t_k\}\ \textbf{implies}\ \mathsf{eval}_\delta(t') \in \mathbb{D}\right]\ \textbf{for all}\ t'\right]$. Then, by applying set theory, conclude $\left[\mathsf{eval}_\delta(t_1) \in \mathbb{D}\ \textbf{and}\ \cdots\ \textbf{and}\ \mathsf{eval}_\delta(t_k) \in \mathbb{D}\right]$.

(Y7) Suppose:

$$t = f(t_1\,,\,\ldots\,,\,t_k)\ \textbf{for some}\ f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$$

Then, by applying (Y5), conclude $\left[t = f(t_1\,,\,\ldots\,,\,t_k)\ \textbf{and}\ \mathcal{I}(f) : \mathbb{D}^k \to \mathbb{D}\right]$. Then, by applying (Y6), conclude $\left[\mathsf{eval}_\delta(t_1) \in \mathbb{D}\ \textbf{and}\ \cdots\ \textbf{and}\ \mathsf{eval}_\delta(t_k)\ \textbf{and}\ \mathcal{I}(f) : \mathbb{D}^k \to \mathbb{D}\right]$. Then, by applying set theory, conclude $\mathcal{I}(f)(\mathsf{eval}_\delta(t_1)\,,\,\ldots\,,\,\mathsf{eval}_\delta(t_k)) \in \mathbb{D}$. Then, by applying Definition 7 of eval, conclude $\mathsf{eval}_\delta(f(t_1\,,\,\ldots t_k)) \in \mathbb{D}$.

Now, prove the inductive step of the induction by the following reduction. Recall from **Step**:

$$t = f(t_1\,,\,\ldots\,,\,t_k)\ \textbf{for some}\ f\,,\,t_1\,,\,\ldots\,,\,t_k\,,\,k$$

Then, by applying (Y7), conclude $\left[t = f(t_1\,,\,\ldots\,,\,t_k)\ \textbf{and}\ \mathsf{eval}_\delta(f(t_1\,,\,\ldots t_k))\right]$. Then, by applying substitution, conclude $\mathsf{eval}_\delta(t) \in \mathbb{D}$.

⟪ QED. ⟫

B.4 Lemma 4

Proof (of Lemma 4). First, assume:

(A1) Pre

Next, observe:

(Z1) Suppose **true**. Then, by applying arithmetic, conclude $1 \geq 1$. Then, by applying replacement, conclude $(1 \geq 1)[P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$. Then, by applying replacement, conclude:

$$(i[i \leftarrow 1] \geq 1)[P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$$

Then, by applying replacement, conclude $(i \geq 1)[i \leftarrow 1][P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$.

(Z2) Suppose:

$$\delta \in [\![\top]\!] \textbf{ for some } \delta$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \models^{\text{dc}} \top$. Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude **true**. Then, by applying standard inference rules, conclude:

$$\delta \models^{\text{dc}} \ell_1 \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \ell_0 \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_0} \not\approx \mathtt{nil}$$

Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude $\delta \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_0 \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_0} \not\approx \mathtt{nil}$. Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_0 \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_0} \not\approx \mathtt{nil}]\!]$.

(Z3) Suppose:

$$\delta \in [\![\top]\!] \textbf{ for some } \delta$$

Then, by a reduction similar to (Z2), conclude $\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_0} \approx \mathcal{I}^{-1}(\delta'(x_0))]\!]$.

(Z4) Recall from (Z2):

$$\big[\delta \in [\![\top]\!] \textbf{ implies } \delta \in [\![\ell_1 \wedge \cdots \wedge \ell_0 \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_0} \not\approx \mathtt{nil}]\!]\big] \textbf{ for all } \delta$$

Then, by applying set theory, conclude $[\![\top]\!] \subseteq [\![\ell_1 \wedge \cdots \wedge \ell_0 \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_0} \not\approx \mathtt{nil}]\!]$. Then, by applying Proposition 3, conclude $\top \Rightarrow \ell_1 \wedge \cdots \wedge \ell_0 \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_0} \not\approx \mathtt{nil}$.

(Z5) By a reduction similar to (Z4), conclude $\top \Rightarrow \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_0} \approx \mathcal{I}^{-1}(\delta'(x_0))$.

(Z6) Recall $\vdash \{\top\}\ \mathtt{skip}\ \{\top\}$ from Axiom–Skip in Figure 7. Then, by introducing (Z4), conclude:

$$\vdash \{\top\}\ \mathtt{skip}\ \{\top\} \textbf{ and } \top \Rightarrow \ell_1 \wedge \cdots \wedge \ell_0 \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_0} \not\approx \mathtt{nil}$$

Then, by applying Rule–Consequence in Figure 7, conclude:

$$\vdash \{\top\}\ \mathtt{skip}\ \{\ell_1 \wedge \cdots \wedge \ell_0 \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_0} \not\approx \mathtt{nil}\}$$

Then, by applying replacement, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_0 \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_0} \not\approx \mathtt{nil}\})[P \leftarrow \mathtt{skip}]$$

Then, by applying arithmetic, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{1-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{1-1}} \not\approx \mathtt{nil}\})[P \leftarrow \mathtt{skip}]$$

Then, by applying replacement, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\})[i \leftarrow 1][P \leftarrow \mathtt{skip}]$$

(Z7) By a reduction similar to (Z6), conclude:

$$(\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\})[i \leftarrow 1][P \leftarrow \mathtt{skip}]$$

(Z8) Recall $(\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\})[i \leftarrow 1][P \leftarrow \mathtt{skip}]$ from (Z7). Then, by applying standard inference rules, conclude:

$$\begin{array}{c}\left[\delta' \overset{\text{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and}\ \text{Dom}(\delta') = \{x_1, \ldots, x_n\}\right]\ \textbf{implies} \\ (\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\})[i \leftarrow 1][P \leftarrow \mathtt{skip}]\end{array}$$

Then, by applying replacement, conclude:

$$\left[\left[\begin{array}{l}\left[\delta' \overset{\text{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and}\ \text{Dom}(\delta') = \{x_1, \ldots, x_n\}\right] \\ \quad \textbf{implies}\ \ \vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{array}{c}\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\end{array}\right\} \\ \qquad\qquad \textbf{for all}\ \delta'\end{array}\right]\right][i \leftarrow 1][P \leftarrow \mathtt{skip}]$$

(Z9) Recall $(i \geq 1)[i \leftarrow 1][P \leftarrow \mathtt{skip}]$ from (Z1). Then, by introducing (Z6), conclude:

$$\begin{array}{l}(i \geq 1)[i \leftarrow 1][P \leftarrow \mathtt{skip}] \\ \textbf{and}\ \ (\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\})[i \leftarrow 1][P \leftarrow \mathtt{skip}]\end{array}$$

Then, by introducing (Z8), conclude:

$$\begin{array}{l}(i \geq 1)[i \leftarrow 1][P \leftarrow \mathtt{skip}] \\ \textbf{and}\ \ (\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\})[i \leftarrow 1][P \leftarrow \mathtt{skip}] \\ \textbf{and}\ \ \left[\left[\begin{array}{l}\left[\delta' \overset{\text{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and}\ \text{Dom}(\delta') = \{x_1, \ldots, x_n\}\right] \\ \quad \textbf{implies}\ \ \vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{array}{c}\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\end{array}\right\} \\ \qquad\qquad \textbf{for all}\ \delta'\end{array}\right]\right][i \leftarrow 1][P \leftarrow \mathtt{skip}]\end{array}$$

Then, by applying replacement, conclude:

$$\left[\begin{array}{l}i \geq 1 \\ \textbf{and}\ \ \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\ \textbf{and}\ \ \left[\left[\begin{array}{l}\left[\delta' \overset{\text{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and}\ \text{Dom}(\delta') = \{x_1, \ldots, x_n\}\right] \\ \quad \textbf{implies}\ \ \vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{array}{c}\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\end{array}\right\} \\ \qquad\qquad \textbf{for all}\ \delta'\end{array}\right]\right]\end{array}\right][i \leftarrow 1][P \leftarrow \mathtt{skip}]$$

Then, by applying the definition of Inv_1 in Figure 10, conclude $\mathsf{Inv}_1[i \leftarrow 1][P \leftarrow \mathtt{skip}]$.

Finally, conclude the lemma by the following reduction. Recall Pre from (A1). Then, by introducing (Z9), conclude $\left[\mathsf{Pre}\ \textbf{and}\ \mathsf{Inv}_1[i \leftarrow 1][P \leftarrow \mathtt{skip}]\right]$.

(⟪ QED. ⟫)

B.5 Lemma 5

Proof (of Lemma 5). First, assume:

(A1) Pre

(A2) Inv_1

(A3) $i \leq n$

(A4) $n - i = z$

(A5) $\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}$

Next, observe:

(Z1) Recall Pre from (A1). Then, by applying the definition of Pre in Figure 10, conclude:

$$\begin{array}{l}\prec_L \text{ is a strict partial order} \\ \textbf{and } \big[\big[\big[\mathtt{d}_x \approx t \in L \textbf{ and } \mathtt{d}_y \in \mathsf{Free}(t)\big] \textbf{ implies } \big[\mathtt{d}_y \approx u \prec_L \mathtt{d}_x \approx t \textbf{ for some } u\big]\big] \textbf{ for all } x, y, t\big] \\ \textbf{and } < \text{ is a linear extension of } \prec_L \\ \textbf{and } L = \{\ell_j \mid 1 \leq j \leq n+m\} \\ \textbf{and } \underbrace{\mathtt{d}_{x_1} \approx t_1}_{\ell_1} < \cdots < \underbrace{\mathtt{d}_{x_n} \approx t_n}_{\ell_n} < \ell_{n+1} < \ldots < \ell_{n+m} \\ \textbf{and } \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} = \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\} \\ \textbf{and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_1) \textbf{ and } \cdots \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_{n+m})\end{array}$$

(Z2) Recall Inv_1 from (A2). Then, by applying the definition of Inv_1 in Figure 10, conclude:

$$\begin{array}{l} i \geq 1 \\ \textbf{and } \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\ \textbf{and } \left[\left[\begin{array}{l}[\delta' \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}] \textbf{ implies} \\ \vdash_{\mathrm{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\}\end{array}\right] \textbf{ for all } \delta'\right]\end{array}$$

(Z3) Recall $i \geq 1$ from (Z2). Then, by applying replacement, conclude $(i \geq 1)[P \leftarrow P\ \mathtt{;\ if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->\ skip\ fi}]$. Then, by applying arithmetic, conclude $(i+1 \geq 1)[P \leftarrow P\ \mathtt{;\ if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->\ skip\ fi}]$. Then, by applying replacement, conclude $(i[i \leftarrow i+1] \geq 1)[P \leftarrow P\ \mathtt{;\ if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->\ skip\ fi}]$. Then, by applying replacement, conclude $(i \geq 1)[i \leftarrow i+1][P \leftarrow P\ \mathtt{;\ if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->\ skip\ fi}]$.

(Z4) Recall $\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}$ from (A5). Then, by applying set theory, conclude:

$$\big[\mathtt{d}_{x_i} = \mathtt{d}_{x_j} \textbf{ and } 1 \leq j < i\big] \textbf{ for some } j$$

Then, by applying arithmetic, conclude $\big[\mathtt{d}_{x_i} = \mathtt{d}_{x_j} \textbf{ and } 1 \leq j \leq i-1\big]$.

(Z5) Suppose:

$$\big[\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\big] \textbf{ for some } \delta$$

Then, by introducing (Z4), conclude:

$$\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \textbf{ and } \big[\big[\mathtt{d}_{x_i} = \mathtt{d}_{x_j} \textbf{ and } 1 \leq j \leq i-1\big] \textbf{ for some } j\big]$$

Then, by applying standard inference rules, conclude:

$$\big[\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \textbf{ and } \mathtt{d}_{x_i} = \mathtt{d}_{x_j} \textbf{ and } 1 \leq j \leq i-1\big] \textbf{ for some } j$$

Then, by applying substitution, conclude $\big[\delta \models^{\mathrm{dc}} \mathtt{d}_{x_j} \not\approx \mathtt{nil} \textbf{ and } \mathtt{d}_{x_i} = \mathtt{d}_{x_j}\big]$. Then, by applying substitution, conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_{x_i} \not\approx \mathtt{nil}$.

(Z6) Suppose:

$$\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i]\!] \textbf{ for some } \delta$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i$. Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude:

$$\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_i} \approx t_i$$

Then, by applying (Z5), conclude:

$$\begin{gathered}\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \\ \textbf{and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_i} \approx t_i \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_i} \not\approx \mathtt{nil}\end{gathered}$$

Then, by applying (Z1), conclude:

$$\begin{gathered}\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \\ \textbf{and } \delta \models^{\mathrm{dc}} \ell_i \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_i} \not\approx \mathtt{nil}\end{gathered}$$

Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude $\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}$. Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}]\!]$.

(Z7) Recall from (Z6):

$$\left[\begin{array}{l}\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i]\!] \\ \quad \textbf{implies } \delta \in [\![\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}]\!]\end{array}\right] \textbf{ for all } \delta$$

Then, by applying set theory, conclude:

$$\begin{gathered}[\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i]\!] \\ \subseteq [\![\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}]\!]\end{gathered}$$

Then, by applying Proposition 3, conclude:

$$\begin{gathered}\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i \\ \Rightarrow \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\end{gathered}$$

Then, by introducing Axiom–Skip in Figure 7, conclude:

$$\left[\begin{array}{l}\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i \\ \Rightarrow \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\end{array}\right]$$

$$\textbf{and} \left[\begin{array}{l}\vdash \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\} \\ \quad \mathtt{skip} \\ \quad \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\}\end{array}\right]$$

Then, by applying Rule–Consequence in Figure 7, conclude:

$$\begin{gathered}\vdash \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i\} \\ \mathtt{skip} \\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\}\end{gathered}$$

Then, by applying Rule–Failure in Figure 7, conclude:

$$\begin{gathered}\vdash \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\ \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi} \\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\}\end{gathered}$$

Z8 Recall from Z2:

$$\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\}$$

Then, by introducing Z7, conclude:

$$\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\}$$
$$\textbf{and} \quad \left[\begin{array}{l} \vdash \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\ \quad \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi} \\ \quad \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\} \end{array}\right]$$

Then, by applying Rule–Composition in Figure 7, conclude:

$$\vdash \{\top\}\ P\ \mathtt{;}\ \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\}$$

Then, by applying replacement, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\})[P \leftarrow P\ \mathtt{;}\ \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$$

Then, by applying arithmetic, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i+1-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i+1-1}} \not\approx \mathtt{nil}\})[P \leftarrow P\ \mathtt{;}\ \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$$

Then, by applying replacement, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\})[i \leftarrow i+1][P \leftarrow P\ \mathtt{;}\ \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$$

Z9 Suppose:

$$\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\ \textbf{for some}\ \delta\,,\ \delta'$$

Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude:

$$\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1))\ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\mathrm{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))$$

Then, by introducing Z4, conclude:

$$\begin{array}{c} \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1))\ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\mathrm{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \textbf{and}\ \left[\left[\mathtt{d}_{x_i} = \mathtt{d}_{x_j}\ \textbf{and}\ 1 \leq j \leq i-1\right]\ \textbf{for some}\ j\right] \end{array}$$

Then, by applying standard inference rules, conclude:

$$\left[\begin{array}{c} \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1))\ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\mathrm{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \textbf{and}\ \mathtt{d}_{x_i} = \mathtt{d}_{x_j}\ \textbf{and}\ 1 \leq j \leq i-1 \end{array}\right]\ \textbf{for some}\ j$$

Then, by applying substitution, conclude $\left[\delta \models^{\mathrm{dc}} \mathtt{d}_{x_j} \approx \mathcal{I}^{-1}(\delta'(x_j))\ \textbf{and}\ \mathtt{d}_{x_i} = \mathtt{d}_{x_j}\right]$. Then, by applying substitution, conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))$.

Z0 Suppose:

$$\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))]\!]\ \textbf{for some}\ \delta\,,\ \delta'$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))$. Then, by applying Z9, conclude $\left[\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\ \textbf{and}\ \delta \models^{\mathrm{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\right]$. Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))$. Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))]\!]$.

Y1 Recall from Z0:

$$\begin{array}{c} \delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))]\!] \\ \textbf{implies}\ \delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))]\!] \end{array}$$

Then, by applying set theory, conclude:

$$[\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))]\!] \subseteq [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))]\!]$$

Then, by applying Proposition 3, conclude:

$$\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \Rightarrow \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))$$

(Y2) Suppose:

$$\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\}\ \textbf{for some}\ \delta'$$

Then, by introducing (Y1), conclude:

$$\begin{array}{l}\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\} \\ \textbf{and}\ \ \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \Rightarrow \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\end{array}$$

Then, by applying Rule–Consequence in Figure 7, conclude:

$$\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\}$$

(Y3) Recall $i \leq n$ from (A3). Then, by introducing (Z2), conclude, $1 \leq i \leq n$.

(Y4) Recall $1 \leq i \leq n$ from (Y3). Then, by applying arithmetic, conclude $1 \leq i \leq n + m$.

(Y5) Suppose:

$$\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))]\!]\ \textbf{for some}\ \delta\,,\ \delta'$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))$ Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude:

$$\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1))\ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))$$

(Y6) Suppose:

$$\mathtt{d}_y \in \mathsf{Free}(t_i)\ \textbf{for some}\ y$$

Then, by introducing (A1)(A2)(A3), conclude $\big[\mathtt{d}_y \in \mathsf{Free}(t_i)\ \textbf{and}\ \mathsf{Pre}\ \textbf{and}\ \mathsf{Inv}_1\ \textbf{and}\ i \leq n\big]$. Then, by applying Lemma 1, conclude $\big[\big[\mathtt{d}_y = \mathtt{d}_{x_j}\ \textbf{and}\ 1 \leq j \leq i-1\big]\ \textbf{for some}\ j\big]$.

(Y7) Suppose:

$$\delta \models^{\text{dc}} \mathtt{d}_{x_j} \approx \mathcal{I}^{-1}(\delta'(x_j))\ \textbf{for some}\ \delta\,,\ \delta'\,,\ j$$

Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude $\mathsf{eval}_\delta(\mathtt{d}_{x_j}) = \mathsf{eval}_\delta(\mathcal{I}^{-1}(\delta'(x_j))) \neq \mathsf{nil}$. Then, by applying Definition 7 of eval, conclude $\delta(x_j) = \mathcal{I}(\mathcal{I}^{-1}(\delta'(x_j)))$. Then, by applying Definition 6 of $\mathcal{I}$, conclude $\delta(x_j) = \delta'(x_j)$.

(Y8) Suppose:

$$\begin{array}{c}\big[\mathtt{d}_y \in \mathsf{Free}(t_i)\ \textbf{and}\ \delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1))\ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\big] \\ \textbf{for some}\ y\,,\ \delta\,,\ \delta'\end{array}$$

Then, by applying (Y6), conclude:

$$\begin{array}{c}\big[\big[\mathtt{d}_y = \mathtt{d}_{x_j}\ \textbf{and}\ 1 \leq j \leq i-1\big]\ \textbf{for some}\ j\big]\ \textbf{and} \\ \delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1))\ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\end{array}$$

Then, by applying standard inference rules, conclude:

$$\left[\begin{array}{c}\mathtt{d}_y = \mathtt{d}_{x_j}\ \textbf{and}\ 1 \leq j \leq i-1\ \textbf{and} \\ \delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1))\ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\end{array}\right]\ \textbf{for some}\ j$$

Then, by applying substitution, conclude $\big[\mathtt{d}_y = \mathtt{d}_{x_j}\ \textbf{and}\ \delta \models^{\text{dc}} \mathtt{d}_{x_j} \approx \mathcal{I}^{-1}(\delta'(x_j))\big]$. Then, by applying (Y7), conclude $\big[\mathtt{d}_y = \mathtt{d}_{x_j}\ \textbf{and}\ \delta(x_j) = \delta'(x_j)\big]$. Then, by applying substitution, conclude $\delta(y) = \delta'(y)$.

(Y9) Suppose:

$$\big[\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1))\ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\big]\ \textbf{for some}\ \delta\,,\ \delta'$$

Then, by applying (Y8), conclude $\big[\big[\mathtt{d}_y \in \mathsf{Free}(t_i)\ \textbf{implies}\ \delta(y) = \delta'(y)\big]\ \textbf{for all}\ y\big]$. Then, by applying Lemma 2, conclude $\mathsf{eval}_\delta(t_i) = \mathsf{eval}_{\delta'}(t_i)$.

(Y0) Suppose:

$$\left[\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\right] \textbf{ for some } \delta\,,\, \delta'$$

Then, by applying standard inference rules, conclude:

$$\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))$$

Then, by applying (Z9), conclude $\delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))$. Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude $\mathsf{eval}_\delta(\mathtt{d}_{x_i}) = \mathsf{eval}_\delta(\mathcal{I}^{-1}(\delta'(x_i))) \neq \mathsf{nil}$. Then, by applying Definition 7 of eval, conclude:

$$\delta(x_i) = \mathcal{I}(\mathcal{I}^{-1}(\delta'(x_i)))$$

Then, by applying Definition 6 of $\mathcal{I}$, conclude $\delta(x_i) = \delta'(x_i)$.

(X1) Suppose:

$$\left[\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\right] \textbf{ for some } \delta\,,\, \delta'$$

Then, by applying (Y9), conclude:

$$\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i)) \textbf{ and } \mathsf{eval}_\delta(t_i) = \mathsf{eval}_{\delta'}(t_i)$$

Then, by applying (Y0), conclude $\left[\delta(x_i) = \delta'(x_i) \textbf{ and } \mathsf{eval}_\delta(t_i) = \mathsf{eval}_{\delta'}(t_i)\right]$.

(X2) Suppose:

$$\delta' \models^{\text{dc}} \ell_i \textbf{ for some } \delta'$$

Then, by applying (Z1), conclude $\delta' \models^{\text{dc}} \mathtt{d}_{x_i} \approx t_i$. Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude:

$$\mathsf{eval}_{\delta'}(\mathtt{d}_{x_i}) = \mathsf{eval}_{\delta'}(t_i) \neq \mathsf{nil}$$

Then, by applying Definition 7 of eval, conclude $\delta'(x_i) = \mathsf{eval}_{\delta'}(t_i) \neq \mathsf{nil}$.

(X3) Suppose:

$$\left[\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))]\!] \textbf{ and } \delta' \models^{\text{dc}} \ell_i\right] \textbf{ for some } \delta\,,\, \delta'$$

Then, by applying (Y5), conclude:

$$\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i)) \textbf{ and } \delta' \models^{\text{dc}} \ell_i$$

Then, by applying (X1), conclude $\left[\delta(x_i) = \delta'(x_i) \textbf{ and } \mathsf{eval}_\delta(t_i) = \mathsf{eval}_{\delta'}(t_i) \textbf{ and } \delta' \models^{\text{dc}} \ell_i\right]$. Then, by applying (X2), conclude $\left[\delta(x_i) = \delta'(x_i) \textbf{ and } \mathsf{eval}_\delta(t_i) = \mathsf{eval}_{\delta'}(t_i) \textbf{ and } \delta'(x_i) = \mathsf{eval}_{\delta'}(t_i) \neq \mathsf{nil}\right]$. Then, by applying substitution, conclude $\left[\delta(x_i) = \delta'(x_i) \textbf{ and } \delta'(x_i) = \mathsf{eval}_\delta(t_i) \neq \mathsf{nil}\right]$. Then, by applying substitution, conclude $\delta(x_i) = \mathsf{eval}_\delta(t_i) \neq \mathsf{nil}$. Then, by applying Definition 7 of eval, conclude $\mathsf{eval}_\delta(\mathtt{d}_{x_i}) = \mathsf{eval}_\delta(t_i) \neq \mathsf{nil}$. Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude $\delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx t_i$. Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \in [\![\mathtt{d}_{x_i} \approx t_i]\!]$.

(X4) Suppose:

$$\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ for some } \delta'$$

Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude $\left[\delta' \models^{\text{dc}} \ell_1 \textbf{ and } \cdots \textbf{ and } \delta' \models^{\text{dc}} \ell_{n+m}\right]$. Then, by introducing (Y4), conclude $\left[\delta' \models^{\text{dc}} \ell_1 \textbf{ and } \cdots \textbf{ and } \delta' \models^{\text{dc}} \ell_{n+m} \textbf{ and } 1 \leq i \leq n+m\right]$. Then, by applying substitution, conclude $\delta' \models^{\text{dc}} \ell_i$. Then, by applying (X3), conclude:

$$\left[\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))]\!] \textbf{ implies } \delta \in [\![\mathtt{d}_{x_i} \approx t_i]\!]\right] \textbf{ for all } \delta$$

Then, by applying set theory, conclude $[\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))]\!] \subseteq [\![\mathtt{d}_{x_i} \approx t_i]\!]$. Then,

by applying Proposition 3, conclude $\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i)) \Rightarrow \mathtt{d}_{x_i} \approx t_i$. Then, by introducing Axiom–Skip in Figure 7, conclude:

$$\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i)) \Rightarrow \mathtt{d}_{x_i} \approx t_i$$
$$\textbf{and} \quad \left[\begin{array}{l} \vdash_{\text{tot}} \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\} \\ \quad \mathtt{skip} \\ \quad \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\} \end{array}\right]$$

Then, by applying Rule–Failure II in Figure 7, conclude:

$$\begin{array}{l} \vdash_{\text{tot}} \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\} \\ \quad \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi} \\ \quad \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\} \end{array}$$

(X5) Suppose:

$$\left[\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and}\ \text{Dom}(\delta') = \{x_1, \ldots, x_n\}\right] \ \textbf{for some}\ \delta'$$

Then, by applying (Z2), conclude:

$$\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and} \ \vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\}$$

Then, by applying (Y2), conclude:

$$\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and} \ \vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\}$$

Then, by applying (X4), conclude:

$$\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\}$$
$$\textbf{and} \quad \left[\begin{array}{l} \vdash_{\text{tot}} \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\} \\ \quad \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi} \\ \quad \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\} \end{array}\right]$$

Then, by applying Rule–Composition in Figure 7, conclude:

$$\vdash_{\text{tot}} \{\top\}\ P\ \mathtt{;}\ \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\}$$

Then, by applying replacement, conclude:

$$(\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\})[P \leftarrow P\ \mathtt{;}\ \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$$

Then, by applying arithmetic, conclude:

$$(\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i+1-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i+1-1}))\})[P \leftarrow P\ \mathtt{;}\ \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$$

Then, by applying replacement, conclude:

$$(\vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{array}{l} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \end{array}\right\})[i \leftarrow i+1][P \leftarrow P\ \mathtt{;}\ \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$$

(X6) Recall from (X5):

$$\left[\begin{array}{c} \left[\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and}\ \text{Dom}(\delta') = \{x_1, \ldots, x_n\}\right] \ \textbf{implies} \\ (\vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{array}{l} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \end{array}\right\})[i \leftarrow i+1][P \leftarrow P\ \mathtt{;}\ \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}] \end{array}\right] \ \textbf{for all}\ \delta'$$

Then, by applying replacement, conclude:

$$\left[\begin{array}{c} \left[\left[\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and}\ \text{Dom}(\delta') = \{x_1, \ldots, x_n\}\right]\right. \\ \left.\textbf{implies}\ \vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{array}{l} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \end{array}\right\}\right] \\ \textbf{for all}\ \delta' \end{array}\right][i \leftarrow i+1]$$
$$[P \leftarrow P\ \mathtt{;}\ \mathtt{if}\ \mathtt{d}_{x_i} \approx t_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$$

(X7) Recall $(i \geq 1)[i \leftarrow i+1][P \leftarrow P \texttt{ ; if } \mathtt{d}_{x_i} \approx t_i \texttt{ -> skip fi}]$ from (Z3). Then, by introducing (Z8), conclude:

$$\begin{array}{l}(i \geq 1)[i \leftarrow i+1][P \leftarrow P \texttt{ ; if } \mathtt{d}_{x_i} \approx t_i \texttt{ -> skip fi}] \\ \textbf{and } (\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\})[i \leftarrow i+1] \\ \qquad\qquad [P \leftarrow P \texttt{ ; if } \mathtt{d}_{x_i} \approx t_i \texttt{ -> skip fi}]\end{array}$$

Then, by introducing (X6), conclude:

$$\begin{array}{l}(i \geq 1)[i \leftarrow i+1][P \leftarrow P \texttt{ ; if } \mathtt{d}_{x_i} \approx t_i \texttt{ -> skip fi}] \\ \textbf{and } (\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\})[i \leftarrow i+1] \\ \qquad\qquad [P \leftarrow P \texttt{ ; if } \mathtt{d}_{x_i} \approx t_i \texttt{ -> skip fi}] \\ \textbf{and } \left[\begin{array}{l}\left[\left[\delta' \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}\right]\right. \\ \quad \textbf{implies } \vdash_{\mathrm{tot}} \{\top\}\ P\ \left\{\begin{array}{l}\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\end{array}\right\}\left.\right] \\ \qquad \textbf{for all } \delta'\end{array}\right][i \leftarrow i+1] \\ \qquad\qquad [P \leftarrow P \texttt{ ; if } \mathtt{d}_{x_i} \approx t_i \texttt{ -> skip fi}]\end{array}$$

Then, by applying replacement, conclude:

$$\left[\begin{array}{l}i \geq 1 \\ \textbf{and } \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\ \textbf{and } \left[\begin{array}{l}\left[\left[\delta' \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}\right]\right. \\ \quad \textbf{implies } \vdash_{\mathrm{tot}} \{\top\}\ P\ \left\{\begin{array}{l}\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\end{array}\right\}\left.\right] \\ \qquad \textbf{for all } \delta'\end{array}\right]\end{array}\right][i \leftarrow i+1] \\ \qquad [P \leftarrow P \texttt{ ; if } \mathtt{d}_{x_i} \approx t_i \texttt{ -> skip fi}]$$

Then, by applying the definition of Inv_1 in Figure 10, conclude:

$$\mathsf{Inv}_1[i \leftarrow i+1][P \leftarrow P \texttt{ ; if } \mathtt{d}_{x_i} \approx t_i \texttt{ -> skip fi}]$$

(X8) Recall $n - i = z$ from (A4). Then, by applying replacement, conclude $n - i = z$. Then, by applying arithmetic, conclude $n - i \leq z$. Then, by applying arithmetic, conclude $n - i - 1 < z$. Then, by applying arithmetic, conclude $n - (i+1) < z$.

