Ordered Linear Logic Programming

Jeff Polakow and Frank Pfenning [1]
December 22, 1998
CMU-CS-98-183

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

We begin with a review of *intuitionistic non-commutative linear logic* (INCLL), a refinement of linear logic with an inherent notion of order proposed by the authors in prior work. We then develop a logic programming interpretation for INCLL in two steps: (1) we give a system of ordered uniform derivations which is sound and complete with respect to INCLL, and (2) we present a model of resource consumption which removes non-determinism from ordered resource allocation during search for uniform derivations. We also illustrate the expressive power of the resulting *ordered linear logic programming language* through some examples, including programs for merge sort, insertion sort, and natural language parsing.

[1] The authors can be reached at *jpolakow@cs.cmu.edu* and *fp@cs.cmu.edu*.

This work was sponsored NSF Grants CCR-9804014 and CCR-9619584.

Keywords: Intuitionistic Non-Commutative Linear Logic, Linear Logic Programming

1 Introduction

Linear logic [Gir87] can be considered a logic of *state*, since we can express many problems involving state in a much more natural and concise manner in linear logic than in traditional logics. It supplements the familiar notion of logical assumption with *linear hypotheses* which must be used exactly once in a derivation. Linear logic has found applications in functional programming, logic programming, logical frameworks, concurrency, and related areas. One of the critical properties of linear logic is that it extends traditional logic *conservatively*, that is, we strictly gain expressive power.

Given the many insights resulting from linear logic, we can ask if other computational phenomena besides state can be captured in a similarly logical manner. The one we consider here is the notion of *order*. There are many situations where computation is naturally subject to ordering constraints. Words in a sentence, for example, are ordered, which has given rise to the Lambek calculus [Lam58], a weak logic with an inherent notion of order for the analysis of natural language.

To fully explore uses of an order-aware logic in computer science, we would like it to conservatively extend both traditional and linear logic. Recently, following early explorations [Abr90, BG91], two such proposals have been made: one by Abrusci and Ruet rooted in a *non-commutative conjunction* [Rue97, AR98], and one by the authors based on *ordered hypotheses* [PP99]. Our system of natural deduction provides a foundation for applications in functional programming, which we investigated via the Curry-Howard isomorphism and properties of an ordered λ-calculus. Subsequently, we exhibited a related sequent calculus [PP98] which can serve as the basis for proof search procedures. We call our calculus *intuitionistic non-commutative linear logic* or *INCLL*.

In this paper we investigate logic programming with ordered hypotheses. We follow the paradigm that logic programming should be understood via an abstract notion of *uniform derivation* [MNPS91] which, in a slight abuse of terminology, we take to encompass *goal-directed search* and *focussed* use of hypotheses [And92]. Somewhat unexpectedly, the extension of these notions from the case of linear logic [HM94] is far from straightforward. The principal contributions of this paper are

1. a system of ordered uniform derivations which is sound and complete with respect to INCLL;
2. a model of resource consumption which removes non-determinism from resource allocation during the search for ordered uniform derivations while remaining sound and complete; and
3. example programs which illustrate the expressive power of the resulting ordered logic programming language.

We have successfully experimented with our design and the example programs through a prototype implementation in the Twelf system [PS98]. In future work we plan to investigate techniques for lower-level efficient implementation of ordered logic programming, which raises a number of new pragmatic and theoretical issues. We also have to gain more experience with the ordered programming techniques—we feel at present we have barely scratched the surface.

The remainder of this extended abstract is organized as follows. We first review the sequent formulation of INCLL in Section 2 and then introduce uniform derivations and show their soundness and completeness in Section 3. We eliminate non-determinism from resource allocation in Section 4 and show some example logic programs in Section 5.

2 Sequent Calculus

We review the sequent calculus for intuitionistic non-commutative linear logic (INCLL) first introduced in [PP98]. It is shown there that cut is admissible in this system, and that there is a strong connection between cut-free sequent derivations and normal natural deduction in INCLL as presented in [PP99].

For the sake of brevity we restrict ourselves to the purely implicational fragment. Extension by additive conjunction $A \& B$, additive truth $\top$ and universal quantification $\forall x.\ A$ preserve the properties we are interested in; occurrences of other connectives must be restricted as in Lolli [HM94].

$$
\begin{array}{llcll}
\textit{Formulas} & A & ::= & P & \text{atomic propositions} \\
 & & \mid & A_1 \rightarrow A_2 & \text{intuitionistic implication} \\
 & & \mid & A_1 \multimap A_2 & \text{linear implication} \\
 & & \mid & A_1 \twoheadrightarrow A_2 & \text{ordered right implication} \\
 & & \mid & A_1 \rightarrowtail A_2 & \text{ordered left implication}
\end{array}
$$

We note that in the Lambek calculus, $A_1 \rightarrowtail A_2$ is written as $A_1 \backslash A_2$ and $A_1 \twoheadrightarrow A_2$ as A_2 / A_1.

Our sequents have the form $\Gamma; \Delta; \Omega \Longrightarrow A$ where Γ, Δ, and Ω are lists of hypothesis interpreted as follows:

- Γ are *unrestricted hypotheses* (they may be used arbitrarily often in any order),
- Δ are *linear hypotheses* (each must be used exactly once, but in no particular order),
- Ω are *ordered hypotheses* (each must be used exactly once, subject to their order).

We use "$\cdot$" to stand for the empty list and juxtaposition for both list concatenation and adjoining an element to a list. Manipulating linear hypotheses requires a non-deterministic merge operation which may interleave the hypotheses in arbitrary order. We denote this operation by $\bowtie$ which is defined inductively as follows:

$$\cdot \bowtie \cdot = \cdot$$

$$\Delta_A A \bowtie \Delta_B = (\Delta_A \bowtie \Delta_B) A$$

$$\Delta_A \bowtie \Delta_B B = (\Delta_A \bowtie \Delta_B) B$$

Lemma 1 $\quad \Delta_A \bowtie \Delta_B = \Delta$ *iff* $\Delta_B \bowtie \Delta_A = \Delta$.

$(\Delta_A \bowtie \Delta_B) \bowtie \Delta_C = \Delta$ *iff* $\Delta_A \bowtie (\Delta_B \bowtie \Delta_C) = \Delta$.

We construct our sequent calculus such that the expected structural rules within each context will be admissible without being explicitly part of the system. So Γ admits exchange, weakening, and contraction, Δ admits exchange, and Ω does not admit any structural rules (except for associativity which is built in the formulation of the context as a list).

One may logically think of the three antecedent contexts as one big context where the ordered hypotheses are in a fixed relative order while the other linear hypotheses formulas may float, and the unrestricted hypotheses may float as well as copy or delete themselves. Generally the right rules are straightforward, while the left rules (which apply only to formulas in the ordered context Ω) require some care.

We start with initial sequents, which encode that all linear and ordered hypotheses must be used, while those in Γ need not be used.

$$\frac{}{\Gamma; \cdot; A \Longrightarrow A}\,\textbf{init}$$

We have two explicit structural rules: **place** which commits a linear hypothesis to a particular place among the ordered hypotheses, and **copy** which duplicates and places an unrestricted hypothesis. The non-determinism implicit in these rules when viewed from the bottom up (*Where do we place A?*) presents a major difficulty in designing a deterministic resource allocation mechanism (discussed in Section 4).

$$\frac{\Gamma_L A \Gamma_R; \Delta; \Omega_L A \Omega_R \longrightarrow B}{\Gamma_L A \Gamma_R; \Delta; \Omega_L \Omega_R \longrightarrow B}\,\textbf{copy} \qquad \frac{\Gamma; \Delta_L \Delta_R; \Omega_L A \Omega_R \longrightarrow B}{\Gamma; \Delta_L A \Delta_R; \Omega_L \Omega_R \longrightarrow B}\,\textbf{place}$$

The following four rules describing the intuitionistic and linear implications translate the standard sequent rules for intuitionistic linear logic into our setting. Note the restrictions on the linear and ordered contexts in the two left rules which are necessary to preserve linearity and order, respectively.

$$\frac{\Gamma A; \Delta; \Omega \Longrightarrow B}{\Gamma; \Delta; \Omega \Longrightarrow A \to B}\to_R \qquad \frac{\Gamma; \Delta; \Omega_L B \Omega_R \Longrightarrow C \qquad \Gamma; \cdot; \cdot \Longrightarrow A}{\Gamma; \Delta; \Omega_L (A \to B) \Omega_R \Longrightarrow C}\to_L$$

$$\frac{\Gamma; \Delta A; \Omega \Longrightarrow B}{\Gamma; \Delta; \Omega \Longrightarrow A \multimap B}\multimap_R \qquad \frac{\Gamma; \Delta_B; \Omega_L B \Omega_R \Longrightarrow C \qquad \Gamma; \Delta_A; \cdot \Longrightarrow A}{\Gamma; \Delta_B \bowtie \Delta_A; \Omega_L (A \multimap B) \Omega_R \Longrightarrow C}\multimap_L$$

The right rule for ordered right implication $A \twoheadrightarrow B$ adds A at the right end of the ordered context. For cut-elimination to hold, the left rule must then take hypotheses immediately to the right of the right implication for deriving the antecedent A. The remaining hypotheses are joined with B (in order!) to derive C. We must also be careful that each linear hypothesis comes from exactly one premise, although their order does not matter (hence the merge operation $\Delta_B \bowtie \Delta_A$).

$$\frac{\Gamma; \Delta; \Omega A \Longrightarrow B}{\Gamma; \Delta; \Omega \Longrightarrow A \twoheadrightarrow B}\twoheadrightarrow_R \qquad \frac{\Gamma; \Delta_B; \Omega_L B \Omega_R \Longrightarrow C \qquad \Gamma; \Delta_A; \Omega_A \Longrightarrow A}{\Gamma; \Delta_B \bowtie \Delta_A; \Omega_L (A \twoheadrightarrow B) \Omega_A \Omega_R \Longrightarrow C}\twoheadrightarrow_L$$

The rules for left implication are symmetric.

$$\frac{\Gamma; \Delta; A\Omega \Longrightarrow B}{\Gamma; \Delta; \Omega \Longrightarrow A \rightarrowtail B}\rightarrowtail_R \qquad \frac{\Gamma; \Delta_B; \Omega_L B \Omega_R \Longrightarrow C \qquad \Gamma; \Delta_A; \Omega_A \Longrightarrow A}{\Gamma; \Delta_B \bowtie \Delta_A; \Omega_L \Omega_A (A \rightarrowtail B) \Omega_R \Longrightarrow C}\rightarrowtail_L$$

To give a feel for how the two ordered implications and the ordered context work, we note the following: $A \twoheadrightarrow (A \twoheadrightarrow B) \twoheadrightarrow B$ is not provable while $A \twoheadrightarrow (A \twoheadrightarrow B) \rightarrowtail B$ is provable. Symmetrically, $A \twoheadrightarrow (A \rightarrowtail B) \twoheadrightarrow B$ is provable while $A \twoheadrightarrow (A \rightarrowtail B) \rightarrowtail B$ is not provable. Furthermore, $A \twoheadrightarrow (B \rightarrowtail C)$ is provable iff $B \rightarrowtail (A \twoheadrightarrow C)$.

We now show a sample derivation which sketches how INCLL can be used for natural language parsing. Suppose $\Gamma = [\texttt{np} \twoheadrightarrow \texttt{vp} \twoheadrightarrow \texttt{snt},\ \texttt{tv} \twoheadrightarrow \texttt{np} \twoheadrightarrow \texttt{vp},\ \texttt{loves} \twoheadrightarrow \texttt{tv},\ \texttt{mary} \twoheadrightarrow \texttt{np},\ \texttt{bob} \twoheadrightarrow \texttt{np}]$ where all the words and grammatical abbreviations are atomic formulas. We may think of the formulas in Γ as a grammar for simple English sentences. The phrase to be parsed with the grammar is in the ordered context. The succedent contains the grammatical pattern with which we are trying to classify the input. Thus to parse the sentence: `mary loves bob`, we would prove:

$\Gamma; \cdot; \texttt{mary loves bob} \Longrightarrow \texttt{snt}$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\Theta \qquad \dfrac{}{\Gamma; \cdot; \texttt{bob} \Longrightarrow \texttt{bob}}\,\textbf{init}}{\Gamma; \cdot; \texttt{np tv (bob} \twoheadrightarrow \texttt{np) bob} \Longrightarrow \texttt{snt}}\,{\twoheadrightarrow_L} \qquad \dfrac{}{\Gamma; \cdot; \texttt{loves} \Longrightarrow \texttt{loves}}\,\textbf{init}}{\Gamma; \cdot; \texttt{np (loves} \twoheadrightarrow \texttt{tv) loves (bob} \twoheadrightarrow \texttt{np) bob} \Longrightarrow \texttt{snt}}\,{\twoheadrightarrow_L} \qquad \dfrac{}{\Gamma; \cdot; \texttt{mary} \Longrightarrow \texttt{mary}}\,\textbf{init}}{\Gamma; \cdot; \texttt{(mary} \twoheadrightarrow \texttt{np) mary (loves} \twoheadrightarrow \texttt{tv) loves (bob} \twoheadrightarrow \texttt{np) bob} \Longrightarrow \texttt{snt}}\,{\twoheadrightarrow_L}}{\Gamma; \cdot; \texttt{mary loves bob} \Longrightarrow \texttt{snt}}\,\textbf{copy} * \textbf{3}$$

where $\Theta =$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma; \cdot; \texttt{snt} \Longrightarrow \texttt{snt}}\,\textbf{init} \qquad \dfrac{}{\Gamma; \cdot; \texttt{vp} \Longrightarrow \texttt{vp}}\,\textbf{init}}{\Gamma; \cdot; \texttt{(vp} \twoheadrightarrow \texttt{snt) vp} \Longrightarrow \texttt{snt}}\,{\twoheadrightarrow_L} \qquad \dfrac{}{\Gamma; \cdot; \texttt{np} \Longrightarrow \texttt{np}}\,\textbf{init}}{\Gamma; \cdot; \texttt{(np} \twoheadrightarrow \texttt{vp} \twoheadrightarrow \texttt{snt) np vp} \Longrightarrow \texttt{snt}}\,{\twoheadrightarrow_L} \qquad \dfrac{}{\Gamma; \cdot; \texttt{np} \Longrightarrow \texttt{np}}\,\textbf{init}}{\Gamma; \cdot; (\mathbf{np} \twoheadrightarrow \texttt{vp} \twoheadrightarrow \texttt{snt) np (np} \twoheadrightarrow \texttt{vp) np} \Longrightarrow \texttt{snt}}\,{\twoheadrightarrow_L} \qquad \dfrac{}{\Gamma; \cdot; \texttt{tv} \Longrightarrow \texttt{tv}}\,\textbf{init}}{\Gamma; \cdot; \texttt{(np} \twoheadrightarrow \texttt{vp} \twoheadrightarrow \texttt{snt) np (tv} \twoheadrightarrow \texttt{np} \twoheadrightarrow \texttt{vp) tv np} \Longrightarrow \texttt{snt}}\,{\twoheadrightarrow_L}}{\Gamma; \cdot; \texttt{np tv np} \Longrightarrow \texttt{snt}}\,\textbf{copy} * \textbf{2}$$

Note that this is not the only way to derive the end-sequent. For instance, we could have moved all instances of **copy** and **place** to the beginning of the derivation; or we could have applied $\twoheadrightarrow_L$ to the formulas in a different order. We leave a more in-depth and interesting example of natural language parsing for section 5.

