Proof-Theoretic Foundation of Hypothetical Reasoning

Pablo López Ernesto Pimentel

Departamento de Lenguajes y Ciencias de la Computación
Universidad de Málaga
Campus de Teatinos. 29071 Málaga
Phone: +34 952 13 13 96 Fax: +34 952 13 13 97
{lopez,ernesto}@lcc.uma.es

Abstract. It is well-known that state change is rather difficult to express within classical or intuitionistic logics. That inability makes those logics inappropriate for applications involving state update such as knowledge representation and non-monotonic reasoning. Modal and temporal logics intend to offer a solution by introducing stages of deduction where the notions of state and change are dealt with. In spite of its merits, that approach heavily relies on model-theoretic methods most of which are known to be mathematically complex and computationally unsuitable. In this paper, we propose a first step to a proof-theoretic foundation of knowledge representation and non-monotonic reasoning. In particular, we argue that substructural logics do provide a convenient framework (devoid of ad hoc constructions) where non-monotonicity is accomplished via resource-sensitivity. To illustrate our approach, we present a proof-theoretic formalization of hypothetical reasoning within intuitionistic linear logic. The new concept of *essential hypothesis* is introduced and its expressive power is illustrated by means of a deductive database supporting hypothetical queries. As for the computational aspects, an efficient proof search strategy which constructs lazy proofs is outlined, and certain proof-theoretic properties of those lazy proofs are applied to further extend the expressive power of the proposed logical framework.

Keywords: proof theory, hypothetical reasoning, non-classical logics, applications of logic-based systems

1 Introduction

As argued in [8], it is worth to formalize some part of human reasoning assuming that it is subject to certain informal rigor. Such a formalization should keep formalized reasoning as close to informal reasoning as possible; that is, the formalization should be direct, concise, and simple. On the one hand, it is clear that an excess of artificial formality should be avoided since that would lead to extremely detailed and verbose reasonings unrelated to the original informal reasoning. On the other hand, human reasoning is subject to frequent changes of

mind which could contradict previous conclusions and hence the formalization framework should be able to accommodate those changes. Finally, it should be clear that most of human reasoning deals with the representation of situations (or knowledge) which change according to a certain number of actions, thus the notions of state and state change play a fundamental role in any plausible formalization.

Classical logic is not a good candidate for this task, since it is monotonic and state-unconscious; i.e. the representation of state and state change usually leads to both a marked verbosity and a lack of adaptability. Consider, for example, the difficulties raised by the frame problem.

Modal and temporal logics intend to offer a solution by introducing stages of deduction where the notions of state and change are dealt with. In spite of its merits, that approach has several drawbacks. From the proof-theoretic viewpoint, the modalities introduced by these logics are too complex and artificial [8]. From the computational viewpoint, these logics are heavily tied to model-theoretic semantics which involve different forms of infinity which usually cannot be represented in a computer [15].

In this paper, we propose a first step to a proof-theoretic foundation of knowledge representation and non-monotonic reasoning, and hence, to the formalization of informal human reasoning. In particular, we argue that substructural logics do provide a convenient (devoid of ad hoc constructions) framework for this task, where non-monotonicity[1] is accomplished via resource-sensitivity.

To illustrate our approach, we present a proof-theoretic formalization of hypothetical reasoning within intuitionistic linear logic. The new concept of *essential hypothesis* is introduced and its expressive power is illustrated by means of a deductive database supporting hypothetical queries. From the computational point of view, we briefly review a resource management system for Lolli and show that, in addition to being a fundamental implementation technique, its proof-theoretic properties can be applied to extend the facilities provided by our formalization of hypothetical reasoning. In particular, the user is no longer required to decide the essentiality of hypotheses a priori; but it is the resource management system which finds out whether the hypotheses are essential or not.

The rest of the paper is organized as follows. Section 2 presents hypothetical reasoning and discusses its formalization through the use of different kinds of implications. Section 3 introduces the concept of *essential hypothesis* and advocates the use of substructural logics to deal with it. In section 4 we present a brief introduction to intuitionsitic linear logic. Next section presents a deductive database example which evidences the gain of expressive power offered by essential hypotheses. Section 6 illustrates the application of proof-theoretic properties of a resource management system to extend the expressive power of our formalization of hypothetical reasoning. Finally, conclusions and further work are outlined.

[1] The fact that substructural logics are non-monotonic or not depends whether the property is applied to formulae or to sequents.

2 Hypothetical Reasoning

In everyday life, people do use to reason about a given situation by assuming certain conditions whose validity is unknown: they first suppose that these conditions hold, and then infer a number of conclusions from them. If the supposed conditions actually hold, the inferred conclusions are taken as valid. This reasoning scheme usually appears structured in an arbitrary number of cases, each of which deals with a particular set of assumptions leading to different, probably incompatible, conclusions. Hypotheses emerge as a versatile and natural way for structuring uncertain knowledge, and for taking decisions on the basis of that assumed knowledge. Both are actually common tasks of human reasoning. As an instance of this kind of conditional reasoning, consider how many times a day you think in the following terms: "if my plane arrives on time, then I will pick a bus, and if my plane arrives late, then I will pick a taxi".

The form of deduction above outlined is formally referred to as *hypothetical reasoning*, an inference mechanism to reason provided that a set of conditions, known as *assumptions* or *hypotheses*, hold. Different kinds of hypothetical reasoning have been extensively used to reason about legal issues [6, 14], natural language processing [5, 9], and deductive databases [2, 3].