Finally, conclude the lemma by the following reduction. Recall Pre from (A1). Then, by introducing (X7), conclude $\left[\mathsf{Pre} \textbf{ and } \mathsf{Inv}_1[i \leftarrow i+1][P \leftarrow P \texttt{ ; if } \mathtt{d}_{x_i} \approx t_i \texttt{ -> skip fi}]\right]$. Then, by introducing (X8), conclude:

$$\mathsf{Pre} \textbf{ and } \mathsf{Inv}_1[i \leftarrow i+1][P \leftarrow P \texttt{ ; if } \mathtt{d}_{x_i} \approx t_i \texttt{ -> skip fi}] \textbf{ and } n - (i+1) < z$$

⟪ QED. ⟫

B.6 Lemma 6

Proof (of Lemma 6). First, assume:

(A1) Pre

(A2) Inv_1

(A3) $i \leq n$

(A4) $n - i = z$

(A5) $\mathtt{d}_{x_i} \notin \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}$

Next, observe:

(Z1) Recall Pre from (A1). Then, by applying the definition of Pre in Figure 10, conclude:

$$\begin{array}{l}
\prec_L \text{ is a strict partial order} \\
\textbf{and } \big[\big[\big[\mathtt{d}_x \approx t \in L \textbf{ and } \mathtt{d}_y \in \mathsf{Free}(t)\big] \textbf{ implies } \big[\mathtt{d}_y \approx u \prec_L \mathtt{d}_x \approx t \textbf{ for some } u\big]\big] \textbf{ for all } x, y, t\big] \\
\textbf{and } < \text{ is a linear extension of } \prec_L \\
\textbf{and } L = \{\ell_j \mid 1 \leq j \leq n+m\} \\
\textbf{and } \underbrace{\mathtt{d}_{x_1} \approx t_1}_{\ell_1} < \cdots < \underbrace{\mathtt{d}_{x_n} \approx t_n}_{\ell_n} < \ell_{n+1} < \ldots < \ell_{n+m} \\
\textbf{and } \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} = \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\} \\
\textbf{and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_1) \textbf{ and } \cdots \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_{n+m})
\end{array}$$

(Z2) Recall Inv_1 from (A2). Then, by applying the definition of Inv_1 in Figure 10, conclude:

$$\begin{array}{l}
i \geq 1 \\
\textbf{and } \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\
\textbf{and } \left[\left[\begin{array}{l}[\delta' \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}] \textbf{ implies} \\ \vdash_{\mathrm{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\}\end{array}\right] \textbf{ for all } \delta'\right]
\end{array}$$

(Z3) Recall $i \geq 1$ from (Z2). Then, by applying replacement, conclude $(i \geq 1)[P \leftarrow P\ ;\ x_i := t_i]$. Then, by applying arithmetic, conclude $(i+1 \geq 1)[P \leftarrow P\ ;\ x_i := t_i]$. Then, by applying replacement, conclude $(i[i \leftarrow i+1] \geq 1)[P \leftarrow P\ ;\ x_i := t_i]$. Then, by applying replacement, conclude:

$$(i \geq 1)[i \leftarrow i+1][P \leftarrow P\ ;\ x_i := t_i]$$

(Z4) Suppose:

$$\mathtt{d}_y \in \mathsf{Free}(t_i) \textbf{ for some } y$$

Then, by introducing (A1)(A2)(A3), conclude $\big[\mathtt{d}_y \in \mathsf{Free}(t_i) \textbf{ and } \mathsf{Pre} \textbf{ and } \mathsf{Inv}_1 \textbf{ and } i \leq n\big]$. Then, by applying Lemma 1, conclude $\big[\big[\mathtt{d}_y = \mathtt{d}_{x_j} \textbf{ and } 1 \leq j \leq i-1\big] \textbf{ for some } j\big]$.

(Z5) Suppose:

$$\big[\mathtt{d}_y \in \mathsf{Free}(t_i) \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\big] \textbf{ for some } y, \delta$$

Then, by applying (Z4), conclude:

$$\big[\big[\mathtt{d}_y = \mathtt{d}_{x_j} \textbf{ and } 1 \leq j \leq i-1\big] \textbf{ for some } j\big] \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}$$

Then, by applying standard inference rules, conclude:

$$\big[\mathtt{d}_y = \mathtt{d}_{x_j} \textbf{ and } 1 \leq j \leq i-1 \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\big] \textbf{ for some } j$$

Then, by applying substitution, conclude $\big[\mathtt{d}_y = \mathtt{d}_{x_j} \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_j} \not\approx \mathtt{nil}\big]$. Then, by applying substitution, conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}$.

(Z6) Recall $i \leq n$ from (A3). Then, by introducing (Z2), conclude, $1 \leq i \leq n$.

(Z7) Recall $1 \leq i \leq n$ from (Z6). Then, by applying arithmetic, conclude $1 \leq i \leq n + m$.

(Z8) Recall $1 \leq i \leq n + m$ from (Z7). Then, by introducing (Z1), conclude:

$$1 \leq i \leq n + m \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_1) \textbf{ and } \cdots \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_{n+m})$$

Then, by applying substitution, conclude $\mathtt{nil}\text{-}\mathsf{Free}(\ell_i)$. Then, by applying (Z1), conclude:

$$\mathtt{nil}\text{-}\mathsf{Free}(\mathtt{d}_{x_i} \approx t_i)$$

Then, by applying Definition 7 of nil-Free, conclude $\mathtt{nil}\text{-}\mathsf{Free}(t_i)$.

(Z9) Suppose:

$$\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \textbf{ for some } \delta$$

Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude $\left[\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\right]$. Then, by applying (Z5), conclude $\left[\left[\mathtt{d}_y \in \mathsf{Free}(t_i) \textbf{ implies } \delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\right] \textbf{ for all } y\right]$. Then, by introducing (Z8), conclude $\left[\left[\left[\mathtt{d}_y \in \mathsf{Free}(t_i) \textbf{ implies } \delta \models^{\mathrm{dc}} \mathtt{d}_y \not\approx \mathtt{nil}\right] \textbf{ for all } y\right] \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(t_i)\right]$. Then, by applying Lemma 3, conclude $\mathsf{eval}_\delta(t_i) \in \mathbb{D}$.

(Z0) Recall $1 \leq i \leq n + m$ from (Z7). Then, by applying set theory, conclude $\ell_i \in \{\ell_j \mid 1 \leq j \leq n + m\}$. Then, by applying (Z1), conclude $\ell_i \in L$. Then, by applying (Z1), conclude $\mathtt{d}_{x_i} \approx t_i \in L$.

(Y1) Recall $1 \leq i \leq n$ from (Z6). Then, by applying (Z1), conclude $\mathtt{d}_{x_1} \approx t_1 < \cdots < \mathtt{d}_{x_{i-1}} \approx t_{i-1} < \mathtt{d}_{x_i} \approx t_i$.

(Y2) Recall $\left[\left[\prec_L \text{ is a strict partial order}\right] \textbf{ and } \left[< \text{ is a linear extension of } \prec_L\right]\right]$ from (Z1). Then, by applying order theory, conclude $\left[\left[< \text{ is a strict total order}\right] \textbf{ and } \prec_L \subseteq <\right]$. Then, by applying order theory, conclude $\left[\left[< \text{ is transitive, asymmetric, and irreflexive}\right] \textbf{ and } \prec_L \subseteq <\right]$.

(Y3) Suppose $\mathtt{d}_{x_i} \in \mathsf{Free}(t_i)$. Then, by introducing (Z0), conclude $\left[\mathtt{d}_{x_i} \in \mathsf{Free}(t_i) \textbf{ and } \mathtt{d}_{x_i} \approx t_i \in L\right]$. Then, by applying (Z1), conclude:

$$\mathtt{d}_{x_i} \approx u \prec_L \mathtt{d}_{x_i} \approx t_i \textbf{ for some } u$$

Then, by applying (Y2), conclude $\mathtt{d}_{x_i} \approx u < \mathtt{d}_{x_i} \approx t_i$. Then, by introducing (Y1), conclude:

$$\mathtt{d}_{x_i} \approx u < \mathtt{d}_{x_i} \approx t_i \textbf{ and } \mathtt{d}_{x_1} \approx t_1 < \cdots < \mathtt{d}_{x_{i-1}} \approx t_{i-1} < \mathtt{d}_{x_i} \approx t_i$$

Then, by applying (Y2), conclude $\left[\mathtt{d}_{x_i} \approx u = \mathtt{d}_{x_1} \approx t_1 \textbf{ or } \cdots \textbf{ or } \mathtt{d}_{x_i} \approx u = \mathtt{d}_{x_{i-1}} \approx t_{i-1}\right]$. Then, by applying standard inference rules, conclude $\left[\mathtt{d}_{x_i} = \mathtt{d}_{x_1} \textbf{ or } \cdots \textbf{ or } \mathtt{d}_{x_i} = \mathtt{d}_{x_{i-1}}\right]$. Then, by applying set theory, conclude $\left[\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_1}\} \textbf{ or } \cdots \textbf{ or } \mathtt{d}_{x_i} \in \{\mathtt{d}_{x_{i-1}}\}\right]$. Then, by applying set theory, conclude $\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_{i-1}}\}$. Then, by applying set theory, conclude $\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}$. Then, by introducing (A5), conclude $\left[\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_j} \mid 1 \leq j < i\} \textbf{ and } \mathtt{d}_{x_i} \notin \{\mathtt{d}_{x_j} \mid 1 \leq j < i\}\right]$. Then, by applying standard inference rules, conclude **false**.

(Y4) Recall $\mathtt{d}_{x_i} \notin \mathsf{Free}(t_i)$ from (Y3). Then, by applying replacement, conclude $t_i = t_i[\mathtt{d}_{x_i} := t_i]$.

(Y5) Suppose:

$$\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \textbf{ for some } \delta$$

Then, by applying (Z9), conclude $\mathsf{eval}_\delta(t_i) \in \mathbb{D}$. Then, by applying Definition 3 of nil, conclude:

$$\mathsf{eval}_\delta(t_i) \neq \mathsf{nil}$$

Then, by applying set theory, conclude $\mathsf{eval}_\delta(t_i) = \mathsf{eval}_\delta(t_i) \neq \mathsf{nil}$. Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude $\delta \models^{\mathrm{dc}} t_i \approx t_i$. Then, by applying replacement, conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_{x_i}[\mathtt{d}_{x_i} := t_i] \approx t_i$. Then, by introducing (Y4), conclude $\left[\delta \models^{\mathrm{dc}} \mathtt{d}_{x_i}[\mathtt{d}_{x_i} := t_i] \approx t_i \textbf{ and } t_i = t_i[\mathtt{d}_{x_i} := t_i]\right]$. Then, by applying substitution, conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_{x_i}[\mathtt{d}_{x_i} := t_i] \approx t_i[\mathtt{d}_{x_i} := t_i]$. Then, by applying replacement, conclude $\delta \models^{\mathrm{dc}} (\mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]$.

(Y6) Suppose:

$$1 \leq j \leq i-1 \text{ for some } j$$

Then, by applying (Z7), conclude $\left[1 \leq j \leq i-1 \text{ and } 1 \leq i \leq n+m\right]$. Then, by applying arithmetic, conclude $1 \leq j \leq n+m$. Then, by applying set theory, conclude $\ell_j \in \{\ell_{j'} \mid 1 \leq j' \leq n+m\}$. Then, by applying (Z1), conclude $\ell_j \in L$.

(Y7) Suppose:

$$1 \leq j \leq i-1 \text{ for some } j$$

Then, by applying (Z6), conclude $\left[1 \leq j \leq i-1 \text{ and } 1 \leq i \leq n\right]$. Then, by applying arithmetic, conclude $1 \leq j \leq n$. Then, by applying (Z1), conclude $\ell_j = \mathtt{d}_{x_j} \approx t_j$.

(Y8) Suppose:

$$1 \leq j \leq i-1 \text{ for some } j$$

Then, by applying (Y6), conclude $\left[1 \leq j \leq i-1 \text{ and } \ell_j \in L\right]$. Then, by applying (Y7), conclude:

$$\ell_j = \mathtt{d}_{x_j} \approx t_j \text{ and } \ell_j \in L$$

Then, by applying substitution, conclude $\mathtt{d}_{x_j} \approx t_j \in L$.

(Y9) Suppose:

$$\left[\mathtt{d}_{x_i} \in \mathsf{Free}(t_j) \text{ and } 1 \leq j \leq i-1\right] \text{ for some } j$$

Then, by applying (Y8), conclude $\left[\mathtt{d}_{x_i} \in \mathsf{Free}(t_j) \text{ and } \mathtt{d}_{x_j} \approx t_j \in L\right]$. Then, by applying (Z1), conclude:

$$\mathtt{d}_{x_i} \approx u \prec_L \mathtt{d}_{x_j} \approx t_j \text{ for some } u$$

Then, by applying (Y2), conclude $\mathtt{d}_{x_i} \approx u < \mathtt{d}_{x_j} \approx t_j$.

(Y0) Suppose:

$$1 \leq j \leq i-1 \text{ for some } j$$

Then, by applying (A3), conclude $1 \leq j \leq n$. Then, by applying (Z1), conclude:

$$\mathtt{d}_{x_1} \approx t_1 < \cdots < \mathtt{d}_{x_{j-1}} \approx t_{j-1} < \mathtt{d}_{x_j} \approx t_j$$

(X1) Suppose:

$$\left[\mathtt{d}_{x_i} \in \mathsf{Free}(t_j) \text{ and } 1 \leq j \leq i-1\right] \text{ for some } j$$

Then, by applying (X1), conclude $\left[1 \leq j \leq i-1 \text{ and } \left[\mathtt{d}_{x_i} \approx u < \mathtt{d}_{x_j} \approx t_j \text{ for some } u\right]\right]$. Then, by applying standard inference rules, conclude:

$$\left[1 \leq j \leq i-1 \text{ and } \mathtt{d}_{x_i} \approx u < \mathtt{d}_{x_j} \approx t_j\right] \text{ for some } u$$

Then, by applying (Y0), conclude $\left[\mathtt{d}_{x_i} \approx u < \mathtt{d}_{x_j} \approx t_j \text{ and } \mathtt{d}_{x_1} \approx t_1 < \cdots < \mathtt{d}_{x_{j-1}} \approx t_{j-1} < \mathtt{d}_{x_j} \approx t_j\right]$. Then, by applying (Y2), conclude $\left[\mathtt{d}_{x_i} \approx u = \mathtt{d}_{x_1} \approx t_1 \text{ or } \cdots \text{ or } \mathtt{d}_{x_i} \approx u = \mathtt{d}_{x_{j-1}} \approx t_{j-1}\right]$. Then, by applying standard inference rules, conclude $\left[\mathtt{d}_{x_i} = \mathtt{d}_{x_1} \text{ or } \cdots \text{ or } \mathtt{d}_{x_i} = \mathtt{d}_{x_{j-1}}\right]$. Then, by applying set theory, conclude $\left[\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_1}\} \text{ or } \cdots \text{ or } \mathtt{d}_{x_i} \in \{\mathtt{d}_{x_{j-1}}\}\right]$. Then, by applying set theory, conclude $\mathtt{d}_{x_i} \in \{x_1, \ldots, x_{j-1}\}$. Then, by applying set theory, conclude $\mathtt{d}_{x_i} \in \{x_1, \ldots, x_j\}$.

(X2) Suppose:

$$\left[\mathtt{d}_{x_i} \in \mathsf{Free}(t_j) \text{ and } 1 \leq j \leq i-1\right] \text{ for some } j$$

Then, by applying (X1), conclude $\left[1 \leq j \leq i-1 \text{ and } \mathtt{d}_{x_i} \in \{x_1, \ldots, x_j\}\right]$. Then, by applying set theory, conclude $\mathtt{d}_{x_i} \in \{x_1, \ldots, x_{i-1}\}$. Then, by applying set theory, conclude $\mathtt{d}_{x_i} \in \{x_{j'} \mid 1 \leq j' < i\}$. Then, by introducing (A5), conclude $\left[\mathtt{d}_{x_i} \in \{x_{j'} \mid 1 \leq j' < i\} \text{ and } \mathtt{d}_{x_i} \notin \{\mathtt{d}_{x_{j'}} \mid 1 \leq j' < i\}\right]$. Then, by applying standard inference rules, conclude **false**.

(X3) Suppose:

$$\mathtt{d}_{x_i} = \mathtt{d}_{x_j} \textbf{ and } 1 \leq j \leq i-1 \textbf{ for some } j$$

Then, by applying set theory, conclude $\left[\mathtt{d}_{x_i} = \mathtt{d}_{x_j} \textbf{ and } \mathtt{d}_{x_j} \in \{\mathtt{d}_{x_{j'}} \mid 1 \leq j' \leq i-1\}\right]$. Then, by applying substitution, conclude $\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_{j'}} \mid 1 \leq j' \leq i-1\}$. Then, by introducing (A5), conclude:

$$\mathtt{d}_{x_i} \in \{\mathtt{d}_{x_{j'}} \mid 1 \leq j' \leq i-1\} \textbf{ and } \mathtt{d}_{x_i} \notin \{\mathtt{d}_{x_{j'}} \mid 1 \leq j' \leq i-1\}$$

Then, by applying standard inference rules, conclude **false**.

(X4) Suppose:

$$1 \leq j \leq i-1 \textbf{ for some } j$$

Then, by applying (X2), conclude $\left[1 \leq j \leq i-1 \textbf{ and } \mathtt{d}_{x_i} \notin \mathsf{Free}(t_j)\right]$. Then, by applying (X3), conclude $\left[\mathtt{d}_{x_i} \neq \mathtt{d}_{x_j} \textbf{ and } \mathtt{d}_{x_i} \notin \mathsf{Free}(t_j)\right]$. Then, by applying replacement, conclude:

$$\mathtt{d}_{x_j} = \mathtt{d}_{x_j}[\mathtt{d}_{x_i} := t_i] \textbf{ and } t_j = t_j[\mathtt{d}_{x_i} := t_i]$$

(X5) Suppose:

$$\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \textbf{ for some } \delta$$

Then, by applying (X4), conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1}[\mathtt{d}_{x_i} := t_i] \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}}[\mathtt{d}_{x_i} := t_i] \not\approx \mathtt{nil}$. Then, by applying replacement, conclude:

$$\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1}[\mathtt{d}_{x_i} := t_i] \not\approx \mathtt{nil}[\mathtt{d}_{x_i} := t_i] \wedge \cdots \wedge \mathtt{d}_{x_{i-1}}[\mathtt{d}_{x_i} := t_i] \not\approx \mathtt{nil}[\mathtt{d}_{x_i} := t_i]$$

Then, by applying replacement, conclude $\delta \models^{\mathrm{dc}} (\mathtt{d}_{x_1} \not\approx \mathtt{nil})[\mathtt{d}_{x_i} := t_i] \wedge \cdots \wedge (\mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil})[\mathtt{d}_{x_i} := t_i]$.

(X6) Suppose:

$$\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \textbf{ for some } \delta$$

Then, by applying (Z1), conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \approx t_1 \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx t_{i-1}$. Then, by applying (X4), conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1}[\mathtt{d}_{x_i} := t_i] \approx t_1[\mathtt{d}_{x_i} := t_i] \wedge \cdots \wedge \mathtt{d}_{x_{i-1}}[\mathtt{d}_{x_i} := t_i] \approx t_{i-1}[\mathtt{d}_{x_i} := t_i]$. Then, by applying replacement, conclude $\delta \models^{\mathrm{dc}} (\mathtt{d}_{x_1} \approx t_1)[\mathtt{d}_{x_i} := t_i] \wedge \cdots \wedge (\mathtt{d}_{x_{i-1}} \approx t_{i-1})[\mathtt{d}_{x_i} := t_i]$. Then, by applying (Z1), conclude $\delta \models^{\mathrm{dc}} \ell_1[\mathtt{d}_{x_i} := t_i] \wedge \cdots \wedge \ell_{i-1}[\mathtt{d}_{x_i} := t_i]$.

(X7) Suppose:

$$\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}]\!] \textbf{ for some } \delta$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}$. Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude:

$$\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}$$

Then, by applying (Y5), conclude:

$$\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \textbf{ and } \delta \models^{\mathrm{dc}} (\mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]$$

Then, by applying (X5), conclude:

$$\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \textbf{ and}$$
$$\delta \models^{\mathrm{dc}} (\mathtt{d}_{x_1} \not\approx \mathtt{nil})[\mathtt{d}_{x_i} := t_i] \wedge \cdots \wedge (\mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil})[\mathtt{d}_{x_i} := t_i] \textbf{ and } \delta \models^{\mathrm{dc}} (\mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]$$

Then, by applying (X6), conclude:

$$\delta \models^{\mathrm{dc}} \ell_1[\mathtt{d}_{x_i} := t_i] \wedge \cdots \wedge \ell_{i-1}[\mathtt{d}_{x_i} := t_i] \textbf{ and}$$
$$\delta \models^{\mathrm{dc}} (\mathtt{d}_{x_1} \not\approx \mathtt{nil})[\mathtt{d}_{x_i} := t_i] \wedge \cdots \wedge (\mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil})[\mathtt{d}_{x_i} := t_i] \textbf{ and } \delta \models^{\mathrm{dc}} (\mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]$$

Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude:

$$\begin{array}{l}\delta \models^{\text{dc}} \ell_1[\mathtt{d}_{x_i} := t_i] \wedge \cdots \wedge \ell_{i-1}[\mathtt{d}_{x_i} := t_i] \\ \quad \wedge (\mathtt{d}_{x_1} \not\approx \mathtt{nil})[\mathtt{d}_{x_i} := t_i] \wedge \cdots \wedge (\mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil})[\mathtt{d}_{x_i} := t_i] \wedge (\mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]\end{array}$$

Then, by applying replacement, conclude:

$$\delta \models^{\text{dc}} (\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude:

$$\delta \in [\![(\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]]\!]$$

(X8) Recall from (X7):

$$\left[\begin{array}{c}\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}]\!] \textbf{ implies} \\ \delta \in [\![(\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]]\!]\end{array}\right] \textbf{ for all } \delta$$

Then, by applying set theory, conclude:

$$\begin{array}{l}[\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}]\!] \\ \subseteq [\![(\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]]\!]\end{array}$$

Then, by applying Proposition 3, conclude:

$$\begin{array}{l}\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \\ \Rightarrow (\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]\end{array}$$

(X9) Suppose:

$$\delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx t_i \textbf{ for some } \delta$$

Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude $\mathsf{eval}_\delta(\mathtt{d}_{x_i}) \neq \mathsf{nil}$. Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude $\delta \models^{\text{dc}} \mathtt{d}_{x_i} \not\approx \mathtt{nil}$.

(X0) Suppose:

$$\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i]\!] \textbf{ for some } \delta$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude:

$$\delta \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i$$

Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude:

$$\delta \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx t_i$$

Then, by applying (X9), conclude:

$$\delta \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_i} \approx t_i \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_i} \not\approx \mathtt{nil}$$

Then, by applying (Z1), conclude:

$$\delta \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \textbf{ and } \delta \models^{\text{dc}} \ell_i \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_i} \not\approx \mathtt{nil}$$

Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude $\delta \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}$. Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}]\!]$.

(W1) Recall from (X0):

$$\left[\begin{array}{l}\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i]\!] \\ \quad \textbf{implies } \delta \in [\![\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}]\!]\end{array}\right] \textbf{ for all } \delta$$

Then, by applying set theory, conclude:

$$\begin{array}{l}[\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i]\!] \\ \subseteq [\![\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}]\!]\end{array}$$

Then, by applying Proposition 3, conclude:

$$\begin{array}{l}\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i \\ \Rightarrow \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\end{array}$$

(W2) Recall from (X8):

$$\begin{array}{l}\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \\ \Rightarrow (\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]\end{array}$$

Then, by applying Axiom–Assignment in Figure 7, conclude:

$$\left[\begin{array}{l}\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \\ \Rightarrow (\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]\end{array}\right]$$

$$\textbf{and} \left[\begin{array}{l}\vdash \{(\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]\} \\ \quad x_i := t_i \\ \quad \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i\}\end{array}\right]$$

Then, by introducing (W1), conclude:

$$\left[\begin{array}{l}\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \\ \Rightarrow (\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]\end{array}\right]$$

$$\textbf{and} \left[\begin{array}{l}\vdash \{(\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i)[\mathtt{d}_{x_i} := t_i]\} \\ \quad x_i := t_i \\ \quad \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i\}\end{array}\right]$$

$$\textbf{and} \left[\begin{array}{l}\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \wedge \mathtt{d}_{x_i} \approx t_i \\ \Rightarrow \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\end{array}\right]$$

Then, by applying Rule–Consequence in Figure 7, conclude:

$$\begin{array}{l}\vdash \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\ \quad x_i := t_i \\ \quad \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\}\end{array}$$

(W3) Recall from (Z2):

$$\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\}$$

Then, by introducing (W2), conclude:

$$\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\}$$

$$\textbf{and} \left[\begin{array}{l}\vdash \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\ \quad x_i := t_i \\ \quad \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\}\end{array}\right]$$

Then, by applying Rule–Composition in Figure 7, conclude:

$$\vdash \{\top\}\ P\ ;\ x_i := t_i\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\}$$

(W4) Recall from (W3):

$$\vdash \{\top\}\ P\ ;\ x_i := t_i\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\}$$

Then, by applying replacement, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\})[P \leftarrow P\ ;\ x_i := t_i]$$

Then, by applying arithmetic, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i+1-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i+1-1}} \not\approx \mathtt{nil}\})[P \leftarrow P\ ;\ x_i := t_i]$$

Then, by applying replacement, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\})[i \leftarrow i+1][P \leftarrow P\ ;\ x_i := t_i]$$

(W5) Suppose:

$$\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))]\!] \textbf{ for some } \delta$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))$. Then, by applying (X4), conclude:

$$\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1}[\mathtt{d}_{x_i} := t_i] \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}}[\mathtt{d}_{x_i} := t_i] \approx \mathcal{I}^{-1}(\delta'(x_{i-1}[\mathtt{d}_{x_i} := t_i]))$$

Then, by applying replacement, conclude:

$$\delta \models^{\mathrm{dc}} (\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)))[\mathtt{d}_{x_i} := t_i] \wedge \cdots \wedge (\mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})))[\mathtt{d}_{x_i} := t_i]$$

Then, by applying replacement, conclude:

$$\delta \models^{\mathrm{dc}} (\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})))[\mathtt{d}_{x_i} := t_i]$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude:

$$\delta \in [\![(\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})))[\mathtt{d}_{x_i} := t_i]]\!]$$

(W6) Recall from (W5):

$$\left[\begin{array}{l} \delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))]\!] \textbf{ implies} \\ \delta \in [\![(\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})))[\mathtt{d}_{x_i} := t_i]]\!] \end{array}\right] \textbf{ for all } \delta$$

Then, by applying set theory, conclude:

$$\begin{array}{l} [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))]\!] \\ \subseteq [\![(\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})))[\mathtt{d}_{x_i} := t_i]]\!] \end{array}$$

Then, by applying Proposition 3, conclude:

$$\begin{array}{l} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \Rightarrow (\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})))[\mathtt{d}_{x_i} := t_i] \end{array}$$

Then, by applying Axiom–Assignment in Figure 7, conclude:

$$\left[\begin{array}{l} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \Rightarrow (\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})))[\mathtt{d}_{x_i} := t_i] \end{array}\right]$$

$$\textbf{and} \quad \left[\begin{array}{l} \vdash_{\mathrm{tot}} \{(\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})))[\mathtt{d}_{x_i} := t_i]\} \\ \quad x_i := t_i \\ \quad \{(\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})))\} \end{array}\right]$$

Then, by applying Rule–Consequence in Figure 7, conclude:

$$\begin{array}{l} \vdash_{\mathrm{tot}} \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\} \\ \quad x_i := t_i \\ \quad \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\} \end{array}$$

(W7) Suppose:

$$\left[\delta' \models^{\mathsf{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}\right] \textbf{ for some } \delta'$$

Then, by applying (Z2), conclude:

$$\vdash_{\mathrm{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\}$$

Then, by introducing (W6), conclude:

$$\vdash_{\mathrm{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\}$$
$$\textbf{and} \quad \left[\begin{array}{l} \vdash_{\mathrm{tot}} \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\} \\ \quad x_i := t_i \\ \quad \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\} \end{array}\right]$$

Then, by applying Rule–Composition in Figure 7, conclude:

$$\vdash_{\mathrm{tot}} \{\top\}\ P\ ;\ x_i := t_i\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\}$$

Then, by introducing (W3), conclude:

$$\vdash_{\mathrm{tot}} \{\top\}\ P\ ;\ x_i := t_i\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\} \textbf{ and}$$
$$\vdash \{\top\}\ P\ ;\ x_i := t_i\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\}$$

Then, by applying Rule–Decomposition in Figure 7, conclude:

$$\vdash_{\mathrm{tot}} \{\top\}\ P\ ;\ x_i := t_i\ \left\{\begin{array}{l} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \quad \wedge \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil} \end{array}\right\}$$

(W8) Suppose:

$$\delta \in \left[\!\!\left[\begin{array}{l} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \quad \wedge \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil} \end{array}\right]\!\!\right] \textbf{ for some } \delta, \delta'$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude:

$$\begin{array}{c} \delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \quad \wedge \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil} \end{array}$$

Then, by applying Definition 8 of $\models^{\mathsf{dc}}$, conclude:

$$\begin{array}{c} \delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathsf{dc}} \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \textbf{and } \delta \models^{\mathsf{dc}} \ell_1 \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathsf{dc}} \ell_i \textbf{ and } \delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil} \end{array}$$

Then, by applying standard inference rules, conclude:

$$\delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathsf{dc}} \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \textbf{ and } \delta \models^{\mathsf{dc}} \ell_i$$

(W9) Suppose:

$$\delta \models^{\mathsf{dc}} \mathtt{d}_{x_j} \approx \mathcal{I}^{-1}(\delta'(x_j)) \textbf{ for some } \delta, \delta', j$$

Then, by applying Definition 8 of $\models^{\mathsf{dc}}$, conclude $\mathsf{eval}_\delta(\mathtt{d}_{x_j}) = \mathsf{eval}_\delta(\mathcal{I}^{-1}(\delta'(x_j))) \neq \mathsf{nil}$. Then, by applying Definition 7 of eval, conclude $\delta(x_j) = \mathcal{I}(\mathcal{I}^{-1}(\delta'(x_j)))$. Then, by applying Definition 6 of $\mathcal{I}$, conclude $\delta(x_j) = \delta'(x_j)$.