We shall now validate our version of the sequent calculus by showing the admissibility of cut in the system. Towards this end, we have the following lemma.

Lemma 2 (Weakening, Contraction, and Exchange) *The following all hold:*

1. $\Gamma; \Delta; \Omega \implies C$ *implies* $\Gamma A; \Delta; \Omega \implies C$.
2. $\Gamma AA; \Delta; \Omega \implies C$ *implies* $\Gamma A; \Delta; \Omega \implies C$.
3. $\Gamma_L AB \Gamma_R; \Delta; \Omega \implies C$ *implies* $\Gamma_L BA \Gamma_R; \Delta; \Omega \implies C$.
4. $\Gamma; \Delta_L AB \Delta_R; \Omega \implies C$ *implies* $\Gamma; \Delta_L BA \Delta_R; \Omega \implies C$.

Proof: By structural induction on the sequent derivation of the given judgement. Note that the structure of the derivation remains the same. □

Theorem 3 (Admissibility of Cut) *The following three statements hold:*

Cut$_\Omega$*:* $\Gamma; \Delta_C; \Omega_C \implies C$ *and* $\Gamma; \Delta; \Omega_L C \Omega_R \implies A$ *implies* $\Gamma; \Delta_C \bowtie \Delta; \Omega_L \Omega_C \Omega_R \implies A$.

Cut$_\Delta$*:* $\Gamma; \Delta_C; \cdot \implies C$ *and* $\Gamma; \Delta_L C \Delta_R; \Omega \implies A$ *implies* $\Gamma; \Delta_L \bowtie \Delta_C \bowtie \Delta_R; \Omega \implies A$.

Cut$_\Gamma$*:* $\Gamma; \cdot; \cdot \implies C$ *and* $\Gamma_L C \Gamma_R; \Delta; \Omega \implies A$ *implies* $\Gamma_L \Gamma \Gamma_R; \Delta; \Omega \implies A$.

The following proof is adapted from [Pfe94].

Proof: By induction on the structure of the cut formula, the type of cut where $\mathbf{Cut}_\Gamma > \mathbf{Cut}_\Delta > \mathbf{Cut}_\Omega$, and the derivations of the given judgements. Therefore we may apply the induction hypothesis in the following cases: 1) the cut formula gets smaller; 2) the same cut formula but we move from $\mathbf{Cut}_\Gamma$ to $\mathbf{Cut}_\Omega$ (or to $\mathbf{Cut}_\Delta$); 2) the same cut formula but we move from $\mathbf{Cut}_\Delta$ to $\mathbf{Cut}_\Omega$; 4) the cut formula and type of cut stay the same but one of the given judgements of the induction hypothesis gets smaller.

There are 4 basic cases to consider: **init** cases where one of the derivations ends in with the **init** rule, essential cases where the principal formula of the end-sequent of both derivations is cut, commutative cases where the cut formula is a side formula on the end-sequent of either given derivation. Note that these cases are not mutually exclusive.

case 1: init cases.

case: $\dfrac{}{\Gamma; \cdot; C \implies C}\,\mathbf{init}$ and $\Gamma; \Delta; \Omega_L C \Omega_R \implies A$ is trivial.

case: $\Gamma; \Delta; \Omega \implies C$ and $\dfrac{}{\Gamma; \cdot; C \implies C}\,\mathbf{init}$ is trivial.

case: $\Gamma; \cdot; \cdot \implies C$ and $\dfrac{}{\Gamma_L C \Gamma_R; \cdot; A \implies A}\,\mathbf{init}$ then $\dfrac{}{\Gamma_L \Gamma \Gamma_R; \cdot; A \implies A}\,\mathbf{init}$

case 2: Essential cases.

case: $\Gamma; \cdot; \cdot \implies C$ and $\dfrac{\Gamma_L C \Gamma_R; \Delta; \Omega_L C \Omega_R \implies A}{\Gamma_L C \Gamma_R; \Delta; \Omega_L \Omega_R \implies A}\,\mathbf{copy}$

Then by ind. hyp. we know (using $\mathbf{Cut}_\Gamma$) $\Gamma_L \Gamma \Gamma_R; \Delta; \Omega_L C \Omega_R \implies A$.
By weakening (part 1 of lemma 2) we know $\Gamma_L \Gamma \Gamma_R; \cdot; \cdot \implies C$.
Then again by ind. hyp. (since $\mathbf{Cut}_\Gamma > \mathbf{Cut}_\Omega$) we know $\Gamma_L \Gamma \Gamma_R; \Delta; \Omega_L \Omega_R \implies A$.

case: $\Gamma; \Delta_C; \cdot \implies C$ and $\dfrac{\Gamma; \Delta_L \Delta_R; \Omega_L C \Omega_R \implies A}{\Gamma; \Delta_L C \Delta_R; \Omega_L \Omega_R \implies A}\,\mathbf{place}$

Then by ind. hyp. we know $\Gamma; \Delta_L \bowtie \Delta_C \bowtie \Delta_R; \Omega_L \Omega_R \implies A$.

case: $\dfrac{\Gamma A; \Delta; \Omega \implies B}{\Gamma; \Delta; \Omega \implies A \to B}\,{\to_R}$ and $\dfrac{\Gamma; \Delta_C; \Omega_L B \Omega_R \implies C \qquad \Gamma; \cdot; \cdot \implies A}{\Gamma; \Delta_C; \Omega_L (A \to B) \Omega_R \implies C}\,{\to_L}$

Then by ind. hyp. (using $\mathbf{Cut}_\Gamma$) we know $\Gamma\Gamma; \Delta; \Omega \implies B$
Then by contraction (part 2 of lemma 2) we know $\Gamma; \Delta; \Omega \implies B$
Then by ind. hyp. (using $\mathbf{Cut}_\Omega$) we know $\Gamma; \Delta \bowtie \Delta_C; \Omega_L \Omega \Omega_R \implies C$

case: $\multimap$ is similar to $\to$.

case: $\dfrac{\Gamma; \Delta; A\Omega \implies B}{\Gamma; \Delta; \Omega \implies A \rightarrowtail B} \rightarrowtail_R$ and $\dfrac{\Gamma; \Delta_B; \Omega_L B \Omega_R \implies C \qquad \Gamma; \Delta_A; \Omega_A \implies A}{\Gamma; \Delta_B \bowtie \Delta_A; \Omega_L \Omega_A (A \rightarrowtail B) \Omega_R \implies C} \rightarrowtail_L$

Then by ind. hyp. we know $\Gamma; \Delta_A \bowtie \Delta; \Omega_A \Omega \implies B$.
Then by ind. hyp. we know $\Gamma; \Delta_A \bowtie \Delta \bowtie \Delta_B; \Omega_L \Omega_A \Omega \Omega_R \implies C$.

case: $\twoheadrightarrow$ is similar to $\rightarrowtail$.

case 3: Commutative cases where cut formula is not principal in first hypothesis (i.e. end-sequent of first given derivation can't end in a right rule).

case: $\dfrac{\Gamma; \Delta_B; \Omega_L B \Omega_R \implies C \qquad \Gamma; \Delta_A; \Omega_A \implies A}{\Gamma; \Delta_B \bowtie \Delta_A; \Omega_L \Omega_A (A \rightarrowtail B) \Omega_R \implies C} \rightarrowtail_L$ and $\Gamma; \Delta; \Omega_{LC} C \Omega_{RC} \implies D$

Then by ind. hyp. $\Gamma; \Delta_B \bowtie \Delta; \Omega_{LC} \Omega_L B \Omega_R \Omega_{RC} \implies D$.
Then by $\rightarrowtail_L$ we know $\Gamma; \Delta_B \bowtie \Delta \bowtie \Delta_A; \Omega_{LC} \Omega_L \Omega_A (A \rightarrowtail B) \Omega_R \Omega_{RC} \implies D$.

The cases for $\twoheadrightarrow_L$, $\multimap_L$, $\rightarrow_L$ are similar.

case: $\dfrac{\Gamma_L A \Gamma_R; \Delta_C; \Omega_{LC} A \Omega_{RC} \implies C}{\Gamma_L A \Gamma_R; \Delta_C; \Omega_{LC} \Omega_{RC} \implies C} \textbf{copy}$ and $\Gamma_L A \Gamma_R; \Delta; \Omega_L C \Omega_R \implies D$

Then by ind. hyp. (using $\mathbf{Cut}_\Omega$) we know $\Gamma_L A \Gamma_R; \Delta_C \bowtie \Delta; \Omega_L \Omega_{LC} A \Omega_{RC} \Omega_R \implies D$.
Then by **copy** we know $\Gamma_L A \Gamma_R; \Delta_C \bowtie \Delta; \Omega_L \Omega_{LC} \Omega_{RC} \Omega_R \implies D$.

The case for **place** is similar.

case 4: commutative cases where cut formula is not principal in end-sequent of second given derivation.

case: $\Gamma; \Delta_C; \Omega_C \implies C$ and $\dfrac{\Gamma; \Delta; A \Omega_L C \Omega_R \implies B}{\Gamma; \Delta; \Omega_L C \Omega_R \implies A \rightarrowtail B} \rightarrowtail_R$

Then by ind. hyp. we know $\Gamma; \Delta_C \bowtie \Delta; A \Omega_L \Omega_C \Omega_R \implies B$.
Then by $\rightarrowtail_R$ we know $\Gamma; \Delta_C \bowtie \Delta; \Omega_L \Omega_C \Omega_R \implies A \rightarrowtail B$.

Cases for $\twoheadrightarrow_R$,$\multimap_R$,$\rightarrow_R$ are similar.

case: $\Gamma; \Delta_C; \Omega_C \implies C$ and $\dfrac{\Gamma; \Delta_B; \Omega_{LL} C \Omega_{LR} B \Omega_R \implies D \qquad \Gamma; \Delta_A; \Omega_A \implies A}{\Gamma; \Delta_B \bowtie \Delta_A; \Omega_{LL} C \Omega_{LR} \Omega_A (A \rightarrowtail B) \Omega_R \implies D} \rightarrowtail_L$

Then by ind. hyp. we know $\Gamma; \Delta_C \bowtie \Delta_B; \Omega_{LL} \Omega_C \Omega_{LR} B \Omega_R \implies D$

Then by $\rightarrowtail_L$ we know $\Gamma; \Delta_C \bowtie \Delta_B \bowtie \Delta_A; \Omega_{LL}\Omega_C\Omega_{LR}\Omega_A(A \rightarrowtail B)\Omega_R \implies D$.

Cases when second given derivation is

$$\frac{\Gamma; \Delta_B; \Omega_L B \Omega_R \implies D \qquad \Gamma; \Delta_A; \Omega_{AL} C \Omega_{AR} \implies A}{\Gamma; \Delta_B \bowtie \Delta_A; \Omega_L \Omega_{AL} C \Omega_{AR}(A \rightarrowtail B)\Omega_R \implies D} \rightarrowtail_L \qquad \text{or}$$

$$\frac{\Gamma; \Delta_B; \Omega_L B \Omega_{RL} C \Omega_{RR} \implies D \qquad \Gamma; \Delta_A; \Omega_A \implies A}{\Gamma; \Delta_B \bowtie \Delta_A; \Omega_L \Omega_A(A \rightarrowtail B)\Omega_{RL} C \Omega_{RR} \implies D} \rightarrowtail_L$$

are similar.

Cases for $\twoheadrightarrow_L$, $\multimap_L$, $\rightarrow_L$, **copy**, **place** are similar.

case: $\Gamma; \Delta_C; \cdot \implies C$ and $\dfrac{\Gamma; \Delta_L C \Delta_R; A\Omega \implies B}{\Gamma; \Delta_L C \Delta_R; \Omega \implies A \rightarrowtail B} \rightarrowtail_R$

Then by ind. hyp. we know $\Gamma; \Delta_L \bowtie \Delta_C \bowtie \Delta_R; A\Omega \implies B$.
Then by $\rightarrowtail_R$ we know $\Gamma; \Delta_L \bowtie \Delta_C \bowtie \Delta_R; \Omega \implies A \rightarrowtail B$.

Cases for $\twoheadrightarrow_R$,$\multimap_R$,$\rightarrow_R$ are similar.

case: $\Gamma; \Delta_C; \cdot \implies C$ and $\dfrac{\Gamma; \Delta_{BL} C \Delta_{BR}; \Omega_L B \Omega_R \implies D \qquad \Gamma; \Delta_A; \Omega_A \implies A}{\Gamma; (\Delta_{BL} C \Delta_{BR}) \bowtie \Delta_A; \Omega_L \Omega_A(A \rightarrowtail B)\Omega_R \implies D} \rightarrowtail_L$

Then by ind. hyp. we know $\Gamma; \Delta_C \bowtie (\Delta_{BL}\Delta_{BR}); \Omega_L B \Omega_R \implies D$
Then by $\rightarrowtail_L$ we know $\Gamma; \Delta_C \bowtie (\Delta_{BL}\Delta_{BR}) \bowtie \Delta_A; \Omega_L \Omega_A(A \rightarrowtail B)\Omega_R \implies D$.

Case when second given derivation is

$$\frac{\Gamma; \Delta_B; \Omega_L B \Omega_R \implies D \qquad \Gamma; \Delta_{AL} C \Delta_{AR}; \Omega_A \implies A}{\Gamma; \Delta_B \bowtie (\Delta_{AL} C \Delta_{AR}); \Omega_L \Omega_A(A \rightarrowtail B)\Omega_R \implies D} \rightarrowtail_L \qquad \text{is similar.}$$

Cases for $\twoheadrightarrow_L$,$\multimap_L$,$\rightarrow_L$, **copy**,**place** are similar.

case: $\Gamma; \cdot; \cdot \implies C$ and $\dfrac{\Gamma_L C \Gamma_R; \Delta; A\Omega \implies B}{\Gamma_L C \Gamma_R; \Delta; \Omega \implies A \rightarrowtail B} \rightarrowtail_R$

Then by ind. hyp. we know $\Gamma_L \Gamma \Gamma_R; \Delta; A\Omega \implies B$.
Then by $\rightarrowtail_R$ we know $\Gamma_L \Gamma \Gamma_R; \Delta; \Omega \implies A \rightarrowtail B$.

Cases for $\twoheadrightarrow_R$,$\multimap_R$,$\rightarrow_R$ are similar.

case: $\Gamma; \cdot; \cdot \implies C$ and $\dfrac{\Gamma_L C \Gamma_R; \Delta_B; \Omega_L B \Omega_R \implies D \qquad \Gamma_L C \Gamma_R; \Delta_A; \Omega_A \implies A}{\Gamma_L C \Gamma_R; \Delta_B \bowtie \Delta_A; \Omega_L \Omega_A (A \rightarrowtail B)\Omega_R \implies D} \rightarrowtail_L$

Then by ind. hyp. we know $\Gamma_L \Gamma \Gamma_R; \Delta_B; \Omega_L B \Omega_R \implies D$ and $\Gamma_L \Gamma \Gamma_R; \Delta_A; \Omega_A \implies A$
Then by $\rightarrowtail_L$ we know $\Gamma_L \Gamma \Gamma_R; \Delta_B \bowtie \Delta_A; \Omega_L \Omega_A (A \rightarrowtail B)\Omega_R \implies D$.

Cases for $\twoheadrightarrow_L$,$\multimap_L$,$\rightarrow_L$, **copy**,**place** are similar.