In order to formalize the notion of hypothetical reasoning we require a formal inference system $\mathcal{I}$, a collection of well-formed formulae Δ [2], and a provability relation $\cdot \vdash_{\mathcal{I}} \cdot$ defined on $\mathcal{I}$, such that the judgement $\Delta \vdash_{\mathcal{I}} \psi$ holds if and only if there exists an $\mathcal{I}$-proof from Δ for the formula ψ. Within this setting, hypothetical reasoning is accomplished by temporarily adding the set of hypotheses $\mathcal{H}$ to the antecedent of the judgement Δ, yielding $\Delta, \mathcal{H} \vdash_{\mathcal{I}} \psi$.

Hitherto, most of the research on hypothetical reasoning has been conducted within classical and intuitionistic (sub)logics. The temporal addition of the set of hypotheses to the antecedent of a judgement is then formalized via implications (i.e. as a direct application of the deduction meta-theorem)

$$\Delta, \mathcal{H} \vdash_{\mathcal{I}} \psi \text{ iff } \Delta \vdash_{\mathcal{I}} \mathcal{H} \rightarrow \psi$$

where $\rightarrow$ stands for either classical (material) or intuitionistic implication. Therefore, it is quite clear that the adequacy of a given formalization of hypothetical reasoning heavily depends on the logical properties of the implication connective involved (material, intuitionistic, linear, etc.). For example, Bonner [1] shows that material implication is inappropriate for hypothetical reasoning due to its non-constructive nature. In doing so, he first defines a criteria named *rulebase independence* and then shows that material implication is unable to fulfill that criteria. Non-monotonicity seems to be a desirable feature as well, as it allows for the addition of hypotheses which invalidate previous conclusions. In the next section, we shall introduce the concept of essential hypothesis and show that classical and intuitionistic logics (i.e. material and intuitionistic implications) are unable to deal with it.

[2] Depending on the application domain, the collection of formulae Δ shall denote a deductive database, a logic theory, or a logic program.

3 Essential Hypotheses and Substructural Logics

Intuitionistic logic is *constructive*; that is, a proposition is true if and only if it is possible to construct a concrete proof for it. In contrast, in classical logic a proposition is known to be true if the negated proposition does not hold [3]. Constructivism is exemplified by the *existential property* of constructive logics: $\vdash \exists x.A$ iff there exists a concrete term t (referred to as the *witness*) for the variable x such that $\vdash A[t/x]$. It is worth noting that is is not enough to know that such a term exists, but a proof must construct a concrete term t.

A constructive proof method constructs a concrete logical object – i.e. a proof – whose very existence proves the proposition considered. From this point of view, intuitionistic formulae (and terms) are interpreted as basic building blocks, glued together by means of inference rules. In building a constructive proof, a given building block can be used as many times as needed, including zero. Thus if a proposition can be proved once from Δ, it can be proved any number of times from $\Delta' \supseteq \Delta$. That amounts to say that intuitionistic logic is monotonic; i.e. it deals with static truth. The "reusability" of intuitionistic building blocks arises from the structural rules of *weakening* and *contraction* shown below

$$\frac{\Delta \vdash B}{\Delta, A \vdash B}\ Weakening \qquad\qquad \frac{\Delta, A, A \vdash B}{\Delta, A \vdash B}\ Contraction$$

As we search for a proof from the bottom up, weakening allows dispensing with a block A which is no longer needed. Contraction allows obtaining a copy of a block A. Should Δ be considered as a knowledge base, weakening means that qualitative increases of knowledge do not invalidate previous conclusions while contraction means that quantitative aspects are irrelevant.

Substructural logics emerge from dispensing with the weakening and/or contraction structural rules[4]. As a result, the number of occurrences of formulae matter. Formulae are no longer considered as reusable building blocks but as finite resources to be carefully invested in the proof of a proposition. Such a distinguishing feature turns substructural logics into resource conscious, non-monotonic logics.

Several substructural logics have been proposed including:

Affine logic dispenses with the contraction rule, and hence, affine resources must be used at most once.

Relevant logic arises from dispensing with the weakening rule. Relevant resources must be used at least once.

Linear logic results from dispensing with both weakening and contraction, therefore, linear resources must be used exactly once.

[3] This is due to the *law of excluded middle* $\vdash A \vee A^{\perp}$; that is, A is either true or false

[4] Actually, these rules are not removed but substituted for exponentials. This will be explained in section 4

Substructural logics define new kinds of implication connectives and provide an explicit control on the use of formulae. These features reveal appropriate for hypothetical reasoning. In particular, they allow us to define the concept of essential hypothesis:

Definition 1 (Essential Hypothesis). *A hypothesis H is* essential *with respect to a fix a context Δ iff $\Delta, H \vdash_{\mathcal{I}} \psi$ and $\Delta \not\vdash_{\mathcal{I}} \psi$.*

That is, a hypothesis is *essential* if and only if the conclusion cannot be reached if the hypothesis does not hold. The finite resource interpretation of formulae provided by relevant and linear logics allows a logical encoding of essentiality. It should be noted that relevant resources can be reused while linear ones cannot; thus these logics do provide different notions of essentiality. However, in section 5 we present an encoding of relevant implication within intuitionistic linear logic, thus showing the greater expressive power of intuitionistic linear logic. It depends on the nature of the resource which logic is best suited for reasoning about it. On the other hand, affine logic does not support essentiality. Essentiality of hypotheses can neither be easily expressed within intuitionistic logic, where hypotheses are dealt with as reusable building blocks.