(W0) Suppose:

$$\left[\mathtt{d}_y \in \mathsf{Free}(t_i) \textbf{ and } \delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\mathsf{dc}} \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\right]$$
$$\textbf{for some } y, \delta, \delta'$$

Then, by applying (Z4), conclude:

$$\begin{array}{c}\big[\big[\mathtt{d}_y = \mathtt{d}_{x_j} \ \textbf{and} \ 1 \leq j \leq i-1\big] \ \textbf{for some} \ j\big] \ \textbf{and} \\ \delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \ \textbf{and} \ \cdots \ \textbf{and} \ \delta \models^{\mathsf{dc}} \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\end{array}$$

Then, by applying standard inference rules, conclude:

$$\left[\begin{array}{c}\mathtt{d}_y = \mathtt{d}_{x_j} \ \textbf{and} \ 1 \leq j \leq i-1 \ \textbf{and} \\ \delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \ \textbf{and} \ \cdots \ \textbf{and} \ \delta \models^{\mathsf{dc}} \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\end{array}\right] \ \textbf{for some} \ j$$

Then, by applying substitution, conclude $\big[\mathtt{d}_y = \mathtt{d}_{x_j} \ \textbf{and} \ \delta \models^{\mathsf{dc}} \mathtt{d}_{x_j} \approx \mathcal{I}^{-1}(\delta'(x_j))\big]$. Then, by applying (W9), conclude $\big[\mathtt{d}_y = \mathtt{d}_{x_j} \ \textbf{and} \ \delta(x_j) = \delta'(x_j)\big]$. Then, by applying substitution, conclude $\delta(y) = \delta'(y)$.

(V1) Suppose:

$$\big[\delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \ \textbf{and} \ \cdots \ \textbf{and} \ \delta \models^{\mathsf{dc}} \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\big] \ \textbf{for some} \ \delta\,,\ \delta'$$

Then, by applying (W0), conclude $\big[\big[\mathtt{d}_y \in \mathsf{Free}(t_i) \ \textbf{implies} \ \delta(y) = \delta'(y)\big] \ \textbf{for all} \ y\big]$. Then, by applying Lemma 2, conclude $\mathsf{eval}_\delta(t_i) = \mathsf{eval}_{\delta'}(t_i)$.

(V2) Suppose:

$$\delta'' \models^{\mathsf{dc}} \ell_i \ \textbf{for some} \ \delta''$$

Then, by applying (Z1), conclude $\delta'' \models^{\mathsf{dc}} \mathtt{d}_{x_i} \approx t_i$. Then, by applying Definition 8 of $\models^{\mathsf{dc}}$, conclude:

$$\delta''(x_i) = \mathsf{eval}_{\delta''}(t_i) \neq \mathsf{nil}$$

(V3) Suppose:

$$\begin{array}{c}\big[\delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \ \textbf{and} \ \cdots \ \textbf{and} \ \delta \models^{\mathsf{dc}} \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \ \textbf{and} \ \delta \models^{\mathsf{dc}} \ell_i \ \textbf{and} \ \delta' \models^{\mathsf{dc}} \ell_i\big] \\ \textbf{for some} \ \delta\,,\ \delta'\end{array}$$

Then, by applying (V1), conclude $\big[\mathsf{eval}_\delta(t_i) = \mathsf{eval}_{\delta'}(t_i) \ \textbf{and} \ \delta \models^{\mathsf{dc}} \ell_i \ \textbf{and} \ \delta' \models^{\mathsf{dc}} \ell_i\big]$. Then, by applying (V2), conclude $\big[\mathsf{eval}_\delta(t_i) = \mathsf{eval}_{\delta'}(t_i) \ \textbf{and} \ \delta(x_i) = \mathsf{eval}_\delta(t_i) \neq \mathsf{nil} \ \textbf{and} \ \delta'(x_i) = \mathsf{eval}_{\delta'}(t_i) \neq \mathsf{nil}\big]$. Then, by applying substitution, conclude $\delta(x_i) = \delta'(x_i) \neq \mathsf{nil}$. Then, by applying Definition 6 of $\mathcal{I}$, conclude:

$$\delta(x_i) = \mathcal{I}(\mathcal{I}^{-1}(\delta'(x_i))) \neq \mathsf{nil}$$

Then, by applying Definition 7 of eval, conclude $\mathsf{eval}_\delta(\mathtt{d}_{x_i}) = \mathsf{eval}(\mathcal{I}^{-1}(\delta'(x_i))) \neq \mathsf{nil}$. Then, by applying Definition 8 of $\models^{\mathsf{dc}}$, conclude $\delta \models^{\mathsf{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))$.

(V4) Suppose:

$$\left[\delta \in \left[\!\!\left[\begin{array}{c}\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \wedge\, \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil}\end{array}\right]\!\!\right] \ \textbf{and} \ \delta' \models^{\mathsf{dc}} \ell_i\right] \ \textbf{for some} \ \delta\,,\ \delta'$$

Then, by applying (W8), conclude:

$$\delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \ \textbf{and} \ \cdots \ \textbf{and} \ \delta \models^{\mathsf{dc}} \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \ \textbf{and} \ \delta \models^{\mathsf{dc}} \ell_i \ \textbf{and} \ \delta' \models^{\mathsf{dc}} \ell_i$$

Then, by applying (V3), conclude

$$\delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \ \textbf{and} \ \cdots \ \textbf{and} \ \delta \models^{\mathsf{dc}} \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \ \textbf{and} \ \delta \models^{\mathsf{dc}} \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))$$

Then, by applying Definition 8 of $\models^{\mathsf{dc}}$, conclude $\delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))$. Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))]\!]$.

(V5) Suppose:

$$\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ for some } \delta'$$

Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude $\left[\delta' \models^{\text{dc}} \ell_1 \textbf{ and } \cdots \textbf{ and } \delta' \models^{\text{dc}} \ell_{n+m}\right]$. Then, by introducing (Z7), conclude $\left[\delta' \models^{\text{dc}} \ell_1 \textbf{ and } \cdots \textbf{ and } \delta' \models^{\text{dc}} \ell_{n+m} \textbf{ and } 1 \leq i \leq n+m\right]$. Then, by applying substitution, conclude $\delta' \models^{\text{dc}} \ell_i$. Then, by applying (V4), conclude:

$$\left[\begin{array}{l} \delta \in \left[\!\!\left[\begin{array}{c} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \wedge\, \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil} \end{array}\right]\!\!\right] \\ \textbf{implies } \delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))]\!] \end{array}\right] \textbf{ for all } \delta$$

Then, by applying set theory, conclude:

$$\left[\!\!\left[\begin{array}{c} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \wedge\, \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil} \end{array}\right]\!\!\right] \subseteq [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))]\!]$$

Then, by applying Proposition 3, conclude:

$$\left[\begin{array}{c} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \wedge\, \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil} \end{array}\right] \Rightarrow \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))$$

(V6) Suppose:

$$\left[\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}\right] \textbf{ for some } \delta'$$

Then, by applying (W7), conclude:

$$\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and}$$
$$\left[\vdash_{\text{tot}} \{\top\}\ P\ ;\ x_i := t_i\ \left\{\begin{array}{c} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \wedge\, \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil} \end{array}\right\}\right]$$

Then, by applying (V5), conclude:

$$\left[\vdash_{\text{tot}} \{\top\}\ P\ ;\ x_i := t_i\ \left\{\begin{array}{c} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \wedge\, \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil} \end{array}\right\}\right] \textbf{ and}$$
$$\left[\begin{array}{c} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \\ \wedge\, \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_i} \not\approx \mathtt{nil} \end{array}\right] \Rightarrow \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))$$

Then, by applying Rule–Consequence in Figure 7, conclude:

$$\vdash_{\text{tot}} \{\top\}\ P\ ;\ x_i := t_i\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\}$$

Then, by applying replacement, conclude:

$$(\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_i} \approx \mathcal{I}^{-1}(\delta'(x_i))\})[P \leftarrow P\ ;\ x_i := t_i]$$

Then, by applying arithmetic, conclude:

$$(\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i+1-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i+1-1}))\})[P \leftarrow P\ ;\ x_i := t_i]$$

Then, by applying replacement, conclude:

$$(\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\})[i \leftarrow i+1][P \leftarrow P\ ;\ x_i := t_i]$$

(V7) Recall from (V6):

$$\left[\begin{array}{l} \left[\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}\right] \textbf{ implies} \\ (\vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{array}{c} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \end{array}\right\})[i \leftarrow i+1][P \leftarrow P\ ;\ x_i := t_i] \end{array}\right] \textbf{ for all } \delta'$$

Then, by applying replacement, conclude:

$$\left[\left[\begin{array}{l}\left[\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and} \ \mathrm{Dom}(\delta') = \{x_1 \,,\, \ldots \,,\, x_n\}\right] \\ \quad \textbf{implies} \ \vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{array}{c}\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\end{array}\right\} \\ \qquad\qquad \textbf{for all} \ \delta'\end{array}\right]\right][i \leftarrow i+1][P \leftarrow P \,;\, x_i := t_i]$$

(V8) Recall $(i \geq 1)[i \leftarrow i+1][P \leftarrow P \,;\, x_i := t_i]$ from (Z3). Then, by introducing (W4), conclude:

$$\begin{array}{l}(i \geq 1)[i \leftarrow i+1][P \leftarrow P \,;\, x_i := t_i] \\ \textbf{and} \ (\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\})[i \leftarrow i+1][P \leftarrow P \,;\, x_i := t_i]\end{array}$$

Then, by introducing (V7), conclude:

$$\begin{array}{l}(i \geq 1)[i \leftarrow i+1][P \leftarrow P \,;\, x_i := t_i] \\ \textbf{and} \ (\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\})[i \leftarrow i+1][P \leftarrow P \,;\, x_i := t_i] \\ \textbf{and} \ \left[\left[\begin{array}{l}\left[\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and} \ \mathrm{Dom}(\delta') = \{x_1 \,,\, \ldots \,,\, x_n\}\right] \\ \quad \textbf{implies} \ \vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{array}{c}\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\end{array}\right\} \\ \qquad\qquad \textbf{for all} \ \delta'\end{array}\right]\right][i \leftarrow i+1][P \leftarrow P \,;\, x_i := t_i]\end{array}$$

Then, by applying replacement, conclude:

$$\left[\begin{array}{l}i \geq 1 \\ \textbf{and} \ \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\ \textbf{and} \ \left[\left[\begin{array}{l}\left[\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and} \ \mathrm{Dom}(\delta') = \{x_1 \,,\, \ldots \,,\, x_n\}\right] \\ \quad \textbf{implies} \ \vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{array}{c}\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\end{array}\right\} \\ \qquad\qquad \textbf{for all} \ \delta'\end{array}\right]\right]\end{array}\right][i \leftarrow i+1][P \leftarrow P \,;\, x_i := t_i]$$

Then, by applying the definition of Inv_1 in Figure 10, conclude:

$$\mathsf{Inv}_1[i \leftarrow i+1][P \leftarrow P \,;\, x_i := t_i]$$

(V9) Recall $n - i = z$ from (A4). Then, by applying replacement, conclude $n - i = z$. Then, by applying arithmetic, conclude $n - i \leq z$. Then, by applying arithmetic, conclude $n - i - 1 < z$. Then, by applying arithmetic, conclude $n - (i+1) < z$.

Finally, conclude the lemma by the following reduction. Recall Pre from (A1). Then, by introducing (V8), conclude $\left[\mathsf{Pre} \ \textbf{and} \ \mathsf{Inv}_1[i \leftarrow i+1][P \leftarrow P \,;\, x_i := t_i]\right]$. Then, by introducing (V9), conclude:

$$\mathsf{Pre} \ \textbf{and} \ \mathsf{Inv}_1[i \leftarrow i+1][P \leftarrow P \,;\, x_i := t_i] \ \textbf{and} \ n - (i+1) < z$$

(⟪ QED. ⟫)

B.7 Lemma 7

Proof (of Lemma 7). First, assume:

(A1) Pre

(A2) Inv_1

(A3) $i > n$

Next, observe:

(Z1) Recall Inv_1 from (A2). Then, by applying the definition of Inv_1 in Figure 10, conclude:

$$\begin{array}{l} i \geq 1 \\ \textbf{and} \;\; \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\ \textbf{and} \;\; \left[\left[\begin{array}{c}[\delta' \models^{\mathsf{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \;\; \textbf{and} \;\; \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}] \;\; \textbf{implies} \\ \vdash_{\mathrm{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\}\end{array}\right] \;\; \textbf{for all} \;\; \delta'\right] \end{array}$$

(Z2) Recall $i > n$ from (A3). Then, by applying arithmetic, conclude $i \geq 1 + n$.

(Z3) Recall $i > n$ from (A3). Then, by applying arithmetic, conclude $n \leq i - 1$.

(Z4) Suppose:

$$\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}]\!] \;\; \textbf{for some} \;\; \delta$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \models^{\mathsf{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}$. Then, by applying Definition 8 of $\models^{\mathsf{dc}}$, conclude:

$$\delta \models^{\mathsf{dc}} \ell_1 \;\; \textbf{and} \;\; \cdots \;\; \textbf{and} \;\; \delta \models^{\mathsf{dc}} \ell_{i-1} \;\; \textbf{and} \;\; \delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \;\; \textbf{and} \;\; \cdots \;\; \textbf{and} \;\; \delta \models^{\mathsf{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}$$

Then, by introducing (Z3), conclude:

$$\delta \models^{\mathsf{dc}} \ell_1 \;\; \textbf{and} \;\; \cdots \;\; \textbf{and} \;\; \delta \models^{\mathsf{dc}} \ell_{i-1} \;\; \textbf{and} \;\; \delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \;\; \textbf{and} \;\; \cdots \;\; \textbf{and} \;\; \delta \models^{\mathsf{dc}} \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \;\; \textbf{and} \;\; n \leq i - 1$$

Then, by applying substitution, conclude:

$$\delta \models^{\mathsf{dc}} \ell_1 \;\; \textbf{and} \;\; \cdots \;\; \textbf{and} \;\; \delta \models^{\mathsf{dc}} \ell_{i-1} \;\; \textbf{and} \;\; \delta \models^{\mathsf{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \;\; \textbf{and} \;\; \cdots \;\; \textbf{and} \;\; \delta \models^{\mathsf{dc}} \mathtt{d}_{x_n} \not\approx \mathtt{nil}$$

Then, by applying Definition 8 of $\models^{\mathsf{dc}}$, conclude $\delta \models^{\mathsf{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}$. Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}]\!]$.

(Z5) Suppose:

$$\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))]\!] \;\; \textbf{for some} \;\; \delta$$

Then, by a reduction similar to (Z2), conclude $\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))]\!]$.

(Z6) Recall from (Z4):

$$\left[\begin{array}{c}\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}]\!] \\ \textbf{implies} \;\; \delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}]\!]\end{array}\right] \;\; \textbf{for all} \;\; \delta$$

Then, by applying set theory, conclude:

$$[\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}]\!] \subseteq [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}]\!]$$

Then, by applying Proposition 3, conclude:

$$\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \Rightarrow \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}$$

(Z7) By a reduction similar to (Z4), conclude:

$$\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \Rightarrow \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))$$

(Z8) Recall $\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\}$ from (Z1). Then, by introducing (Z6), conclude:

$$\begin{array}{l}\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil}\} \\ \textbf{and}\ \ \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \not\approx \mathtt{nil} \Rightarrow \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\end{array}$$

Then, by applying Rule–Consequence in Figure 7, conclude:

$$\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\}$$

(Z9) Suppose:

$$\left[\delta' \overset{\text{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m}\ \textbf{and}\ \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}\right]\ \textbf{for some}\ \delta'$$

Then, by applying (Z1), conclude $\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\}$. Then, by introducing (Z7), conclude:

$$\begin{array}{l}\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\} \\ \textbf{and}\ \ \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1})) \Rightarrow \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\end{array}$$

Then, by applying Rule–Consequence in Figure 7, conclude:

$$\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}$$

(Z0) Recall $i \geq 1 + n$ from (Z2). Then, by introducing (Z8), conclude:

$$\begin{array}{l} i \geq 1 + n \\ \textbf{and}\ \ \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\}\end{array}$$

Then, by introducing (Z9), conclude:

$$\begin{array}{l} i \geq 1 + n \\ \textbf{and}\ \ \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \\ \left[\left[\begin{array}{l}\left[\delta' \overset{\text{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m}\ \textbf{and}\ \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}\right]\ \textbf{implies} \\ \quad \vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}\end{array}\right]\ \textbf{for all}\ \delta'\right]\end{array}$$

Then, by applying the definition of Inv_2 in Figure 10, conclude Inv_2.

Finally, conclude the lemma by the following reduction. Recall Pre from (A1). Then, by introducing (Z0), conclude $\left[\mathsf{Pre}\ \textbf{and}\ \mathsf{Inv}_2\right]$.

(⫷ QED. ⫸)

B.8 Lemma 8

Proof (of Lemma 8). First, assume:

(A1) Pre

(A2) Inv_2

(A3) $i \leq n + m$

(A4) $n + m - i = z$

Next, observe:

(Z1) Recall Pre from (A1). Then, by applying the definition of Pre in Figure 10, conclude:

$$\begin{array}{l}
\prec_L \text{ is a strict partial order} \\
\textbf{and } \left[\left[\left[\mathtt{d}_x \approx t \in L \textbf{ and } \mathtt{d}_y \in \mathsf{Free}(t)\right] \textbf{ implies } \left[\mathtt{d}_y \approx u \prec_L \mathtt{d}_x \approx t \textbf{ for some } u\right]\right] \textbf{ for all } x, y, t\right] \\
\textbf{and } < \text{ is a linear extension of } \prec_L \\
\textbf{and } L = \{\ell_j \mid 1 \leq j \leq n + m\} \\
\textbf{and } \underbrace{\mathtt{d}_{x_1} \approx t_1}_{\ell_1} < \cdots < \underbrace{\mathtt{d}_{x_n} \approx t_n}_{\ell_n} < \ell_{n+1} < \ldots < \ell_{n+m} \\
\textbf{and } \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} = \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n + m\} \\
\textbf{and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_1) \textbf{ and } \cdots \textbf{ and } \mathtt{nil}\text{-}\mathsf{Free}(\ell_{n+m})
\end{array}$$

(Z2) Recall Inv_2 from (A2). Then, by applying the definition of Inv_2 in Figure 10, conclude:

$$\begin{array}{l}
i \geq 1 + n \\
\textbf{and } \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \\
\textbf{and } \left[\left[\begin{array}{l}[\delta' \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}] \textbf{ implies} \\ \vdash_{\mathrm{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}\end{array}\right] \textbf{ for all } \delta'\right]
\end{array}$$

(Z3) Recall $i \geq 1$ from (Z2). Then, by applying replacement, conclude $(i \geq 1)[P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$. Then, by applying arithmetic, conclude $(i + 1 \geq 1)[P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$. Then, by applying replacement, conclude $(i[i \leftarrow i + 1] \geq 1)[P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$. Then, by applying replacement, conclude $(i \geq 1)[i \leftarrow i + 1][P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$.

(Z4) Suppose:

$$\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil} \wedge \ell_i]\!] \textbf{ for some } \delta$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil} \wedge \ell_i$. Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude:

$$\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \textbf{ and } \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil} \textbf{ and } \delta \models^{\mathrm{dc}} \ell_i$$

Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude $\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}$. Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}]\!]$.

(Z5) Recall from (Z4):

$$\left[\begin{array}{l}\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil} \wedge \ell_i]\!] \\ \textbf{implies } \delta \in [\![\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}]\!]\end{array}\right] \textbf{ for all } \delta$$

Then, by applying set theory, conclude:

$$[\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil} \wedge \ell_i]\!] \subseteq [\![\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}]\!]$$

Then, by applying Proposition 3, conclude:

$$\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil} \wedge \ell_i \Rightarrow \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}$$

Then, by introducing Axiom–Skip in Figure 7, conclude:

$$\begin{bmatrix} \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil} \wedge \ell_i \\ \Rightarrow \ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil} \end{bmatrix}$$

$$\textbf{and} \begin{bmatrix} \vdash \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \\ \mathtt{skip} \\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \end{bmatrix}$$

Then, by applying Rule–Consequence in Figure 7, conclude:

$$\begin{array}{l} \vdash \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil} \wedge \ell_i\} \\ \mathtt{skip} \\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \end{array}$$

Then, by applying Rule–Failure in Figure 7, conclude:

$$\begin{array}{l} \vdash \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \\ \mathtt{if}\ \ell_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi} \\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \end{array}$$

(Z6) Recall from (Z2):

$$\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\}$$

Then, by introducing (Z5), conclude:

$$\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\}$$

$$\textbf{and} \begin{bmatrix} \vdash \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \\ \mathtt{if}\ \ell_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi} \\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \end{bmatrix}$$

Then, by applying Rule–Composition in Figure 7, conclude:

$$\vdash \{\top\}\ P\ \mathtt{;}\ \mathtt{if}\ \ell_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\}$$

Then, by applying replacement, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_i \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\})[P \leftarrow P\ \mathtt{;}\ \mathtt{if}\ \ell_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$$

Then, by applying arithmetic, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i+1-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\})[P \leftarrow P\ \mathtt{;}\ \mathtt{if}\ \ell_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$$

Then, by applying replacement, conclude:

$$(\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\})[i \leftarrow i+1][P \leftarrow P\ \mathtt{;}\ \mathtt{if}\ \ell_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$$

(Z7) Recall $i \leq n$ from (A3). Then, by introducing (Z2), conclude, $1 \leq i \leq n$.

(Z8) Recall $1 \leq i \leq n$ from (Z7). Then, by applying arithmetic, conclude $1 \leq i \leq n+m$.

(Z9) Suppose:

$$\delta' \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{for some}\ \delta'$$

Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude $\left[\delta' \models^{\mathrm{dc}} \ell_1\ \textbf{and}\ \cdots\ \textbf{and}\ \delta' \models^{\mathrm{dc}} \ell_{n+m}\right]$. Then, by introducing (Z8), conclude $\left[\delta' \models^{\mathrm{dc}} \ell_1\ \textbf{and}\ \cdots\ \textbf{and}\ \delta' \models^{\mathrm{dc}} \ell_{n+m}\ \textbf{and}\ 1 \leq i \leq n+m\right]$. Then, by applying substitution, conclude $\delta' \models^{\mathrm{dc}} \ell_i$.

(Z0) Suppose:

$$\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))]\!] \textbf{ for some } \delta\,,\ \delta'$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))$ Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude:

$$\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))$$

(Y1) Suppose:

$$\big[\mathrm{Dom}(\delta') = \{x_1\,,\ \ldots\,,\ x_n\} \textbf{ and } x \in \mathrm{Dom}(\delta')\big] \textbf{ for some } \delta'\,,\ x$$

Then, by applying substitution, conclude $x \in \{x_1\,,\ \ldots\,,\ x_n\}$. Then, by applying set theory, conclude $\big[\big[x = x_j \textbf{ and } 1 \leq j \leq n\big] \textbf{ for some } j\big]$.

(Y2) Suppose:

$$\left[\begin{array}{c}\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \\ \textbf{and } \mathrm{Dom}(\delta') = \{x_1\,,\ \ldots\,,\ x_n\} \textbf{ and } x \in \mathrm{Dom}(\delta')\end{array}\right] \textbf{ for some } \delta\,,\ \delta'\,,\ x$$

Then, by applying (Y1), conclude:

$$\begin{array}{c}\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \\ \textbf{and } \big[\big[x = x_j \textbf{ and } 1 \leq j \leq n\big] \textbf{ for some } j\big]\end{array}$$

Then, by applying standard inference rules, conclude:

$$\left[\begin{array}{c}\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \\ \textbf{and } x = x_j \textbf{ and } 1 \leq j \leq n\end{array}\right] \textbf{ for some } j$$

Then, by applying substitution, conclude $\big[\delta \models^{\text{dc}} \mathtt{d}_{x_j} \approx \mathcal{I}^{-1}(\delta'(x_j)) \textbf{ and } x = x_j\big]$. Then, by applying substitution, conclude $\delta \models^{\text{dc}} \mathtt{d}_x \approx \mathcal{I}^{-1}(\delta'(x))$. Then, by applying Definition 8 of $\models^{\text{dc}}$, conclude:

$$\mathsf{eval}_\delta(\mathtt{d}_x) = \mathsf{eval}_\delta(\mathcal{I}^{-1}(\delta'(x))) \neq \mathsf{nil}$$

Then, by applying Definition 7 of eval, conclude $\delta(x) = \mathcal{I}(\mathcal{I}^{-1}(\delta'(x)))$. Then, by applying Definition 6 of $\mathcal{I}$, conclude $\delta(x) = \delta'(x)$.

(Y3) Suppose:

$$\big[\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))]\!] \textbf{ and } \mathrm{Dom}(\delta') = \{x_1\,,\ \ldots\,,\ x_n\}\big] \textbf{ for some } \delta\,,\ \delta'$$

Then, by applying (Z0), conclude:

$$\delta \models^{\text{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \textbf{ and } \cdots \textbf{ and } \delta \models^{\text{dc}} \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \textbf{ and } \mathrm{Dom}(\delta') = \{x_1\,,\ \ldots\,,\ x_n\}$$

Then, by applying (Y2), conclude $\big[\big[x \in \mathrm{Dom}(\delta') \textbf{ implies } \delta(x) = \delta'(x)\big] \textbf{ for all } x\big]$. Then, by applying set theory, conclude $\delta' \subseteq \delta$.