□

3 Uniform Derivations

Now that we have a suitable sequent system for INCLL we begin analyzing proof structure with an eye towards achieving a logic programming language, where we view computation as the bottom-up construction of a derivation. We refer to the succedent of a given sequent as the *goal*. The difficulty with the sequent system is that in any situation, many left or right rules could be applied, leading to unacceptable non-determinism. To solve this problem, we design an alternative, more restricted system with the following properties (which are enforced *syntactically*):

- Derivations are *goal-directed* in that a sequent with a non-atomic goal always ends in a right rule. This allows us to view logical connectives in goals as search instructions.
- Derivations are *focussed* in that when deriving a sequent with an atomic goal we single out a particular hypothesis and apply a sequence of left rules until it is also atomic and immediately implies the goal. This allows us to view atomic goals as procedure calls.

In a minor departure from [MNPS91] we call derivations which are both goal-directed and focussed *uniform* and write

$\Gamma; \Delta; \Omega \longrightarrow A$	goal A is uniformly derivable, and
$\Gamma; \Delta; (\Omega_L; \Omega_R) \longrightarrow A \gg P$	hypothesis A immediately entails atomic goal P,

where Γ, Δ and Ω are unrestricted, linear, and ordered hypotheses, respectively. In the latter judgment the ordered hypotheses are syntactically divided into a left part Ω_L and a right part Ω_R. It corresponds to the sequent

$$\Gamma; \Delta; (\Omega_L A \Omega_R) \implies P$$

so that the split in the ordered context tracks the location of the hypothesis we have focused on. This correspondence is stated formally in the soundness and completeness theorems for uniform derivations below.

All of the right rules are exactly the same as in the sequent calculus. Since no left rules apply when the goal is non-atomic, the derivation is completely determined by the structure of the goal, as desired.

$$\frac{\Gamma A;\Delta;\Omega \longrightarrow B}{\Gamma;\Delta;\Omega \longrightarrow A \rightarrow B}\rightarrow_R \qquad \frac{\Gamma;\Delta A;\Omega \longrightarrow B}{\Gamma;\Delta;\Omega \longrightarrow A \multimap B}\multimap_R$$

$$\frac{\Gamma;\Delta;A\Omega \longrightarrow B}{\Gamma;\Delta;\Omega \longrightarrow A \rightarrowtail B}\rightarrowtail_R \qquad \frac{\Gamma;\Delta;\Omega A \longrightarrow B}{\Gamma;\Delta;\Omega \longrightarrow A \twoheadrightarrow B}\twoheadrightarrow_R$$

When the goal has become atomic, we need to single out a hypothesis and determine if it immediately entails the goal. This is achieved by the three **choice** rules which apply to unrestricted, linear, or ordered hypotheses.

$$\frac{\Gamma_L A \Gamma_R;\Delta;(\Omega_L;\Omega_R) \longrightarrow A \gg P}{\Gamma_L A \Gamma_R;\Delta;\Omega_L\Omega_R \longrightarrow P}\textbf{choice}_\Gamma$$

$$\frac{\Gamma;\Delta_L\Delta_R;(\Omega_L;\Omega_R) \longrightarrow A \gg P}{\Gamma;\Delta_L A \Delta_R;\Omega_L\Omega_R \longrightarrow P}\textbf{choice}_\Delta \qquad \frac{\Gamma;\Delta;(\Omega_L;\Omega_R) \longrightarrow A \gg P}{\Gamma;\Delta;\Omega_L A \Omega_R \longrightarrow P}\textbf{choice}_\Omega$$

$\textbf{choice}_\Gamma$ is justified by **copy** in the sequent calculus, and $\textbf{choice}_\Delta$ by **place**. The premise and conclusion of $\textbf{choice}_\Omega$ correspond to identical sequents. An initial sequent corresponds to an immediate entailment between identical atomic formulas.

$$\frac{}{\Gamma;\cdot;(\cdot;\cdot) \longrightarrow P \gg P}\textbf{init}$$

The remaining left rules for immediate entailment directly correspond to the left sequent rules, keeping in mind that we have to consider the focussing formula as being between the left and right parts of the ordered context.

$$\frac{\Gamma;\Delta;(\Omega_L;\Omega_R) \longrightarrow B \gg P \qquad \Gamma;\cdot;\cdot \longrightarrow A}{\Gamma;\Delta;(\Omega_L;\Omega_R) \longrightarrow A \rightarrow B \gg P}\rightarrow_L$$

$$\frac{\Gamma;\Delta_B;(\Omega_L;\Omega_R) \longrightarrow B \gg P \qquad \Gamma;\Delta_A;\cdot \longrightarrow A}{\Gamma;\Delta_A \bowtie \Delta_B;(\Omega_L;\Omega_R) \longrightarrow A \multimap B \gg P}\multimap_L$$

$$\frac{\Gamma;\Delta_B;(\Omega_L;\Omega_R) \longrightarrow B \gg P \qquad \Gamma;\Delta_A;\Omega_A \longrightarrow A}{\Gamma;\Delta_A \bowtie \Delta_B;(\Omega_L\Omega_A;\Omega_R) \longrightarrow A \rightarrowtail B \gg P}\rightarrowtail_L$$

$$\frac{\Gamma;\Delta_B;(\Omega_L;\Omega_R) \longrightarrow B \gg P \qquad \Gamma;\Delta_A;\Omega_A \longrightarrow A}{\Gamma;\Delta_A \bowtie \Delta_B;(\Omega_L;\Omega_A\Omega_R) \longrightarrow A \twoheadrightarrow B \gg P}\twoheadrightarrow_L$$

Note that in the last two rules, Ω_A is some initial or final segment of the right or left part of the ordered context, respectively.

In the uniform system, we rewrite our sample parsing proof from the previous section as follows:

$$
\dfrac{\dfrac{\dfrac{}{\Gamma;\cdot;(\cdot;\cdot)\longrightarrow \texttt{snt}\gg\texttt{snt}}\,\mathbf{init}\qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma;\cdot;(\cdot;\cdot)\longrightarrow \texttt{vp}\gg\texttt{vp}}\,\mathbf{init}\qquad \Theta_{\texttt{bob}}}{\Gamma;\cdot;(\cdot;\texttt{bob})\longrightarrow \texttt{np}\twoheadrightarrow\texttt{vp}\gg\texttt{vp}}\twoheadrightarrow_L\qquad \Theta_{\texttt{loves}}}{\Gamma;\cdot;(\cdot;\texttt{loves bob})\longrightarrow \texttt{tv}\twoheadrightarrow\texttt{np}\twoheadrightarrow\texttt{vp}\gg\texttt{vp}}\twoheadrightarrow_L}{\Gamma;\cdot;\texttt{loves bob}\longrightarrow\texttt{vp}}\,\mathbf{choice}_\Gamma}{\dfrac{\Gamma;\cdot;(\cdot;\texttt{loves bob})\longrightarrow(\texttt{vp}\twoheadrightarrow\texttt{snt})\gg\texttt{snt}\qquad\Theta_{\texttt{mary}}}{\dfrac{\Gamma;\cdot;(\cdot;\texttt{mary loves bob})\longrightarrow\texttt{np}\twoheadrightarrow\texttt{vp}\twoheadrightarrow\texttt{snt}\gg\texttt{snt}}{\Gamma;\cdot;\texttt{mary loves bob}\longrightarrow\texttt{snt}}\,\mathbf{choice}_\Gamma}\twoheadrightarrow_L}\twoheadrightarrow_L
$$

where $\Theta_{\texttt{bob}}$ =

$$
\dfrac{\dfrac{\dfrac{}{\Gamma;\cdot;(\cdot;\cdot)\longrightarrow\texttt{np}\gg\texttt{np}}\,\mathbf{init}\qquad\dfrac{\dfrac{}{\Gamma;\cdot;(\cdot;\cdot)\longrightarrow\texttt{bob}\gg\texttt{bob}}\,\mathbf{init}}{\Gamma;\cdot;\texttt{bob}\longrightarrow\texttt{bob}}\,\mathbf{choice}_\Omega}{\Gamma;\cdot;(\cdot;\texttt{bob})\longrightarrow\texttt{bob}\twoheadrightarrow\texttt{np}\gg\texttt{np}}\twoheadrightarrow_L}{\Gamma;\cdot;\texttt{bob}\longrightarrow\texttt{np}}\,\mathbf{choice}_\Gamma
$$

and $\Theta_{\texttt{mary}}, \Theta_{\texttt{loves}}$ are similar.

Unlike the example given in the previous section, this is the only proof of the end-sequent. The first choice is forced since $\texttt{np}\twoheadrightarrow\texttt{vp}\twoheadrightarrow\texttt{snt}$ is the only formula in Γ whose head, $\texttt{snt}$, matches the goal. The same is true for all the other choices made.

We now show that uniform derivations are sound and complete with respect to the sequent calculus. We begin with a lemma which shows that the sequent calculus presented in section 2 can be restricted to a goal-directed system without affecting the provability of any formula.

The soundness result is easy to show.

Theorem 4 (Soundness of Uniform Derivations)

1. *If* $\Gamma;\Delta;\Omega\ \longrightarrow\ A$ *then* $\Gamma;\Delta;\Omega\ \Longrightarrow\ A$.
2. *If* $\Gamma;\Delta;(\Omega_L;\Omega_R)\ \longrightarrow\ A\gg P$ *then* $\Gamma;\Delta;\Omega_L A\Omega_R\ \Longrightarrow\ P$.

Proof: By mutual structural induction on the sequent derivations of the given judgements. □

The completeness results is harder, but largely follows techniques of [And92] and [MNPS91], adapted to the ordered case.

We begin with the following lemmas.

Lemma 5 (Goal Directedness) *The following all hold:*

1. $\Gamma;\Delta;\Omega\Longrightarrow A\twoheadrightarrow B$ *implies* $\Gamma;\Delta;\Omega A\longrightarrow B$
2. $\Gamma;\Delta;\Omega\Longrightarrow A\rightarrowtail B$ *implies* $\Gamma;\Delta;A\Omega\longrightarrow B$
3. $\Gamma;\Delta;\Omega\Longrightarrow A\multimap B$ *implies* $\Gamma;\Delta A;\Omega\longrightarrow B$
4. $\Gamma;\Delta;\Omega\Longrightarrow A\rightarrow B$ *implies* $\Gamma A;\Delta;\Omega\longrightarrow B$

Proof: By induction on the formula in the succedent in the case of **init**; for the other rules by induction on the sequent derivation of the given judgement making use of the left sequent rules. □

The next lemmas concern inversion principles for uniform derivations. Lemma 6 states the inversion principle for left implication ($\rightarrowtail$)— given $\Gamma; \Delta_A; \Omega_A \longrightarrow A$ and $\Gamma; \Delta; \Omega_L B \Omega_R \longrightarrow P$ we can conclude $\Gamma; \Delta \bowtie \Delta_A; \Omega_L \Omega_A (A \rightarrowtail B) \Omega_R \longrightarrow P$. Lemma 7 states all the inversion principles we need for the proof of completeness.

Lemma 6 $\Gamma; \Delta_A; \Omega_A \longrightarrow A$ *implies all of the following hold:*

1. $\Gamma; \Delta; \Omega_L B \Omega_R \longrightarrow P$ *implies* $\Gamma; \Delta \bowtie \Delta_A; \Omega_L \Omega_A (A \rightarrowtail B) \Omega_R \longrightarrow P$.
2. $\Gamma; \Delta; (\Omega_{LL} B \Omega_{LR}; \Omega_R) \longrightarrow C \gg P$ *implies* $\Gamma; \Delta \bowtie \Delta_A; (\Omega_{LL} \Omega_A (A \rightarrowtail B) \Omega_{LR}; \Omega_R) \longrightarrow C \gg P$.
3. C, P, $\Gamma; \Delta; (\Omega_L; \Omega_{RL} B \Omega_{RR}) \longrightarrow C \gg P$ *implies* $\Gamma; \Delta \bowtie \Delta_A; (\Omega_L; \Omega_{RL} \Omega_A (A \rightarrowtail B) \Omega_{RR}) \longrightarrow C \gg P$.

Proof: By mutual induction on the structure of the given derivations. Assume $\Gamma; \Delta_A; \Omega_A \longrightarrow A$.

part 1: Assume $\Gamma; \Delta; \Omega_L B \Omega_R \overset{\Pi}{\longrightarrow} C$. Then there are 7 possibilities for Π:

case 1: Π ends with **choice**$_\Omega$ and $\Gamma; \Delta; (\Omega_L; \Omega_R) \longrightarrow B \gg P$
Then by $\rightarrowtail_L$ we know $\Gamma; \Delta \bowtie \Delta_A; (\Omega_L \Omega_A; \Omega_R) \longrightarrow A \rightarrowtail B \gg P$.
Then $\Gamma; \Delta \bowtie \Delta_A; \Omega_L \Omega_A (A \rightarrowtail B) \Omega_R \longrightarrow P$.

case 2: Π ends with **choice**$_\Omega$ and $\Gamma; \Delta; (\Omega_{LL}; \Omega_{LR} B \Omega_R) \longrightarrow C \gg P$ where $\Omega_L = \Omega_{LL} C \Omega_{LR}$.
By ind. hyp. (using part 3) we know $\Gamma; \Delta \bowtie \Delta_A; (\Omega_{LL}; \Omega_{LR} \Omega_A (A \rightarrowtail B) \Omega_R) \longrightarrow C \gg P$.
Then $\Gamma; \Delta \bowtie \Delta_A; \Omega_{LL} C \Omega_{LR} \Omega_A (A \rightarrowtail B) \Omega_R \longrightarrow P$.

case 3: Π ends with $\Gamma; \Delta; (\Omega_L B \Omega_{RL}; \Omega_{RR}) \longrightarrow C \gg P$ where $\Omega_R = \Omega_{RL} C \Omega_{RR}$.
By ind. hyp. (using part 2) we know $\Gamma; \Delta \bowtie \Delta_A; (\Omega_L \Omega_A (A \rightarrowtail B) \Omega_{RL}; \Omega_{RR}) \longrightarrow C \gg P$.
Then $\Omega_L \Omega_A (A \rightarrowtail B) \Omega_{RL} C \Omega_{RR} \longrightarrow P$.

cases 4,5,6,7 : Π ends with **choice**$_\Delta$ (2 cases) or **choice**$_\Gamma$ (2 cases).
Similar to previous two cases.

part 2: Assume $\Gamma; \Delta; (\Omega_{LL} B \Omega_{LR}; \Omega_R) \longrightarrow C \gg P$. Note that C cannot be atomic. Then there are 5 possibilities:

case 1: $C = C_1 \rightarrowtail C_2$ and $\Delta = \Delta_2 \bowtie \Delta_1$ and $\Omega_{LR} = \Omega_{LRL} \Omega_{LRR}$ and
$\Gamma; \Delta_2; (\Omega_{LL} B \Omega_{LRL}; \Omega_R) \longrightarrow C_2 \gg P$ and $\Gamma; \Delta_1; \Omega_{LRR} \longrightarrow C_1$.

By ind. hyp. we know: $\Gamma; \Delta_1 \bowtie \Delta_A; (\Omega_{LL} \Omega_A (A \rightarrowtail B) \Omega_{LRL}; \Omega_R) \longrightarrow C_2 \gg P$.
Then $\Gamma; \Delta \bowtie \Delta_A; (\Omega_{LL} \Omega_A (A \rightarrowtail B) \Omega_{LRL} \Omega_{LRR}; \Omega_R) \longrightarrow C_1 \rightarrowtail C_2 \gg P$.

case 2: $C = C_1 \rightarrowtail C_2$ and $\Delta = \Delta_2 \bowtie \Delta_1$ and $\Omega_{LL} = \Omega_{LLL} \Omega_{LLR}$ and
$\Gamma; \Delta_2; (\Omega_{LLL}; \Omega_R) \longrightarrow C_2 \gg P$ and $\Gamma; \Delta_1; \Omega_{LLR} B \Omega_{LR} \longrightarrow C_1$.