Hypothetical reasoning within substructural logics has been successfully applied to natural language processing. Hodas [9] defines a linear logic extension of phrase structure grammars for parsing relative clauses. Tarau *et al.* [5] uses affine logic to define assumption grammars. However, these works are very much of a specific application and do not dwell to the appropriateness of substructural logics for hypothetical reasoning in general. In the following sections we shall introduce intuitionistic linear logic and present an example that demonstrates that intuitionistic linear logic exceeds the capabilities of intuitionistic logic for hypothetical reasoning.

4 Intuitionistic Linear Logic

Linear logic [7] is the substructural logic which results from restricting the use of both the weakening and contraction rules. This turns linear logic into a "resource conscious logic": linear formulae are interpreted as finite resources which cannot be replicated or discarded, but assigned once within a proof. In a sense, linear logic endows each occurrence of a given formula with a certain individuality which distinguishes it, by its very use, from the rest of occurrences.

Dispensing with the weakening and contraction structural rules has an important effect on the definition of the connectives. Consider, for instance, the additive and multiplicative versions of conjunction as defined in intuitionistic logic:

$$\frac{\Delta \vdash A \quad \Delta \vdash B}{\Delta \vdash A \wedge B} \wedge R \qquad\qquad \frac{\Delta_1 \vdash A \quad \Delta_2 \vdash B}{\Delta_1, \Delta_2 \vdash A \wedge B} \wedge R$$

where $\Delta = \Delta_1 \cup \Delta_2$. As shown above, the additive version replicates Δ while the multiplicative version splits Δ. In the presence of weakening and contraction

these two definitions of conjunction are equivalent. However, in linear logic they correspond to two different connectives, the additive & and multiplicative $\otimes$ conjunctions:

$$\frac{\Delta \vdash A \quad \Delta \vdash B}{\Delta \vdash A \,\&\, B} \;\&R \qquad\qquad \frac{\Delta_1 \vdash A \quad \Delta_2 \vdash B}{\Delta_1, \Delta_2 \vdash A \otimes B} \;\otimes R$$

Similarly, intuitionistic linear logic also includes an additive disjunction $\oplus$, an additive falsehood 0, additive $\top$ and multiplicative 1 truths, and a linear implication $\multimap$ which is multiplicative.

The simple removal of the weakening and contraction rules yields a too weak logic where each formula must be used exactly once. Instead, linear logic defines an exponential operator ! (pronounced *bang* or *of course*), and replaces these structural rules by the following specific ones:

$$\frac{\Delta \vdash C}{\Delta, !A \vdash C} \;Weakening \qquad\qquad \frac{\Delta, !A, !A \vdash C}{\Delta, !A \vdash C} \;Contraction$$

thus restoring the capability of weakening and contracting in a controlled way: the formulae explicitly marked with ! are intuitionistic, and hence, behave as reusable building blocks.

In light of this exponential, the intuitionistic implication $A \Rightarrow B$, originally intended as "from the proof of A, I can obtain the proof of B", is reviewed[5] and further decomposed into two primitive connectives yielding $!A \multimap B$. Here, the intended reading of $!A$ is "I have lots of proofs for A" whereas the implication $A \multimap B$ is interpreted as "from one proof of A, I can obtain one proof of B". In the context of hypothetical reasoning, the coexistence of linear and intuitionistic implications allows us to cope with both essential and non-essential hypotheses within the same logic.

The inclusion of the exponential operator ! allows to define a provability-preserving translation – the Girard translation [7] – embedding intuitionistic logic into intuitionistic linear logic. From this follows that the expressive power of intuitionistic linear logic is an extension of that of intuitionistic logic.

Finally, first and higher order linear logics have the typical universal $\forall$ and existential $\exists$ quantifiers, defined as in intuitionistic logic. Figure 1 shows the proof system for intuitionistic linear logic.

5 A Deductive Database with Linear Hypothetical Reasoning

Girard's translation ensures that linear intuitionistic logic extends the expressive power of intuitionistic logic, as every intuitionistic proof can be mapped into a linear one. Thus it follows, for example, that it satisfies Bonner's criteria of rulebase independence [1] which was fulfilled by intuitionistic logic. In addition,

[5] Linear logic actually emerged from a deeper insight into the semantics of the λ-calculus which refined the functional arrow type $A \Rightarrow B$ into $!A \multimap B$.

The axioms:

$$\frac{}{A \vdash A}\ \textit{Identity} \qquad \frac{}{\Delta, 0 \vdash C}\ 0L \qquad \frac{}{\Delta \vdash \top}\ \top R$$

$$\frac{\Delta \vdash C}{\Delta, 1 \vdash C}\ 1L \qquad \frac{}{\vdash 1}\ 1R$$

The additives:

$$\frac{\Delta, A_i \vdash B}{\Delta, A_1 \,\&\, A_2 \vdash B}\ \&\, L_i\ (i = 1, 2) \qquad \frac{\Delta \vdash A \quad \Delta \vdash B}{\Delta \vdash A \,\&\, B}\ \&\, R$$

$$\frac{\Delta, A \vdash C \quad \Delta, B \vdash C}{\Delta, A \oplus B \vdash C}\ \oplus L \qquad \frac{\Delta \vdash A_i}{\Delta \vdash A_1 \oplus A_2}\ \oplus R_i\ (i = 1, 2)$$

$$\frac{\Delta \vdash A \quad \Delta, B \vdash C}{\Delta, A \Rightarrow B \vdash C}\ \Rightarrow L \qquad \frac{\Delta, !A \vdash B}{\Delta \vdash A \Rightarrow B}\ \Rightarrow R$$