(Y4) Recall $\{\mathtt{d}_{x_1}\,,\ \ldots\,,\ \mathtt{d}_{x_n}\} = \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\}$ from (Z1). Then, by applying set theory, conclude $\big[\big[X \in \{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\} \textbf{ implies } X \subseteq \{\mathtt{d}_{x_1}\,,\ \ldots\,,\ \mathtt{d}_{x_n}\}\big] \textbf{ for all } X\big]$. Then, by applying set theory, conclude:

$$\big[\big[\big[X = \mathsf{Free}(\ell_j) \textbf{ and } 1 \leq j \leq n+m\big] \textbf{ for some } j\big] \textbf{ implies } X \subseteq \{\mathtt{d}_{x_1}\,,\ \ldots\,,\ \mathtt{d}_{x_n}\}\big] \textbf{ for all } X$$

Then, by applying standard inference rules, conclude:

$$\big[\big[X = \mathsf{Free}(\ell_j) \textbf{ and } 1 \leq j \leq n+m\big] \textbf{ implies } X \subseteq \{\mathtt{d}_{x_1}\,,\ \ldots\,,\ \mathtt{d}_{x_n}\}\big] \textbf{ for all } X\,,\ j$$

Then, by applying substitution, conclude:

$$\big[\big[X = \mathsf{Free}(\ell_j) \ \textbf{and} \ 1 \leq j \leq n+m\big] \ \textbf{implies} \ \mathsf{Free}(\ell_j) \subseteq \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}\big] \ \textbf{for all} \ X, j$$

Then, by applying standard inference rules, conclude:

$$\big[1 \leq j \leq n+m \ \textbf{implies} \ \mathsf{Free}(\ell_j) \subseteq \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}\big] \ \textbf{for all} \ j$$

Then, by introducing (Z8), conclude:

$$\big[\big[1 \leq j \leq n+m \ \textbf{implies} \ \mathsf{Free}(\ell_j) \subseteq \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}\big] \ \textbf{for all} \ j\big] \ \textbf{and} \ 1 \leq i \leq n+m$$

Then, by applying standard inference rules, conclude $\mathsf{Free}(\ell_i) \subseteq \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}$. Then, by applying set theory, conclude $\mathsf{Free}(\ell_i) \subseteq \{\mathtt{d}_x \mid x \in \{x_1, \ldots, x_n\}\}$.

(Y5) Suppose:

$$\mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\} \ \textbf{for some} \ \delta'$$

Then, by applying (Y4), conclude $\big[\mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\} \ \textbf{and} \ \mathsf{Free}(\ell_i) \subseteq \{\mathtt{d}_x \mid x \in \{x_1, \ldots, x_n\}\}\big]$. Then, by applying substitution, conclude $\mathsf{Free}(\ell_i) \subseteq \{\mathtt{d}_x \mid x \in \mathrm{Dom}(\delta')\}$

(Y6) Suppose:

$$\begin{bmatrix}\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))]\!] \\ \textbf{and} \ \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\} \ \textbf{and} \ \delta' \models^{\mathrm{dc}} \ell_i\end{bmatrix} \ \textbf{for some} \ \delta, \delta'$$

Then, by applying (Y3), conclude $\big[\mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\} \ \textbf{and} \ \delta' \models^{\mathrm{dc}} \ell_i \ \textbf{and} \ \delta' \subseteq \delta\big]$. Then, by applying (Y5), conclude $\big[\mathsf{Free}(\ell_i) \subseteq \{\mathtt{d}_x \mid x \in \mathrm{Dom}(\delta')\} \ \textbf{and} \ \delta' \models^{\mathrm{dc}} \ell_i \ \textbf{and} \ \delta' \subseteq \delta\big]$ Then, by applying Proposition 2, conclude $\delta \models^{\mathrm{dc}} \ell_i$. Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \in [\![\ell_i]\!]$.

(Y7) Suppose:

$$\big[\delta' \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and} \ \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}\big] \ \textbf{for some} \ \delta'$$

Then, by applying (Z9), conclude $\big[\mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\} \ \textbf{and} \ \delta' \models^{\mathrm{dc}} \ell_i\big]$. Then, by applying (Y6), conclude $\big[\big[\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))]\!] \ \textbf{implies} \ \delta \in [\![\ell_i]\!]\big] \ \textbf{for all} \ \delta\big]$. Then, by applying set theory, conclude $[\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))]\!] \subseteq [\![\ell_i]\!]$. Then, by applying Proposition 3, conclude $\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \Rightarrow \ell_i$. Then, by introducing Axiom–Skip in Figure 7, conclude:

$$\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \Rightarrow \ell_i$$
$$\textbf{and} \ \begin{bmatrix}\vdash_{\mathrm{tot}} \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\} \\ \quad \mathtt{skip} \\ \quad \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}\end{bmatrix}$$

Then, by applying Rule–Failure II in Figure 7, conclude:

$$\begin{array}{l}\vdash_{\mathrm{tot}} \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\} \\ \quad \mathtt{if} \ \ell_i \ \mathtt{->} \ \mathtt{skip} \ \mathtt{fi} \\ \quad \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}\end{array}$$

(Y8) Suppose:

$$\big[\delta' \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and} \ \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}\big] \ \textbf{for some} \ \delta'$$

Then, by applying (Y7), conclude:

$$\delta' \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and} \ \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}$$
$$\textbf{and} \ \begin{bmatrix}\vdash_{\mathrm{tot}} \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\} \\ \quad \mathtt{if} \ \ell_i \ \mathtt{->} \ \mathtt{skip} \ \mathtt{fi} \\ \quad \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}\end{bmatrix}$$

Then, by applying (Z2), conclude:

$$\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}$$
$$\textbf{and} \begin{bmatrix} \vdash_{\text{tot}} \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\} \\ \texttt{if } \ell_i \texttt{ -> skip fi} \\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\} \end{bmatrix}$$

Then, by applying Rule–Composition in Figure 7, conclude:

$$\vdash_{\text{tot}} \{\top\}\ P\ \texttt{; if } \ell_i \texttt{ -> skip fi}\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}$$

Then, by applying replacement, conclude:

$$(\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\})[P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$$

Then, by applying arithmetic, conclude:

$$(\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\})[P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$$

Then, by applying replacement, conclude:

$$(\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\})[i \leftarrow i+1][P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$$

(Y9) Recall from (Y8):

$$\begin{bmatrix} \left[\delta' \overset{\text{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \text{Dom}(\delta') = \{x_1, \ldots, x_n\}\right] \textbf{ implies} \\ (\vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{matrix} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \end{matrix}\right\})[i \leftarrow i+1][P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}] \end{bmatrix} \textbf{ for all } \delta'$$

Then, by applying replacement, conclude:

$$\left[\left[\begin{array}{c} \left[\delta' \overset{\text{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \text{Dom}(\delta') = \{x_1, \ldots, x_n\}\right] \\ \textbf{implies } \vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{matrix} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \end{matrix}\right\} \\ \textbf{for all } \delta' \end{array}\right]\right][i \leftarrow i+1]$$
$$[P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$$

(Y0) Recall $(i \geq 1)[i \leftarrow i+1][P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$ from (Z3). Then, by introducing (Z6), conclude:

$$(i \geq 1)[i \leftarrow i+1][P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$$
$$\textbf{and } (\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \texttt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \texttt{nil}\})[i \leftarrow i+1]$$
$$[P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$$

Then, by introducing (Y9), conclude:

$$(i \geq 1)[i \leftarrow i+1][P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$$
$$\textbf{and } (\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \texttt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \texttt{nil}\})[i \leftarrow i+1]$$
$$[P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$$
$$\textbf{and } \left[\left[\begin{array}{c} \left[\delta' \overset{\text{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \text{Dom}(\delta') = \{x_1, \ldots, x_n\}\right] \\ \textbf{implies } \vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{matrix} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \end{matrix}\right\} \\ \textbf{for all } \delta' \end{array}\right]\right][i \leftarrow i+1]$$
$$[P \leftarrow P \texttt{ ; if } \ell_i \texttt{ -> skip fi}]$$

Then, by applying replacement, conclude:

$$\left[\begin{array}{l} i \geq 1 \\ \textbf{and} \;\; \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \\ \textbf{and} \;\; \left[\left[\begin{array}{c} \left[\delta' \overset{\text{dc}}{\models} \ell_1 \wedge \cdots \wedge \ell_{n+m} \;\; \textbf{and} \;\; \mathrm{Dom}(\delta') = \{x_1\,,\,\ldots\,,\,x_n\}\right] \\ \textbf{implies} \;\; \vdash_{\text{tot}} \{\top\}\ P\ \left\{\begin{array}{c} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \\ \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \end{array}\right\} \\ \textbf{for all} \;\; \delta' \end{array}\right]\right] \end{array}\right] [i \leftarrow i+1] \; [P \leftarrow P \;\mathtt{;}\; \mathtt{if}\ \ell_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$$

Then, by applying the definition of Inv_2 in Figure 10, conclude $\mathsf{Inv}_2[i \leftarrow i+1][P \leftarrow P \;\mathtt{;}\; \mathtt{if}\ \ell_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]$.

(X1) Recall $n+m-i=z$ from (A4). Then, by applying replacement, conclude $n+m-i=z$. Then, by applying arithmetic, conclude $n+m-i \leq z$. Then, by applying arithmetic, conclude $n+m-i-1 < z$. Then, by applying arithmetic, conclude $n+m-(i+1) < z$.

Finally, conclude the lemma by the following reduction. Recall Pre from (A1). Then, by introducing (Y0), conclude $\left[\mathsf{Pre} \;\textbf{and}\; \mathsf{Inv}_2[i \leftarrow i+1][P \leftarrow P \;\mathtt{;}\; \mathtt{if}\ \ell_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}]\right]$. Then, by introducing (X1), conclude:

$$\mathsf{Pre} \;\textbf{and}\; \mathsf{Inv}_2[i \leftarrow i+1][P \leftarrow P \;\mathtt{;}\; \mathtt{if}\ \ell_i\ \mathtt{->}\ \mathtt{skip}\ \mathtt{fi}] \;\textbf{and}\; n+m-(i+1) < z$$

⟪ QED. ⟫

B.9 Lemma 9

Proof (of Lemma 9). First, assume:

(A1) Pre

(A2) Inv_2

(A3) $i > n + m$

Next, observe:

(Z1) Recall Inv_2 from (A2). Then, by applying the definition of Inv_2 in Figure 10, conclude:

$$\begin{array}{l} i \geq 1 + n \\ \textbf{and} \ \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \\ \textbf{and} \ \left[\left[\begin{array}{c}[\delta' \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \ \textbf{and}\ \mathrm{Dom}(\delta') = \{x_1, \ldots, x_n\}] \ \textbf{implies} \\ \vdash_{\mathrm{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}\end{array}\right] \ \textbf{for all}\ \delta'\right] \end{array}$$

(Z2) Recall $i > n + m$ from (A3). Then, by applying arithmetic, conclude $n + m \leq i - 1$.

(Z3) Suppose:

$$\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}]\!] \ \textbf{for some}\ \delta$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}$. Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude:

$$\delta \models^{\mathrm{dc}} \ell_1 \ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\mathrm{dc}} \ell_{i-1} \ \textbf{and}\ \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\mathrm{dc}} \mathtt{d}_{x_n} \not\approx \mathtt{nil}$$

Then, by applying standard inference rules, conclude $[\delta \models^{\mathrm{dc}} \ell_1 \ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\mathrm{dc}} \ell_{i-1}]$. Then, by introducing (Z2), conclude $[\delta \models^{\mathrm{dc}} \ell_1 \ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\mathrm{dc}} \ell_{i-1} \ \textbf{and}\ n + m \leq i - 1]$. Then, by applying substitution, conclude $[\delta \models^{\mathrm{dc}} \ell_1 \ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\mathrm{dc}} \ell_{n+m}]$. Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude $\delta \models^{\mathrm{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m}$. Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{n+m}]\!]$.

(Z4) Recall from (Z3):

$$\left[\begin{array}{c}\delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}]\!] \\ \textbf{implies}\ \delta \in [\![\ell_1 \wedge \cdots \wedge \ell_{n+m}]\!]\end{array}\right] \ \textbf{for all}\ \delta$$

Then, by applying set theory, conclude $[\![\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}]\!] \subseteq [\![\ell_1 \wedge \cdots \wedge \ell_{n+m}]\!]$. Then, by applying Proposition 3, conclude $\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil} \Rightarrow \ell_1 \wedge \cdots \wedge \ell_{n+m}$.

(Z5) Recall $\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\}$ from (Z1). Then, by introducing (Z4), conclude:

$$\begin{array}{l} \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil}\} \\ \textbf{and}\ \ell_1 \wedge \cdots \wedge \ell_{i-1} \wedge \mathtt{d}_{x_1} \not\approx \mathtt{nil} \wedge \cdots \wedge \mathtt{d}_{x_n} \not\approx \mathtt{nil} \Rightarrow \ell_1 \wedge \cdots \wedge \ell_{n+m} \end{array}$$

Then, by applying Rule–Consequence in Figure 7, conclude $\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{n+m}\}$.

(Z6) Suppose:

$$\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))]\!] \ \textbf{for some}\ \delta$$

Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))$. Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude:

$$\delta \models^{\mathrm{dc}} \ell_1 \ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\mathrm{dc}} \ell_{i-1} \ \textbf{and}\ \delta \models^{\mathrm{dc}} \mathtt{d}_{x_1} \not\approx \mathtt{nil} \ \textbf{and}\ \cdots\ \textbf{and}\ \delta \models^{\mathrm{dc}} \mathtt{d}_{x_n} \not\approx \mathtt{nil}$$

Then, by applying standard inference rules, conclude **true**. Then, by applying Definition 8 of $\models^{\mathrm{dc}}$, conclude $\delta \models^{\mathrm{dc}} \top$. Then, by applying Definition 8 of $[\![\cdot]\!]$, conclude $\delta \in [\![\top]\!]$.

(Z7) Recall from (Z6):

$$[\delta \in [\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))]\!] \ \textbf{implies}\ \delta \in [\![\top]\!]] \ \textbf{for all}\ \delta$$

Then, by applying set theory, conclude $[\![\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))]\!] \subseteq [\![\top]\!]$. Then, by applying Proposition 3, conclude $\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \Rightarrow \top$.

(Z8) Suppose:

$$\left[\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1 , \ldots , x_n\}\right] \textbf{ for some } \delta'$$

Then, by applying (Z1), conclude $\vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n))\}$. Then, by introducing (Z7), conclude:

$$\begin{array}{l} \vdash_{\text{tot}} \{\top\}\ P\ \{\mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_{i-1}} \approx \mathcal{I}^{-1}(\delta'(x_{i-1}))\} \\ \textbf{and } \mathtt{d}_{x_1} \approx \mathcal{I}^{-1}(\delta'(x_1)) \wedge \cdots \wedge \mathtt{d}_{x_n} \approx \mathcal{I}^{-1}(\delta'(x_n)) \Rightarrow \top \end{array}$$

Then, by applying Rule–Consequence in Figure 7, conclude $\vdash_{\text{tot}} \{\top\}\ P\ \{\top\}$.

Finally, conclude the lemma by the following reduction. Recall $\vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{n+m}\}$ from (Z5). Then, by introducing (Z8), conclude:

$$\begin{array}{l} \vdash \{\top\}\ P\ \{\ell_1 \wedge \cdots \wedge \ell_{n+m}\} \\ \textbf{and } \left[\left[\begin{array}{c}\left[\delta' \models^{\text{dc}} \ell_1 \wedge \cdots \wedge \ell_{n+m} \textbf{ and } \mathrm{Dom}(\delta') = \{x_1 , \ldots , x_n\}\right] \textbf{ implies} \\ \vdash_{\text{tot}} \{\top\}\ P\ \{\top\}\end{array}\right] \textbf{ for all } \delta'\right] \end{array}$$

Then, by applying the definition of Post in Figure 10, conclude Post.

(⟪ QED. ⟫)

B.10 Lemma 10

Proof (of Lemma 10).

1. First, assume:

   (A1) $\ell \prec_L^{\text{arb}} \ell''$

   Now, prove the lemma by induction on $|\prec_L|$.

   – **Base:** $|\prec_L^{\text{arb}}| = 0$
   First, observe:

   (Z1) Recall $|\prec_L^{\text{arb}}| = 0$ from **Base**. Then, by applying set theory, conclude $\prec_L^{\text{arb}} = \emptyset$.

   Now, prove the base case by the following reduction. Recall $\ell \prec_L^{\text{arb}} \ell''$ from (A1). Then, by applying set theory, conclude $(\ell\,,\,\ell'') \in \prec_L^{\text{arb}}$. Then, by applying (Z1), conclude $(\ell\,,\,\ell'') \in \emptyset$. Then, by applying set theory, conclude **false**.

   – **IH:**

   $$\left[\left[|\hat{\prec}_L^{\text{arb}}| < |\prec_L^{\text{arb}}| \text{ and } \hat{\ell}\hat{\prec}_L^{\text{arb}}\hat{\ell}''\right] \text{ implies } \left[\begin{array}{l}\left[\begin{array}{l}L^1 \blacktriangleleft_L^{\text{arb}} \ell^2 \text{ and } \cdots \text{ and } L^{k-1} \blacktriangleleft_L^{\text{arb}} \ell^k \\ \text{and } \ell^1 \in L^1 \text{ and } \cdots \text{ and } \ell^{k-1} \in L^{k-1} \\ \text{and } \ell^1 = \ell_1 \text{ and } \ell^k = \ell_3\end{array}\right] \\ \quad \text{for some } L^1\,,\,\ldots\,,\,L^k\,,\,\ell^1\,,\,\ldots\ell^k\,,\,k\end{array}\right]\right]$$
   $$\text{for all } \hat{\prec}_L^{\text{arb}}\,,\,\hat{\ell}\,,\,\hat{\ell}''$$

   – **Step:** $|\prec_L^{\text{arb}}| > 0$
   First, observe:

   (Y1) Suppose:

   $$\left[\{\ell_1\,,\,\ldots\,,\,\ell_k\} \blacktriangleleft_L^{\text{arb}} \ell'' \text{ and } \ell_i \in \{\ell_1\,,\,\ldots\,,\,\ell_k\}\right] \text{ for some } \ell_1\,,\,\ldots\,,\,\ell_k\,,\,i\,,\,k$$

   Then, by applying standard inference rules, conclude:

   $$\{\ell_1\,,\,\ldots\,,\,\ell_k\} \blacktriangleleft_L^{\text{arb}} \ell'' \text{ and } \ell_i \in \{\ell_1\,,\,\ldots\,,\,\ell_k\} \text{ and } \left[L' = \{\ell_1\,,\,\ldots\,,\,\ell_k\} \text{ for some } L'\right]$$

   Then, by applying standard inference rules, conclude:

   $$\left[\{\ell_1\,,\,\ldots\,,\,\ell_k\} \blacktriangleleft_L^{\text{arb}} \ell'' \text{ and } \ell_i \in \{\ell_1\,,\,\ldots\,,\,\ell_k\} \text{ and } L' = \{\ell_1\,,\,\ldots\,,\,\ell_k\}\right] \text{ for some } L'$$

   Then, by applying substitution, conclude $\left[L' \blacktriangleleft_L^{\text{arb}} \ell'' \text{ and } \ell_i \in L'\right]$.

   (Y2) Suppose:

   $$\left[\{\ell_1\,,\,\ldots\,,\,\ell_k\} \blacktriangleleft_L^{\text{arb}} \ell'' \text{ and } 1 \leq i \leq k \text{ and } \ell_i = \ell\right] \text{ for some } \ell_1\,,\,\ldots\,,\,\ell_k\,,\,i\,,\,k$$

   Then, by applying set theory, conclude:

   $$\{\ell_1\,,\,\ldots\,,\,\ell_k\} \blacktriangleleft_L^{\text{arb}} \ell'' \text{ and } \ell_i \in \{\ell_1\,,\,\ldots\,,\,\ell_k\} \text{ and } \ell_i = \ell$$

   Then, by applying (Y1), conclude $\left[\left[\left[L' \blacktriangleleft_L^{\text{arb}} \ell'' \text{ and } \ell_i \in L'\right] \text{ for some } L'\right] \text{ and } \ell_i = \ell\right]$. Then, by applying standard inference rules, conclude:

   $$\left[L^1 \blacktriangleleft_L^{\text{arb}} \ell'' \text{ and } \ell_i \in L^1 \text{ and } \ell_i = \ell\right] \text{ for some } L^1$$

   Then, by applying substitution, conclude $\left[L^1 \blacktriangleleft_L^{\text{arb}} \ell'' \text{ and } \ell \in L^1\right]$. Then, by applying standard inference rules, conclude:

   $$L^1 \blacktriangleleft_L^{\text{arb}} \ell'' \text{ and } \ell \in L^1 \text{ and } \left[\ell^1 = \ell \text{ for some } \ell^1\right] \text{ and } \left[\ell^2 = \ell'' \text{ for some } \ell^2\right]$$

   Then, by applying standard inference rules, conclude:

   $$\left[L^1 \blacktriangleleft_L^{\text{arb}} \ell'' \text{ and } \ell \in L^1 \text{ and } \ell^1 = \ell \text{ and } \ell^2 = \ell''\right] \text{ for some } \ell^1\,,\,\ell^2$$

   Then, by applying substitution, conclude $\left[L^1 \blacktriangleleft_L^{\text{arb}} \ell^2 \text{ and } \ell^1 \in L^1 \text{ and } \ell^1 = \ell \text{ and } \ell^2 = \ell''\right]$.

(Y3) Suppose:

$$\left[(\ell_1\,,\,\ell_2) \in \prec_L^{\text{arb}} \text{ and } \ell_2 \neq \ell_3\right] \text{ for some } \ell_1\,,\,\ell_2\,,\,\ell_3$$

Then, by applying standard inference rules, conclude $\left[(\ell_1\,,\,\ell_2) \in \prec_L^{\text{arb}} \text{ and } (\ell_1\,,\,\ell_2) \neq (\ell_2\,,\,\ell_3)\right]$. Then, by applying set theory, conclude $(\ell_1\,,\,\ell_2) \in \prec_L^{\text{arb}} \setminus \{(\ell_2\,,\,\ell_3)\}$.

(Y4) Suppose:

$$\left[(\ell_2\,,\,\ell_3) \in \prec_L^{\text{arb}} \text{ and } \ell_2 \neq \ell_1\right] \text{ for some } \ell_1\,,\,\ell_2\,,\,\ell_3$$

Then, by a reduction similar to (Y3), conclude $(\ell_2\,,\,\ell_3) \in \prec_L^{\text{arb}} \setminus \{(\ell_1\,,\,\ell_2)\}$.

(Y5) Suppose:

$$(\ell_\dagger\,,\,\ell_\ddagger) \in \prec_L^{\text{arb}} \text{ for some } \ell_\dagger\,,\,\ell_\ddagger$$

Then, by applying set theory, conclude $|\prec_L^{\text{arb}} \setminus \{(\ell_\dagger\,,\,\ell_\ddagger)\}| = |\prec_L^{\text{arb}}| - 1$. Then, by applying arithmetic, conclude $|\prec_L^{\text{arb}} \setminus \{(\ell_\dagger\,,\,\ell_\ddagger)\}| < |\prec_L^{\text{arb}}|$.

(Y6) Suppose:

$$\left[\ell_1 \prec_L^{\text{arb}} \ell_2 \prec_L^{\text{arb}} \ell_3 \text{ and } \ell_2 \neq \ell_3\right] \text{ for some } \ell_1\,,\,\ell_2\,,\,\ell_3$$

Then, by applying (Y3), conclude $\left[\ell_2 \prec_L^{\text{arb}} \ell_3 \text{ and } (\ell_1\,,\,\ell_2) \in \prec_L^{\text{arb}} \setminus \{(\ell_2\,,\,\ell_3)\}\right]$. Then, by applying (Y5), conclude $\left[|\prec_L^{\text{arb}} \setminus \{(\ell_2\,,\,\ell_3)\}| < |\prec_L^{\text{arb}}| \text{ and } (\ell_1\,,\,\ell_2) \in \prec_L^{\text{arb}} \setminus \{(\ell_2\,,\,\ell_3)\}\right]$. Then, by applying **IH**, conclude:

$$\left[\begin{array}{l} L^1 \blacktriangleleft_L^{\text{arb}} \ell^2 \text{ and } \cdots \text{ and } L^{k-1} \blacktriangleleft_L^{\text{arb}} \ell^k \\ \text{and } \ell^1 \in L^1 \text{ and } \cdots \text{ and } \ell^{k-1} \in L^{k-1} \\ \text{and } \ell^1 = \ell_1 \text{ and } \ell^k = \ell_2 \end{array}\right] \text{ for some } L^1\,,\,\ldots\,,\,L^k\,,\,\ell^1\,,\,\ldots\,\ell^k\,,\,k$$

(Y7) Suppose:

$$\left[\ell_1 \prec_L^{\text{arb}} \ell_2 \prec_L^{\text{arb}} \ell_3 \text{ and } \ell_2 \neq \ell_1\right] \text{ for some } \ell_1\,,\,\ell_2\,,\,\ell_3$$

Then, by a reduction similar to (Y6), conclude:

$$\left[\begin{array}{l} L^1 \blacktriangleleft_L^{\text{arb}} \ell^2 \text{ and } \cdots \text{ and } L^{k-1} \blacktriangleleft_L^{\text{arb}} \ell^k \\ \text{and } \ell^1 \in L^1 \text{ and } \cdots \text{ and } \ell^{k-1} \in L^{k-1} \\ \text{and } \ell^1 = \ell_2 \text{ and } \ell^k = \ell_3 \end{array}\right] \text{ for some } L^1\,,\,\ldots\,,\,L^k\,,\,\ell^1\,,\,\ldots\,\ell^k\,,\,k$$

(Y8) Suppose:

$$\left[\ell_1 \prec_L^{\text{arb}} \ell_2 \prec_L^{\text{arb}} \ell_3 \text{ and } \ell_2 \neq \ell_1 \text{ and } \ell_2 \neq \ell_3\right] \text{ for some } \ell_1\,,\,\ell_2\,,\,\ell_3$$

Then, by applying (Y6), conclude:

$$\ell_1 \prec_L^{\text{arb}} \ell_2 \prec_L^{\text{arb}} \ell_3 \text{ and } \ell_2 \neq \ell_1 \text{ and}$$
$$\left[\begin{array}{l} L^1 \blacktriangleleft_L^{\text{arb}} \ell^2 \text{ and } \cdots \text{ and } L^{k-1} \blacktriangleleft_L^{\text{arb}} \ell^k \\ \text{and } \ell^1 \in L^1 \text{ and } \cdots \text{ and } \ell^{k-1} \in L^{k-1} \\ \text{and } \ell^1 = \ell_1 \text{ and } \ell^k = \ell_2 \end{array}\right] \text{ for some } L^1\,,\,\ldots\,,\,L^k\,,\,\ell^1\,,\,\ldots\,\ell^k\,,\,k$$

Then, by applying (Y7), conclude:

$$\left[\begin{array}{l} L^1 \blacktriangleleft_L^{\text{arb}} \ell^2 \text{ and } \cdots \text{ and } L^{k-1} \blacktriangleleft_L^{\text{arb}} \ell^k \\ \text{and } \ell^1 \in L^1 \text{ and } \cdots \text{ and } \ell^{k-1} \in L^{k-1} \\ \text{and } \ell^1 = \ell_1 \text{ and } \ell^k = \ell_2 \end{array}\right] \text{ and}$$
$$\left[\begin{array}{l} \left[\begin{array}{l} L^{k+1} \blacktriangleleft_L^{\text{arb}} \ell^{k+2} \text{ and } \cdots \text{ and } L^{k+k'-1} \blacktriangleleft_L^{\text{arb}} \ell^{k+k'} \\ \text{and } \ell^{k+1} \in L^{k+1} \text{ and } \cdots \text{ and } \ell^{k+k'-1} \in L^{k+k'-1} \\ \text{and } \ell^{k+1} = \ell_2 \text{ and } \ell^{k+k'} = \ell_3 \end{array}\right] \\ \quad \text{for some } L^{k+1}\,,\,\ldots\,,\,L^{k+k'}\,,\,\ell^{k+1}\,,\,\ldots\,\ell^{k+k'}\,,\,k' \end{array}\right]$$

Then, by applying standard inference rules, conclude:

$$\left[\begin{array}{l}\left[\begin{array}{l}L^1 \blacktriangleleft_L^{\text{arb}} \ell^2 \text{ and } \cdots \text{ and } L^{k-1} \blacktriangleleft_L^{\text{arb}} \ell^k \\ \text{and } \ell^1 \in L^1 \text{ and } \cdots \text{ and } \ell^{k-1} \in L^{k-1} \\ \text{and } \ell^1 = \ell_1 \text{ and } \ell^k = \ell_2\end{array}\right] \\ \text{and } \left[\begin{array}{l}L^{k+1} \blacktriangleleft_L^{\text{arb}} \ell^{k+2} \text{ and } \cdots \text{ and } L^{k+k'-1} \blacktriangleleft_L^{\text{arb}} \ell^{k+k'} \\ \text{and } \ell^{k+1} \in L^{k+1} \text{ and } \cdots \text{ and } \ell^{k+k'-1} \in L^{k+k'-1} \\ \text{and } \ell^{k+1} = \ell_2 \text{ and } \ell^{k+k'} = \ell_3\end{array}\right]\end{array}\right] \text{ for some } L^{k+1}, \ldots, L^{k+k'}, \ell^{k+1}, \ldots \ell^{k+k'}, k'$$