By ind. hyp. (using part 1) we know $\Gamma; \Delta_1 \bowtie \Delta_A; \Omega_{LLR} \Omega_A (A \rightarrowtail B) \Omega_{LR} \longrightarrow C_1$.
Then $\Gamma; \Delta \bowtie \Delta_A; (\Omega_{LLL} \Omega_{LLR} \Omega_A (A \rightarrowtail B) \Omega_{LR}; \Omega_R) \longrightarrow C_1 \rightarrowtail C_2 \gg P$.

case 3: $C = C_1 \twoheadrightarrow C_2$ and $\Delta = \Delta_2 \bowtie \Delta_1$ and $\Omega_R = \Omega_{RL}\Omega_{RR}$ and
$\Gamma; \Delta_2; (\Omega_{LL}B\Omega_{LR}; \Omega_{RR}) \longrightarrow C_2 \gg P$ and $\Gamma; \Delta_1; \Omega_{RL} \longrightarrow C_1$.

By ind. hyp. we know $\Gamma; \Delta_2 \bowtie \Delta_A; (\Omega_{LL}\Omega_A(A \rightarrowtail B)\Omega_{LR}; \Omega_{RR}) \longrightarrow C_2 \gg P$
Then $\Gamma; \Delta \bowtie \Delta_A; (\Omega_{LL}\Omega_A(A \rightarrowtail B)\Omega_{LR}; \Omega_{RL}\Omega_{RR}) \longrightarrow C_1 \twoheadrightarrow C_2 \gg P$.

cases 4,5: $C = C_1 \multimap C_2$, $C = C_1 \rightarrow C_2$.
Similar to previous case.

part 3: Assume $\Omega_L; \Omega_{RL}B\Omega_{RR} \longrightarrow C \gg P$.
Then can be proven with symmetric reasoning to part 2.

□

Lemma 7 (Inversion) *The following all hold:*

1. $\Gamma; \Delta; \Omega_L B \Omega_R \longrightarrow P$ *and* $\Gamma; \Delta_A; \Omega_A \longrightarrow A$ *imply* $\Gamma; \Delta \bowtie \Delta_A; \Omega_L\Omega_A(A \rightarrowtail B)\Omega_R \longrightarrow P$.
2. $\Gamma; \Delta; \Omega_L B \Omega_R \longrightarrow P$ *and* $\Gamma; \Delta_A; \Omega_A \longrightarrow A$ *imply* $\Gamma; \Delta \bowtie \Delta_A; \Omega_L\Omega_A(A \twoheadrightarrow B)\Omega_R \longrightarrow P$.
3. $\Gamma; \Delta; \Omega_L B \Omega_R \longrightarrow P$ *and* $\Gamma; \Delta_A; \cdot \longrightarrow A$ *imply* $\Gamma; \Delta \bowtie \Delta_A; \Omega_L(A \multimap B)\Omega_R \longrightarrow P$.
4. $\Gamma; \Delta; \Omega_L B \Omega_R \longrightarrow P$ *and* $\Gamma; \cdot; \cdot \longrightarrow A$ *imply* $\Gamma; \Delta; \Omega_L(A \rightarrow B)\Omega_R \longrightarrow P$.
5. $\Gamma; \Delta_L\Delta_R; \Omega_L A \Omega_R \longrightarrow P$ *implies* $\Gamma; \Delta_L A \Delta_R; \Omega_L\Omega_R \longrightarrow P$.
6. $\Gamma_L A \Gamma_R; \Delta_L A \Delta_R; \Omega \longrightarrow P$ *implies* $\Gamma_L A \Gamma_R; \Delta_L\Delta_R; \Omega \longrightarrow P$.

Proof: Part 1 is immediate from the previous lemma. The other parts are similarly proved. □

We may now easily prove the completeness result.

Theorem 8 (Completeness of Uniform Derivations) *:*
If $\Gamma; \Delta; \Omega \Longrightarrow A$ *then* $\Gamma; \Delta; \Omega \longrightarrow A$.

Proof: By induction on the structure of the given judgement:

case: $\dfrac{}{\Gamma; \cdot; P \Longrightarrow P}\,\textbf{init}$ then $\dfrac{\dfrac{}{\Gamma; \cdot; (\cdot; \cdot) \longrightarrow P \gg P}\,\textbf{init}}{\Gamma; \cdot; P \longrightarrow P}\,\textbf{choice}_\Omega$

case: $\dfrac{\Gamma; \Delta; A\Omega \Longrightarrow B}{\Gamma; \Delta; \Omega \Longrightarrow A \rightarrowtail B} \rightarrowtail_R$

By ind. hyp. we know $\Gamma; \Delta; A\Omega \longrightarrow B$. Then $\Gamma; \Delta; \Omega \longrightarrow A \rightarrowtail B$
Note we need only consider this derivation of the judgement by lemma 5.

cases: $\twoheadrightarrow_R$,$\multimap_R$,$\rightarrow_R$ all similar to previous case.

case: $$\frac{\Gamma; \Delta_B; \Omega_L B \Omega_R \implies P \qquad \Gamma; \Delta_A; \Omega_A \implies A}{\Gamma; \Delta_B \bowtie \Delta_A; \Omega_L \Omega_A (A \rightarrowtail B) \Omega_R \implies P} \rightarrowtail_L$$

by ind. hyp. we know $\Gamma; \Delta_B; \Omega_L B \Omega_R \longrightarrow P$ and $\Gamma; \Delta_A; \Omega_A \longrightarrow A$.
Then by lemma 7, we know $\Gamma; \Delta_B \bowtie \Delta_A; \Omega_L \Omega_A (A \rightarrowtail B) \Omega_R \longrightarrow P$.

cases: $\twoheadrightarrow_L$,$\multimap_L$,$\rightarrow_L$,**copy**,**place** all similar to previous case.

□

We have now shown that INCLL qualifies as an abstract logic programming language in the sense of [MNPS91]. However, uniform derivations as given above are not yet suitable for a logic programming interpreter, since there is an enormous amount of non-determinism in the system arising from the need to split the linear and ordered contexts in various left rules.

4 Ordered Resource Management

There are several sources of non-determinism in uniform derivations which must be resolved in order to obtain a predictable operational behavior. Fortunately, standard solutions suffice for most of them. The selection of hypothesis implicit in the **choice** rules is resolved by scanning the context in a fixed order and backtracking. The selection of subgoals in the rules with two premises proceeds from left to right. When universal quantifiers are added, their instantiation is postponed and the **init** rule performs unification.

What remains are issues of resource management. Unrestricted hypotheses are propagated to all subgoals without difficulty. Linear hypotheses can be treated as in the so-called IO system of Hodas and Miller [HM94]. The rules $\multimap_L$, $\twoheadrightarrow_L$, and $\rightarrowtail_L$ propagate all linear hypotheses to the first premise which returns the list of unused hypotheses when it has been solved successfully. These are then passed on to the second premise. The hypotheses used in neither premise are then returned as unused in the conclusion.

This model of deterministic *resource consumption* is intuitively attractive and easy to reason about for the programmer. However, its extension to the ordered context requires some care in order to maintain the ordering constraints on the ordered context splits in the **choice** and left rules. Our system requires three different judgements. We shall motivate the development of the three types of judgements in subsection 4.1. We then give the complete resource management system in subsection 4.2.

4.1 Development of Ordered Resource Management System

For the main judgment of uniform derivability, adding input and output contexts is straightforward.

$$\Gamma; \Delta_I \backslash \Delta_O; \Omega_I \backslash \Omega_O \longrightarrow A$$

During the search, the *input contexts* Γ, Δ_I, and Ω_I and the goal A are given, while the *output contexts* Δ_O and Ω_O are returned. In the interest of economy (both for the presentation of the rules and the implementation) we do not actually delete formulas from Δ_I and Ω_I but replace them with a placeholder $\Box$. For the remainder of this paper, contexts may contain formulas and placeholders. In order to state the invariants relating input and output contexts we define the context difference $\Omega_I - \Omega_O$:

$$\begin{array}{rcl}
\cdot - \cdot & = & \cdot \\
\Omega_I A - \Omega_O A & = & \Omega_I - \Omega_O \\
\Omega_I \Box - \Omega_O \Box & = & \Omega_I - \Omega_O \\
\Omega_I A - \Omega_O \Box & = & (\Omega_I - \Omega_O) A \\
\Omega_I \Box - \Omega_O A & = & \text{undefined}
\end{array}$$

$\Omega_I - \Omega_O$ is also undefined when $\|\Omega_I\| \neq \|\Omega_O\|$ where $\|\Omega\|$ denotes the length of list Ω. The resource management judgements are constructed so that the context difference $\Omega_I - \Omega_O$ and $\Delta_I - \Delta_O$ is always defined in valid derivations.

The right rules for the ordered resource management judgment are easy to construct.

$$\frac{\Gamma A \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash \Omega_O \;\longrightarrow\; B}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash \Omega_O \;\longrightarrow\; A \to B} \to_R \qquad \frac{\Gamma \,;\, \Delta_I A \backslash \Delta_O \Box \,;\, \Omega_I \backslash \Omega_O \;\longrightarrow\; B}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash \Omega_O \;\longrightarrow\; A \multimap B} \multimap_R$$

$$\frac{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, A\Omega_I \backslash \Box\Omega_O \;\longrightarrow\; B}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash \Omega_O \;\longrightarrow\; A \rightarrowtail B} \rightarrowtail_R \qquad \frac{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I A \backslash \Omega_O \Box \;\longrightarrow\; B}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash \Omega_O \;\longrightarrow\; A \twoheadrightarrow B} \twoheadrightarrow_R$$

We require the $\Box$ in the output contexts for linear and ordered implications to make sure the hypothesis has actually been used.

Next we come to the $\mathbf{choice}_\Omega$ rule, that is, we chose to focus on an ordered assumption. This determines the division of the remaining ordered hypotheses unambiguously. We therefore divide the input contexts and join the output contexts at the chosen assumption. The new judgment reads

$$\Gamma; \Delta_I \backslash \Delta_O; (\Omega_{LI} \backslash \Omega_{LO} \,;\, \Omega_{RI} \backslash \Omega_{RO}) \;\longrightarrow\; A \gg P$$

where Ω_{LI} and Ω_{RI} are the parts to the left and right of the focussed formula A, and Ω_{LO} and Ω_{RO} are the corresponding output contexts. The $\mathbf{choice}_\Omega$ rule for this system then looks as follows:

$$\frac{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, (\Omega_{LI} \backslash \Omega_{LO} \,;\, \Omega_{RI} \backslash \Omega_{RO}) \;\longrightarrow\; A \gg P}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_{LI} A \Omega_{RI} \backslash \Omega_{LO} \Box \Omega_{RO} \;\longrightarrow\; P} \mathbf{choice}_\Omega$$

Replacing A from the input context with $\Box$ in the output context indicates that A was consumed. We postpone dealing with the other choice rules.

The $\mathbf{init}_1$ rule does not consume any resources except for the focus formula. Therefore all input resources are passed on.

$$\frac{}{\Gamma; \Delta \backslash \Delta; (\Omega_L \backslash \Omega_L; \Omega_R \backslash \Omega_R) \longrightarrow P \gg P} \mathbf{init}_1$$

This effectively states that the linear and ordered contexts of the initial sequent should be empty. The unrestricted and linear left rules for this judgment, $\to_{L1}$ and $\multimap_{L1}$ introduce no new ideas.

We now focus on the left rule for right implication, which means we are trying to derive a judgment of the form

$$(\Omega_{LI} \backslash \mathbf{?} \,;\, \Omega_{RI} \backslash \mathbf{?}) \longrightarrow A \twoheadrightarrow B \gg P$$

where we have omitted the distracting, but trivial unrestricted and linear hypotheses, and the output contexts denoted by **?** have yet to be computed. Because $A \twoheadrightarrow B$ is a right implication situated between Ω_{LI} and Ω_{RI}, the derivation of A must consume some initial segment of Ω_{RI}. Before that, we need to see if B immediately entails P (the left premise of the $\twoheadrightarrow_L$ rule)[1] which we obtain from

$$(\Omega_{LI}\backslash\Omega_{LO}\ ;\ \Omega_{RI}\backslash\Omega) \longrightarrow B \gg P$$

Then we need to take the unconsumed parts at the left end of Ω, denoted by Ω_{AI} (we shall denote the remainder as Ω_{RO} so $\Omega = \Omega_{AI}\Omega_{RO}$), and allow them as the input context for the solution to A.

$$\Omega_{AI}\backslash\Omega_{AO} \longrightarrow A$$

Now we can fill the holes in the conclusion with Ω_{LO} and $\Omega_{AO}\Omega_{RO}$, respectively. In summary, the rule reads

$$\frac{(\Omega_{LI}\backslash\Omega_{LO}\ ;\ \Omega_{RI}\backslash\Omega)\ \longrightarrow\ B \gg P \qquad \Omega_{AI}\backslash\Omega_{AO}\ \longrightarrow\ A}{(\Omega_{LI}\backslash\Omega_{LO}\ ;\ \Omega_{RI}\backslash\Omega_{AO}\Omega_{RO})\ \longrightarrow\ A \twoheadrightarrow B \gg P}\twoheadrightarrow_{L1}$$

where Ω_{AI} is the longest prefix of Ω not containing $\Box$, and Ω_{RO} the remainder (so $\Omega = \Omega_{AI}\Omega_{RO}$).

The left rule for left ordered implication is symmetric.

A difficulty remains, however, in that when a formula is directly chosen from the unrestricted or linear context, its exact position in the ordered context is undetermined. As before, we would like the **choice** rule applications in the derivation of the premise to implicitly determine where the formula might have been placed. This is captured in the judgment

$$\Gamma\ ;\ \Delta_I\backslash\Delta_O\ ;\ \Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ A \gg P$$

where $\Omega_I - \Omega_L\Omega_M\Omega_R$ is defined and Ω_M does not contain any $\Box$. Since no formula in Ω_M is actually consumed in the derivation of $A \gg P$, the whole subcontext marks the place where A occurs in the ordered context in the sequent calculus.

The unrestricted choice rule

$$\frac{\Gamma_L A \Gamma_R\ ;\ \Delta_I\backslash\Delta_O\ ;\ \Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ A \gg P}{\Gamma_L A \Gamma_R\ ;\ \Delta_I\backslash\Delta_O\ ;\ \Omega_I\backslash\Omega_L\Omega_M\Omega_R\ \longrightarrow\ P}\mathbf{choice}_\Gamma$$

then just passes the whole input context Ω_I. The $\mathbf{choice}_\Delta$ rule is similar.

We need corresponding initial and left rules for this second type of immediate entailment. The initial rule, $\mathbf{init}_2$, is easy to construct. Since it consumes no resources, it must place all the ordered resources in the middle slot of the output context.

$$\frac{}{\Gamma;\Delta\backslash\Delta;\Omega\backslash(\cdot|\Omega|\cdot) \longrightarrow P \gg P}\mathbf{init}_2$$

Then we apply similar reasoning as above to deduce the correct form of the left rules. We need to derive a judgement of the form (again ignoring the unrestricted and linear contexts)

$$\Omega_I\backslash(\mathbf{?}|\mathbf{?}|\mathbf{?}) \longrightarrow A \twoheadrightarrow B \gg P$$

Again we first need to see if B immediately entails P

$$\Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R) \longrightarrow B \gg P$$

[1] In Prolog terminology: we need to unify the clause head with P before solving any subgoals.