The multiplicatives:

$$\frac{\Delta, A, B \vdash C}{\Delta, A \otimes B \vdash C}\ \otimes L \qquad \frac{\Delta_1 \vdash A, \quad \Delta_2 \vdash B}{\Delta_1, \Delta_2 \vdash A \otimes B}\ \otimes R$$

$$\frac{\Delta_1 \vdash A \quad \Delta_2, B \vdash C}{\Delta, A \multimap B \vdash C}\ \multimap L \qquad \frac{\Delta, A \vdash B}{\Delta \vdash A \multimap B}\ \multimap R$$

The exponential:

$$\frac{\Delta \vdash B}{\Delta, !A \vdash B}\ \textit{Weakening} \qquad \frac{\Delta, !A, !A \vdash B}{\Delta, !A \vdash B}\ \textit{Contraction}$$

$$\frac{\Delta, A \vdash B}{\Delta, !A \vdash B}\ !L \qquad \frac{!\Delta \vdash A}{!\Delta \vdash !A}\ !R$$

The quantifiers:

$$\frac{\Delta, A[t/x] \vdash B}{\Delta, \forall x A \vdash B}\ \forall L \qquad \frac{\Delta \vdash A[y/x]}{\Delta \vdash \forall x A}\ \forall R$$

$$\frac{\Delta, A[y/x] \vdash B}{\Delta, \exists x A \vdash B}\ \exists L \qquad \frac{\Delta \vdash A[t/x]}{\Delta \vdash \exists x A}\ \exists R$$

Fig. 1. $\mathcal{ILL}$ Proof System for Intuitionistic Linear Logic

intuitionistic linear logic offers a logical encoding of essential hypotheses, which are out of the scope of intuitionistic logic. In this section we build on a deductive database example originally proposed by Bonner [1, 2]. We show that intuitionistic linear logic is able to express database queries and rules which intuitionistic logic cannot. Our example centers on the graduation and grant policies of a "hypothetical" university. We shall assume a deductive database storing a large number of facts of the form $passes(S, C)$, meaning that student S passed course C. The graduation policy is reflected in a set of rules of the form $grad(St, Su)$ that collects the courses a student St is required to pass for graduating in a certain subject Su. For instance, the rule stating the requirements for graduating in business would have the form below

$$grad(S, business) \circ\!\!- passes(S, market137) \otimes \cdots \otimes passes(S, stat304)$$

where $\circ\!\!-$ stands for reversed linear implication. We shall show the gain of expressive power provided by essential hypotheses by posing hypothetical queries and specifying hypothetical rules which cannot be dealt with non-essential – i.e. intuitionistic – hypotheses. Recall that essentiality of hypotheses is supported by both relevant and linear logics. In the following we shall assume a rather strict university graduation policy: a given course cannot be used to graduate in different subjects, thus linear hypotheses are the right choice for this application.

Assume we want to ask whether Marisol is eligible for graduation in case she passed a given course; e.g. $his102$. We can then pose the hypothetical query

$$\vdash passes(marisol, his102) \Rightarrow grad(marisol, business)$$

However, recall that intuitionistic implication is monotonic and therefore the query above holds even if $\vdash grad(marisol, business)$ does. Thus it is not possible to know whether passing $his102$ actually contributed to Marisol's graduation. Alternatively, we can pose the query using linear implication

$$\vdash passes(marisol, his102) \multimap grad(marisol, business)$$

since the hypothesis $passes(marisol, his102)$ must be used in the deduction, this enforces that passing $his102$ is essential to Marisol's graduation. Of course, both kinds of conditions can be combined in a single query. For instance, the query

$$\vdash passes(marisol, his102) \multimap passes(marisol, lang307) \Rightarrow grad(marisol, business)$$

asks whether Marisol would be eligible for graduation iff she passed $his102$ (mandatory) and if she passed $lang307$ (optional). Most importantly, the use of the same hypothesis as antecedent of both implications leads to a logical encoding of relevant implication within intuitionistic linear logic[6]. In particular, we introduce a new logical symbol $\multimap_R$ for relevant implication, and define it as

$$A \multimap_R B \equiv A \Rightarrow A \multimap B$$

[6] This encoding was previously proposed in [11]

Such an encoding adds relevant hypothetical reasoning to our linear framework. For example, in the query

$$\vdash passes(marisol, his102) \multimap_R grad(marisol, business)$$

the hypothesis $passes(marisol, his102)$ must be used at least once.

In addition to implications, other linear connectives can be used to pose interesting hypothetical queries. For instance, assume that Marisol is within one course of graduation and she is doubtful about attending either $his102$ or $lang307$. Of course, she wants to attend that course leading to her graduation. The use of additive disjunction $\oplus$ and conjunction & can help Marisol to configure her curriculum. The hypothetical query

$$\vdash (passes(marisol, his102) \oplus passes(marisol, lang307)) \multimap grad(marisol, business)$$

asks whether any of the two courses would lead to graduation; thus Marisol can choose on her own. On the other hand, the hypothetical query

$$\vdash (passes(marisol, his102) \& passes(marisol, lang307)) \multimap grad(marisol, business)$$

tells Marisol that maybe only one course meets her graduation requirements[7], thus maybe she is forced to attend a course in particular.