Then, by applying substitution, conclude:

$$\begin{array}{l}L^1 \blacktriangleleft_L^{\text{arb}} \ell^2 \text{ and } \cdots \text{ and } L^{k-1} \blacktriangleleft_L^{\text{arb}} \ell^k \\ \text{and } L^{k+1} \blacktriangleleft_L^{\text{arb}} \ell^{k+2} \text{ and } \cdots \text{ and } L^{k+k'-1} \blacktriangleleft_L^{\text{arb}} \ell^{k+k'} \\ \text{and } \ell^1 \in L^1 \text{ and } \cdots \text{ and } \ell^{k-1} \in L^{k-1} \\ \text{and } \ell^{k+1} \in L^{k+1} \text{ and } \cdots \text{ and } \ell^{k+k'-1} \in L^{k+k'-1} \\ \text{and } \ell^1 = \ell_1 \text{ and } \ell^{k+k'} = \ell_3 \text{ and } \ell^k = \ell^{k+1}\end{array}$$

Then, by applying substitution, conclude:

$$\begin{array}{l}L^1 \blacktriangleleft_L^{\text{arb}} \ell^2 \text{ and } \cdots \text{ and } L^{k-1} \blacktriangleleft_L^{\text{arb}} \ell^k \\ \text{and } L^{k+1} \blacktriangleleft_L^{\text{arb}} \ell^{k+2} \text{ and } \cdots \text{ and } L^{k+k'-1} \blacktriangleleft_L^{\text{arb}} \ell^{k+k'} \\ \text{and } \ell^1 \in L^1 \text{ and } \cdots \text{ and } \ell^{k-1} \in L^{k-1} \\ \text{and } \ell^k \in L^{k+1} \text{ and } \ell^{k+2} \in L^{k+2} \text{ and } \cdots \text{ and } \ell^{k+k'-1} \in L^{k+k'-1} \\ \text{and } \ell^1 = \ell_1 \text{ and } \ell^{k+k'} = \ell_3\end{array}$$

(Y9) Suppose:

$$\left[\ell_1 \prec_L^{\text{arb}} \ell_2 \prec_L^{\text{arb}} \ell_3 \text{ and } \ell_2 \notin \{\ell_1, \ell_3\} \text{ and } \ell_1 = \ell \text{ and } \ell_3 = \ell''\right] \text{ for some } \ell_1, \ell_2, \ell_3$$

Then, by applying set theory, conclude:

$$\ell_1 \prec_L^{\text{arb}} \ell_2 \prec_L^{\text{arb}} \ell_3 \text{ and } \ell_2 \neq \ell_1 \text{ and } \ell_2 \neq \ell_3 \text{ and } \ell_1 = \ell \text{ and } \ell_3 = \ell''$$

Then, by applying (Y8), conclude:

$$\left[\left[\begin{array}{l}\hat{L}^1 \blacktriangleleft_L^{\text{arb}} \hat{\ell}^2 \text{ and } \cdots \text{ and } \hat{L}^{\hat{k}-1} \blacktriangleleft_L^{\text{arb}} \hat{\ell}^{\hat{k}} \\ \text{and } \hat{\ell}^1 \in \hat{L}^1 \text{ and } \cdots \text{ and } \hat{\ell}^{\hat{k}-1} \in \hat{L}^{\hat{k}-1} \\ \text{and } \hat{\ell}^1 = \ell_1 \text{ and } \hat{\ell}^{\hat{k}} = \ell_3\end{array}\right] \text{ for some } \hat{L}^1, \ldots, \hat{L}^{\hat{k}}, \hat{\ell}^1, \ldots \hat{\ell}^{\hat{k}}, \hat{k}\right]$$
$$\text{and } \ell_1 = \ell \text{ and } \ell_3 = \ell''$$

Then, by applying standard inference rules, conclude:

$$\left[\begin{array}{l}\hat{L}^1 \blacktriangleleft_L^{\text{arb}} \hat{\ell}^2 \text{ and } \cdots \text{ and } \hat{L}^{\hat{k}-1} \blacktriangleleft_L^{\text{arb}} \hat{\ell}^{\hat{k}} \\ \text{and } \hat{\ell}^1 \in \hat{L}^1 \text{ and } \cdots \text{ and } \hat{\ell}^{\hat{k}-1} \in \hat{L}^{\hat{k}-1} \\ \text{and } \hat{\ell}^1 = \ell_1 \text{ and } \hat{\ell}^{\hat{k}} = \ell_3 \\ \text{and } \ell_1 = \ell \text{ and } \ell_3 = \ell''\end{array}\right] \text{ for some } \hat{L}^1, \ldots, \hat{L}^{\hat{k}}, \hat{\ell}^1, \ldots \hat{\ell}^{\hat{k}}, \hat{k}$$

Then, by applying substitution, conclude:

$$\hat{L}^1 \blacktriangleleft_L^{\text{arb}} \hat{\ell}^2 \text{ and } \cdots \text{ and } \hat{L}^{\hat{k}-1} \blacktriangleleft_L^{\text{arb}} \hat{\ell}^{\hat{k}} \text{ and } \hat{\ell}^1 \in \hat{L}^1 \text{ and } \cdots \text{ and } \hat{\ell}^{\hat{k}-1} \in \hat{L}^{\hat{k}-1}$$
$$\text{and } \hat{\ell}^1 = \ell \text{ and } \hat{\ell}^{\hat{k}} = \ell''$$

Now, prove the inductive step by the following reduction. Recall $\ell \prec_L^{\mathrm{arb}} \ell''$ from (A1). Then, by Definition 15 of $\prec_L^{\mathrm{arb}}$, conclude:

$$\begin{array}{l}\big[\big[\{\ell_1\,,\,\dots\,,\,\ell_k\} \blacktriangleleft_L^{\mathrm{arb}} \ell'' \ \textbf{and}\ 1 \leq i \leq k \ \textbf{and}\ \ell_i = \ell\big] \ \textbf{for some}\ \ell_1\,,\,\dots\,,\,\ell_k\,,\,k\big] \\ \textbf{or}\ \big[\big[\ell_1 \prec_L^{\mathrm{arb}} \ell_2 \prec_L^{\mathrm{arb}} \ell_3 \ \textbf{and}\ \ell_2 \notin \{\ell_1\,,\,\ell_3\} \ \textbf{and}\ \ell_1 = \ell \ \textbf{and}\ \ell_3 = \ell''\big] \ \textbf{for some}\ \ell_1\,,\,\ell_2\,,\,\ell_3\big]\end{array}$$

Then, by applying (Y2), conclude:

$$\begin{array}{l}\left[\begin{bmatrix}\hat{L}^1 \blacktriangleleft_L^{\mathrm{arb}} \hat{\ell}^2 \ \textbf{and}\ \cdots \ \textbf{and}\ \hat{L}^{\hat{k}-1} \blacktriangleleft_L^{\mathrm{arb}} \hat{\ell}^{\hat{k}} \\ \textbf{and}\ \hat{\ell}^1 \in \hat{L}^1 \ \textbf{and}\ \cdots \ \textbf{and}\ \hat{\ell}^{\hat{k}-1} \in \hat{L}^{\hat{k}-1} \\ \textbf{and}\ \hat{\ell}^1 = \ell \ \textbf{and}\ \hat{\ell}^{\hat{k}} = \ell''\end{bmatrix} \textbf{for some}\ \hat{L}^1\,,\,\dots\,,\,\hat{L}^{\hat{k}}\,,\,\hat{\ell}^1\,,\,\dots\hat{\ell}^{\hat{k}}\,,\,\hat{k}\right] \\ \textbf{or}\ \big[\big[\ell_1 \prec_L^{\mathrm{arb}} \ell_2 \prec_L^{\mathrm{arb}} \ell_3 \ \textbf{and}\ \ell_2 \notin \{\ell_1\,,\,\ell_3\} \ \textbf{and}\ \ell_1 = \ell \ \textbf{and}\ \ell_3 = \ell''\big] \ \textbf{for some}\ \ell_1\,,\,\ell_2\,,\,\ell_3\big]\end{array}$$

Then, by applying (Y9), conclude:

$$\begin{array}{l}\left[\begin{bmatrix}\hat{L}^1 \blacktriangleleft_L^{\mathrm{arb}} \hat{\ell}^2 \ \textbf{and}\ \cdots \ \textbf{and}\ \hat{L}^{\hat{k}-1} \blacktriangleleft_L^{\mathrm{arb}} \hat{\ell}^{\hat{k}} \\ \textbf{and}\ \hat{\ell}^1 \in \hat{L}^1 \ \textbf{and}\ \cdots \ \textbf{and}\ \hat{\ell}^{\hat{k}-1} \in \hat{L}^{\hat{k}-1} \\ \textbf{and}\ \hat{\ell}^1 = \ell \ \textbf{and}\ \hat{\ell}^{\hat{k}} = \ell''\end{bmatrix} \textbf{for some}\ \hat{L}^1\,,\,\dots\,,\,\hat{L}^{\hat{k}}\,,\,\hat{\ell}^1\,,\,\dots\hat{\ell}^{\hat{k}}\,,\,\hat{k}\right] \\ \textbf{or}\ \left[\begin{bmatrix}\hat{L}^1 \blacktriangleleft_L^{\mathrm{arb}} \hat{\ell}^2 \ \textbf{and}\ \cdots \ \textbf{and}\ \hat{L}^{\hat{k}-1} \blacktriangleleft_L^{\mathrm{arb}} \hat{\ell}^{\hat{k}} \\ \textbf{and}\ \hat{\ell}^1 \in \hat{L}^1 \ \textbf{and}\ \cdots \ \textbf{and}\ \hat{\ell}^{\hat{k}-1} \in \hat{L}^{\hat{k}-1} \\ \textbf{and}\ \hat{\ell}^1 = \ell \ \textbf{and}\ \hat{\ell}^{\hat{k}} = \ell''\end{bmatrix} \textbf{for some}\ \hat{L}^1\,,\,\dots\,,\,\hat{L}^{\hat{k}}\,,\,\hat{\ell}^1\,,\,\dots\hat{\ell}^{\hat{k}}\,,\,\hat{k}\right]\end{array}$$

Then, by applying standard inference rules, conclude:

$$\begin{bmatrix}\hat{L}^1 \blacktriangleleft_L^{\mathrm{arb}} \hat{\ell}^2 \ \textbf{and}\ \cdots \ \textbf{and}\ \hat{L}^{\hat{k}-1} \blacktriangleleft_L^{\mathrm{arb}} \hat{\ell}^{\hat{k}} \\ \textbf{and}\ \hat{\ell}^1 \in \hat{L}^1 \ \textbf{and}\ \cdots \ \textbf{and}\ \hat{\ell}^{\hat{k}-1} \in \hat{L}^{\hat{k}-1} \\ \textbf{and}\ \hat{\ell}^1 = \ell \ \textbf{and}\ \hat{\ell}^{\hat{k}} = \ell''\end{bmatrix} \textbf{for some}\ \hat{L}^1\,,\,\dots\,,\,\hat{L}^{\hat{k}}\,,\,\hat{\ell}^1\,,\,\dots\hat{\ell}^{\hat{k}}\,,\,\hat{k}$$

(| QED. |)

2. First, observe:

(X1) Suppose:

$$\ell \prec_L^{\mathrm{arb}} \ell \ \textbf{for some}\ \ell$$

Then, by applying Lemma 10:1, conclude:

$$\begin{bmatrix}L^1 \blacktriangleleft_L^{\mathrm{arb}} \ell^2 \ \textbf{and}\ \cdots \ \textbf{and}\ L^{k-1} \blacktriangleleft_L^{\mathrm{arb}} \ell^k \\ \textbf{and}\ \ell^1 \in L^1 \ \textbf{and}\ \cdots \ \textbf{and}\ \ell^{k-1} \in L^{k-1} \\ \textbf{and}\ \ell^1 = \ell \ \textbf{and}\ \ell^k = \ell\end{bmatrix} \textbf{for some}\ L^1\,,\,\dots\,,\,L^k\,,\,\ell^1\,,\,\dots\ell^k\,,\,k$$

Then, by applying substitution, conclude:

$$\begin{bmatrix}L^1 \blacktriangleleft_L^{\mathrm{arb}} \ell^2 \ \textbf{and}\ \cdots \ \textbf{and}\ L^{k-1} \blacktriangleleft_L^{\mathrm{arb}} \ell^k \ \textbf{and} \\ \ell^1 \in L^1 \ \textbf{and}\ \cdots \ \textbf{and}\ \ell^{k-1} \in L^{k-1} \ \textbf{and}\ \ell^1 = \ell^k\end{bmatrix} \textbf{for some}\ L^1\,,\,\dots\,,\,L^k\,,\,\ell^1\,,\,\dots\ell^k\,,\,k$$

Then, by introducing Proposition 5:2, conclude:

$$\begin{array}{l}\left[\begin{array}{c}\begin{bmatrix}L^1 \blacktriangleleft_L^{\mathrm{arb}} \ell^2 \ \textbf{and}\ \cdots \ \textbf{and}\ L^{k-1} \blacktriangleleft_L^{\mathrm{arb}} \ell^k \ \textbf{and} \\ \ell^1 \in L^1 \ \textbf{and}\ \cdots \ \textbf{and}\ \ell^{k-1} \in L^{k-1} \ \textbf{and}\ \ell^1 = \ell^k\end{bmatrix} \\ \textbf{for some}\ L^1\,,\,\dots\,,\,L^k\,,\,\ell^1\,,\,\dots\ell^k\,,\,k\end{array}\right] \textbf{and} \\ \left[\textbf{not}\ \left[\begin{array}{c}\begin{bmatrix}L^1 \blacktriangleleft_L^{\mathrm{arb}} \ell^2 \ \textbf{and}\ \cdots \ \textbf{and}\ L^{k-1} \blacktriangleleft_L^{\mathrm{arb}} \ell^k \ \textbf{and} \\ \ell^1 \in L^1 \ \textbf{and}\ \cdots \ \textbf{and}\ \ell^{k-1} \in L^{k-1} \ \textbf{and}\ \ell^1 = \ell^k\end{bmatrix} \\ \textbf{for some}\ L^1\,,\,\dots\,,\,L^k\,,\,\ell^1\,,\,\dots\ell^k\,,\,k\end{array}\right]\right]\end{array}$$

Then, by applying standard inference rules, conclude **false**.

Now, prove the lemma by the following reduction. Recall $\left[\ell \not\prec_L^{\mathrm{arb}} \ell \ \textbf{for all} \ \ell\right]$ from (X1). Then, by applying order theory, conclude $\left[\prec_L^{\mathrm{arb}} \text{ is irreflexive}\right]$.

⟪ QED. ⟫

B.11 Lemma 11

Proof (of Lemma 11).

1. First, assume:

   (A1) $\ell \prec_L^{\text{arb}} \ell''$

   Now, prove the lemma by induction on $|\prec_L|$.

   – **Base:** $|\prec_L^{\text{arb}}| = 0$
   First, observe:

   (Z1) Recall $|\prec_L^{\text{arb}}| = 0$ from **Base**. Then, by applying set theory, conclude $\prec_L^{\text{arb}} = \emptyset$.

   Now, prove the base case by the following reduction. Recall $\ell \prec_L^{\text{arb}} \ell''$ from (A1). Then, by applying set theory, conclude $(\ell\,,\,\ell'') \in \prec_L^{\text{arb}}$. Then, by applying (Z1), conclude $(\ell\,,\,\ell'') \in \emptyset$. Then, by applying set theory, conclude **false**.

   – **IH:** $\big[\big[|\hat{\prec}_L^{\text{arb}}| < |\prec_L^{\text{arb}}|$ **and** $\hat{\ell} \hat{\prec}_L^{\text{arb}} \hat{\ell}''\big]$ **implies** $\big[\hat{\ell} = \mathtt{d}_x \approx t$ **for some** $x\,,\,t\big]\big]$ **for all** $\hat{\prec}_L^{\text{arb}}\,,\,\hat{\ell}\,,\,\hat{\ell}''$

   – **Step:** $|\prec_L^{\text{arb}}| > 0$
   First, observe:

   (Y1) Suppose:

   $$\big[\{\ell_1\,,\,\ldots\,,\,\ell_k\} \blacktriangleleft_L^{\text{arb}} \ell''$$ **and** $$1 \leq i \leq k$$ **and** $$\ell_i = \ell\big]$$ **for some** $\ell_1\,,\,\ldots\,,\,\ell_k\,,\,i\,,\,k$

   Then, by applying Definition 13 of $\blacktriangleleft_L$, conclude:

   $$1 \leq i \leq k \text{ and } \ell_i = \ell \text{ and}$$
   $$\big[\big[\ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_k = \mathtt{d}_{x_k} \approx t_k\big] \text{ for some } x_1\,,\,\ldots\,,\,x_k\,,\,t_1\,,\,\ldots\,,\,t_k\big]$$

   Then, by applying standard inference rules, conclude:

   $$\left[\begin{array}{c} 1 \leq i \leq k \text{ and } \ell_i = \ell \text{ and} \\ \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_k = \mathtt{d}_{x_k} \approx t_k \end{array}\right] \text{ for some } x_1\,,\,\ldots\,,\,x_k\,,\,t_1\,,\,\ldots\,,\,t_k$$

   Then, by applying substitution, conclude $\big[\ell_i = \mathtt{d}_{x_i} \approx t_i$ **and** $\ell_i = \ell\big]$. Then, by applying substitution, conclude $\ell = \mathtt{d}_{x_i} \approx t_i$.

   (Y2) Suppose:

   $$\big[(\ell_1\,,\,\ell_2) \in \prec_L^{\text{arb}}$$ **and** $$\ell_2 \neq \ell_3\big]$$ **for some** $\ell_1\,,\,\ell_2\,,\,\ell_3$

   Then, by applying standard inference rules, conclude $\big[(\ell_1\,,\,\ell_2) \in \prec_L^{\text{arb}}$ **and** $(\ell_1\,,\,\ell_2) \neq (\ell_2\,,\,\ell_3)\big]$. Then, by applying set theory, conclude $(\ell_1\,,\,\ell_2) \in \prec_L^{\text{arb}} \setminus \{(\ell_2\,,\,\ell_3)\}$.

   (Y3) Suppose:

   $$(\ell_\dagger\,,\,\ell_\ddagger) \in \prec_L^{\text{arb}} \text{ for some } \ell_\dagger\,,\,\ell_\ddagger$$

   Then, by applying set theory, conclude $|\prec_L^{\text{arb}} \setminus \{(\ell_\dagger\,,\,\ell_\ddagger)\}| = |\prec_L^{\text{arb}}| - 1$. Then, by applying arithmetic, conclude $|\prec_L^{\text{arb}} \setminus \{(\ell_\dagger\,,\,\ell_\ddagger)\}| < |\prec_L^{\text{arb}}|$.

   (Y4) Suppose:

   $$\big[\ell_1 \prec_L^{\text{arb}} \ell_2 \prec_L^{\text{arb}} \ell_3$$ **and** $$\ell_2 \neq \ell_3\big]$$ **for some** $\ell_1\,,\,\ell_2\,,\,\ell_3$

   Then, by applying (Y2), conclude $\big[\ell_2 \prec_L^{\text{arb}} \ell_3$ **and** $(\ell_1\,,\,\ell_2) \in \prec_L^{\text{arb}} \setminus \{(\ell_2\,,\,\ell_3)\}\big]$. Then, by applying (Y3), conclude $\big[|\prec_L^{\text{arb}} \setminus \{(\ell_2\,,\,\ell_3)\}| < |\prec_L^{\text{arb}}|$ **and** $(\ell_1\,,\,\ell_2) \in \prec_L^{\text{arb}} \setminus \{(\ell_2\,,\,\ell_3)\}\big]$. Then, by applying **IH**, conclude $\big[\ell_1 = \mathtt{d}_x \approx t$ **for some** $x\,,\,t\big]$.

(Y5) Suppose:

$$\big[\ell_1 \prec_L^{\text{arb}} \ell_2 \prec_L^{\text{arb}} \ell_3 \ \textbf{and}\ \ell_2 \notin \{\ell_1\,,\,\ell_3\} \ \textbf{and}\ \ell_1 = \ell\big] \ \textbf{for some}\ \ell_1\,,\,\ell_2\,,\,\ell_3$$

Then, by applying set theory, conclude $\big[\ell_1 \prec_L^{\text{arb}} \ell_2 \prec_L^{\text{arb}} \ell_3 \ \textbf{and}\ \ell_2 \neq \ell_3 \ \textbf{and}\ \ell_1 = \ell\big]$. Then, by applying (Y4), conclude $\big[\big[\ell_1 = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big] \ \textbf{and}\ \ell_1 = \ell\big]$. Then, by applying substitution, conclude $\big[\ell = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big]$.

Now, prove the inductive step by the following reduction. Recall $\ell \prec_L^{\text{arb}} \ell''$ from (A1). Then, by Definition 15 of $\prec_L^{\text{arb}}$, conclude:

$$\begin{array}{l}\big[\big[\{\ell_1\,,\ldots,\ell_k\} \blacktriangleleft_L^{\text{arb}} \ell'' \ \textbf{and}\ 1 \leq i \leq k \ \textbf{and}\ \ell_i = \ell\big] \ \textbf{for some}\ \ell_1\,,\ldots,\ell_k\,,k\big] \\ \textbf{or}\ \big[\big[\ell_1 \prec_L^{\text{arb}} \ell_2 \prec_L^{\text{arb}} \ell_3 \ \textbf{and}\ \ell_2 \notin \{\ell_1\,,\ell_3\} \ \textbf{and}\ \ell_1 = \ell\big] \ \textbf{for some}\ \ell_1\,,\ell_2\,,\ell_3\big]\end{array}$$

Then, by applying (Y1), conclude:

$$\begin{array}{l}\big[\ell = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big] \\ \textbf{or}\ \big[\big[\ell_1 \prec_L^{\text{arb}} \ell_2 \prec_L^{\text{arb}} \ell_3 \ \textbf{and}\ \ell_2 \notin \{\ell_1\,,\ell_3\} \ \textbf{and}\ \ell_1 = \ell \ \textbf{and}\ \ell_3 = \ell''\big] \ \textbf{for some}\ \ell_1\,,\ell_2\,,\ell_3\big]\end{array}$$

Then, by applying (Y5), conclude $\big[\big[\ell = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big] \ \textbf{or}\ \big[\ell = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big]\big]$. Then, by applying standard inference rules, conclude $\big[\ell = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big]$.

(⟪ QED. ⟫)

2. First, assume:

(B1) $\ell \prec_L \ell''$

Now, prove the lemma by induction on $|\prec_L|$.

– **Base:** $|\prec_L| = 0$
First, observe:

(X1) Recall $|\prec_L| = 0$ from **Base**. Then, by applying set theory, conclude $\prec_L = \emptyset$.

Now, prove the base case by the following reduction. Recall $\ell \prec_L \ell''$ from (A1). Then, by applying set theory, conclude $(\ell\,,\ell'') \in \prec_L$. Then, by applying (X1), conclude $(\ell\,,\ell'') \in \emptyset$. Then, by applying set theory, conclude **false**.

– **IH:** $\big[\big[|\hat{\prec}_L| < |\prec_L| \ \textbf{and}\ \hat{\ell}\hat{\prec}_L\hat{\ell}''\big] \ \textbf{implies}\ \big[\hat{\ell} = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big]\big] \ \textbf{for all}\ \hat{\prec}_L\,,\hat{\ell}\,,\hat{\ell}''$

– **Step:** $|\prec_L| > 0$
First, observe:

(W1) Suppose:

$$\big[(\ell_1\,,\ell_2) \in \prec_L \ \textbf{and}\ \ell_2 \neq \ell_3\big] \ \textbf{for some}\ \ell_1\,,\ell_2\,,\ell_3$$

Then, by applying standard inference rules, conclude $\big[(\ell_1\,,\ell_2) \in \prec_L \ \textbf{and}\ (\ell_1\,,\ell_2) \neq (\ell_2\,,\ell_3)\big]$. Then, by applying set theory, conclude $(\ell_1\,,\ell_2) \in \prec_L \setminus \{(\ell_2\,,\ell_3)\}$.

(W2) Suppose:

$$(\ell_\dagger\,,\ell_\ddagger) \in \prec_L \ \textbf{for some}\ \ell_\dagger\,,\ell_\ddagger$$

Then, by applying set theory, conclude $|\prec_L \setminus \{(\ell_\dagger\,,\ell_\ddagger)\}| = |\prec_L| - 1$. Then, by applying arithmetic, conclude $|\prec_L \setminus \{(\ell_\dagger\,,\ell_\ddagger)\}| < |\prec_L|$.

(W3) Suppose:

$$\big[\ell_1 \prec_L \ell_2 \prec_L \ell_3 \ \textbf{and}\ \ell_2 \neq \ell_3\big] \ \textbf{for some}\ \ell_1\,,\ell_2\,,\ell_3$$

Then, by applying (W1), conclude $\big[\ell_2 \prec_L \ell_3 \ \textbf{and}\ (\ell_1\,,\ell_2) \in \prec_L \setminus \{(\ell_2\,,\ell_3)\}\big]$. Then, by applying (W2), conclude $\big[|\prec_L \setminus \{(\ell_2\,,\ell_3)\}| < |\prec_L| \ \textbf{and}\ (\ell_1\,,\ell_2) \in \prec_L \setminus \{(\ell_2\,,\ell_3)\}\big]$. Then, by applying **IH**, conclude $\big[\ell_1 = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big]$

(W4) Suppose:

$$\big[\ell_1 \prec_L \ell_2 \prec_L \ell_3 \ \textbf{and}\ \ell_2 \notin \{\ell_1\,,\ell_3\} \ \textbf{and}\ \ell_1 = \ell\big] \ \textbf{for some}\ \ell_1\,,\ell_2\,,\ell_3$$

Then, by applying set theory, conclude $\big[\ell_1 \prec_L \ell_2 \prec_L \ell_3 \ \textbf{and}\ \ell_2 \neq \ell_3 \ \textbf{and}\ \ell_1 = \ell\big]$. Then, by applying (W3), conclude $\big[\big[\ell_1 = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big] \ \textbf{and}\ \ell_1 = \ell\big]$. Then, by applying substitution, conclude $\big[\ell = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big]$.

Now, prove the inductive step by the following reduction. Recall $\ell \prec_L \ell''$ from (B1). Then, by Definition 16 of $\prec_L$, conclude:

$$\begin{array}{l}\ell \prec_L^{\text{arb}} \ell'' \\ \textbf{or}\ \big[\mathtt{d}_x \approx t = \ell \ \textbf{for some}\ x\,,t\big] \\ \textbf{or}\ \big[\big[\ell_1 \prec_L \ell_2 \prec_L \ell_3 \ \textbf{and}\ \ell_2 \notin \{\ell_1\,,\ell_3\} \ \textbf{and}\ \ell_1 = \ell\big] \ \textbf{for some}\ \ell_1\,,\ell_2\,,\ell_3\big]\end{array}$$

Then, by Lemma 11:1, conclude:

$$\begin{array}{l}\big[\ell = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big] \\ \textbf{or}\ \big[\mathtt{d}_x \approx t = \ell \ \textbf{for some}\ x\,,t\big] \\ \textbf{or}\ \big[\big[\ell_1 \prec_L \ell_2 \prec_L \ell_3 \ \textbf{and}\ \ell_2 \notin \{\ell_1\,,\ell_3\} \ \textbf{and}\ \ell_1 = \ell\big] \ \textbf{for some}\ \ell_1\,,\ell_2\,,\ell_3\big]\end{array}$$

Then, by applying (W4), conclude:

$$\big[\ell = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big] \ \textbf{or}\ \big[\mathtt{d}_x \approx t = \ell \ \textbf{for some}\ x\,,t\big] \ \textbf{or}\ \big[\ell = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big]$$

Then, by applying standard inference rules, conclude $\big[\ell = \mathtt{d}_x \approx t \ \textbf{for some}\ x\,,t\big]$.

(⟪ QED. ⟫)

3. First, assume:

(C1) $\ell \prec_L \ell''$

Now, prove the lemma by induction on $|\prec_L|$.

– **Base:** $|\prec_L| = 0$
First, observe:

(V1) Recall $|\prec_L| = 0$ from **Base**. Then, by applying set theory, conclude $\prec_L = \emptyset$.