Then we need to take the unconsumed parts of the output, Ω_M, and allow them as the input context for the solution to A.

$$\Omega_M \backslash \Omega_{MO} \longrightarrow A$$

Now we know that $A \twoheadrightarrow B$ must be placed to the left of any resources consumed by the derivation of A. This observation is all we need to complete the output context. The complete rule reads

$$\frac{\Omega_I \backslash (\Omega_L | \Omega_M | \Omega_R) \longrightarrow B \gg P \qquad \Omega_M \backslash \Omega_{MO} \longrightarrow A}{\Omega_I \backslash (\Omega_L | \Omega_{ML} | \Omega_{MR} \Omega_R) \longrightarrow A \twoheadrightarrow B \gg P} \twoheadrightarrow_{L2}$$

where Ω_{ML} is the longest prefix of Ω_{MO} not containing $\Box$, and Ω_{MR} the remainder (so $\Omega_{MO} = \Omega_{ML}\Omega_{MR}$).

Again, the left rule for left ordered implication is symmetric.

To summarize, our context management system is based on three judgments.

$$\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash \Omega_O \longrightarrow A$$

$$\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, (\Omega_{LI} \backslash \Omega_{LO} \,;\, \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow A \gg P$$

$$\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash (\Omega_L | \Omega_M | \Omega_R) \longrightarrow A \gg P$$

where contexts Δ and Ω may contain $\Box$ as placeholder for a consumed formula. The first two judgments mirror the behavior of the uniform sequents where the ordered context split is always known. The third sequent is used when a focus formula is chosen from the intuitionistic or linear contexts and the splitting of the ordered context is to be determined lazily.

4.2 Ordered Resource Management Judgements

Here is the complete resource management system:

$$\frac{\Gamma A \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash \Omega_O \longrightarrow B}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash \Omega_O \longrightarrow A \rightarrow B} \rightarrow_R \qquad \frac{\Gamma \,;\, \Delta_I A \backslash \Delta_O \Box \,;\, \Omega_I \backslash \Omega_O \longrightarrow B}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash \Omega_O \longrightarrow A \multimap B} \multimap_R$$

$$\frac{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, A\Omega_I \backslash \Box \Omega_O \longrightarrow B}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash \Omega_O \longrightarrow A \rightarrowtail B} \rightarrowtail_R \qquad \frac{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I A \backslash \Omega_O \Box \longrightarrow B}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_I \backslash \Omega_O \longrightarrow A \twoheadrightarrow B} \twoheadrightarrow_R$$

$$\frac{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, (\Omega_{LI} \backslash \Omega_{LO} \,;\, \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow A \gg P}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, \Omega_{LI} A \Omega_{RI} \backslash \Omega_{LO} \Box \Omega_{RO} \longrightarrow P} \mathbf{choice}_\Omega$$

$$\frac{}{\Gamma \,;\, \Delta \backslash \Delta \,;\, (\Omega_L \backslash \Omega_L \,;\, \Omega_R \backslash \Omega_R) \longrightarrow P \gg P} \mathbf{init}_1$$

$$\frac{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, (\Omega_{LI} \backslash \Omega_{LO} \,;\, \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow B \gg P \qquad \Gamma \,;\, \cdot \backslash \cdot \,;\, \cdot \backslash \cdot \longrightarrow A}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, (\Omega_{LI} \backslash \Omega_{LO} \,;\, \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow A \rightarrow B \gg P} \rightarrow_{L1}$$

$$\frac{\Gamma \,;\, \Delta_I \backslash \Delta_M \,;\, (\Omega_{LI} \backslash \Omega_{LO} \,;\, \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow B \gg P \qquad \Gamma \,;\, \Delta_M \backslash \Delta_O \,;\, \cdot \backslash \cdot \longrightarrow A}{\Gamma \,;\, \Delta_I \backslash \Delta_O \,;\, (\Omega_{LI} \backslash \Omega_{LO} \,;\, \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow A \multimap B \gg P} \multimap_{L1}$$

$$\frac{\Gamma\,;\,\Delta_I\backslash\Delta_M\,;\,(\Omega_{LI}\backslash\Omega_{LO}\,;\,\Omega_{RI}\backslash\Omega_{AI}\Omega_{RO})\ \longrightarrow\ B \gg P \qquad \Gamma\,;\,\Delta_M\backslash\Delta_O\,;\,\Omega_{AI}\backslash\Omega_{AO}\ \longrightarrow\ A}{\Gamma\,;\,\Delta_I\backslash\Delta_O\,;\,(\Omega_{LI}\backslash\Omega_{LO}\,;\,\Omega_{RI}\backslash\Omega_{AO}\Omega_{RO})\ \longrightarrow\ A \twoheadrightarrow B \gg P}\twoheadrightarrow_{L1}$$

where Ω_{AI} contains no occurrence of $\Box$ and $\Omega_{RO} = \cdot$ or $\Omega_{RO} = \Box\Omega_{RO1}$ (i.e. if $\Omega = \Omega_{AI}\Omega_{RO}$ then Ω_{AI} is the leftmost portion of Ω not containing $\Box$.)

$$\frac{\Gamma\,;\,\Delta_I\backslash\Delta_M\,;\,(\Omega_{LI}\backslash\Omega_{LO}\Omega_{AI}\,;\,\Omega_{RI}\backslash\Omega_{RO})\ \longrightarrow\ B \gg P \qquad \Gamma\,;\,\Delta_M\backslash\Delta_O\,;\,\Omega_{AI}\backslash\Omega_{AO}\ \longrightarrow\ A}{\Gamma\,;\,\Delta_I\backslash\Delta_O\,;\,(\Omega_{LI}\backslash\Omega_{LO}\Omega_{AO}\,;\,\Omega_{RI}\backslash\Omega_{RO})\ \longrightarrow\ A \rightarrowtail B \gg P}\rightarrowtail_{L1}$$

where Ω_{AI} contains no occurrence of $\Box$ and $\Omega_{LO} = \cdot$ or $\Omega_{LO} = \Omega_{LO1}\Box$ (i.e. if $\Omega = \Omega_{LO}\Omega_{AI}$ then Ω_{AI} is the rightmost portion of Ω not containing $\Box$.)

$$\frac{\Gamma_L A \Gamma_R\,;\,\Delta_I\backslash\Delta_O\,;\,\Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ A \gg P}{\Gamma_L A \Gamma_R\,;\,\Delta_I\backslash\Delta_O\,;\,\Omega_I\backslash\Omega_L\Omega_M\Omega_R\ \longrightarrow\ P}\mathbf{choice}_\Gamma$$

$$\frac{\Gamma\,;\,\Delta_{LI}\Delta_{RI}\backslash\Delta_{LO}\Delta_{RO}\,;\,\Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ A \gg P}{\Gamma\,;\,\Delta_{LI}A\Delta_{RI}\backslash\Delta_{LO}\Box\Delta_{RO}\,;\,\Omega_I\backslash\Omega_L\Omega_M\Omega_R\ \longrightarrow\ P}\mathbf{choice}_\Delta\ (\|\Delta_{LI}\| = \|\Delta_{LO}\|)$$

$$\frac{}{\Gamma\,;\,\Delta\backslash\Delta\,;\,\Omega\backslash(\cdot|\Omega|\cdot)\ \longrightarrow\ P \gg P}\mathbf{init}_2$$

$$\frac{\Gamma\,;\,\Delta_I\backslash\Delta_O\,;\,\Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ B \gg P \qquad \Gamma\,;\,\cdot\backslash\cdot\,;\,\cdot\backslash\cdot\ \longrightarrow\ A}{\Gamma\,;\,\Delta_I\backslash\Delta_O\,;\,\Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ A \rightarrow B \gg P}\rightarrow_{L2}$$

$$\frac{\Gamma\,;\,\Delta_I\backslash\Delta_M\,;\,\Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ B \gg P \qquad \Gamma\,;\,\Delta_M\backslash\Delta_O\,;\,\cdot\backslash\cdot\ \longrightarrow\ A}{\Gamma\,;\,\Delta_I\backslash\Delta_O\,;\,\Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ A \multimap B \gg P}\multimap_{L2}$$

$$\frac{\Gamma\,;\,\Delta_I\backslash\Delta_M\,;\,\Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ B \gg P \qquad \Gamma\,;\,\Delta_M\backslash\Delta_O\,;\,\Omega_M\backslash\Omega_{ML}\Omega_{MR}\ \longrightarrow\ A}{\Gamma\,;\,\Delta_I\backslash\Delta_O\,;\,\Omega_I\backslash(\Omega_L|\Omega_{ML}|\Omega_{MR}\Omega_R)\ \longrightarrow\ A \twoheadrightarrow B \gg P}\twoheadrightarrow_{L2}$$

where Ω_{ML} contains no occurrence of $\Box$ and $\Omega_{MR} = \cdot$ or $\Omega_{MR} = \Box\Omega_{MR1}$ (i.e. if $\Omega = \Omega_{ML}\Omega_{MR}$ then Ω_{ML} is the leftmost portion of Ω not containing $\Box$.)

$$\frac{\Gamma\,;\,\Delta_I\backslash\Delta_M\,;\,\Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ B \gg P \qquad \Gamma\,;\,\Delta_M\backslash\Delta_O\,;\,\Omega_M\backslash\Omega_{ML}\Omega_{MR}\ \longrightarrow\ A}{\Gamma\,;\,\Delta_I\backslash\Delta_O\,;\,\Omega_I\backslash(\Omega_L\Omega_{ML}|\Omega_{MR}|\Omega_R)\ \longrightarrow\ A \rightarrowtail B \gg P}\rightarrowtail_{L2}$$

where Ω_{MR} contains no occurrence of $\Box$ and $\Omega_{ML} = \cdot$ or $\Omega_{ML} = \Omega_{ML1}\Box$ (i.e. if $\Omega = \Omega_{ML}\Omega_{MR}$ then Ω_{MR} is the rightmost portion of Ω not containing $\Box$.)

Appendix A contains the example parsing derivation written out using the resource managment system.

We prove the correctness of the resource management system with respect to the uniform system by separately proving a completeness and a soundness result.

Lemma 9 (Resource Invariants) *If* $\Omega_I \backslash \Omega_O$ *or* $\Omega_I \backslash (\Omega_L | \Omega_M | \Omega_R)$ *occurs in a valid derivation then:*

1. $\Omega_I - \Omega_O$ *and* $\Omega_I - \Omega_L \Omega_M \Omega_R$ *are defined.*
2. Ω_M *does not contain* $\Box$.
3. Ω_L *is* $\cdot$ *or ends with* $\Box$.
4. Ω_R *is* $\cdot$ *or begins with* $\Box$.

Theorem 10 (Soundness of Ordered Resource Management)

1. *If* $\Gamma; \Delta_I \backslash \Delta_O; \Omega_I \backslash \Omega_O \longrightarrow A$ *then* $\Gamma; \Delta_I - \Delta_O; \Omega_I - \Omega_O \longrightarrow A$,
2. *if* $\Gamma; \Delta_I \backslash \Delta_O; (\Omega_{LI} \backslash \Omega_{LO};\ \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow B \gg P$
   then $\Gamma; \Delta_I - \Delta_O; (\Omega_{LI} - \Omega_{LO}\ ;\ \Omega_{RI} - \Omega_{RO}) \longrightarrow B \gg P$, *and*
3. *if* $\Gamma; \Delta_I \backslash \Delta_O; \Omega_I \backslash (\Omega_L | \Omega_M | \Omega_R) \longrightarrow A \gg P$
   then $\Gamma; \Delta_I - \Delta_O; (\Omega_{IL} - \Omega_L; \Omega_{IR} - \Omega_R) \longrightarrow A \gg P$,

where $\Omega_I, \Omega_{LI}, \Omega_{RI}$ *do not contain* $\Box$ *and* $\Omega_I = \Omega_{IL}\Omega_M\Omega_{IR}$ *and* $\|\Omega_{IL}\| = \|\Omega_L\|$ *and* $\|\Omega_{IR}\| = \|\Omega_R\|$.

Proof: By mutual induction on the structure of the given derivations. Appeals to lemma 9 and elementary properties of context difference $(-)$ are left implicit.

case: $\dfrac{}{\Gamma; \Delta_I \backslash \Delta_I; (\Omega_L \backslash \Omega_L\ ;\ \Omega_R \backslash \Omega_R) \longrightarrow P \gg P}\,\textbf{init}$ then $\dfrac{}{\Gamma; \cdot; (\cdot\ ;\ \cdot) \longrightarrow P \gg P}\,\textbf{init}$

case: $\dfrac{}{\Gamma; \Delta_I \backslash \Delta_I; \Omega_I \backslash (\cdot | \Omega_I | \cdot) \longrightarrow P \gg P}\,\textbf{init}$ then $\dfrac{}{\Gamma; \cdot; (\cdot\ ;\ \cdot) \longrightarrow P \gg P}\,\textbf{init}$

case: $$\dfrac{\Gamma; \Delta_I \backslash \Delta_M; (\Omega_{LI} \backslash \Omega_{LO}\Omega_{AI}\ ;\ \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow B \gg P \qquad \Gamma; \Delta_M \backslash \Delta_O; \Omega_{AI} \backslash \Omega_{AO} \longrightarrow A}{\Gamma; \Delta_I \backslash \Delta_O; (\Omega_{LI} \backslash \Omega_{LO}\Omega_{AO}\ ;\ \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow A \rightarrowtail B \gg P}\,{\rightarrowtail}_{L1}$$

By assumption, Ω_{AI} does not contain $\Box$.
We know by ind. hyp.:

1. $\Gamma; \Delta_I - \Delta_M; (\Omega_{LI} - \Omega_{LO}\Omega_{AI}\ ;\ \Omega_{RI} - \Omega_{RO}) \longrightarrow B \gg P$
2. $\Gamma; \Delta_M - \Delta_O; \Omega_{AI} - \Omega_{AO} \longrightarrow A$

Then $\Gamma; \Delta_I - \Delta_O; ((\Omega_{LI} - \Omega_{LO}\Omega_{AI})(\Omega_{AI} - \Omega_{AO})\ ;\ \Omega_{RI} - \Omega_{RO}) \longrightarrow A \rightarrowtail B \gg P$.
We will now show $\Omega_{LI} - \Omega_{LO}\Omega_{AO} = (\Omega_{LI} - \Omega_{LO}\Omega_{AI})(\Omega_{AI} - \Omega_{AO})$.
We know from the hypothesis that $\Omega_{LI} = \Omega_{LIL}\Omega_{AI}$.
Then $\Omega_{LI} - \Omega_{LO}\Omega_{AO} = \Omega_{LIL}\Omega_{AI} - \Omega_{LO}\Omega_{AO} = (\Omega_{LIL} - \Omega_{LO})(\Omega_{AI} - \Omega_{AO})$.
Then $(\Omega_{LIL} - \Omega_{LO})(\Omega_{AI} - \Omega_{AO}) = (\Omega_{LIL}\Omega_{AI} - \Omega_{LO}\Omega_{AI})(\Omega_{AI} - \Omega_{AO}) = (\Omega_{LI} - \Omega_{LO}\Omega_{AI})(\Omega_{AI} - \Omega_{AO})$.
Then $\Gamma; \Delta_I - \Delta_O; (\Omega_{LI} - \Omega_{LO}\Omega_{AO}\ ;\ \Omega_{RI} - \Omega_{RO}) \longrightarrow A \rightarrowtail B \gg P$

cases: ${\twoheadrightarrow}_{L1}, {\multimap}_{L1}, {\rightarrow}_{L1}$
Similar to previous case.