All of the previous examples suppose that we exactly know the courses to be passed by Marisol (i.e. the hypotheses to assume). Now consider we want to know if Marisol is within one course C at most of her graduation. Therefore, we pose our query as follows

$$\vdash \exists C.(passes(marisol, C) \Rightarrow grad(marisol, business))$$

so that the hypothesis $passes(marisol, C)$ can be used several times – including zero – but C is constrained to denote a single definite term within each proof. We could be tempted to universally quantify the hypothesis yielding

$$\vdash \forall C.(passes(marisol, C)) \Rightarrow grad(marisol, business)$$

but this amounts to add the hypothesis $\forall C.(passes(marisol, C))$ to the database, stating that Marisol has passed every course offered by our university. Clearly, this was not the desired effect. Though this example suggests that universally quantified hypotheses are useless for hypothetical reasoning, we shall illustrate that actually they add extra expressiveness when combined with essential hypotheses. By applying essential hypotheses, we can ask whether Marisol is *exactly within one* course of her graduation

$$\vdash \exists C.(passes(marisol, C) \multimap grad(marisol, business))$$

That is, course C is essential to Marisol's graduation. Such a query cannot be posed in intuitionistic logic. Most importantly, the interaction between intuitionistic and linear implications and quantifiers can be exploited to find interesting

[7] The inference mechanism of the deductive database can easily be adapted to show the course(s) actually used in the deduction.

applications for apparently useless universal hypotheses. For instance, the query below

$$\vdash \forall X.(passes(marisol, X)) \Rightarrow passes(marisol, C) \multimap grad(marisol, business))$$

enforces the use of the essential hypothesis $passes(marisol, C)$ whereas the universal hypothesis $\forall X.(passes(marisol, X)$ can be used several times. Hence it asks whether Marisol is *within one course C at least* of her graduation. Note that this linear query is not equivalent to the relevant hypothetical query

$$\vdash passes(marisol, C) \multimap_R grad(marisol, business)$$

as the universal hypothesis $\forall X.(passes(marisol, X))$ can be used for different values of X. The previous queries illustrate both the expressive power of hypothetical reasoning with essential hypotheses and the adequacy of substructural logics to support them. Hypothetical rules can also take advantage of essential hypotheses as illustrated in the following example. Assume that our university grant policy regulates that a student is eligible for a reduced grant iff she meets certain income limit requirements and is within one course of graduation. The hypothetical rule below implements such a regulation

$$reducedGrant(S) \ \circ\!\!- \ incomeLimit(S) \ \otimes \ (passes(S, C) \multimap grad(S, Sb))$$

thus the query $\vdash reducedGrant(marisol)$ asks whether Marisol would be granted to take exactly one course that would lead to her graduation. It should be clear that for this rule to work properly the hypothesis $passes(S, C)$ must be essential; as the otherwise flawed rule

$$reducedGrant(S) \ \circ\!\!- \ incomeLimit(S) \ \otimes \ (passes(S, C) \Rightarrow grad(S, Sb))$$

states that any graduate satisfying $incomeLimit(S)$ would be granted for attending an extra course.

6 Resource Management and Hypothetical Reasoning

One of the main advantages of proof-theoretic approaches is that proof theory and computation have been shown to be closely related subjects. From a theoretical point of view, computational interpretations of proof search and reduction provide a solid foundation for the design of programming languages and elegant logical encodings of computational models such as formal machines and process calculi. From a practical point of view, the automatization of proof search and reduction has been tackled, yielding a variety of well-known and efficient implementation techniques[8]. In this section, we briefly revise a proof system for Lolli [10, 11] – a uniform fragment of intuitionistic linear logic – which embodies an

[8] This is in contrast with model-theoretic approaches which usually are not computationally oriented.

efficient proof search strategy, and apply certain of its proof-theoretic properties to further extend the expressive power of hypothetical reasoning.

It is well-known that proof search in linear logic arises major problems concerning the management of linear formulae. Consider, for example, the rule

$$\frac{\Psi; \Delta_1 \vdash G_1 \qquad \Psi; \Delta_2 \vdash G_2}{\Psi; \underbrace{\Delta_1 \uplus \Delta_2}_{\Delta} \vdash G_1 \otimes G_2} \otimes\ R.$$

When applying this rule bottom-up the context Δ must be split into Δ_1 and Δ_2. The problem here is that we do not know in advance how this split should be done. As the number of possible splits is exponential with respect to the cardinality of Δ, it is quite clear that a trivial implementation which simply backtracks through all these splits is hopelessly inefficient. A commonly adopted solution is to split the contexts lazily; that is, we do not split Δ but pass the whole of it to the proof of G_1 as *input*, which first consumes the formulae it needs – Δ_1 – and then returns the unused formulae – Δ_2 – as *output* to be consumed by the proof of G_2. The strategy outlined is known as the *input-output model of resource consumption* and was first proposed by Hodas and Miller for the Lolli language [10]; and further refined to deal with $\top$ in [11]. Several alternative proof search strategies have been proposed since then [16, 4, 13], however, they all are equivalent in spirit to [10, 11] except for the addition of certain optimizations and the stucture of the sequents and/or contexts involved. On the other hand, it should be noted that the structure of the contexts have a deep impact on the efficiency of this proof search strategy [13].