Now, prove the base case by the following reduction. Recall $\ell \prec_L \ell''$ from (C1). Then, by applying set theory, conclude $(\ell\,,\ell'') \in \prec_L$. Then, by applying (V1), conclude $(\ell\,,\ell'') \in \emptyset$. Then, by applying set theory, conclude **false**.

– **IH:**

$$\big[\big[|\hat{\prec}_L| < |\prec_L| \ \textbf{and}\ \hat{\ell}\,\hat{\prec}_L\hat{\ell}''\big] \ \textbf{implies}\ \big[\hat{\ell} \prec_L^{\text{arb}} \hat{\ell}'' \ \textbf{or}\ \big[\hat{\ell}'' \neq \mathtt{d}_x \approx t \ \textbf{for all}\ x\,,t\big]\big]\big] \ \textbf{for all}\ \hat{\prec}_L\,,\hat{\ell}\,,\hat{\ell}''$$

– **Step:** $|\prec_L| > 0$
First, observe:

(U1) Suppose:

$$\big[(\ell_1\,,\ell_2) \in \prec_L \ \textbf{and}\ \ell_2 \neq \ell_3\big] \ \textbf{for some}\ \ell_1\,,\ell_2\,,\ell_3$$

Then, by applying standard inference rules, conclude $\big[(\ell_1\,,\ell_2) \in \prec_L \ \textbf{and}\ (\ell_1\,,\ell_2) \neq (\ell_2\,,\ell_3)\big]$. Then, by applying set theory, conclude $(\ell_1\,,\ell_2) \in \prec_L \setminus \{(\ell_2\,,\ell_3)\}$.

(U2) Suppose:

$$\big[(\ell_2\,,\ell_3) \in \prec_L \ \textbf{and}\ \ell_2 \neq \ell_1\big] \ \textbf{for some}\ \ell_1\,,\ell_2\,,\ell_3$$

Then, by a reduction similar to (U1), conclude $(\ell_2\,,\ell_3) \in \prec_L \setminus \{(\ell_1\,,\ell_2)\}$.

U3 Suppose:

$$(\ell_\dagger\,,\,\ell_\ddagger) \in \prec_L \textbf{ for some } \ell_\dagger\,,\,\ell_\ddagger$$

Then, by applying set theory, conclude $|\prec_L \setminus \{(\ell_\dagger\,,\,\ell_\ddagger)\}| = |\prec_L| - 1$. Then, by applying arithmetic, conclude $|\prec_L \setminus \{(\ell_\dagger\,,\,\ell_\ddagger)\}| < |\prec_L|$.

U4 Suppose:

$$\big[\ell_1 \prec_L \ell_2 \prec_L \ell_3 \textbf{ and } \ell_2 \neq \ell_3\big] \textbf{ for some } \ell_1\,,\,\ell_2\,,\,\ell_3$$

Then, by applying U1, conclude $\big[\ell_2 \prec_L \ell_3 \textbf{ and } (\ell_1\,,\,\ell_2) \in \prec_L \setminus \{(\ell_2\,,\,\ell_3)\}\big]$. Then, by applying U3, conclude $\big[|\prec_L \setminus \{(\ell_2\,,\,\ell_3)\}| < |\prec_L| \textbf{ and } (\ell_1\,,\,\ell_2) \in \prec_L \setminus \{(\ell_2\,,\,\ell_3)\}\big]$. Then, by applying **IH**, conclude $\big[\ell_1 \prec_L^{\text{arb}} \ell_2 \textbf{ or } \big[\ell_2 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big]\big]$.

U5 Suppose:

$$\big[\ell_1 \prec_L \ell_2 \prec_L \ell_3 \textbf{ and } \ell_2 \neq \ell_1\big] \textbf{ for some } \ell_1\,,\,\ell_2\,,\,\ell_3$$

Then, by a reduction similar to U4, conclude $\big[\ell_2 \prec_L^{\text{arb}} \ell_3 \textbf{ or } \big[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big]\big]$.

U6 Suppose:

$$\big[\ell_1 \prec_L^{\text{arb}} \ell_2 \textbf{ and } \ell_2 \prec_L^{\text{arb}} \ell_3\big] \textbf{ for some } \ell_1\,,\,\ell_2\,,\,\ell_3$$

Then, by applying Lemma 10:2, conclude:

$$\ell_1 \prec_L^{\text{arb}} \ell_2 \textbf{ and } \ell_2 \prec_L^{\text{arb}} \ell_3 \textbf{ and } \ell_2 \neq \ell_1 \textbf{ and } \ell_2 \neq \ell_3$$

Then, by applying set theory, conclude $\big[\ell_1 \prec_L^{\text{arb}} \ell_2 \textbf{ and } \ell_2 \prec_L^{\text{arb}} \ell_3 \textbf{ and } \ell_2 \notin \{\ell_1\,,\,\ell_3\}\big]$. Then, by applying Definition 15 of $\prec_L^{\text{arb}}$, conclude $\ell_1 \prec_L^{\text{arb}} \ell_3$.

U7 Suppose:

$$\big[\big[\ell_2 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big] \textbf{ and } \ell_2 \prec_L^{\text{arb}} \ell_3\big] \textbf{ for some } \ell_1\,,\,\ell_2\,,\,\ell_3$$

Then, by applying Lemma 11:1, conclude:

$$\big[\ell_2 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big] \textbf{ and } \big[\ell_2 = \mathtt{d}_x \approx t \textbf{ for some } x\,,\,t\big]$$

Then, by applying standard inference rules, conclude **false**.

U8 Suppose:

$$\big[\ell_1 \prec_L \ell_2 \prec_L \ell_3 \textbf{ and } \ell_2 \neq \ell_1 \textbf{ and } \ell_2 \neq \ell_3\big] \textbf{ for some } \ell_1\,,\,\ell_2\,,\,\ell_3$$

Then, by applying U4, conclude:

$$\ell_1 \prec_L \ell_2 \prec_L \ell_3 \textbf{ and } \ell_2 \neq \ell_1 \textbf{ and } \big[\ell_1 \prec_L^{\text{arb}} \ell_2 \textbf{ or } \big[\ell_2 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big]\big]$$

Then, by applying U5, conclude:

$$\big[\ell_1 \prec_L^{\text{arb}} \ell_2 \textbf{ or } \big[\ell_2 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big]\big] \textbf{ and } \big[\ell_2 \prec_L^{\text{arb}} \ell_3 \textbf{ or } \big[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big]\big]$$

Then, by applying standard inference rules, conclude:

$$\begin{array}{l}\big[\ell_1 \prec_L^{\text{arb}} \ell_2 \textbf{ and } \ell_2 \prec_L^{\text{arb}} \ell_3\big] \\ \textbf{or } \big[\ell_1 \prec_L^{\text{arb}} \ell_2 \textbf{ and } \big[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big]\big] \\ \textbf{or } \big[\big[\ell_2 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big] \textbf{ and } \ell_2 \prec_L^{\text{arb}} \ell_3\big] \\ \textbf{or } \big[\big[\ell_2 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big] \textbf{ and } \big[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big]\big]\end{array}$$

Then, by applying U6, conclude:

$$\begin{array}{l}\ell_1 \prec_L^{\text{arb}} \ell_3 \\ \textbf{or } \big[\ell_1 \prec_L^{\text{arb}} \ell_2 \textbf{ and } \big[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big]\big] \\ \textbf{or } \big[\big[\ell_2 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big] \textbf{ and } \ell_2 \prec_L^{\text{arb}} \ell_3\big] \\ \textbf{or } \big[\big[\ell_2 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big] \textbf{ and } \big[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\,t\big]\big]\end{array}$$

Then, by applying standard inference rules, conclude:

$$\begin{array}{l}\ell_1 \prec_L^{\text{arb}} \ell_3 \\ \textbf{or } \left[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\, t\right] \\ \textbf{or } \left[\left[\ell_2 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\, t\right] \textbf{ and } \ell_2 \prec_L^{\text{arb}} \ell_3\right] \\ \textbf{or } \left[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\, t\right]\end{array}$$

Then, by applying (U7), conclude:

$$\ell_1 \prec_L^{\text{arb}} \ell_3 \textbf{ or } \left[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\, t\right] \textbf{ or false or } \left[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\, t\right]$$

Then, by applying standard inference rules, conclude:

$$\ell_1 \prec_L^{\text{arb}} \ell_3 \textbf{ or } \left[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\, t\right]$$

(U9) Suppose:

$$\left[\ell_1 \prec_L \ell_2 \prec_L \ell_3 \textbf{ and } \ell_2 \notin \{\ell_1\,,\, \ell_3\} \textbf{ and } \ell_1 = \ell \textbf{ and } \ell_3 = \ell''\right] \textbf{ for some } \ell_1\,,\, \ell_2\,,\, \ell_3$$

Then, by applying set theory, conclude:

$$\ell_1 \prec_L \ell_2 \prec_L \ell_3 \textbf{ and } \ell_2 \neq \ell_1 \textbf{ and } \ell_2 \neq \ell_3 \textbf{ and } \ell_1 = \ell \textbf{ and } \ell_3 = \ell''$$

Then, by applying (U8), conclude:

$$\ell_1 \prec_L^{\text{arb}} \ell_3 \textbf{ or } \left[\ell_3 \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\, t\right] \textbf{ and } \ell_1 = \ell \textbf{ and } \ell_3 = \ell''$$

Then, by applying substitution, conclude $\left[\ell \prec_L^{\text{arb}} \ell'' \textbf{ or } \left[\ell'' \neq \mathtt{d}_x \approx t \textbf{ for all } x\,,\, t\right]\right]$.

Now, prove the inductive step by the following reduction. Recall $\ell \prec_L \ell''$ from (B1). Then, by Definition 16 of $\prec_L$, conclude:

$$\begin{array}{l}\ell \prec_L^{\text{arb}} \ell'' \\ \textbf{or } \left[\ell'' \neq \mathtt{d}_{x'} \approx t' \textbf{ for all } x'\,,\, t'\right] \\ \textbf{or } \left[\left[\ell_1 \prec_L \ell_2 \prec_L \ell_3 \textbf{ and } \ell_2 \notin \{\ell_1\,,\, \ell_3\} \textbf{ and } \ell_1 = \ell \textbf{ and } \ell_3 = \ell'\right] \textbf{ for some } \ell_1\,,\, \ell_2\,,\, \ell_3\right]\end{array}$$

Then, by applying (U9), conclude:

$$\ell \prec_L^{\text{arb}} \ell'' \textbf{ or } \left[\ell'' \neq \mathtt{d}_{x'} \approx t' \textbf{ for all } x'\,,\, t'\right] \textbf{ or } \ell \prec_L^{\text{arb}} \ell'' \textbf{ or } \left[\ell'' \neq \mathtt{d}_{x'} \approx t' \textbf{ for all } x'\,,\, t'\right]$$

Then, by applying standard inference rules, conclude $\left[\ell \prec_L^{\text{arb}} \ell'' \textbf{ or } \left[\ell'' \neq \mathtt{d}_{x'} \approx t' \textbf{ for all } x'\,,\, t'\right]\right]$.

(⟪ QED. ⟫)

4. First, observe:

(T1) Suppose:

$$\ell \prec_L \ell \textbf{ for some } \ell$$

Then, by applying Lemma 11:3, conclude $\left[\ell \prec_L^{\text{arb}} \ell \textbf{ or } \left[\ell \neq \mathtt{d}_{x'} \approx t' \textbf{ for all } x'\,,\, t'\right]\right]$. Then, by applying Lemma 10:1, conclude $\left[\textbf{false or } \left[\ell \neq \mathtt{d}_{x'} \approx t' \textbf{ for all } x'\,,\, t'\right]\right]$. Then, by applying standard inference rules, conclude $\left[\ell \neq \mathtt{d}_{x'} \approx t' \textbf{ for all } x'\,,\, t'\right]$.

(T2) Suppose:

$$\ell \prec_L \ell \textbf{ for some } \ell$$

Then, by applying (T1), conclude $\left[\ell \prec_L \ell \textbf{ and } \left[\ell \neq \mathtt{d}_{x'} \approx t' \textbf{ for all } x'\,,\, t'\right]\right]$ Then, by applying Lemma 11:2, conclude $\left[\left[\ell = \mathtt{d}_x \approx t \textbf{ for some } x\,,\, t\right] \textbf{ and } \left[\ell \neq \mathtt{d}_{x'} \approx t' \textbf{ for all } x'\,,\, t'\right]\right]$. Then, by applying standard inference rules, conclude **false**.

Now, prove the lemma by the following reduction. Recall $\left[\ell \not\prec_L \ell \textbf{ for all } \ell\right]$ from (T2). Then, by applying order theory, conclude $\left[\prec_L \text{ is irreflexive}\right]$.

(⟪ QED. ⟫)

B.12 Lemma 12

Proof (of Lemma 12). First, observe:

(Z1) Suppose:

$$\big[\ell_1 \prec_L \ell_2 \textbf{ and } \ell_2 \prec_L \ell_3\big] \textbf{ for some } \ell_1\,,\ \ell_2\,,\ \ell_3$$

Then, by applying Lemma 11:4, conclude $\big[\ell_1 \prec_L \ell_2 \textbf{ and } \ell_2 \prec_L \ell_3 \textbf{ and } \ell_2 \neq \ell_1 \textbf{ and } \ell_2 \neq \ell_3\big]$. Then, by applying set theory, conclude $\big[\ell_1 \prec_L \ell_2 \textbf{ and } \ell_2 \prec_L \ell_3 \textbf{ and } \ell_2 \notin \{\ell_1\,, \ell_3\}\big]$. Then, by applying Definition 16 of $\prec_L$, conclude $\ell_1 \prec_L \ell_3$.

(Z2) Recall $\big[\big[\big[\ell_1 \prec_L \ell_2 \textbf{ and } \ell_2 \prec_L \ell_3\big] \textbf{ implies } \ell_1 \prec_L \ell_3\big] \textbf{ for all } \ell_1\,, \ell_2\,, \ell_3\big]$. Then, by applying order theory, conclude $\big[\prec_L \text{ is transitive}\big]$.

(Z3) Suppose:

$$\big[\ell_1 \prec_L \ell_2 \textbf{ and } \ell_2 \prec_L \ell_1\big] \textbf{ for some } \ell_1\,,\ \ell_2$$

Then, by applying (Z2), conclude $\ell_1 \prec_L \ell_1$. Then, by applying Lemma 11:4, conclude **false**.

(Z4) Recall $\big[\big[\textbf{not } \big[\ell_1 \prec_L \ell_2 \textbf{ and } \ell_2 \prec_L \ell_1\big]\big] \textbf{ for all } \ell_1\,, \ell_2\big]$ from (Z3). Then, by applying standard inference rules, conclude $\big[\big[\ell_1 \not\prec_L \ell_2 \textbf{ or } \ell_2 \not\prec_L \ell_1\big] \textbf{ for all } \ell_1\,, \ell_2\big]$. Then, by applying standard inference rules, conclude $\big[\big[\ell_1 \prec_L \ell_2 \textbf{ implies } \ell_2 \not\prec_L \ell_1\big] \textbf{ for all } \ell_1\,, \ell_2\big]$. Then, by applying order theory, conclude $\big[\prec_L \text{ is asymmetric}\big]$.

Now, prove the lemma by the following reduction. Recall $\big[\prec_L \text{ is transitive}\big]$ from (Z2). Then, by introducing (Z4), conclude $\big[\prec_L \text{ is transitive and asymmetric}\big]$. Then, by introducing Lemma 11:4, conclude:

$$\prec_L \text{ is transitive, asymmetric, and irreflexive}$$

Then, by applying order theory, conclude $\big[\prec_L \text{ is a strict partial order}\big]$.

((QED.))

B.13 Lemma 13

Proof (of Lemma 13). First, assume:

(A1) $\mathtt{d}_x \approx t \in L$

(A2) $\mathtt{d}_y \in \mathsf{Free}(t)$

Next, observe:

(Z1) Suppose $\mathsf{Free}(\mathtt{d}_x \approx t) = \emptyset$. Then, by applying Definition 8 of Free, conclude $\mathsf{Free}(\mathtt{d}_x) \cup \mathsf{Free}(t) = \emptyset$. Then, by applying Definition 7 of Free, conclude $\{\mathtt{d}_x\} \cup \mathsf{Free}(t) = \emptyset$. Then, by applying set theory, conclude $\big[\mathtt{d}_x \in \{\mathtt{d}_x\} \cup \mathsf{Free}(t)$ **and** $\{\mathtt{d}_x\} \cup \mathsf{Free}(t) = \emptyset\big]$. Then, by applying set theory, conclude **false**.

(Z2) Suppose $\mathsf{Free}(t) = \emptyset$. Then, by introducing (A2), conclude $\big[\mathsf{Free}(t) = \emptyset$ **and** $\mathtt{d}_y \in \mathsf{Free}(t)\big]$. Then, by applying set theory, conclude **false**.

(Z3) Recall $\mathtt{d}_y \in \mathsf{Free}(t)$ from (A2). Then, by applying set theory, conclude $\mathtt{d}_y \in \{\mathtt{d}_x\} \cup \mathsf{Free}(t)$. Then, by applying Definition 7 of Free, conclude $\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_x \approx t)$.

(Z4) Suppose:

$$\mathsf{Free}(\mathtt{d}_x \approx t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\} \textbf{ for some } x_1, \ldots, x_k, k$$

Then, by introducing (Z3), conclude $\big[\mathsf{Free}(\mathtt{d}_x \approx t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\}$ **and** $\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_x \approx t)\big]$. Then, by applying substitution, conclude $\mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\}$. Then, by applying set theory, conclude:

$$\big[\mathtt{d}_y = \mathtt{d}_{x_i} \textbf{ and } 1 \leq i \leq k\big] \textbf{ for some } i$$

(Z5) Suppose:

$$\mathsf{Free}(t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\} \textbf{ for some } x_1, \ldots, x_k, k$$

Then, by introducing (A2), conclude $\big[\mathsf{Free}(\mathtt{d}_x \approx t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\}$ **and** $\mathtt{d}_y \in \mathsf{Free}(t)\big]$. Then, by applying substitution, conclude $\mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\}$. Then, by applying set theory, conclude:

$$\big[\mathtt{d}_y = \mathtt{d}_{x_i} \textbf{ and } 1 \leq i \leq k\big] \textbf{ for some } i$$

(Z6) Suppose:

$$L' \blacktriangleleft_L^{\text{arb}} \mathtt{d}_x \approx t \textbf{ for some } L'$$

Then, by applying Definition 13 of $\blacktriangleleft_L$, conclude:

$$\begin{array}{l}
\mathsf{Free}(\mathtt{d}_x \approx t) = \emptyset \\
\textbf{or } \mathsf{Free}(t) = \emptyset \\
\textbf{or } \left[\begin{array}{c}\big[\mathsf{Free}(\mathtt{d}_x \approx t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\} \textbf{ and } L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\big] \\ \textbf{for some } x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}\right] \\
\textbf{or } \left[\begin{array}{c}\big[\mathsf{Free}(t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\} \textbf{ and } L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\big] \\ \textbf{for some } x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}\right]
\end{array}$$

Then, by applying (Z1), conclude:

$$\begin{array}{l}
\textbf{false} \\
\textbf{or } \mathsf{Free}(t) = \emptyset \\
\textbf{or } \left[\begin{array}{c}\big[\mathsf{Free}(\mathtt{d}_x \approx t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\} \textbf{ and } L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\big] \\ \textbf{for some } x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}\right] \\
\textbf{or } \left[\begin{array}{c}\big[\mathsf{Free}(t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\} \textbf{ and } L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\big] \\ \textbf{for some } x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}\right]
\end{array}$$

Then, by applying (Z2), conclude:

$$\begin{array}{l}\textbf{false}\\ \textbf{or}\ \ \textbf{false}\\ \textbf{or}\ \ \left[\begin{array}{c}\left[\mathsf{Free}(\mathtt{d}_x \approx t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\}\ \textbf{and}\ L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\right]\\ \textbf{for some}\ x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}\right]\\ \textbf{or}\ \ \left[\begin{array}{c}\left[\mathsf{Free}(t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\}\ \textbf{and}\ L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\right]\\ \textbf{for some}\ x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}\right]\end{array}$$

Then, by applying (Z4), conclude:

$$\begin{array}{l}\textbf{false}\\ \textbf{or}\ \ \textbf{false}\\ \textbf{or}\ \ \left[\begin{array}{c}\left[\left[\left[\mathtt{d}_y = \mathtt{d}_{x_i}\ \textbf{and}\ 1 \le i \le k\right]\ \textbf{for some}\ i\right]\ \textbf{and}\ L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\right]\\ \textbf{for some}\ x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}\right]\\ \textbf{or}\ \ \left[\begin{array}{c}\left[\mathsf{Free}(t) = \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_k}\}\ \textbf{and}\ L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\right]\\ \textbf{for some}\ x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}\right]\end{array}$$

Then, by applying (Z5), conclude:

$$\begin{array}{l}\textbf{false}\\ \textbf{or}\ \ \textbf{false}\\ \textbf{or}\ \ \left[\begin{array}{c}\left[\left[\left[\mathtt{d}_y = \mathtt{d}_{x_i}\ \textbf{and}\ 1 \le i \le k\right]\ \textbf{for some}\ i\right]\ \textbf{and}\ L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\right]\\ \textbf{for some}\ x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}\right]\\ \textbf{or}\ \ \left[\begin{array}{c}\left[\left[\left[\mathtt{d}_y = \mathtt{d}_{x_i}\ \textbf{and}\ 1 \le i \le k\right]\ \textbf{for some}\ i\right]\ \textbf{and}\ L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\right]\\ \textbf{for some}\ x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}\right]\end{array}$$

Then, by applying standard inference rules, conclude:

$$\begin{array}{c}\left[\left[\left[\mathtt{d}_y = \mathtt{d}_{x_i}\ \textbf{and}\ 1 \le i \le k\right]\ \textbf{for some}\ i\right]\ \textbf{and}\ L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\right]\\ \textbf{for some}\ x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}$$

Then, by applying standard inference rules, conclude:

$$\left[\mathtt{d}_y = \mathtt{d}_{x_i}\ \textbf{and}\ 1 \le i \le k\ \textbf{and}\ L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\right]\ \textbf{for some}\ i$$

Now, prove the lemma by the following reduction. Recall $\mathtt{d}_x \approx t \in L$ from (A1). Then, by applying set theory, conclude $\mathtt{d}_x \approx t \in L \setminus \{\bigstar\}$. Then, by applying Proposition 5:1, conclude:

$$L' \blacktriangleleft_L^{\text{arb}} \mathtt{d}_x \approx t\ \textbf{for some}\ L'$$

Then, by applying (Z6), conclude:

$$L' \blacktriangleleft_L^{\text{arb}} \mathtt{d}_x \approx t\ \textbf{and}\ \left[\begin{array}{c}\left[\mathtt{d}_y = \mathtt{d}_{x_i}\ \textbf{and}\ 1 \le i \le k\ \textbf{and}\ L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\right]\\ \textbf{for some}\ i, x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}\right]$$

Then, by applying standard inference rules, conclude:

$$\begin{array}{c}\left[L' \blacktriangleleft_L^{\text{arb}} \mathtt{d}_x \approx t\ \textbf{and}\ \mathtt{d}_y = \mathtt{d}_{x_i}\ \textbf{and}\ 1 \le i \le k\ \textbf{and}\ L' = \{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\}\right]\\ \textbf{for some}\ i, x_1, \ldots, x_k, t_1, \ldots, t_k, k\end{array}$$

Then, by applying substitution, conclude:

$$\{\mathtt{d}_{x_1} \approx t_1, \ldots, \mathtt{d}_{x_k} \approx t_k\} \blacktriangleleft_L^{\text{arb}} \mathtt{d}_x \approx t\ \textbf{and}\ \mathtt{d}_y = \mathtt{d}_{x_i}\ \textbf{and}\ 1 \le i \le k$$

Then, by applying Definition 15 of $\prec_L^{\text{arb}}$, conclude $\left[\mathtt{d}_{x_i} \approx t_i \prec_L^{\text{arb}} \mathtt{d}_x \approx t\ \textbf{and}\ \mathtt{d}_y = \mathtt{d}_{x_i}\right]$. Then, by applying substitution, conclude $\mathtt{d}_y \approx t_i \prec_L^{\text{arb}} \mathtt{d}_x \approx t$. Then, by applying Definition 16 of $\prec_L$, conclude:

$$\mathtt{d}_y \approx t_i \prec_L \mathtt{d}_x \approx t$$

⟨ QED. ⟩

B.14 Lemma 14

Proof (of Lemma 14). First, observe:

(Z1) Recall $\left[\prec_L \text{ is a strict partial order}\right]$ from Lemma 12. Then, by applying order theory, conclude:

$$\left[< \text{ is a linear extension of } \prec_L\right] \textbf{ for some } <$$

(Z2) Suppose:

$$\left[< \text{ is a strict total order on } L\right] \textbf{ for some } <$$

Then, by applying set theory, conclude:

$$\left[< \text{ is a strict total order on } L\right] \textbf{ and } \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} \subseteq L$$

Then, by applying order theory, conclude $\left[< \text{ is a strict total order on } \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\}\right]$. Then, by applying order theory, conclude:

$$\left[\{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \textbf{ and } \ell_1 < \cdots < \ell_n\right] \textbf{ for some } \ell_1, \ldots, \ell_n, n$$

(Z3) Suppose:

$$\left[< \text{ is a strict total order on } L\right] \textbf{ for some } <$$

Then, by applying set theory, conclude:

$$\left[< \text{ is a strict total order on } L\right] \textbf{ and } L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} \subseteq L$$

Then, by applying order theory, conclude $\left[< \text{ is a strict total order on } L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\}\right]$. Then, by applying order theory, conclude:

$$\left[L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_m\} \textbf{ and } \ell_1 < \cdots < \ell_m\right] \textbf{ for some } \ell_1, \ldots, \ell_m, m$$

(Z4) Suppose:

$$\left[< \text{ is a linear extension of } \prec_L\right] \textbf{ for some } <$$

Then, by applying order theory, conclude $\left[< \text{ is a strict total order on } L\right]$. Then, by applying (Z2):

$$\begin{array}{l}\left[< \text{ is a strict total order on } L\right] \\ \textbf{and } \left[\left[\{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \textbf{ and } \ell_1 < \cdots < \ell_n\right] \textbf{ for some } \ell_1, \ldots, \ell_n, n\right]\end{array}$$

Then, by applying standard inference rules, conclude:

$$\begin{array}{c}\left[\left[< \text{ is a strict total order on } L\right] \textbf{ and } \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \textbf{ and } \ell_1 < \cdots < \ell_n\right] \\ \textbf{for some } \ell_1, \ldots, \ell_n, n\end{array}$$

Then, by applying (Z3), conclude:

$$\begin{array}{c}\{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \textbf{ and } \ell_1 < \cdots < \ell_n \textbf{ and} \\ \left[\begin{array}{c}\left[L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_{n+1}, \ldots, \ell_{n+m}\} \textbf{ and } \ell_{n+1} < \cdots < \ell_{n+m}\right] \\ \textbf{for some } \ell_{n+1}, \ldots, \ell_{n+m}, m\end{array}\right]\end{array}$$

Then, by applying standard inference rules, conclude:

$$\begin{array}{c}\left[\begin{array}{c}\{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \textbf{ and } \ell_1 < \cdots < \ell_n \textbf{ and} \\ L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_{n+1}, \ldots, \ell_{n+m}\} \textbf{ and } \ell_{n+1} < \cdots < \ell_{n+m}\end{array}\right] \\ \textbf{for some } \ell_{n+1}, \ldots, \ell_{n+m}, m\end{array}$$

(Z5) Recall **true**. Then, by applying set theory, conclude $\{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} \subseteq L$. Then, by applying set theory, conclude $L = \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} \cup L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\}$.

(Z6) Suppose:

$$\begin{bmatrix}\{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_1 , \ldots , \ell_n\} \\ \text{and } L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_{n+1} , \ldots , \ell_{n+m}\}\end{bmatrix} \text{ for some } \ell_1 , \ldots , \ell_{n+m} , n , m$$

Then, by introducing (Z5), conclude:

$$\begin{array}{l}\{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_1 , \ldots , \ell_n\} \\ \text{and } L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_{n+1} , \ldots , \ell_{n+m}\} \\ \text{and } L = \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} \cup L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\}\end{array}$$

Then, by applying substitution, conclude $L = \{\ell_1 , \ldots , \ell_n\} \cup \{\ell_{n+1} , \ldots , \ell_{n+m}\}$. Then, by applying set theory, conclude $L = \{\ell_1 , \ldots , \ell_{n+m}\}$. Then, by applying set theory, conclude $L = \{\ell_j \mid 1 \leq j \leq n+m\}$.