case:
$$\frac{\Gamma;\Delta_I\backslash\Delta_M;\Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ B\gg P \qquad \Gamma;\Delta_M\backslash\Delta_O;\Omega_M\backslash\Omega_{ML}\Omega_{MR}\ \longrightarrow\ A}{\Gamma;\Delta_I\backslash\Delta_O;\Omega_I\backslash(\Omega_L\Omega_{ML}|\Omega_{MR}|\Omega_R)\ \longrightarrow\ A\rightarrowtail B\gg P}\rightarrowtail_{L2}$$

By assumption, Ω_M,Ω_{MR} do not contain $\Box$.
Let $\Omega_{M_{ML}}\Omega_{MR}=\Omega_M$ (then $\|\Omega_{M_{ML}}\|=\|\Omega_{ML}\|$).
We know by ind. hyp.:

1. $\Gamma;\Delta_I-\Delta_M;(\Omega_{IL}-\Omega_L;\Omega_{IR}-\Omega_R)\ \longrightarrow\ B\gg P$
2. $\Gamma;\Delta_M-\Delta_O;\Omega_M-\Omega_{ML}\Omega_{MR}\ \longrightarrow\ A$

where $\Omega_{IL}\Omega_M\Omega_{IR}=\Omega_I$ and $\|\Omega_{IL}\|=\|\Omega_L\|$.
Then $\Gamma;\Delta_I-\Delta_O;((\Omega_{IL}-\Omega_L)(\Omega_M-\Omega_{ML}\Omega_{MR});\Omega_{IR}-\Omega_R)\ \longrightarrow\ A\rightarrowtail B\gg P$.
And $\Omega_M-\Omega_{ML}\Omega_{MR}=\Omega_{M_{ML}}\Omega_{MR}-\Omega_{ML}\Omega_{MR}=\Omega_{M_{ML}}-\Omega_{ML}$.
Then $\Gamma;\Delta_I-\Delta_O;(\Omega_{IL}\Omega_{M_{ML}}-\Omega_L\Omega_{ML};\Omega_{IR}-\Omega_R)\ \longrightarrow\ A\rightarrowtail B\gg P$.
And $\Omega_I=\Omega_{IL}\Omega_{M_{ML}}\Omega_{MR}\Omega_{IR}$.

cases: $\twoheadrightarrow_{L2}$, $\multimap_{L2}$, $\rightarrow_{L2}$
Similar to previous case.

case:
$$\frac{\Gamma;\Delta_I\backslash\Delta_O;A\Omega_I\backslash\Box\Omega_O\ \longrightarrow\ B}{\Gamma;\Delta_I\backslash\Delta_O;\Omega_I\backslash\Omega_O\ \longrightarrow\ A\rightarrowtail B}\rightarrowtail_R$$

We know by ind. hyp. that $\Gamma;\Delta_I-\Delta_O;A\Omega_I-\Box\Omega_O\ \longrightarrow\ B$.
Then $\Gamma;\Delta_I-\Delta_O;A(\Omega_I-\Omega_O)\ \longrightarrow\ B$.
Then $\Gamma;\Delta_I-\Delta_O;\Omega_I-\Omega_O\ \longrightarrow\ A\rightarrowtail B$.

cases: $\twoheadrightarrow_R$, $\multimap_R$, $\rightarrow_R$
Similar reasoning as previous case.

case:
$$\frac{\Gamma;\Delta_I\backslash\Delta_O;(\Omega_{LI}\backslash\Omega_{LO}\ ;\ \Omega_{RI}\backslash\Omega_{RO})\ \longrightarrow\ A\gg P}{\Gamma;\Delta_I\backslash\Delta_O;\Omega_{LI}A\Omega_{RI}\backslash\Omega_{LO}\Box\Omega_{RO}\ \longrightarrow\ P}\,\mathbf{choice}_\Omega$$

We know by ind. hyp. $\Gamma;\Delta_I-\Delta_O;(\Omega_{LI}-\Omega_{LO};\Omega_{RI}-\Omega_{RO})\ \longrightarrow\ A\gg P$
Then $\Gamma;\Delta_I-\Delta_O;(\Omega_{LI}-\Omega_{LO})A(\Omega_{RI}-\Omega_{RO})\ \longrightarrow\ P$.
Then $\Gamma;\Delta_I-\Delta_O;\Omega_{LI}A\Omega_{RI}-\Omega_{LO}\Box\Omega_{RO}\ \longrightarrow\ P$.

case:
$$\frac{\Gamma;\Delta_{LI}\Delta_{RI}\backslash\Delta_{LO}\Delta_{RO};\Omega_I\backslash(\Omega_L|\Omega_M|\Omega_R)\ \longrightarrow\ A\gg P}{\Gamma;\Delta_{LI}A\Delta_{RI}\backslash\Delta_{LO}\Box\Delta_{RO};\Omega_I\backslash\Omega_L\Omega_M\Omega_R\ \longrightarrow\ P}\,\mathbf{choice}_\Delta$$

Let $\Omega_{IL}\Omega_M\Omega_{IR}=\Omega_I$ where $\|\Omega_{IL}\|=\|\Omega_L\|$.
We know by ind. hyp. $\Gamma;\Delta_{LI}\Delta_{RI}-\Delta_{LO}\Delta_{RO};(\Omega_{IL}-\Omega_L;\Omega_{IR}-\Omega_R)\ \longrightarrow\ A\gg P$
Then $\Gamma;\Delta_{LI}A\Delta_{RI}-\Delta_{LO}\Box\Delta_{RO};(\Omega_{IL}-\Omega_L)(\Omega_{IR}-\Omega_R)\ \longrightarrow\ P$.

Then $\Gamma; \Delta_{LI} A \Delta_{RI} - \Delta_{LO} \Box \Delta_{RO}; \Omega_{IL} \Omega_M \Omega_{IR} - \Omega_L \Omega_M \Omega_R \longrightarrow P$.

case: $$\frac{\Gamma_L A \Gamma_R; \Delta_I \backslash \Delta_O; \Omega_I \backslash (\Omega_L | \Omega_M | \Omega_R) \longrightarrow A \gg P}{\Gamma_L A \Gamma_R; \Delta_I \backslash \Delta_O; \Omega_I \backslash \Omega_L \Omega_M \Omega_R \longrightarrow P}\ \mathbf{choice}_\Gamma$$

Similar to previous case.

$\Box$

Theorem 11 (Completeness of Ordered Resource Management)

1. *For all* $\Delta_I, \Delta_O, \Omega_I, \Omega_O$ *such that* $\Delta_I - \Delta_O = \Delta$ *and* $\Omega_I - \Omega_O = \Omega$ *we have that* $\Gamma; \Delta; \Omega \longrightarrow A$ *implies* $\Gamma; \Delta_I \backslash \Delta_O; \Omega_I \backslash \Omega_O \longrightarrow A$.

2. *For all* $\Delta_I, \Delta_O, \Omega_{LI}, \Omega_{LO}, \Omega_{RI}, \Omega_{RO}$ *such that* $\Delta_I - \Delta_O = \Delta$, $\Omega_{LI} - \Omega_{LO} = \Omega_L$ *and* $\Omega_{RI} - \Omega_{RO} = \Omega_R$ *we have that* $\Gamma; \Delta; (\Omega_L; \Omega_R) \longrightarrow A \gg P$ *implies* $\Gamma; \Delta_I \backslash \Delta_O; (\Omega_{LI} \backslash \Omega_{LO}; \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow A \gg P$ *where* $\Omega_I, \Omega_{LI}, \Omega_{RI}$ *do not contain* $\Box$.

3. *For all* $\Delta_I, \Delta_O, \Omega_{LI}, \Omega_{LO}, \Omega_{RI}, \Omega_{RO}, \Omega_M$ *such that* $\Delta_I - \Delta_O = \Delta$, $\Omega_{LI} - \Omega_{LO} = \Omega_L$ *and* $\Omega_{RI} - \Omega_{RO} = \Omega_R$ *we have that* $\Gamma; \Delta; (\Omega_L; \Omega_R) \longrightarrow A \gg P$ *implies* $\Gamma; \Delta_I \backslash \Delta_O; \Omega_{LI} \Omega_M \Omega_{RI} \backslash (\Omega_{LO} | \Omega_M | \Omega_{RO}) \longrightarrow A \gg P$ *where* $\Omega_I, \Omega_M, \Omega_{LI}, \Omega_{RI}$ *do not contain* $\Box$, Ω_{LO} *ends in* $\Box$ *or is* $\cdot$, *and* Ω_{RO} *begins with* $\Box$ *or is* $\cdot$.

Proof: By mutual induction on the structure of the given derivations. Appeals to lemma 9 and elementary properties of context difference $(-)$ are left implicit.

case: $$\frac{\Gamma; \Delta; A\Omega \longrightarrow B}{\Gamma; \Delta; \Omega \longrightarrow A \rightarrowtail B}\ \rightarrowtail_R$$

Let $\Delta_I - \Delta_O = \Delta$ and $\Omega_I - \Omega_O = \Omega$ where Ω_I does not contain $\Box$.
Then, by ind. hyp., $\Gamma; \Delta_I \backslash \Delta_O; A\Omega_I \backslash \Box \Omega_O \longrightarrow B$
Then $\Gamma; \Delta_I \backslash \Delta_O; \Omega_I \backslash \Omega_O \longrightarrow A \rightarrowtail B$.

cases: $\twoheadrightarrow_R, \multimap_R, \rightarrow_R$
Similar to previous case.

case: $$\frac{\Gamma; \Delta; (\Omega_L\ ;\ \Omega_R) \longrightarrow A \gg P}{\Gamma; \Delta; \Omega_L A \Omega_R \longrightarrow P}\ \mathbf{choice}_\Omega$$

Let $\Delta_I - \Delta_O = \Delta$ and $\Omega_I - \Omega_O = \Omega_L A \Omega_R$ where Ω_I does not contain $\Box$.
Then $\Omega_I = \Omega_{IL} A \Omega_{IR}$ and $\Omega_O = \Omega_{OL} \Box \Omega_{OR}$ where $\Omega_{IL} - \Omega_{OL} = \Omega_L$ and $\Omega_{IR} - \Omega_{OR} = \Omega_R$.
We know by ind. hyp. $\Gamma; \Delta_I \backslash \Delta_O; (\Omega_{IL} \backslash \Omega_{OL}; \Omega_{IR} \backslash \Omega_{OR}) \longrightarrow A \gg P$.

Then $\Gamma; \Delta_I \backslash \Delta_O; \Omega_{IL} A \Omega_{IR} \backslash \Omega_{OL} \Box \Omega_{OR} \longrightarrow P$.
Then $\Gamma; \Delta_I \backslash \Delta_O; \Omega_I \backslash \Omega_O \longrightarrow P$.

case: $$\frac{\Gamma; \Delta_L \Delta_R; (\Omega_L \; ; \; \Omega_R) \longrightarrow A \gg P}{\Gamma; \Delta_L A \Delta_R; \Omega_L \Omega_R \longrightarrow P}\,\textbf{choice}_\Delta$$

Let $\Delta_I - \Delta_O = \Delta_L A \Delta_R$ and $\Omega_I - \Omega_O = \Omega_L \Omega_R$ where Ω_I does not contain $\Box$.
Then $\Delta_I = \Delta_{IL} A \Delta_{IR}$ and $\Delta_O = \Delta_{OL} \Box \Delta_{OR}$ where $\Delta_{IL} - \Delta_{OL} = \Delta_L$ and $\Delta_{IR} - \Delta_{OR} = \Delta_R$.
Then $\Omega_I = \Omega_{IL} \Omega_M \Omega_{IR}$ and $\Omega_O = \Omega_{OL} \Omega_M \Omega_{OR}$ where $\Omega_{IL} - \Omega_{OL} = \Omega_L$ and $\Omega_{IR} - \Omega_{OR} = \Omega_R$.
We know by ind. hyp. $\Gamma; \Delta_{IL} \Delta_{IR} \backslash \Delta_{OL} \Delta_{OR}; \Omega_I \backslash (\Omega_{OL} | \Omega_M | \Omega_{OR}) \longrightarrow A \gg P$.
Then $\Gamma; \Delta_{IL} A \Delta_{IR} \backslash \Delta_{OL} \Box \Delta_{OR}; \Omega_I \backslash \Omega_{OL} \Omega_M \Omega_{OR} \longrightarrow P$.
Then $\Gamma; \Delta_I \backslash \Delta_O; \Omega_I \backslash \Omega_O \longrightarrow P$.

case: $\textbf{choice}_\Gamma$ is similar to previous.

case: $$\frac{\Gamma; \Delta_B; (\Omega_L \; ; \; \Omega_R) \longrightarrow B \gg P \qquad \Gamma; \Delta_A; \Omega_A \longrightarrow A}{\Gamma; \Delta_B \bowtie \Delta_A; (\Omega_L \Omega_A \; ; \; \Omega_R) \longrightarrow A \rightarrowtail B \gg P}\rightarrowtail_L$$

Let $\Delta_I - \Delta_O = \Delta_B \bowtie \Delta_A$.
Let Δ_M be a list of size $\|\Delta_I\|$ where (if $(\Delta_M)_i \in \Delta_B$ then $(\Delta_M)_i = \Box$ else $(\Delta_M)_i = (\Delta_I)_i$) and the number of $\Box$ in Δ_M is $\|\Delta_B\|$.
Then $\Delta_I - \Delta_M = \Delta_B$ and $\Delta_M - \Delta_O = \Delta_A$.
Let $\Omega_{LI} - \Omega_{LO} = \Omega_L \Omega_A$ and $\Omega_{RI} - \Omega_{RO} = \Omega_R$ where Ω_{LI}, Ω_{RI} do not contain $\Box$.
Let $\Omega_{LIL} \Omega_{LIA} = \Omega_{LI}$ and $\Omega_{LOL} \Omega_{LOA} = \Omega_{LO}$ where $\Omega_{LIL} - \Omega_{LOL} = \Omega_L$ and $\Omega_{LIA} - \Omega_{LOA} = \Omega_A$ and $\Omega_{LOL} = \Omega_{LOL1} \Box$ (if $\Omega_L \neq \cdot$ else $\Omega_{LOL} = \cdot$).
We know by ind. hyp.:

1. $\Gamma; \Delta_I \backslash \Delta_M; (\Omega_{LI} \backslash \Omega_{LOL} \Omega_{LIA}; \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow B \gg P$.
2. $\Gamma; \Delta_M \backslash \Delta_O; \Omega_{LIA} \backslash \Omega_{LOA} \longrightarrow A$

Then $\Gamma; \Delta_I \backslash \Delta_O; (\Omega_{LI} \backslash \Omega_{LOL} \Omega_{LOA}; \Omega_{RI} \backslash \Omega_{RO}) \longrightarrow A \rightarrowtail B \gg P$.