The $\mathcal{LL}$ proof system was proposed in [13], where it is proved to be sound and complete w.r.t. Lolli, and its algorithmic complexity is measured and carefully compared to those of similar calculi [11, 4]. This proof system uses a new structure we call *frame* to represent returnable input contexts and output contexts, defined as follows

Definition 2 (Frame). *A* frame *is a list of multisets. The empty frame is denoted by the constant* `nil`*, while* $\Delta :: \Pi$ *denotes the frame obtained by concatenating the multiset* Δ *and the frame* Π*.*

The formulation of our frame based resource management system requires the definition of a number of relations and operators on frames. The equality on frames is recursively defined upon the equality of multisets:

1. $\texttt{nil} = \texttt{nil}$
2. $\Delta_1 :: \Pi_1 = \Delta_2 :: \Pi_2 \Leftrightarrow (\Delta_1 = \Delta_2) \wedge (\Pi_1 = \Pi_2)$

The inclusion relation is recursively defined upon the inclusion relation on multisets; that is:

1. $\texttt{nil} \sqsubseteq \texttt{nil}$
2. $\Delta_1 :: \Pi_1 \sqsubseteq \Delta_2 :: \Pi_2 \Leftrightarrow (\Delta_1 \subseteq \Delta_2) \wedge (\Pi_1 \sqsubseteq \Pi_2)$

Given two frames, their union is defined upon the multiset union $\uplus$ as follows:

$$\frac{}{\Psi;\Delta;\Pi \to \top \;/\; \Pi;1}\mathcal{LL}\ \top\text{-R} \qquad \frac{}{\Psi;\emptyset;\Pi \to 1 \;/\; \Pi;0}\mathcal{LL}\ 1\text{-R}$$

$$\frac{\Psi;\Delta;\Pi \to G_1 \;/\; \Pi';0 \quad \Psi;\Delta \uplus (\Pi - \Pi');\mathtt{nil} \to G_2 \;/\; \mathtt{nil};d}{\Psi;\Delta;\Pi \to G_1 \,\&\, G_2 \;/\; \Pi';0}\mathcal{LL}\ \&_0\text{-R}$$

$$\frac{\Psi;\Delta;\Pi \to G_1 \;/\; \Pi';1 \quad \Psi;\Delta \uplus (\Pi - \Pi');\Pi' \to G_2 \;/\; \Pi'';d}{\Psi;\Delta;\Pi \to G_1 \,\&\, G_2 \;/\; \Pi'';d}\mathcal{LL}\ \&_1\text{-R}$$

$$\frac{\Psi;\emptyset;\mathtt{nil} \to G \;/\; \mathtt{nil};d}{\Psi;\emptyset;\Pi \to !G \;/\; \Pi;0}\mathcal{LL}\ !\text{-R} \qquad \frac{\Psi;\Delta;\Pi \to G_i \;/\; \Pi';d}{\Psi;\Delta;\Pi \to G_1 \oplus G_2 \;/\; \Pi';d}\mathcal{LL}\ \oplus\text{-R}\ (i=1,2)$$

$$\frac{\Psi;\Delta \uplus \{D\};\Pi \to G \;/\; \Pi';d}{\Psi;\Delta;\Pi \to D \multimap G \;/\; \Pi';d}\mathcal{LL}\ \multimap\text{-R} \qquad \frac{\Psi \cup \{D\};\Delta;\Pi \to G \;/\; \Pi';d}{\Psi;\Delta;\Pi \to D \Rightarrow G \;/\; \Pi';d}\mathcal{LL}\ \Rightarrow\text{-R}$$

$$\frac{\Psi;\emptyset;\Delta :: \Pi \to G_1 \;/\; \Delta' :: \Pi';0 \quad \Psi;\Delta';\Pi' \to G_2 \;/\; \Pi'';d}{\Psi;\Delta;\Pi \to G_1 \otimes G_2 \;/\; \Pi'';d}\mathcal{LL}\ \otimes_0\text{-R}$$

$$\frac{\Psi;\emptyset;\Delta :: \Pi \to G_1 \;/\; \Delta' :: \Pi';1 \quad \Psi;\emptyset;\Delta' :: \Pi' \to G_2 \;/\; \Delta'' :: \Pi'';d}{\Psi;\Delta;\Pi \to G_1 \otimes G_2 \;/\; \Pi'';1}\mathcal{LL}\ \otimes_1\text{-R}$$

$$\frac{\Psi;\Delta;\Pi \to G[t/x] \;/\; \Pi';d}{\Psi;\Delta;\Pi \to \exists x.G \;/\; \Pi';d}\mathcal{LL}\ \exists\text{-R} \qquad \frac{\Psi;\Delta;\Pi \to G[y/x] \;/\; \Pi';d}{\Psi;\Delta;\Pi \to \forall x.G \;/\; \Pi';d}\mathcal{LL}\ \forall\text{-R}$$

$$\frac{\Psi \cup \{D\};\Delta;\Pi \overset{D}{\to} A \;/\; \Pi';d}{\Psi \cup \{D\};\Delta;\Pi \to A \;/\; \Pi';d}\mathcal{LL}\ \textbf{decide}\ \Psi \qquad \frac{\Psi;\Delta;\Pi \overset{D}{\to} A \;/\; \Pi';d}{\Psi;\Delta \uplus \{D\};\Pi \to A \;/\; \Pi';d}\mathcal{LL}\ \textbf{decide}\ \Delta$$

$$\frac{\Psi;\Delta;\Pi \overset{D}{\to} A \;/\; \Pi';d}{\Psi;\Delta;\Pi \triangleleft D \to A \;/\; \Pi';d}\mathcal{LL}\ \textbf{decide}\ \Pi$$

$$\frac{}{\Psi;\emptyset;\Pi \overset{A}{\to} A \;/\; \Pi;0}\mathcal{LL}\ \textbf{initial}$$