(Z7) Suppose:

$$\ell' \in \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} \text{ for some } \ell'$$

Then, by applying set theory, conclude $\big[\big[\ell' = \mathtt{d}_x \approx t \text{ and } \ell' \in L\big] \text{ for some } x , t\big]$. Then, by applying substitution, conclude $\big[\big[\ell' = \mathtt{d}_x \approx t \text{ and } \mathtt{d}_x \approx t \in L\big] \text{ for some } x , t\big]$.

(Z8) Suppose:

$$\ell'' \notin \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} \text{ for some } \ell''$$

Then, by applying set theory, conclude $\big[\text{not } \big[\big[\ell'' = \mathtt{d}_x \approx t \text{ and } \ell'' \in L\big] \text{ for some } x , t\big]\big]$. Then, by applying standard inference rules, conclude $\big[\text{not } \big[\big[\ell'' = \mathtt{d}_x \approx t \text{ for some } x , t\big] \text{ and } \ell'' \in L\big]\big]$. Then, by applying standard inference rules, conclude $\big[\big[\text{not } \big[\ell'' = \mathtt{d}_x \approx t \text{ for some } x , t\big]\big] \text{ or } \ell'' \notin L\big]$. Then, by applying standard inference rules, conclude $\big[\big[\ell'' \neq \mathtt{d}_x \approx t \text{ for all } x , t\big] \text{ or } \ell'' \notin L\big]$.

(Z9) Suppose:

$$\ell'' \in L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} \text{ for some } \ell''$$

Then, by applying set theory, conclude $\big[\ell'' \in L \text{ and } \ell'' \notin \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\}\big]$. Then, by applying (Z8), conclude $\big[\ell'' \in L \text{ and } \big[\big[\ell'' \neq \mathtt{d}_x \approx t \text{ for all } x , t\big] \text{ or } \ell'' \notin L\big]\big]$. Then, by applying standard inference rules, conclude $\big[\big[\ell'' \in L \text{ and } \big[\ell'' \neq \mathtt{d}_x \approx t \text{ for all } x , t\big]\big] \text{ or } \big[\ell'' \in L \text{ and } \ell'' \notin L\big]\big]$. Then, by applying set theory, conclude $\big[\big[\ell'' \in L \text{ and } \big[\ell'' \neq \mathtt{d}_x \approx t \text{ for all } x , t\big]\big] \text{ or false}\big]$. Then, by applying standard inference rules, conclude $\big[\ell'' \in L \text{ and } \big[\ell'' \neq \mathtt{d}_x \approx t \text{ for all } x , t\big]\big]$.

(Z0) Suppose:

$$\big[\ell' \in \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} \text{ and } \ell'' \in L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\}\big] \text{ for some } \ell' , \ell''$$

Then, by applying (Z7), conclude:

$$\big[\big[\ell' = \mathtt{d}_x \approx t \text{ and } \mathtt{d}_x \approx t \in L\big] \text{ for some } x , t\big] \text{ and } \ell'' \in L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\}$$

Then, by applying (Z9), conclude:

$$\big[\big[\ell' = \mathtt{d}_x \approx t \text{ and } \mathtt{d}_x \approx t \in L\big] \text{ for some } x , t\big] \text{ and } \ell'' \in L \text{ and } \big[\ell'' \neq \mathtt{d}_{x'} \approx t' \text{ for all } x' , t'\big]$$

Then, by applying standard inference rules, conclude:

$$\big[\ell' = \mathtt{d}_x \approx t \text{ and } \mathtt{d}_x \approx t \in L \text{ and } \ell'' \in L \text{ and } \big[\ell'' \neq \mathtt{d}_{x'} \approx t' \text{ for all } x' , t'\big]\big] \text{ for some } x , t$$

Then, by applying Definition 16 of $\prec_L$, conclude $\big[\ell' = \mathtt{d}_x \approx t \text{ and } \mathtt{d}_x \approx t \prec_L \ell''\big]$. Then, by applying substitution, conclude $\ell' \prec_L \ell''$.

(Y1) Suppose:

$$\left[\begin{array}{l}[< \text{ is a linear extension of } \prec_L] \\ \textbf{and } \ell' \in \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} \\ \textbf{and } \ell'' \in L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\}\end{array}\right] \textbf{ for some } <, \ell', \ell''$$

Then, by applying (Z0), conclude:

$$\begin{array}{l}[< \text{ is a linear extension of } \prec_L] \\ \textbf{and } [[\left[\begin{array}{l}\ell' \in \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} \textbf{ and} \\ \ell'' \in L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\}\end{array}\right] \textbf{ implies } \ell' \prec_L \ell''] \textbf{ for all } \ell', \ell'']\end{array}$$

Then, by applying order theory, conclude:

$$\prec_L \subseteq < \textbf{ and } [[\left[\begin{array}{l}\ell' \in \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} \textbf{ and} \\ \ell'' \in L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\}\end{array}\right] \textbf{ implies } \ell' \prec_L \ell''] \textbf{ for all } \ell', \ell'']$$

Then, by applying set theory, conclude:

$$\prec_L \subseteq < \textbf{ and } [[\left[\begin{array}{l}\ell' \in \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} \textbf{ and} \\ \ell'' \in L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\}\end{array}\right] \textbf{ implies } \ell' < \ell''] \textbf{ for all } \ell', \ell'']$$

(Y2) Suppose:

$$\left[\begin{array}{l}[< \text{ is a linear extension of } \prec_L] \\ \textbf{and } \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \textbf{and } L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_{n+1}, \ldots, \ell_{n+m}\}\end{array}\right] \textbf{ for some } <, \ell_1, \ldots, \ell_{n+m}, n, m$$

Then, by introducing (Y1), conclude:

$$\begin{array}{l}\{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \textbf{and } L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_{n+1}, \ldots, \ell_{n+m}\} \\ \textbf{and } [[\left[\begin{array}{l}\ell' \in \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} \textbf{ and} \\ \ell'' \in L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\}\end{array}\right] \textbf{ implies } \ell' < \ell''] \textbf{ for all } \ell', \ell'']\end{array}$$

Then, by applying substitution, conclude:

$$[[\ell' \in \{\ell_1, \ldots, \ell_n\} \textbf{ and } \ell'' \in \{\ell_{n+1}, \ldots, \ell_{n+m}\}] \textbf{ implies } \ell' < \ell''] \textbf{ for all } \ell', \ell''$$

Then, by applying set theory, conclude:

$$[[\left[\begin{array}{l}\ell' = \ell_1 \textbf{ or } \cdots \textbf{ or } \ell' = \ell_n] \textbf{ and} \\ [\ell'' = \ell_{n+1} \textbf{ or } \cdots \textbf{ or } \ell'' = \ell_{n+m}]\end{array}\right] \textbf{ implies } \ell' < \ell''] \textbf{ for all } \ell', \ell''$$

Then, by applying standard inference rules, conclude:

$$[\left[\begin{array}{l}[[\ell' = \ell_1 \textbf{ and } \ell'' = \ell_{n+1}] \textbf{ or } \cdots \textbf{ or } [\ell' = \ell_1 \textbf{ and } \ell'' = \ell_{n+m}]] \\ \qquad\qquad \textbf{or } \cdots \textbf{ or} \\ [[\ell' = \ell_n \textbf{ and } \ell'' = \ell_{n+1}] \textbf{ or } \cdots \textbf{ or } [\ell' = \ell_n \textbf{ and } \ell'' = \ell_{n+m}]]\end{array}\right] \textbf{ implies } \ell' < \ell''] \textbf{ for all } \ell', \ell''$$

Then, by applying standard inference rules, conclude:

$$\left[\begin{array}{l}[[\left[\begin{array}{l}\ell' = \ell_1 \textbf{ and} \\ \ell'' = \ell_{n+1}\end{array}\right] \textbf{ implies } \ell' < \ell''] \textbf{ and } \cdots \textbf{ and } [\left[\begin{array}{l}\ell' = \ell_1 \textbf{ and} \\ \ell'' = \ell_{n+m}\end{array}\right] \textbf{ implies } \ell' < \ell'']] \\ \qquad\qquad \textbf{and } \cdots \textbf{ and} \\ [[\left[\begin{array}{l}\ell' = \ell_n \textbf{ and} \\ \ell'' = \ell_{n+1}\end{array}\right] \textbf{ implies } \ell' < \ell''] \textbf{ and } \cdots \textbf{ and } [\left[\begin{array}{l}\ell' = \ell_n \textbf{ and} \\ \ell'' = \ell_{n+m}\end{array}\right] \textbf{ implies } \ell' < \ell'']]\end{array}\right] \textbf{ for all } \ell', \ell''$$

Then, by applying standard inference rules, conclude:

$$\left[\left[\begin{bmatrix}\ell' = \ell_1 \text{ and}\\ \ell'' = \ell_{n+1}\end{bmatrix} \text{ implies } \ell' < \ell''\right] \text{ for all } \ell', \ell'\right] \text{ and } \cdots \text{ and } \left[\left[\begin{bmatrix}\ell' = \ell_1 \text{ and}\\ \ell'' = \ell_{n+m}\end{bmatrix} \text{ implies } \ell' < \ell''\right] \text{ for all } \ell', \ell'\right]$$
$$\text{and } \cdots \text{ and}$$
$$\left[\left[\begin{bmatrix}\ell' = \ell_n \text{ and}\\ \ell'' = \ell_{n+1}\end{bmatrix} \text{ implies } \ell' < \ell''\right] \text{ for all } \ell', \ell'\right] \text{ and } \cdots \text{ and } \left[\left[\begin{bmatrix}\ell' = \ell_n \text{ and}\\ \ell'' = \ell_{n+m}\end{bmatrix} \text{ implies } \ell' < \ell''\right] \text{ for all } \ell', \ell'\right]$$

Then, by applying substitution, conclude:

$$\left[\left[\begin{bmatrix}\ell' = \ell_1 \text{ and}\\ \ell'' = \ell_{n+1}\end{bmatrix} \text{ implies } \ell_1 < \ell_{n+1}\right] \text{ for all } \ell', \ell'\right] \text{ and } \cdots \text{ and } \left[\left[\begin{bmatrix}\ell' = \ell_1 \text{ and}\\ \ell'' = \ell_{n+m}\end{bmatrix} \text{ implies } \ell_1 < \ell_{n+m}\right] \text{ for all } \ell', \ell'\right]$$
$$\text{and } \cdots \text{ and}$$
$$\left[\left[\begin{bmatrix}\ell' = \ell_n \text{ and}\\ \ell'' = \ell_{n+1}\end{bmatrix} \text{ implies } \ell_n < \ell_{n+1}\right] \text{ for all } \ell', \ell'\right] \text{ and } \cdots \text{ and } \left[\left[\begin{bmatrix}\ell' = \ell_n \text{ and}\\ \ell'' = \ell_{n+m}\end{bmatrix} \text{ implies } \ell_n < \ell_{n+m}\right] \text{ for all } \ell', \ell'\right]$$

Then, by applying standard inference rules, conclude:

$$\left[\ell_1 < \ell_{n+1} \text{ and } \cdots \text{ and } \ell_1 < \ell_{n+m}\right] \text{ and } \cdots \text{ and } \left[\ell_n < \ell_{n+1} \text{ and } \cdots \text{ and } \ell_n < \ell_{n+m}\right]$$

(Y3) Suppose:

$$\begin{bmatrix}\left[< \text{ is a linear extension of } \prec_L\right]\\ \text{and } \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\}\\ \text{and } L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_{n+1}, \ldots, \ell_{n+m}\}\\ \text{and } \ell_1 < \cdots < \ell_n\\ \text{and } \ell_{n+1} < \cdots < \ell_{n+m}\end{bmatrix} \text{ for some } <, \ell_1, \ldots, \ell_{n+m}, n, m$$

Then, by applying (Y2), conclude:

$$\ell_1 < \cdots < \ell_n$$
$$\text{and } \ell_{n+1} < \cdots < \ell_{n+m}$$
$$\text{and } \left[\ell_1 < \ell_{n+1} \text{ and } \cdots \text{ and } \ell_1 < \ell_{n+m}\right] \text{ and } \cdots \text{ and } \left[\ell_n < \ell_{n+1} \text{ and } \cdots \text{ and } \ell_n < \ell_{n+m}\right]$$

Then, by applying order theory, conclude:

$$\begin{bmatrix}\left[\ell_1 < \ell_2 \text{ and } \ell_1 < \ell_3 \text{ and } \cdots \text{ and } \ell_1 < \ell_{n-1} \text{ and } \ell_1 < \ell_n\right]\\ \qquad \text{and } \left[\ell_2 < \ell_3 \text{ and } \cdots \text{ and } \ell_2 < \ell_{n-1} \text{ and } \ell_2 < \ell_n\right]\\ \qquad\qquad \vdots\\ \qquad\qquad \text{and } \ell_{n-1} < \ell_n\end{bmatrix}$$
$$\text{and } \begin{bmatrix}\left[\ell_{n+1} < \ell_{n+2} \text{ and } \ell_{n+1} < \ell_{n+3} \text{ and } \cdots \text{ and } \ell_{n+1} < \ell_{n+m-1} \text{ and } \ell_1 < \ell_{n+m}\right]\\ \qquad \text{and } \left[\ell_{n+2} < \ell_{n+3} \text{ and } \cdots \text{ and } \ell_{n+2} < \ell_{n+m-1} \text{ and } \ell_{n+2} < \ell_{n+m}\right]\\ \qquad\qquad \vdots\\ \qquad\qquad \text{and } \ell_{n+m-1} < \ell_{n+m}\end{bmatrix}$$
$$\text{and } \left[\ell_1 < \ell_{n+1} \text{ and } \cdots \text{ and } \ell_1 < \ell_{n+m}\right] \text{ and } \cdots \text{ and } \left[\ell_n < \ell_{n+1} \text{ and } \cdots \text{ and } \ell_n < \ell_{n+m}\right]$$

Then, by applying order theory, conclude $\ell_1 < \cdots < \ell_n < \ell_{n+1} < \cdots < \ell_{n+m}$.

(Y4) Suppose:

$$\ell \in \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} \text{ for some } \ell$$

Then, by applying set theory, conclude $\big[\ell = \mathtt{d}_x \approx t \text{ for some } x\,,\, t\big]$.

(Y5) Suppose:

$$\big[\{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_1\,,\,\ldots\,,\,\ell_n\}\big] \text{ for some } \ell_1\,,\,\ldots\,,\,\ell_n\,,\,n$$

Then, by introducing (Y4), conclude:

$$\begin{array}{l}\{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_1\,,\,\ldots\,,\,\ell_n\} \\ \text{and } \big[\big[\ell \in \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} \text{ implies } \big[\ell = \mathtt{d}_x \approx t \text{ for some } x\,,\,t\big]\big] \text{ for all } \ell\big]\end{array}$$

Then, by applying substitution, conclude:

$$\big[\ell \in \{\ell_1\,,\,\ldots\,,\,\ell_n\} \text{ implies } \big[\ell = \mathtt{d}_x \approx t \text{ for some } x\,,\,t\big]\big] \text{ for all } \ell$$

Then, by applying set theory, conclude:

$$\big[\big[\ell = \ell_1 \text{ or } \cdots \text{ or } \ell = \ell_n\big] \text{ implies } \big[\ell = \mathtt{d}_x \approx t \text{ for some } x\,,\,t\big]\big] \text{ for all } \ell$$

Then, by applying standard inference rules, conclude:

$$\left[\begin{array}{c}\big[\ell = \ell_1 \text{ implies } \big[\ell = \mathtt{d}_x \approx t \text{ for some } x\,,\,t\big]\big] \\ \text{and } \cdots \text{ and} \\ \big[\ell = \ell_n \text{ implies } \big[\ell = \mathtt{d}_x \approx t \text{ for some } x\,,\,t\big]\big]\end{array}\right] \text{ for all } \ell$$

Then, by applying standard inference rules, conclude:

$$\begin{array}{c}\big[\big[\ell = \ell_1 \text{ implies } \big[\ell = \mathtt{d}_x \approx t \text{ for some } x\,,\,t\big]\big] \text{ for all } \ell\big] \\ \text{and } \cdots \text{ and} \\ \big[\big[\ell = \ell_n \text{ implies } \big[\ell = \mathtt{d}_x \approx t \text{ for some } x\,,\,t\big]\big] \text{ for all } \ell\big]\end{array}$$

Then, by applying substitution, conclude:

$$\begin{array}{c}\big[\big[\ell = \ell_1 \text{ implies } \big[\ell_1 = \mathtt{d}_x \approx t \text{ for some } x\,,\,t\big]\big] \text{ for all } \ell\big] \\ \text{and } \cdots \text{ and} \\ \big[\big[\ell = \ell_n \text{ implies } \big[\ell_n = \mathtt{d}_x \approx t \text{ for some } x\,,\,t\big]\big] \text{ for all } \ell\big]\end{array}$$

Then, by applying standard inference rules, conclude:

$$\big[\big[\ell_1 = \mathtt{d}_x \approx t \text{ for some } x\,,\,t\big] \text{ and } \cdots \text{ and } \big[\ell_n = \mathtt{d}_x \approx t \text{ for some } x\,,\,t\big]\big]$$

Then, by applying standard inference rules, conclude:

$$\big[\big[\ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ for some } x_1\,,\,t_1\big] \text{ and } \cdots \text{ and } \big[\ell_n = \mathtt{d}_{x_n} \approx t_n \text{ for some } x_n\,,\,t_n\big]\big]$$

Then, by applying standard inference rules, conclude

$$\big[\ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\big] \text{ for some } x_1\,,\,\ldots\,,\,x_n\,,\,t_1\,,\,\ldots\,,\,t_n$$

(Y6) Suppose:

$$\mathtt{d}_y \in \bigcup\{\mathsf{Free}(\ell_{j'}) \mid 1 \leq j' \leq n+m\} \text{ for some } y\,,\,\ell_1\,,\,\ldots\,,\,\ell_{n+m}\,,\,n\,,\,m$$

Then, by applying set theory, conclude:

$$\big[\mathtt{d}_y \in \mathsf{Free}(\ell_j) \text{ and } 1 \leq j \leq n+m\big] \text{ for some } j$$

Then, by applying set theory, conclude $\big[\mathtt{d}_y \in \mathsf{Free}(\ell_j) \text{ and } \ell_j \in \{\ell_{j'} \mid 1 \leq j' \leq n+m\}\big]$.

(Y7) Suppose:

$$\ell \in L \text{ for some } \ell$$

Then, by applying Proposition 5:1, conclude:

$$L' \blacktriangleleft_L^{\text{arb}} \ell \text{ for some } L'$$

Then, by applying Definition 13 of $\blacktriangleleft_L$, conclude:

$$\begin{array}{l}
\mathsf{Free}(\ell) = \emptyset \\
\textbf{or } \left[\left[\mathtt{d}_x \approx t \in L \textbf{ and } \mathsf{Free}(t) = \emptyset \textbf{ and } \ell = \mathtt{d}_x \approx t\right] \textbf{ for some } x, t\right] \\
\textbf{or } \left[\begin{array}{c}\left[\mathsf{Free}(\ell) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \textbf{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\right] \\ \textbf{for some } x'_1, \ldots, x'_k, t'_1, \ldots, t'_k, k\end{array}\right] \\
\textbf{or } \left[\begin{array}{c}\left[\begin{array}{c}\mathtt{d}_x \approx t \in L \textbf{ and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \textbf{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L \\ \textbf{and } \ell = \mathtt{d}_x \approx t\end{array}\right] \\ \textbf{for some } x, x'_1, \ldots, x'_k, t, t'_1, \ldots, t'_k, k\end{array}\right]
\end{array}$$

(Y8) Suppose:

$$\left[\begin{array}{c}L = \{\ell_j \mid 1 \leq j \leq n+m\} \textbf{ and} \\ \mathtt{d}_y \in \bigcup\{\mathsf{Free}(\ell_{j'}) \mid 1 \leq j' \leq n+m\}\end{array}\right] \text{ for some } \ell_1, \ldots, \ell_{n+m}, n, m, y$$

Then, by applying (Y6), conclude:

$$L = \{\ell_j \mid 1 \leq j \leq n+m\} \textbf{ and } \left[\left[\mathtt{d}_y \in \mathsf{Free}(\ell_j) \textbf{ and } \ell_j \in \{\ell_{j'} \mid 1 \leq j' \leq n+m\}\right] \textbf{ for some } j\right]$$

Then, by applying standard inference rules, conclude:

$$\left[L = \{\ell_j \mid 1 \leq j \leq n+m\} \textbf{ and } \mathtt{d}_y \in \mathsf{Free}(\ell_j) \textbf{ and } \ell_j \in \{\ell_{j'} \mid 1 \leq j' \leq n+m\}\right] \text{ for some } j$$

Then, by applying substitution, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(\ell_j) \textbf{ and } \ell_j \in L\right]$. Then, by applying (Y7), conclude:

$$\begin{array}{l}
\mathtt{d}_y \in \mathsf{Free}(\ell_j) \\
\textbf{and } \left[\begin{array}{l}
\mathsf{Free}(\ell_j) = \emptyset \\
\textbf{or } \left[\left[\mathtt{d}_x \approx t \in L \textbf{ and } \mathsf{Free}(t) = \emptyset \textbf{ and } \ell_j = \mathtt{d}_x \approx t\right] \textbf{ for some } x, t\right] \\
\textbf{or } \left[\begin{array}{c}\left[\mathsf{Free}(\ell_j) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \textbf{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\right] \\ \textbf{for some } x'_1, \ldots, x'_k, t'_1, \ldots, t'_k, k\end{array}\right] \\
\textbf{or } \left[\begin{array}{c}\left[\begin{array}{c}\mathtt{d}_x \approx t \in L \textbf{ and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \textbf{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L \\ \textbf{and } \ell_j = \mathtt{d}_x \approx t\end{array}\right] \\ \textbf{for some } x, x'_1, \ldots, x'_k, t, t'_1, \ldots, t'_k, k\end{array}\right]
\end{array}\right]
\end{array}$$

(Y9) Suppose:

$$\left[\mathtt{d}_y \in \mathsf{Free}(\ell) \textbf{ and } \mathsf{Free}(\ell) = \emptyset\right] \text{ for some } y, \ell$$

Then, by applying substitution, conclude $\mathtt{d}_y \in \emptyset$. Then, by applying set theory, conclude **false**.

(Y0) Suppose:

$$\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_x) \text{ for some } x, y$$

Then, by applying Definition 7 of Free, conclude $\mathtt{d}_y \in \{\mathtt{d}_x\}$. Then, by applying set theory, conclude $\mathtt{d}_y = \mathtt{d}_x$.

(X1) Suppose:

$$\left[\mathtt{d}_y \in \mathsf{Free}(\ell) \textbf{ and } \mathsf{Free}(t) = \emptyset \textbf{ and } \ell = \mathtt{d}_x \approx t\right] \text{ for some } x, y, t$$

Then, by applying substitution, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_x \approx t) \textbf{ and } \mathsf{Free}(t) = \emptyset\right]$. Then, by applying Definition 8 of Free, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_x) \cup \mathsf{Free}(t) \textbf{ and } \mathsf{Free}(t) = \emptyset\right]$. Then, by applying substitution, conclude $\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_x) \cup \emptyset$. Then, by applying set theory, conclude $\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_x)$. Then, by applying (Y0), conclude $\mathtt{d}_y = \mathtt{d}_x$.

(X2) Suppose:

$$\big[\mathtt{d}_x \approx t \in L \text{ and } \ell = \mathtt{d}_x \approx t\big] \text{ for some } x\,,\, t\,,\, \ell$$

Then, by applying substitution, conclude $\big[\ell \in L \text{ and } \ell = \mathtt{d}_x \approx t\big]$. Then, by applying set theory, conclude $\ell \in \{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\}$.

(X3) Suppose:

$$\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1\,,\,\ldots\,,\,\ell_n\}\\ \text{and } \mathtt{d}_x \approx t \in L \text{ and } \ell = \mathtt{d}_x \approx t\end{array}\right] \text{ for some } \ell\,,\,\ell_1\,,\,\ldots\,,\,\ell_n\,,\,n\,,\,x\,,\,t$$

Then, by applying (X2), conclude:

$$\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1\,,\,\ldots\,,\,\ell_n\} \text{ and } \ell \in \{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\}$$

Then, by applying substitution, conclude $\ell \in \{\ell_1\,,\,\ldots\,,\,\ell_n\}$. Then, by applying set theory, conclude $\big[\ell = \ell_1 \text{ or } \cdots \text{ or } \ell = \ell_n\big]$.

(X4) Suppose:

$$\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1\,,\,\ldots\,,\,\ell_n\}\\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\\ \text{and } \mathtt{d}_x \approx t \in L \text{ and } \ell = \mathtt{d}_x \approx t\end{array}\right]$$

$$\text{for some } \ell\,,\,\ell_1\,,\,\ldots\,,\,\ell_n\,,\,n\,,\,x\,,\,x_1\,,\,\ldots\,,\,x_n\,,\,t\,,\,t_1\,,\,\ldots\,,\,t_n$$

Then, by applying (X3), conclude:

$$\ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \text{ and } \big[\ell = \ell_1 \text{ or } \cdots \text{ or } \ell = \ell_n\big]$$

Then, by applying standard inference rules, conclude:

$$\big[\ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \ell = \ell_1\big] \text{ or } \cdots \text{ or } \big[\ell_n = \mathtt{d}_{x_n} \approx t_n \text{ and } \ell = \ell_n\big]$$

Then, by applying substitution, conclude $\big[\ell = \mathtt{d}_{x_1} \approx t_1 \text{ or } \cdots \text{ or } \ell = \mathtt{d}_{x_n} \approx t_n\big]$.

(X5) Suppose:

$$\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1\,,\,\ldots\,,\,\ell_n\}\\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\\ \text{and } \mathtt{d}_x \approx t \in L \text{ and } \ell = \mathtt{d}_x \approx t \text{ and } \mathtt{d}_y = \mathtt{d}_x\end{array}\right]$$

$$\text{for some } \ell_1\,,\,\ldots\,,\,\ell_n\,,\,n\,,\,x\,,\,x_1\,,\,\ldots\,,\,x_n\,,\,t\,,\,t_1\,,\,\ldots\,,\,t_n$$

Then, by applying (X4), conclude:

$$\ell = \mathtt{d}_x \approx t \text{ and } \mathtt{d}_y = \mathtt{d}_x \text{ and } \big[\ell = \mathtt{d}_{x_1} \approx t_1 \text{ or } \cdots \text{ or } \ell = \mathtt{d}_{x_n} \approx t_n\big]$$

Then, by applying substitution, conclude $\big[\ell = \mathtt{d}_y \approx t \text{ and } \big[\ell = \mathtt{d}_{x_1} \approx t_1 \text{ or } \cdots \text{ or } \ell = \mathtt{d}_{x_n} \approx t_n\big]\big]$. Then, by applying substitution, conclude $\big[\mathtt{d}_y \approx t = \mathtt{d}_{x_1} \approx t_1 \text{ or } \cdots \text{ or } \mathtt{d}_y \approx t = \mathtt{d}_{x_n} \approx t_n\big]$. Then, by applying standard inference rules, conclude $\big[\mathtt{d}_y = \mathtt{d}_{x_1} \text{ or } \cdots \text{ or } \mathtt{d}_y = \mathtt{d}_{x_n}\big]$. Then, by applying set theory, conclude $\mathtt{d}_y \in \{\mathtt{d}_{x_1}\,,\,\ldots\,,\,\mathtt{d}_{x_n}\}$.

(X6) Suppose:

$$\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1\,,\,\ldots\,,\,\ell_n\}\\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\\ \text{and } \mathtt{d}_y \in \mathsf{Free}(\ell) \text{ and } \mathtt{d}_x \approx t \in L \text{ and } \mathsf{Free}(t) = \emptyset \text{ and } \ell = \mathtt{d}_x \approx t\end{array}\right]$$

$$\text{for some } \ell\,,\,\ell_1\,,\,\ldots\,,\,\ell_n\,,\,n\,,\,x\,,\,x_1\,,\,\ldots\,,\,x_n\,,\,y\,,\,t\,,\,t_1\,,\,\ldots\,,\,t_n$$

Then, by applying (X1), conclude:

$$\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1\,,\,\ldots\,,\,\ell_n\}\\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\\ \text{and } \mathtt{d}_x \approx t \in L \text{ and } \ell = \mathtt{d}_x \approx t \text{ and } \mathtt{d}_y = \mathtt{d}_x\end{array}$$

Then, by applying (X5), conclude $\mathtt{d}_y \in \{\mathtt{d}_{x_1}\,,\,\ldots\,,\,\mathtt{d}_{x_n}\}$.