Let $\Delta_I, \Delta_O, \Delta_M$ be defined as above.
Let Ω_* also be defined as above except that Ω_{LO} ends in $\Box$ or is $\cdot$, and Ω_{RO} begins with $\Box$ or is $\cdot$.
Let Ω_M be a list not containing $\Box$.
Then we know by ind. hyp.:

1. $\Gamma; \Delta_I \backslash \Delta_M; \Omega_{LIL} \Omega_{LIA} \Omega_M \Omega_{RI} \backslash (\Omega_{LOL} | \Omega_{LIA} \Omega_M | \Omega_{RO}) \longrightarrow B \gg P$.
2. $\Gamma; \Delta_M \backslash \Delta_O; \Omega_{LIA} \Omega_M \backslash \Omega_{LOA} \Omega_M \longrightarrow A$

Then $\Gamma; \Delta_I \backslash \Delta_O; \Omega_{LIL} \Omega_{LIA} \Omega_M \Omega_{RI} \backslash (\Omega_{LOL} \Omega_{LOA} | \Omega_M | \Omega_{RO}) \longrightarrow A \rightarrowtail B \gg P$.
Then $\Gamma; \Delta_I \backslash \Delta_O; \Omega_{LI} \Omega_M \Omega_{RI} \backslash (\Omega_{LO} | \Omega_M | \Omega_{RO}) \longrightarrow A \rightarrowtail B \gg P$.

cases: $\twoheadrightarrow_L, \multimap_L, \rightarrow_L$

Similar to previous case.

case: $$\frac{}{\Gamma; \cdot; (\cdot\ ;\ \cdot) \ \longrightarrow\ P \gg P}\ \mathbf{init}$$

Then $\dfrac{}{\Gamma; \cdot; (\Omega_L \backslash \Omega_L\ ;\ \Omega_R \backslash \Omega_R) \ \longrightarrow\ P \gg P}\ \mathbf{init}_1$ and $\dfrac{}{\Gamma; \cdot; \Omega_I \backslash (\cdot|\Omega_I|\cdot) \ \longrightarrow\ P \gg P}\ \mathbf{init}_2$.

□

5 Sample Programs

Using the strategy described at the beginning of the preceding section, together with the system of ordered resource management described above, we have arrived at a logic programming language. It remains to clarify the order in which the clauses in a context are considered when the goal has become atomic. We employ here the same strategy as the Lolli language [HM94] and linear LF [CP98] by dividing the hypotheses into a static program and dynamic assumptions made during search. We first scan all dynamic assumptions from right to left, where $\rightarrow_R$, $\multimap_R$, $\twoheadrightarrow_R$ add assumptions on the right, and $\rightarrowtail_R$ adds assumptions on the left. Then we attempt each formula in the static program from first to last. Note that we have to be aware of the status of each hypothesis (unrestricted, linear, or ordered) as we consider it so we can apply the correct **choice** rule.

In the examples below we assume implicit universal quantification over free variables just as in Prolog. We also write $B \twoheadleftarrow A$ for $A \twoheadrightarrow B$ and $B \leftarrowtail A$ for $A \rightarrowtail B$ in the manner of Prolog. We use *italics* for meta-variables which stand for ground terms and `typewriter font` for program code, including logic variables.

5.1 Lists

We begin by considering various programs concerned with lists. `::` is the list constructor and `nil` is the empty list.

The following program, which does not make use of the ordered fragment of the logic, can be used to permute a list.

```
perm (X::L) K
   o— (elem X —o perm L K).

perm nil (X::K)
   o— elem X
   o— perm nil K.

perm nil nil.
```

The program works on a query `perm` l `K` by first assuming `elem` x for every element x of l. This is achieved by the first clause. Then the assumptions are consumed one by one through uses of the second clause and added to the output list. The tail of the output list is instantiated to `nil` when there are no further linear assumptions, and the last clause can therefore succeed. Because the linear context is unordered, every possible order of linear resource consumption constitutes a valid proof where the result variable, `K`, becomes instantiated to a different permutation of the input list l.

Since linear assumptions are unordered, the program will enumerate all possible permutations of a list. If we replace the linear implications by right order implication, only one order remains possible: the one that reverses the list.

$$\begin{array}{l}\texttt{rev (X::L) K}\\ \quad \twoheadleftarrow (\texttt{elem X} \twoheadrightarrow \texttt{rev L K}).\\[1em] \texttt{rev nil (X::K)}\\ \quad \twoheadleftarrow \texttt{elem X}\\ \quad \twoheadleftarrow \texttt{rev nil K}.\\[1em] \texttt{rev nil nil}.\end{array}$$

Now the first appeal to the second clause can pick up and consume only the most recently made assumption, which is the last element of the input list l. Since this is added to the front of K, this operation reverses the list.

In contrast, the following program represents the identity relation, since the elements are added to the left end of the ordered assumptions.

$$\begin{array}{l}\texttt{id (X::L) K}\\ \quad \twoheadleftarrow (\texttt{elem X} \rightarrowtail \texttt{id L K}).\\[1em] \texttt{id nil (X::K)}\\ \quad \twoheadleftarrow \texttt{elem X}\\ \quad \twoheadleftarrow \texttt{id nil K}.\\[1em] \texttt{id nil nil}.\end{array}$$

5.2 Sorting

Our next example is a direct coding of merge sort, where $\texttt{mergeSort}\, k\, \texttt{L}$ expects an input list k and computes an output list L by mergesort. The computation proceeds in two phases. In the first phase, assuming an input list $x_1 \texttt{::} \cdots \texttt{::} x_n \texttt{::nil}$ we want to reduce solving

$$\cdot \longrightarrow \texttt{mergeSort}\,(x_1 \texttt{::} \cdots \texttt{::} x_n \texttt{::nil})\,\texttt{L}$$

to solving

$$\texttt{srt}(x_1 \texttt{ :: nil}), \ldots, \texttt{srt}(x_n \texttt{ :: nil}) \longrightarrow \texttt{msort L}$$

where we have separated ordered hypotheses by commas and omitted the unrestricted context (which contains only the static program and does not change) and the linear context (which is always empty). We achieve this with the two clauses

$$\begin{array}{l}\texttt{mergeSort (H::T) L}\\ \quad \twoheadleftarrow (\texttt{srt (H::nil)} \twoheadrightarrow \texttt{mergeSort T L}).\\[0.5em] \texttt{mergeSort nil L} \twoheadleftarrow \texttt{msort L}.\end{array}$$

In the second phase we assume a general situation of the form

$$\texttt{srt}\, l_n, \ldots, \texttt{srt}\, l_2, \texttt{srt}\, l_1 \longrightarrow \texttt{msort L}$$

where the l_i are already sorted and L is still to be computed. Starting from the right, we merge l_1 and l_2 and add the result to the *left* end of the ordered context, in effect using it as a work queue. The resulting situation will be

$$\texttt{srt}\, l_{12}, \ldots, \texttt{srt}\, l_4, \texttt{srt}\, l_3 \longrightarrow \texttt{msort L}$$

which is then treated the same way, merging l_3 and l_4. We finish when there is only one element $\texttt{srt}\, k$ in the ordered context and unify L with k. This is expressed by the following two clauses.

$$\begin{array}{l}
\texttt{msort L} \\
\quad \twoheadleftarrow \texttt{srt L1} \\
\quad \twoheadleftarrow \texttt{srt L2} \\
\quad \leftarrow \texttt{merge L1 L2 L12} \\
\quad \leftarrowtail (\texttt{srt L12} \rightarrowtail \texttt{msort L}).
\end{array}$$

$$\texttt{msort L} \twoheadleftarrow \texttt{srt L}.$$

We elide the standard Prolog-like `merge` predicate which, given two sorted lists l_1 and l_2 returns a sorted merge l_{12}. It does not use the ordered context, which is enforced by using an unrestricted implication $\leftarrow$.

Since all ordered assumptions must be used, the final clause above can succeed only if the complete list has indeed been sorted. The merge sort program is therefore completely deterministic when called as indicated: L1 and L2 *must* be taken from the right, and L12 *must* be added to the left.

If we change the first `msort` clause to assume L12 on the right, by writing ($\texttt{srt L12} \twoheadrightarrow \texttt{msort L}$) instead of ($\texttt{srt L12} \rightarrowtail \texttt{msort L}$) then we obtain an insertion sort because after one step we arrive at

$$\texttt{srt}\, l_n, \ldots, \texttt{srt}\, l_3, \texttt{srt}\, l_{12}$$

which will next merge l_3 into l_{12}, etc.

5.3 Mini-ML Abstract Machine

Our next example shows how a continuation based abstract machine for evaluating Mini-Ml can be directly encoded in INCLL. The basic idea is to use the ordered context as a stack of continuations to be evaluated. We assume a standard version of Mini-Ml constructed using higher-order abstract syntax [Pfe92]. Values are distinguished from terms by an asterisk; so z is a term while $\texttt{z}^*$ is a value. We have the following signature for our abstract machine (where o is the type of propositions):

```
z :  exp.
s :  exp -> exp.
case :  exp -> exp -> (val -> exp) -> exp.
lam :  (val -> exp) -> exp.
app :  exp -> exp -> exp.
vl :  exp -> val.
z* :  val.
s* :  val -> val.
lam* :  (val -> exp) -> val.
eval :  exp -> val -> o.
ev :  exp -> o.
return :  val -> o.
case1 :  val -> exp -> ( val -> exp) -> o.
appm1 :  val -> exp -> o.
appm2 :  val -> val -> o.
```

Given the goal: `return V` $\twoheadrightarrow$ `ev` e, our program will evaluate the expression e and instantiate `V` with the resulting value. The intended reading of this query is: evaluate e with the identity continuation (the continuation which just returns its value). A goal of `ev` e is intended to mean: evaluate e. A goal of `return V` is intended to mean: pass `V` to the top continuation on the stack (i.e. the rightmost element in the ordered context).

The following program clauses specify the abstract machine:

$$\texttt{eval E V} \twoheadleftarrow (\texttt{return V} \twoheadrightarrow \texttt{ev E})$$

(* Natural Numbers *)

$$
\begin{array}{l}
\texttt{ev z} \twoheadleftarrow \texttt{return z}^*. \\
\texttt{ev (s E)} \\
\quad \twoheadleftarrow ((\forall \texttt{V}.\, \texttt{return V} \leftarrowtail \texttt{return (s}^*\ \texttt{V)}) \twoheadrightarrow \texttt{ev E}). \\
\texttt{ev (case E1 E2 E3)} \\
\quad \twoheadleftarrow ((\forall \texttt{V}.\, \texttt{return V} \leftarrowtail \texttt{case1 V E2 E3}) \twoheadrightarrow \texttt{ev E1}). \\
\texttt{case1 z}^*\ \texttt{E2 E3} \twoheadleftarrow \texttt{ev E2}. \\
\texttt{case1 (s}^*\ \texttt{V) E2 E3} \twoheadleftarrow \texttt{ev (E3 V)}.
\end{array}
$$

(* Functions *)

$$
\begin{array}{l}
\texttt{ev (lam E)} \twoheadleftarrow \texttt{return (lam}^*\ \texttt{E)}. \\
\texttt{ev (app E1 E2)} \\
\quad \twoheadleftarrow ((\forall \texttt{V1}.\, \texttt{return V1} \leftarrowtail \texttt{app1 V1 E2}) \twoheadrightarrow \texttt{ev E1}). \\
\texttt{app1 V1 E2} \\
\quad \twoheadleftarrow (\forall \texttt{V2}.\, \texttt{return V2} \leftarrowtail \texttt{app2 V1 V2}) \twoheadrightarrow \texttt{ev E2}). \\
\texttt{app2 (lam}^*\ \texttt{E1') V2} \twoheadleftarrow \texttt{ev (E1' V2)}.
\end{array}
$$

(* Values *)

$$\texttt{ev (vl V)} \twoheadleftarrow \texttt{return V}.$$

The intended reading of the `ev (s E)` clause (the second program clause) is this: to evaluate `(s E)` evaluate `E` under the continuation which takes its value, `V`, and passes the value $\texttt{s}^*$ `V` to the next continuation on the stack. Note the use of $\rightarrowtail$ nested inside $\twoheadrightarrow$. This forces the continuations put into the ordered context to be evaluated in stack fashion. When the goal is `return V` the only choice of formula to focus on will be the rightmost formula in the ordered context.

To better illustrate this point, consider the evaluation of `(s (s z))`. The initial goal will be $\texttt{return V} \twoheadrightarrow \texttt{ev (s (s z))}$ after which `return V` will be immediately added to the previously empty ordered context. Next the `ev (s E)` clause will be chosen to focus on and will result in $\forall\texttt{V}.\ \texttt{return V} \leftarrowtail \texttt{return (s}^*\ \texttt{V)}$ being added to the right end (because of $\twoheadrightarrow$) of the ordered context. The new goal will be `ev (s z)` and the previous step will be repeated. At this point, the goal will be `ev z` which will cause the appropriate progam clause (the first in the preceding program listing) to be focused on and a new goal of $\texttt{return z}^*$.

The ordered context now consists of:

$$(\texttt{return V}) \quad (\forall\texttt{V}.\ \texttt{return V} \leftarrowtail \texttt{return (s}^*\ \texttt{V)}) \quad (\forall\texttt{V}.\ \texttt{return V} \leftarrowtail \texttt{return (s}^*\ \texttt{V)}).$$

Since there is no program clause whose head matches the goal, one of the clauses in the ordered context must be focused on. Although all the ordered clauses match the goal, only the rightmost one can be successfully chosen. The leftmost clause obviously cannot work since it is atomic and the other clauses are also in the ordered context. The middle clause also does not work because the $\rightarrowtail$ requires that the body of the clause be solved with resources to the left of the clause which would prevent the rightmost clause from being consumed.

5.4 Natural Langauge Parsing

The following example is a fragment of a parser. This example shows how INCLL can be used to directly parse grammatical constructions with unbounded dependencies such as relative clauses.

$$\begin{array}{rl}
1: & \texttt{snt} \twoheadleftarrow \texttt{vp} \twoheadleftarrow \texttt{np}. \\
2: & \texttt{vp} \twoheadleftarrow \texttt{np} \twoheadleftarrow \texttt{tv}. \\
3: & \texttt{rel} \leftarrowtail \texttt{whom} \twoheadleftarrow (\texttt{np} \multimap \texttt{snt}). \\
4: & \texttt{np} \twoheadleftarrow \texttt{jill}. \\
5: & \texttt{tv} \twoheadleftarrow \texttt{married}.
\end{array}$$

We may intuitively read the formulas in the following manner: $\texttt{snt} \twoheadleftarrow \texttt{vp} \twoheadleftarrow \texttt{np}$ states that a sentence is a verb phrase to the right of a noun phrase. We can use these formulas to parse a phrase by putting each word of the sentence into the ordered context and trying to derive the atomic formula corresponding to the phrase type.

We may interpret clause 3 as: a relative clause is `whom` to the left of a sentence missing a noun phrase. As explained in [Hod94] this is a standard interpretation of relative clauses. By putting a `np` into the linear context, the sentence after `whom` will only be successfully parsed if it is indeed missing a noun phrase. Note that by using the linear context, the location of the missing `np` is unconstrained.

We now show a trace of the above formulas parsing a relative clause, showing at each step the resource sequent (including the current goal), the pending goals, and the the rule applied. The unrestricted context containing the above program is left implicit. Note that due to the ordering constraints each step is essentially deterministic.