$$\frac{\Psi;\Delta;\Pi \overset{D[t/x]}{\to} A \;/\; \Pi';d}{\Psi;\Delta;\Pi \overset{\forall x.D}{\to} A \;/\; \Pi';d}\mathcal{LL}\ \forall\text{-L} \qquad \frac{\Psi;\Delta;\Pi \overset{D_i}{\to} A \;/\; \Pi';d}{\Psi;\Delta;\Pi \overset{D_1 \& D_2}{\to} A \;/\; \Pi';d}\mathcal{LL}\ \&\text{-L}(i=1,2)$$

$$\frac{\Psi;\Delta;\Pi \overset{D}{\to} A \;/\; \Pi';d_1 \quad \Psi;\emptyset;\mathtt{nil} \to G \;/\; \mathtt{nil};d_2}{\Psi;\Delta;\Pi \overset{G \Rightarrow D}{\to} A \;/\; \Pi';d_1}\mathcal{LL}\ \Rightarrow\text{-L}$$

$$\frac{\Psi;\emptyset;\Delta :: \Pi \overset{D}{\to} A \;/\; \Delta' :: \Pi';0 \quad \Psi;\Delta';\Pi' \to G \;/\; \Pi'';d}{\Psi;\Delta;\Pi \overset{G \multimap D}{\to} A \;/\; \Pi'';d}\mathcal{LL}\ \multimap_0\text{-L}$$

$$\frac{\Psi;\emptyset;\Delta :: \Pi \overset{D}{\to} A \;/\; \Delta' :: \Pi';1 \quad \Psi;\emptyset;\Delta' :: \Pi' \to G \;/\; \Delta'' :: \Pi'';d}{\Psi;\Delta;\Pi \overset{G \multimap D}{\to} A \;/\; \Pi'';1}\mathcal{LL}\ \multimap_1\text{-L}$$

Fig. 2. Lolli proof system $\mathcal{LL}$

1. $\texttt{nil} \sqcup \texttt{nil} =_{def} \texttt{nil}$
2. $\Delta_1 :: \Pi_1 \sqcup \Delta_2 :: \Pi_2 =_{def} (\Delta_1 \uplus \Delta_2) :: (\Pi_1 \sqcup \Pi_2)$

The multiset difference of two frames is also defined:

1. $\texttt{nil} - \texttt{nil} =_{def} \emptyset$
2. $\Delta_1 :: \Pi_1 - \Delta_2 :: \Pi_2 =_{def} (\Delta_1 \backslash \Delta_2) \uplus (\Pi_1 - \Pi_2)$

where $\backslash$ denotes the multiset difference. Finally, given a frame Π and a formula F, $\Pi \triangleleft F$ denotes a frame obtained by inserting one occurrence of F into a multiset of the frame Π.

Figure 2 shows the $\mathcal{LL}$ proof system. The sequents of this system have the form

$$\Psi; \Delta; \Pi \rightarrow G \ / \ \Pi'; d \quad \text{and} \quad \Psi; \Delta; \Pi \stackrel{D}{\rightarrow} G \ / \ \Pi'; d$$

These sequents include input and output portions separated by the symbol $/$. Ψ is a set of reusable hypotheses whose content is augmented through intuitionistic implication. Δ is a multiset of essential hypotheses which are added via linear implication. Π and Π' are frames not affected by the use of implications: their contents are automatically updated by the $\mathcal{LL}$ rules to implement the input/output model of resource consumption. In particular, Π contains returnable input; that is, hypotheses whose use is optional, while Π' is intended to contain the output; that is, the portion of Π which was not used to prove G. Finally, d is a boolean flag for managing $\top$ lazily, which is irrelevant for our current purposes. The technical details of $\mathcal{LL}$ are subtle and the interested reader is referred to [13]. However, in the following we shall employ two important properties of $\mathcal{LL}$-proofs. The inclusion proposition states that scope of formulae is preserved.

Proposition 1 (inclusion).

1. *If* $\Psi; \Delta; \Pi \rightarrow G \ / \ \Pi'; d$ *then* $\Pi' \sqsubseteq \Pi$
2. *If* $\Psi; \Delta; \Pi \stackrel{D}{\rightarrow} G \ / \ \Pi'; d$ *then* $\Pi' \sqsubseteq \Pi$

The augmentation lemma below – necessary to prove the completeness of $\mathcal{LL}$ w.r.t. Lolli – states that input contexts of provable sequents can arbitrarily be augmented without compromising provability; that is:

Lemma 1 (augmentation of input).

1. *if* $\Psi; \Delta; \Pi \rightarrow G \ / \ \Pi'; 0$ *then* $\Psi, \bar{\Psi}; \Delta; \Pi \sqcup \bar{\Pi} \rightarrow G \ / \ \Pi' \sqcup \bar{\Pi}; 0$
2. *if* $\Psi; \Delta; \Pi \stackrel{D}{\rightarrow} G \ / \ \Pi'; 0$ *then* $\Psi, \bar{\Psi}; \Delta; \Pi \sqcup \bar{\Pi} \stackrel{D}{\rightarrow} G \ / \ \Pi' \sqcup \bar{\Pi}; 0$
3. *if* $\Psi; \Delta; \Pi \rightarrow G \ / \ \Pi'; 1$ *then* $\Psi, \bar{\Psi}; \Delta, \bar{\Delta}; \Pi \sqcup \bar{\Pi} \rightarrow G \ / \ \Pi' \sqcup \bar{\Pi}; 1$
4. *if* $\Psi; \Delta; \Pi \stackrel{D}{\rightarrow} G \ / \ \Pi'; 1$ *then* $\Psi, \bar{\Psi}; \Delta, \bar{\Delta}; \Pi \sqcup \bar{\Pi} \stackrel{D}{\rightarrow} G \ / \ \Pi' \sqcup \bar{\Pi}; 1$