(X7) Suppose:

$$\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1 , \ldots , \ell_n\} \\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \text{and } \mathtt{d}_{x'_1} \approx t'_1 , \ldots , \mathtt{d}_{x'_k} \approx t'_k \in L\end{array}\right]$$

$$\text{for some } \ell_1 , \ldots , \ell_n , n , k , x_1 , \ldots , x_n , x'_1 , \ldots , x'_k , t_1 , \ldots , t_n , t'_1 , \ldots , t'_k$$

Then, by applying (X4), conclude:

$$\begin{array}{c}\left[\mathtt{d}_{x'_1} \approx t'_1 = \mathtt{d}_{x_1} \approx t_1 \text{ or } \cdots \text{ or } \mathtt{d}_{x'_1} \approx t'_1 = \mathtt{d}_{x_n} \approx t_n\right] \\ \text{and } \cdots \text{ and} \\ \left[\mathtt{d}_{x'_k} \approx t'_k = \mathtt{d}_{x_1} \approx t_1 \text{ or } \cdots \text{ or } \mathtt{d}_{x'_k} \approx t'_k = \mathtt{d}_{x_n} \approx t_n\right]\end{array}$$

Then, by applying standard inference rules, conclude:

$$\left[\mathtt{d}_{x'_1} = \mathtt{d}_{x_1} \text{ or } \cdots \text{ or } \mathtt{d}_{x'_1} = \mathtt{d}_{x_n}\right] \text{ and } \cdots \text{ and } \left[\mathtt{d}_{x'_k} = \mathtt{d}_{x_1} \text{ or } \cdots \text{ or } \mathtt{d}_{x'_k} = \mathtt{d}_{x_n}\right]$$

Then, by applying set theory, conclude $\left[\mathtt{d}_{x'_1} \in \{\mathtt{d}_{x_1} , \ldots , \mathtt{d}_{x_n}\} \text{ and } \cdots \text{ and } \mathtt{d}_{x'_k} \in \{\mathtt{d}_{x_1} , \ldots , \mathtt{d}_{x_n}\}\right]$.

(X8) Suppose:

$$\left[\mathtt{d}_y \in \mathsf{Free}(\ell) \text{ and } \mathsf{Free}(\ell) = \{\mathtt{d}_{x'_1} , \ldots , \mathtt{d}_{x'_k}\}\right] \text{ for some } x'_1 , \ldots , x'_k , y , \ell , k$$

Then, by applying substitution, conclude $\mathtt{d}_y \in \{\mathtt{d}_{x'_1} , \ldots , \mathtt{d}_{x'_k}\}$. Then, by applying standard inference rules, conclude $\left[\mathtt{d}_y = \mathtt{d}_{x'_1} \text{ or } \cdots \text{ or } \mathtt{d}_y = \mathtt{d}_{x'_k}\right]$.

(X9) Suppose:

$$\left[\mathtt{d}_y \in \mathsf{Free}(t) \text{ and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1} , \ldots , \mathtt{d}_{x'_k}\}\right] \text{ for some } x'_1 , \ldots , x'_k , y , t , k$$

Then, by a reduction similar to (X8), conclude $\left[\mathtt{d}_y = \mathtt{d}_{x'_1} \text{ or } \cdots \text{ or } \mathtt{d}_y = \mathtt{d}_{x'_k}\right]$.

(X0) Suppose:

$$\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1 , \ldots , \ell_n\} \\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \text{and } \mathtt{d}_y \in \mathsf{Free}(\ell) \text{ and } \mathsf{Free}(\ell) = \{\mathtt{d}_{x'_1} , \ldots , \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1 , \ldots , \mathtt{d}_{x'_k} \approx t'_k \in L\end{array}\right]$$

$$\text{for some } \ell , \ell_1 , \ldots , \ell_n , n , k , x_1 , \ldots , x_n , x'_1 , \ldots , x'_k , y , t_1 , \ldots , t_n , t'_1 , \ldots , t'_k$$

Then, by applying (X7), conclude:

$$\begin{array}{c}\mathtt{d}_y \in \mathsf{Free}(\ell) \text{ and } \mathsf{Free}(\ell) = \{\mathtt{d}_{x'_1} , \ldots , \mathtt{d}_{x'_k}\} \text{ and} \\ \mathtt{d}_{x'_1} \in \{\mathtt{d}_{x_1} , \ldots , \mathtt{d}_{x_n}\} \text{ and } \cdots \text{ and } \mathtt{d}_{x'_k} \in \{\mathtt{d}_{x_1} , \ldots , \mathtt{d}_{x_n}\}\end{array}$$

Then, by applying (X8), conclude:

$$\mathtt{d}_{x'_1} \in \{\mathtt{d}_{x_1} , \ldots , \mathtt{d}_{x_n}\} \text{ and } \cdots \text{ and } \mathtt{d}_{x'_k} \in \{\mathtt{d}_{x_1} , \ldots , \mathtt{d}_{x_n}\} \text{ and } \left[\mathtt{d}_y = \mathtt{d}_{x'_1} \text{ or } \cdots \text{ or } \mathtt{d}_y = \mathtt{d}_{x'_k}\right]$$

Then, by applying standard inference rules, conclude:

$$\left[\mathtt{d}_{x'_1} \in \{\mathtt{d}_{x_1} , \ldots , \mathtt{d}_{x_n}\} \text{ and } \mathtt{d}_y = \mathtt{d}_{x'_1}\right] \text{ or } \cdots \text{ or } \left[\mathtt{d}_{x'_k} \in \{\mathtt{d}_{x_1} , \ldots , \mathtt{d}_{x_n}\} \text{ and } \mathtt{d}_y = \mathtt{d}_{x'_k}\right]$$

Then, by applying substitution, conclude $\left[\mathtt{d}_y \in \{\mathtt{d}_{x_1} , \ldots , \mathtt{d}_{x_n}\} \text{ and } \cdots \text{ and } \mathtt{d}_y \in \{\mathtt{d}_{x_1} , \ldots , \mathtt{d}_{x_n}\}\right]$. Then, by applying standard inference rules, conclude $\mathtt{d}_y \in \{\mathtt{d}_{x_1} , \ldots , \mathtt{d}_{x_n}\}$.

(W1) Suppose:

$$\begin{bmatrix}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \text{and } \mathtt{d}_y \in \mathsf{Free}(t) \text{ and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\end{bmatrix}$$
$$\text{for some } \ell_1, \ldots, \ell_n, n, k, x_1, \ldots, x_n, x'_1, \ldots, x'_k, y, t, t_1, \ldots, t_n, t'_1, \ldots, t'_k$$

Then, by a reduction similar to (X0), conclude $\mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}$.

(W2) Suppose:

$$\left[\mathtt{d}_y \in \mathsf{Free}(\ell) \text{ and } \ell = \mathtt{d}_x \approx t\right] \text{ for some } x, y, \ell, t$$

Then, by applying substitution, conclude $\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_x \approx t)$. Then, by applying Definition 8 of Free, conclude $\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_x) \cup \mathsf{Free}(t)$. Then, by applying set theory, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_x) \text{ or } \mathtt{d}_y \in \mathsf{Free}(t)\right]$. Then, by applying (Y0), conclude $\left[\mathtt{d}_y = \mathtt{d}_x \text{ or } \mathtt{d}_y \in \mathsf{Free}(t)\right]$.

(W3) Suppose:

$$\begin{bmatrix}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \text{and } \mathtt{d}_y \in \mathsf{Free}(\ell) \\ \text{and } \mathtt{d}_x \approx t \in L \text{ and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L \text{ and } \ell = \mathtt{d}_x \approx t\end{bmatrix}$$
$$\text{for some } \ell, \ell_1, \ldots, \ell_n, n, k, x, x_1, \ldots, x_n, x'_1, \ldots, x'_k, y, t, t_1, \ldots, t_n, t'_1, \ldots, t'_k$$

Then, by applying (W2), conclude:

$$\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \text{and } \mathtt{d}_x \approx t \in L \text{ and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L \text{ and } \ell = \mathtt{d}_x \approx t \\ \text{and } \left[\mathtt{d}_y = \mathtt{d}_x \text{ or } \mathtt{d}_y \in \mathsf{Free}(t)\right]\end{array}$$

Then, by applying standard inference rules, conclude:

$$\begin{bmatrix}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \text{and } \mathtt{d}_x \approx t \in L \text{ and } \ell = \mathtt{d}_x \approx t \text{ and } \mathtt{d}_y = \mathtt{d}_x\end{bmatrix}$$
$$\text{or } \begin{bmatrix}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \text{and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L \\ \text{and } \mathtt{d}_y \in \mathsf{Free}(t)\end{bmatrix}$$

Then, by applying (X5), conclude:

$$\mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} \text{ or } \begin{bmatrix}\{\ell' \mid \ell' = \mathtt{d}_{x'} \approx t' \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \text{and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L \\ \text{and } \mathtt{d}_y \in \mathsf{Free}(t)\end{bmatrix}$$

Then, by applying (W1), conclude $\left[\mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} \text{ or } \mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}\right]$. Then, by applying standard inference rules, conclude $\mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}$.

(W4) Suppose:

$$\begin{bmatrix}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } L = \{\ell_j \mid 1 \leq j \leq n + m\} \\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \text{and } \mathtt{d}_y \in \bigcup\{\mathsf{Free}(\ell_{j'}) \mid 1 \leq j' \leq n + m\}\end{bmatrix}$$
$$\text{for some } \ell_1, \ldots, \ell_{n+m}, n, m, x_1, \ldots, x_n, y, t_1, \ldots, t_n$$

Then, by applying (Y8), conclude:

$$
\begin{array}{l}
\{\ell' \mid \ell' = \mathtt{d}_x \approx t \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\
\text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\
\text{and} \left[\begin{array}{l}
\left[\begin{array}{c}
\mathtt{d}_y \in \mathsf{Free}(\ell_j) \text{ and} \\
\left[\begin{array}{l}
\mathsf{Free}(\ell_j) = \emptyset \\
\text{or } \left[\left[\mathtt{d}_x \approx t \in L \text{ and } \mathsf{Free}(t) = \emptyset \text{ and } \ell_j = \mathtt{d}_x \approx t\right] \text{ for some } x, t\right] \\
\text{or } \left[\begin{array}{c}\left[\mathsf{Free}(\ell_j) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\right] \\ \text{for some } x'_1, \ldots, x'_k, t'_1, \ldots, t'_k, k\end{array}\right] \\
\text{or } \left[\begin{array}{c}\left[\mathtt{d}_x \approx t \in L \text{ and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\right] \\ \text{and } \ell_j = \mathtt{d}_x \approx t \\ \text{for some } x, x'_1, \ldots, x'_k, t, t'_1, \ldots, t'_k, k\end{array}\right]
\end{array}\right]
\end{array}\right] \\
\text{for some } j
\end{array}\right]
\end{array}
$$

Then, by applying standard inference rules, conclude:

$$
\begin{array}{l}
\left[\begin{array}{l}
\{\ell' \mid \ell' = \mathtt{d}_x \approx t \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\
\text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\
\text{and } \mathtt{d}_y \in \mathsf{Free}(\ell_j) \\
\text{and} \left[\begin{array}{l}
\mathsf{Free}(\ell_j) = \emptyset \\
\text{or } \left[\left[\mathtt{d}_x \approx t \in L \text{ and } \mathsf{Free}(t) = \emptyset \text{ and } \ell_j = \mathtt{d}_x \approx t\right] \text{ for some } x, t\right] \\
\text{or } \left[\begin{array}{c}\left[\mathsf{Free}(\ell_j) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\right] \\ \text{for some } x'_1, \ldots, x'_k, t'_1, \ldots, t'_k, k\end{array}\right] \\
\text{or } \left[\begin{array}{c}\left[\mathtt{d}_x \approx t \in L \text{ and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\right] \\ \text{and } \ell_j = \mathtt{d}_x \approx t \\ \text{for some } x, x'_1, \ldots, x'_k, t, t'_1, \ldots, t'_k, k\end{array}\right]
\end{array}\right]
\end{array}\right] \\
\text{for some } j
\end{array}
$$

Then, by applying standard inference rules, conclude:

$$
\begin{array}{l}
\left[\mathtt{d}_y \in \mathsf{Free}(\ell_j) \text{ and } \mathsf{Free}(\ell_j) = \emptyset\right] \\
\text{or} \left[\begin{array}{l}
\{\ell' \mid \ell' = \mathtt{d}_x \approx t \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\
\text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\
\text{and } \mathtt{d}_y \in \mathsf{Free}(\ell_j) \\
\text{and } \left[\left[\mathtt{d}_x \approx t \in L \text{ and } \mathsf{Free}(t) = \emptyset \text{ and } \ell_j = \mathtt{d}_x \approx t\right] \text{ for some } x, t\right]
\end{array}\right] \\
\text{or} \left[\begin{array}{l}
\{\ell' \mid \ell' = \mathtt{d}_x \approx t \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\
\text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\
\text{and } \mathtt{d}_y \in \mathsf{Free}(\ell_j) \\
\text{and} \left[\begin{array}{c}\left[\mathsf{Free}(\ell_j) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\right] \\ \text{for some } x'_1, \ldots, x'_k, t'_1, \ldots, t'_k, k\end{array}\right]
\end{array}\right] \\
\text{or} \left[\begin{array}{l}
\{\ell' \mid \ell' = \mathtt{d}_x \approx t \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\
\text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\
\text{and } \mathtt{d}_y \in \mathsf{Free}(\ell_j) \\
\text{and} \left[\begin{array}{c}\left[\mathtt{d}_x \approx t \in L \text{ and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \text{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\right] \\ \text{and } \ell_j = \mathtt{d}_x \approx t \\ \text{for some } x, x'_1, \ldots, x'_k, t, t'_1, \ldots, t'_k, k\end{array}\right]
\end{array}\right]
\end{array}
$$

Then, by applying standard inference rules, conclude:

$$\begin{array}{l}
\left[\mathtt{d}_y \in \mathsf{Free}(\ell_j) \ \textbf{ and } \ \mathsf{Free}(\ell_j) = \emptyset\right] \\
\textbf{or } \left[\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \ \textbf{ and } \ \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \textbf{and } \ \ell_1 = \mathtt{d}_{x_1} \approx t_1 \ \textbf{ and } \ \cdots \ \textbf{ and } \ \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \textbf{and } \ \mathtt{d}_y \in \mathsf{Free}(\ell_j) \\ \textbf{and } \ \mathtt{d}_x \approx t \in L \ \textbf{ and } \ \mathsf{Free}(t) = \emptyset \ \textbf{ and } \ \ell_j = \mathtt{d}_x \approx t\end{array}\right] \ \textbf{for some } \ x, t\right] \\
\textbf{or } \left[\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \ \textbf{ and } \ \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \textbf{and } \ \ell_1 = \mathtt{d}_{x_1} \approx t_1 \ \textbf{ and } \ \cdots \ \textbf{ and } \ \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \textbf{and } \ \mathtt{d}_y \in \mathsf{Free}(\ell_j) \ \textbf{ and } \ \mathsf{Free}(\ell_j) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \ \textbf{ and } \ \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\end{array}\right] \right. \\
\qquad\qquad \left. \textbf{for some } \ x'_1, \ldots, x'_k, t'_1, \ldots, t'_k, k\right] \\
\textbf{or } \left[\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \ \textbf{ and } \ \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \textbf{and } \ \ell_1 = \mathtt{d}_{x_1} \approx t_1 \ \textbf{ and } \ \cdots \ \textbf{ and } \ \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \textbf{and } \ \mathtt{d}_y \in \mathsf{Free}(\ell_j) \\ \textbf{and } \ \mathtt{d}_x \approx t \in L \ \textbf{ and } \ \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \ \textbf{ and } \ \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L \\ \textbf{and } \ \ell_j = \mathtt{d}_x \approx t\end{array}\right] \right. \\
\qquad\qquad \left. \textbf{for some } \ x, x'_1, \ldots, x'_k, t, t'_1, \ldots, t'_k, k\right]
\end{array}$$

Then, by applying (Y9), conclude:

$$\begin{array}{l}
\textbf{false} \\
\textbf{or } \left[\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \ \textbf{ and } \ \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \textbf{and } \ \ell_1 = \mathtt{d}_{x_1} \approx t_1 \ \textbf{ and } \ \cdots \ \textbf{ and } \ \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \textbf{and } \ \mathtt{d}_y \in \mathsf{Free}(\ell_j) \\ \textbf{and } \ \mathtt{d}_x \approx t \in L \ \textbf{ and } \ \mathsf{Free}(t) = \emptyset \ \textbf{ and } \ \ell_j = \mathtt{d}_x \approx t\end{array}\right] \ \textbf{for some } \ x, t\right] \\
\textbf{or } \left[\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \ \textbf{ and } \ \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \textbf{and } \ \ell_1 = \mathtt{d}_{x_1} \approx t_1 \ \textbf{ and } \ \cdots \ \textbf{ and } \ \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \textbf{and } \ \mathtt{d}_y \in \mathsf{Free}(\ell_j) \ \textbf{ and } \ \mathsf{Free}(\ell_j) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \ \textbf{ and } \ \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\end{array}\right] \right. \\
\qquad\qquad \left. \textbf{for some } \ x'_1, \ldots, x'_k, t'_1, \ldots, t'_k, k\right] \\
\textbf{or } \left[\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \ \textbf{ and } \ \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \textbf{and } \ \ell_1 = \mathtt{d}_{x_1} \approx t_1 \ \textbf{ and } \ \cdots \ \textbf{ and } \ \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \textbf{and } \ \mathtt{d}_y \in \mathsf{Free}(\ell_j) \\ \textbf{and } \ \mathtt{d}_x \approx t \in L \ \textbf{ and } \ \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \ \textbf{ and } \ \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L \\ \textbf{and } \ \ell_j = \mathtt{d}_x \approx t\end{array}\right] \right. \\
\qquad\qquad \left. \textbf{for some } \ x, x'_1, \ldots, x'_k, t, t'_1, \ldots, t'_k, k\right]
\end{array}$$

Then, by applying (X6), conclude:

$$\begin{array}{l}
\textbf{false } \textbf{ or } \ \mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} \\
\textbf{or } \left[\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \ \textbf{ and } \ \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \textbf{and } \ \ell_1 = \mathtt{d}_{x_1} \approx t_1 \ \textbf{ and } \ \cdots \ \textbf{ and } \ \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \textbf{and } \ \mathtt{d}_y \in \mathsf{Free}(\ell_j) \ \textbf{ and } \ \mathsf{Free}(\ell_j) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \ \textbf{ and } \ \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\end{array}\right] \right. \\
\qquad\qquad \left. \textbf{for some } \ x'_1, \ldots, x'_k, t'_1, \ldots, t'_k, k\right] \\
\textbf{or } \left[\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \ \textbf{ and } \ \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \textbf{and } \ \ell_1 = \mathtt{d}_{x_1} \approx t_1 \ \textbf{ and } \ \cdots \ \textbf{ and } \ \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \textbf{and } \ \mathtt{d}_y \in \mathsf{Free}(\ell_j) \\ \textbf{and } \ \mathtt{d}_x \approx t \in L \ \textbf{ and } \ \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \ \textbf{ and } \ \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L \\ \textbf{and } \ \ell_j = \mathtt{d}_x \approx t\end{array}\right] \right. \\
\qquad\qquad \left. \textbf{for some } \ x, x'_1, \ldots, x'_k, t, t'_1, \ldots, t'_k, k\right]
\end{array}$$

Then, by applying (X0), conclude:

$$\begin{aligned}&\textbf{false or } \mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} \textbf{ or } \mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}\\ &\textbf{or } \left[\begin{array}{l}\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \textbf{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\}\\ \textbf{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \textbf{ and } \cdots \textbf{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\\ \textbf{and } \mathtt{d}_y \in \mathsf{Free}(\ell_j)\\ \textbf{and } \mathtt{d}_x \approx t \in L \textbf{ and } \mathsf{Free}(t) = \{\mathtt{d}_{x'_1}, \ldots, \mathtt{d}_{x'_k}\} \textbf{ and } \mathtt{d}_{x'_1} \approx t'_1, \ldots, \mathtt{d}_{x'_k} \approx t'_k \in L\\ \textbf{and } \ell_j = \mathtt{d}_x \approx t\end{array}\right]\\ \qquad\qquad \textbf{for some } x, x'_1, \ldots, x'_k, t, t'_1, \ldots, t'_k, k\end{array}\right]\end{aligned}$$

Then, by applying (W3), conclude:

$$\textbf{false or } \mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} \textbf{ or } \mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} \textbf{ or } \mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}$$

Then, by applying standard inference rules, conclude $\mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}$.

(W5) Suppose:

$$\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \textbf{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\}\\ \textbf{and } L = \{\ell_j \mid 1 \leq j \leq n+m\}\\ \textbf{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \textbf{ and } \cdots \textbf{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\end{array}\right]$$

$$\textbf{for some } \ell_1, \ldots, \ell_{n+m}, n, m, x_1, \ldots, x_n, t_1, \ldots, t_n$$

Then, by applying (W4), conclude:

$$\left[\mathtt{d}_y \in \bigcup\{\mathsf{Free}(\ell_{j'}) \mid 1 \leq j' \leq n+m\} \textbf{ implies } \mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}\right] \textbf{ for all } y$$

Then, by applying set theory, conclude $\bigcup\{\mathsf{Free}(\ell_{j'}) \mid 1 \leq j' \leq n+m\} \subseteq \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}$.

(W6) Suppose:

$$\mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} \textbf{ for some } x_1, \ldots, x_n, y, n, t_1, \ldots, t_n$$

Then, by applying set theory, conclude $\left[\mathtt{d}_y \in \{\mathtt{d}_{x_1}\} \textbf{ or } \cdots \textbf{ or } \mathtt{d}_y \in \{\mathtt{d}_{x_n}\}\right]$. Then, by applying Definition 7 of Free, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_{x_1}) \textbf{ or } \cdots \textbf{ or } \mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_{x_n})\right]$. Then, by applying set theory, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_{x_1}) \cup \mathsf{Free}(t_1) \textbf{ or } \cdots \textbf{ or } \mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_{x_n}) \cup \mathsf{Free}(t_n)\right]$. Then, by applying Definition 8 of Free, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_{x_1} \approx t_1) \textbf{ or } \cdots \textbf{ or } \mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_{x_n} \approx t_n)\right]$.

(W7) Suppose:

$$\left[\begin{array}{l}\ell_1 = \mathtt{d}_{x_1} \approx t_1 \textbf{ and } \cdots \textbf{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\\ \textbf{and } \mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}\end{array}\right]$$

$$\textbf{for some } \ell_1, \ldots, \ell_{n+m}, x_1, \ldots, x_n, y, t_1, \ldots, t_n, n, m$$

Then, by applying (W6), conclude:

$$\ell_1 = \mathtt{d}_{x_1} \approx t_1 \textbf{ and } \cdots \textbf{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \textbf{ and } \left[\mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_{x_1} \approx t_1) \textbf{ or } \cdots \textbf{ or } \mathtt{d}_y \in \mathsf{Free}(\mathtt{d}_{x_n} \approx t_n)\right]$$

Then, by applying substitution, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(\ell_1) \textbf{ or } \cdots \textbf{ or } \mathtt{d}_y \in \mathsf{Free}(\ell_n)\right]$. Then, by applying standard inference rules, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(\ell_1) \textbf{ or } \cdots \textbf{ or } \mathtt{d}_y \in \mathsf{Free}(\ell_{n+m})\right]$. Then, by applying set theory, conclude $\left[\mathtt{d}_y \in \mathsf{Free}(\ell_1) \cup \cdots \cup \mathsf{Free}(\ell_{n+m})\right]$. Then, by applying set theory, conclude:

$$\mathtt{d}_y \in \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\}$$

(W8) Suppose:

$$\left[\ell_1 = \mathtt{d}_{x_1} \approx t_1 \textbf{ and } \cdots \textbf{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\right] \textbf{ for some } \ell_1, \ldots, \ell_{n+m}, x_1, \ldots, x_n, t_1, \ldots, t_n, n, m$$

Then, by applying (W7), conclude:

$$\left[\mathtt{d}_y \in \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} \textbf{ implies } \mathtt{d}_y \in \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\}\right] \textbf{ for all } y$$

Then, by applying set theory, conclude $\{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} \subseteq \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\}$.

(W9) Suppose:

$$\left[\begin{array}{l}\{\ell' \mid \ell' = \mathtt{d}_x \approx t \text{ and } \ell' \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } L = \{\ell_j \mid 1 \leq j \leq n+m\} \\ \text{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\end{array}\right] \text{ for some } \ell_1, \ldots, \ell_{n+m}, n, m, x_1, \ldots, x_n, t_1, \ldots, t_n$$

Then, by applying (W5), conclude:

$$\begin{array}{l}\ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n \\ \text{and } \bigcup\{\mathsf{Free}(\ell_{j'}) \mid 1 \leq j' \leq n+m\} \subseteq \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\}\end{array}$$

Then, by applying (W8), conclude:

$$\bigcup\{\mathsf{Free}(\ell_{j'}) \mid 1 \leq j' \leq n+m\} \subseteq \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} \text{ and } \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} \subseteq \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\}$$

Then, by applying set theory, conclude $\{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} = \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\}$.

Now, prove the lemma by the following reduction. Recall from (Z1):

$$[< \text{ is a linear extension of } \prec_L] \text{ for some } <$$

Then, by applying (Z4), conclude:

$$[< \text{ is a linear extension of } \prec_L] \text{ and } \left[\left[\begin{array}{l}\{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_{n+1}, \ldots, \ell_{n+m}\} \\ \text{and } \ell_1 < \cdots < \ell_n \\ \text{and } \ell_{n+1} < \cdots < \ell_{n+m}\end{array}\right] \text{ for some } \ell_1, \ldots, \ell_{n+m}, n, m\right]$$

Then, by applying standard inference rules, conclude:

$$\left[\begin{array}{l}[< \text{ is a linear extension of } \prec_L] \\ \text{and } \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_{n+1}, \ldots, \ell_{n+m}\} \\ \text{and } \ell_1 < \cdots < \ell_n \\ \text{and } \ell_{n+1} < \cdots < \ell_{n+m}\end{array}\right] \text{ for some } \ell_1, \ldots, \ell_{n+m}, n, m$$

Then, by applying (Z6), conclude:

$$\begin{array}{l}[< \text{ is a linear extension of } \prec_L] \\ \text{and } \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } L \setminus \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_{n+1}, \ldots, \ell_{n+m}\} \\ \text{and } \ell_1 < \cdots < \ell_n \\ \text{and } \ell_{n+1} < \cdots < \ell_{n+m} \\ \text{and } L = \{\ell_j \mid 1 \leq j \leq n+m\}\end{array}$$

Then, by applying (Y3), conclude:

$$\begin{array}{l}[< \text{ is a linear extension of } \prec_L] \\ \text{and } \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } L = \{\ell_j \mid 1 \leq j \leq n+m\} \\ \text{and } \ell_1 < \cdots < \ell_n < \ell_{n+1} < \cdots < \ell_{n+m}\end{array}$$

Then, by applying (Y5), conclude:

$$\begin{array}{l}[< \text{ is a linear extension of } \prec_L] \\ \text{and } \{\ell \mid \ell = \mathtt{d}_x \approx t \text{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\} \\ \text{and } L = \{\ell_j \mid 1 \leq j \leq n+m\} \\ \text{and } \ell_1 < \cdots < \ell_n < \ell_{n+1} < \cdots < \ell_{n+m} \\ \text{and } [[\ell_1 = \mathtt{d}_{x_1} \approx t_1 \text{ and } \cdots \text{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n] \text{ for some } x_1, \ldots, x_n, t_1, \ldots, t_n]\end{array}$$

Then, by applying standard inference rules, conclude:

$$\left[\begin{array}{l}\big[< \text{ is a linear extension of } \prec_L\big]\\ \textbf{and } \{\ell \mid \ell = \mathtt{d}_x \approx t \textbf{ and } \ell \in L\} = \{\ell_1, \ldots, \ell_n\}\\ \textbf{and } L = \{\ell_j \mid 1 \leq j \leq n+m\}\\ \textbf{and } \ell_1 < \cdots < \ell_n < \ell_{n+1} < \cdots < \ell_{n+m}\\ \textbf{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \textbf{ and } \cdots \textbf{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\end{array}\right] \textbf{ for some } x_1, \ldots, x_n, t_1, \ldots, t_n$$

Then, by applying (W9), conclude:

$$\begin{array}{l}\big[< \text{ is a linear extension of } \prec_L\big]\\ \textbf{and } L = \{\ell_j \mid 1 \leq j \leq n+m\}\\ \textbf{and } \ell_1 < \cdots < \ell_n < \ell_{n+1} < \cdots < \ell_{n+m}\\ \textbf{and } \ell_1 = \mathtt{d}_{x_1} \approx t_1 \textbf{ and } \cdots \textbf{ and } \ell_n = \mathtt{d}_{x_n} \approx t_n\\ \textbf{and } \{\mathtt{d}_{x_1}, \ldots, \mathtt{d}_{x_n}\} = \bigcup\{\mathsf{Free}(\ell_j) \mid 1 \leq j \leq n+m\}\end{array}$$

(⟪ QED. ⟫)