Action	Active hypotheses and goal	Goals pending
	$\cdot$; `whom jill married` $\longrightarrow$ `rel`	none
reduce by 3	$\cdot$; `whom jill married` $\longrightarrow$ `whom`	`np` $\multimap$ `snt`
solved, restore pending goal	$\cdot$; `jill married` $\longrightarrow$ `np` $\multimap$ `snt`	none
assume	`np` ; `jill married` $\longrightarrow$ `snt`	none
reduce by 1	`np` ; `jill married` $\longrightarrow$ `vp`	`np`
reduce by 2	`np` ; `jill married` $\longrightarrow$ `np`	`tv` , `np`
solved, restore pending goal	$\cdot$; `jill married` $\longrightarrow$ `tv`	`np`
reduce by 5	$\cdot$; `jill married` $\longrightarrow$ `married`	`np`
solved, restore pending goal	$\cdot$; `jill` $\longrightarrow$ `np`	none
reduce by 4	$\cdot$; `jill` $\longrightarrow$ `jill`	none
solved		

Using the linear context in manner described above has some limitations which should be pointed out. The correct parsing of dependent clauses typically constrains where the relative pronoun may fill in for a missing noun phrase. Most relative clauses, rather than being sentences missing noun phrases are really sentences whose verb phrase is missing a noun phrase.

If we changed the relative clause to be **whom married jill** we would not have a grammatically correct relative noun. However the parser given above will be able to parse the modified phrase since the location of the missing noun phrase is not constrained. There are a variety of simple ways to fix this problem for the small parser given above. For instance, we could define a new type of sentence in which the verb phrase is missing a noun phrase. This basically amounts to using gap-locator rules as described in [Par89] [Hod94].

The parsers given in [Hod94], which logically handle some constraints on the placement of dependencies, are constructed quite differently from the INCLL parser we have presented. Rather than placing the input to be parsed into the context, they pass it around as a list. They do however use the linear context to store fillers— empty noun phrase predicates which can be used when an actual noun phrase is missing in the sentence— in the same manner as the above parser does. We point out that all of the (pure) Lolli parsers are also valid INCLL programs since pure-Lolli is a subset of INCLL.

In fact with this threaded style of parser, INCLL is able to correctly handle at least one natural language phenomenom which could not be done in pure Lolli. When a relative clause occurs inside a relative clause, correct parses of the sentence should associate the inner relative pronoun with the first missing noun phrase in the clause and the second with the second. In other words, dependencies should not cross inside a nested relative clause. Consider the phrase: **the book that the man whom Jane likes GAP wrote GAP** where **GAP** denotes a missing noun phrase. The first **GAP** should correspond to **the man** and not to **the book**.

Since the Lolli parser (as well as the INCLL parser given above) introduces fillers into the linear context, there is no way to force the parser to use one or the other of two suitable fillers. Thus the Lolli parser would parse the preceding sentence in two ways, only one of which would be correct. Hodas' solution was to rely on a non-logical control construct and the operational semantics of Lolli to prevent the bad parse. One can easily see (as Hodas remarked) that such a situation can be directly handled in INCLL by putting the fillers into the ordered context.

6 Conclusion

The succession of systems developed in this paper form a promising basis for incorporating order into linear logic programming in a logical and efficient manner. In [PP98] we give the full complement of connectives for INCLL. Of those, only universal quantification, &, and $\top$ preserve the completeness of uniform derivations (which is easy to show). At present we have not checked every detail, but we strongly conjecture that the resource management system developed for Lolli in [CHP98] can also be straightforwardly combined with the scheme for managing ordered resources detailed in this paper, leading to a conservative extension of Lolli. That is, every legal Lolli program would remain legal and have the same operational behavior.

There are many examples where order can be exploited, such as algorithms which employ various forms of stacks or queues. Natural language parsing, the original motivation for the Lambek calculus [Lam58] is a further rich source of examples. Using our prototype implementation we have programmed the two styles of parsers sketched above. In many of our examples, the ordered connectives allow a more concise, logical specification of algorithms than possible in other languages.

However, there are also some difficulties. If an algorithm requires more than one work queue or stack they can interfere, since we have only one ordered context. Another problem is that we sometimes have to write "glue" code which initializes or erases the ordered context prior to a subcomputation. Despite these current limitations, we feel that ordered logic programming is an interesting paradigm which can shed light on the concept of order in computation and warrants further investigation from all angles.

Besides further work on language design, implementation, and programming methodology, we also plan to investigate the potential applications of our particular approach to order in concurrent constraint programming as proposed by Ruet [Rue97]. Finally, the existence of canonical forms for the natural deduction system of INCLL [PP99] means that it might be possible to add it to the linear logical framework [CP98] in a conservative manner. However, we have not yet considered such issues as type reconstruction or unification.

References

[Abr90] V. M. Abrusci. Sequent calculus for intuitionistic linear propositional logic. In P. P. Petkov, editor, *Mathematical Logic*, pages 223–242, New York, London, 1990. Plenum Press. Proceedings of the Summer School and Conference on Mathematical Logic, honourably dedicated to the 90th Anniversary of Arend Heyting (1898–1980), Chaika, Bulgaria, 1988.

[And92] J.-M. Andreoli. Logic programming with focusing proofs in linear logic. *Journal of Logic and Computation*, 2(3):297–347, 1992.

[AR98] V. Michele Abrusci and Paul Ruet. Non-commutative logic I: The multiplicative fragment. Submitted, May 1998.

[BG91] C. Brown and D. Gurr. Relations and non-commutative linear logic. Technical Report DAIMI PB-372, Computer Science Department, Aarhus University, November 1991.

[CHP98] Iliano Cervesato, Joshua S. Hodas, and Frank Pfenning. Efficient resource management for linear logic proof search. To appear in the special issue of TCS on Proof Search in Type-Theoretic Languages. Revised version of paper in the Proceedings of the 5th International Workshop on Extensions of Logic Programming, Leipzig, Germany, March 1996, 1998.

[CP98] Iliano Cervesato and Frank Pfenning. A linear logical framework. *Information and Computation*, 1998. Accepted to the special issue with invited papers from LICS'96, E. Clarke, editor.

[Gir87] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[HM94] Joshua Hodas and Dale Miller. Logic programming in a fragment of intuitionistic linear logic. *Information and Computation*, 110(2):327–365, 1994. A preliminary version appeared in the Proceedings of the Sixth Annual IEEE Symposium on Logic in Computer Science, pages 32–42, Amsterdam, The Netherlands, July 1991.

[Hod94] Joshua S. Hodas. *Logic Programming in Intuitionistic Linear Logic: Theory, Design, and Implementation.* PhD thesis, University of Pennsylvania, Department of Computer and Information Science, 1994.

[Lam58] Joachim Lambek. The mathematics of sentence structure. *American Mathematical Monthly*, 65:363–386, 1958.

[MNPS91] Dale Miller, Gopalan Nadathur, Frank Pfenning, and Andre Scedrov. Uniform proofs as a foundation for logic programming. *Annals of Pure and Applied Logic*, 51:125–157, 1991.

[Par89] Remo Pareschi. *Type-Driven Natural Language Analysis.* PhD thesis, University of Edinburgh, Edinburgh, Scotland, July 1989. Available as technical report MS-CIS-89-45, Department of Computer and Information Sciences, University of Pennsylvania.

[Pfe92] Frank Pfenning. Computation and deduction. Unpublished lecture notes, 277 pp. Revised May 1994, April 1996, May 1992.

[Pfe94] F. Pfenning. Structural cut elimination in linear logic. Technical Report CMU-CS-94-222, Carnegie Mellon University, Department of Computer Science, December 1994.

[PP98] Jeff Polakow and Frank Pfenning. Relating natural deduction and sequent calculus for intuitionistic non-commutative linear logic. Submitted, November 1998.

[PP99] Jeff Polakow and Frank Pfenning. Natural deduction for intuitionistic non-commutative linear logic. In J.-Y. Girard, editor, *Proceedings of the Fourth International Conference on Typed Lambda Calculi and Applications (TLCA'99)*, l'Aquila, Italy, April 1999. Springer-Verlag LNCS. To appear.

[PS98] Frank Pfenning and Carsten Schürmann. *Twelf User's Guide*, 1.2 edition, September 1998. Available as Technical Report CMU-CS-98-173, Carnegie Mellon University.

[Rue97] Paul Ruet. *Logique non-commutative et programmation concurrente par contraintes.* PhD thesis, Universite Denis Diderot, Paris 7, 1997.

Appendix A

We present our example parsing derivation in the resource management system. As before, $\Gamma =$

$$
\begin{array}{ll}
1: & \mathtt{snt} \twoheadleftarrow \mathtt{vp} \twoheadleftarrow \mathtt{np}. \\
2: & \mathtt{vp} \twoheadleftarrow \mathtt{np} \twoheadleftarrow \mathtt{tv}. \\
3: & \mathtt{tv} \twoheadleftarrow \mathtt{loves}. \\
4: & \mathtt{np} \twoheadleftarrow \mathtt{mary}. \\
5: & \mathtt{np} \twoheadleftarrow \mathtt{bob}.
\end{array}
$$

Here is an operational trace of the derivation:

Action	Active hypotheses and goal	Goals pending
	$\cdot\,; \mathtt{mary\ loves\ bob} \longrightarrow \mathtt{snt}$	none
reduce by 1	$\cdot\,; \mathtt{mary\ loves\ bob} \longrightarrow \mathtt{vp}$	$\mathtt{np}$
reduce by 2	$\cdot\,; \mathtt{mary\ loves\ bob} \longrightarrow \mathtt{np}$	$\mathtt{tv}$, $\mathtt{np}$
reduce by 5	$\cdot\,; \mathtt{mary\ loves\ bob} \longrightarrow \mathtt{bob}$	$\mathtt{tv}$, $\mathtt{np}$
solved, restore pending goal	$\cdot\,; \mathtt{mary\ loves} \longrightarrow \mathtt{tv}$	$\mathtt{np}$
reduce by 3	$\cdot\,; \mathtt{mary\ loves} \longrightarrow \mathtt{loves}$	$\mathtt{np}$
solved, restore pending goal	$\cdot\,; \mathtt{mary} \longrightarrow \mathtt{np}$	none
reduce by 4	$\cdot\,; \mathtt{mary} \longrightarrow \mathtt{mary}$	none
solved		

The following page contains the complete derivation. As with the uniform system, this is the only proof of the given end-sequent. Notice how the structure of this proof exactly matches the structure of the previous uniform version. Additionally, we note that the resources consumed in a branch of this proof exactly match the resources passed into the same branch of the uniform version. For instance $\Theta_{\mathtt{bob}}$ in this proof begins with a sequent whose input ordered context contains `mary loves bob` and whose output ordered context contains $\mathtt{mary\ loves}\square$. Thus `bob` was the only resource consumed in that proof branch. In the $\Theta_{\mathtt{bob}}$ of the uniform proof, `bob` is the only resource passed to that proof branch.

$$
\dfrac{\dfrac{}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash(\cdot|\texttt{mary loves bob}|\cdot)\longrightarrow\texttt{snt}\gg\texttt{snt}}\,\textbf{init}\qquad\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash(\cdot|\texttt{mary loves bob}|\cdot)\longrightarrow\texttt{vp}\gg\texttt{vp}}\,\textbf{init}\qquad\Theta_{\texttt{bob}}}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash(\cdot|\texttt{mary loves}|\square)\longrightarrow\texttt{np}\twoheadrightarrow\texttt{vp}\gg\texttt{vp}}\,\twoheadrightarrow_{L2}\qquad\Theta_{\texttt{loves}}}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash(\cdot|\texttt{mary}|\square\ \square)\longrightarrow\texttt{tv}\twoheadrightarrow\texttt{np}\twoheadrightarrow\texttt{vp}\gg\texttt{vp}}\,\twoheadrightarrow_{L2}}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash(\cdot|\texttt{mary}|\square\ \square)\longrightarrow\texttt{vp}}\,\textbf{choice}_{\Gamma}}{\dfrac{\dfrac{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash(\cdot|\texttt{mary}|\square\ \square)\longrightarrow(\texttt{vp}\twoheadrightarrow\texttt{snt})\gg\texttt{snt}\qquad\Theta_{\texttt{mary}}}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash(\cdot|\cdot|\square\ \square\ \square)\longrightarrow\texttt{np}\twoheadrightarrow\texttt{vp}\twoheadrightarrow\texttt{snt}\gg\texttt{snt}}\,\twoheadrightarrow_{L2}}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash\square\ \square\ \square\longrightarrow\texttt{snt}}\,\textbf{choice}_{\Gamma}}\,\twoheadrightarrow_{L2}
$$

where $\Theta_{\texttt{bob}} =$

$$
\dfrac{\dfrac{\dfrac{}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash(\cdot|\texttt{mary loves bob}|\cdot)\longrightarrow\texttt{np}\gg\texttt{np}}\,\textbf{init}\qquad\dfrac{\dfrac{}{\Gamma;\cdot\backslash\cdot;(\texttt{mary loves}\backslash\texttt{mary loves};\cdot\backslash\cdot)\longrightarrow\texttt{bob}\gg\texttt{bob}}\,\textbf{init}}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash\texttt{mary loves}\ \square\longrightarrow\texttt{bob}}\,\textbf{choice}_{\Omega}}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash(\cdot|\texttt{mary loves}|\square)\longrightarrow\texttt{bob}\twoheadrightarrow\texttt{np}\gg\texttt{np}}\,\twoheadrightarrow_{L2}}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves bob}\backslash\texttt{mary loves}\ \square\longrightarrow\texttt{np}}\,\textbf{choice}_{\Gamma}
$$

$\Theta_{\texttt{loves}} =$

$$
\dfrac{\dfrac{\dfrac{}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves}\backslash(\cdot|\texttt{mary loves}|\cdot)\longrightarrow\texttt{tv}\gg\texttt{tv}}\,\textbf{init}\qquad\dfrac{\dfrac{}{\Gamma;\cdot\backslash\cdot;(\texttt{mary}\backslash\texttt{mary};\cdot\backslash\cdot)\longrightarrow\texttt{loves}\gg\texttt{loves}}\,\textbf{init}}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves}\backslash\square\longrightarrow\texttt{loves}}\,\textbf{choice}_{\Omega}}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves}\backslash(\cdot|\texttt{mary}|\square)\longrightarrow\texttt{loves}\twoheadrightarrow\texttt{tv}\gg\texttt{tv}}\,\twoheadrightarrow_{L2}}{\Gamma;\cdot\backslash\cdot;\texttt{mary loves}\backslash\texttt{mary}\ \square\longrightarrow\texttt{tv}}\,\textbf{choice}_{\Gamma}
$$

$\Theta_{\texttt{mary}} =$

$$
\dfrac{\dfrac{\dfrac{}{\Gamma;\cdot\backslash\cdot;\texttt{mary}\backslash(\cdot|\texttt{mary}|\cdot)\longrightarrow\texttt{np}\gg\texttt{np}}\,\textbf{init}\qquad\dfrac{\dfrac{}{\Gamma;\cdot\backslash\cdot;(\cdot\backslash\cdot;\cdot\backslash\cdot)\longrightarrow\texttt{mary}\gg\texttt{mary}}\,\textbf{init}}{\Gamma;\cdot\backslash\cdot;\texttt{mary}\backslash\square\longrightarrow\texttt{np}}\,\textbf{choice}_{\Omega}}{\Gamma;\cdot\backslash\cdot;\texttt{mary}\backslash(\cdot|\cdot|\square)\longrightarrow\texttt{mary}\twoheadrightarrow\texttt{np}\gg\texttt{np}}\,\twoheadrightarrow_{L2}}{\Gamma;\cdot\backslash\cdot;\texttt{mary}\backslash\square\longrightarrow\texttt{np}}\,\textbf{choice}_{\Gamma}
$$