For all $\bar{\Psi}, \bar{\Delta}$, *and* $\bar{\Pi}$

Resource management systems [4] emerged to improve the efficiency of proof search in linear logic; and hence, they have been thus far considered no more than sophisticated implementation techniques. However, in this section we show that,

by exploiting certain proof-theoretic properties of resource management systems, it is possible to find new applications for them far beyond implementation issues. In particular, we can further extend the expressive power of hypothetical reasoning by allowing to set the initial contents of Π when posing a query. The formal results above assure that the use of those hypotheses is optional and that they are correctly handled by the resource management system. In particular, the augmentation lemma ensures that hypotheses so added to Π can either be used or returned without compromising provability, while the inclusion lemma ensures that the output can only contain some of those hypotheses. It should be noted that these hypotheses are provided when posing the query only. However, it is also possible to add a new connective $\multimap_\Pi$ to $\mathcal{LL}$ to allow augmenting the contents of Π at any proof step as follows

$$\frac{\Psi; \Delta; \{D\} :: \Pi \to G \ / \ _ :: \Pi'; d}{\Psi; \Delta; \Pi \to D \multimap_\Pi G \ / \ \Pi'; d}$$

At first sight, D can be used at most once and hence this connective seems to be a convenient encoding of affine implication in $\mathcal{LL}$, since it requires no modification to the resource management system. In addition, a $\mathcal{LL}$ **decide** $!\,\Pi$ rule can also be added to deal with !'ed hypotheses

$$\frac{\Psi \cup \{D\}; \Delta; \Pi \stackrel{!D}{\to} A \ / \ \Pi'; d}{\Psi; \Delta; \Pi \triangleleft !\,D \to A \ / \ \Pi'; d}\mathcal{LL}\,\textbf{decide}\ !\,\Pi$$

so that $!\,D \multimap_\Pi$ defines a hypothesis D which is reusable but is returned if it is never used. Note that this behavior is not captured by any of the aforementioned implications. However, for the rest of the paper we shall assume that optional hypotheses can only be added at the initial query.

Thus when posing a query, we are not forced to decide a priori whether a hypothesis is essential or not. On the contrary, the $\mathcal{LL}$ proof will use the hypotheses added to Π as needed and the final output will contain the irrelevant hypotheses Π'. For example, the hypothetical query below

$$\Psi; \Delta; \{h_1, h_2\} :: \texttt{nil} \to grad(marisol, business) \ / \ \Pi' :: \texttt{nil}; d$$

where h_1, h_2 stand for $passes(marisol, his102)$ and $passes(marisol, lang307)$ respectively, asks whether passing $his102$ and/or $lang307$ contributed to Marisol's graduation. An empty Π' means that Marisol is forced to pass both courses, while $\Pi' = \{his102\}$ means that Marisol is required to pass $lang307$ only. The development of interactive automated tools similar to proof editors is suggested by the fact that the final output Π' and the output of intermediate proof tree nodes do provide a trace of the use of hypotheses along the deduction.

7 Conclusions and Further Work

In this paper, we have introduced the concept of *essential hypothesis* and have shown it provides a considerable gain of expressive power for hypothetical reasoning. We have also proved that intuitionistic logic is unable to deal with essential

hypotheses, while relevant and linear logics provide a natural support for them. In particular, intuitionistic linear logic is shown to express both linear and relevant hypothetical reasonings. We have also shown that resource management systems provide facilities unrelated to that of efficiency. We intend to extend these ideas concerning the addition of the new connective in such a way that the $\mathcal{LL}$ resource management system deals with affine, relevant, and linear resources.

References

1. A.J. Bonner. A Logic for Hypothetical Reasoning. *Proceedings of the Seventh National Conference on Artificial Intelligence*, pp. 480–484, 1988. Published in expanded form as Technical Report TR-DCS-230, Department of Computer Science, Rutgers University.
2. A.J. Bonner, L. McCarty, and K. Vadaparty. Expressing Database Queries with Intuitionistic Logic. *Proceedings of the North America Conference on Logic Programming (NACLP)*, pp. 831–850, 1989.
3. A.J. Bonner. Hypothetical Datalog: Complexity and expressibility. *Theoretical Computer Science*, 76:3–51, 1990.
4. I. Cervesato, J. Hodas, and F. Pfenning. Efficient Resource Management for Linear Logic Proof Search. *Proceedings of the 1996 International Workshop on Extensions of Logic Programming*, 1996.
5. V. Dahl, A. Fall, S. Rochefort, and Paul Tarau. A Hypothetical Reasoning Framework for NL Processing. *Proceedings of 8th IEEE International Conference on Tools with Artificial Intelligence*,1996.
6. D.M. Gabbay, and U. Reyle. N-Prolog: An extension of Prolog with hypothetical implications, I. *Journal of Logic Programming*, 1(4):319–355, 1984.
7. J.-Y. Girard. Linear Logic. *Theoretical Computer Science*, 50, pp. 1–102. 1987.
8. J.-Y. Girard. Logic and Exceptions: A Few Remarks. *Journal of Logic and Computation*, vol. 2, num. 2, pp. 111-118. 1992.
9. J.S. Hodas. Specifying Filler-Gap Dependency Parsers in a Linear-Logic Programming Language, *Proceedings of the 1992 Joint International Conference and Symposium on Logic Programming*, 1992.
10. J.S. Hodas and D. Miller. Logic Programming in a Fragment of Intuitionistic Linear Logic. *Information and Computation*, 110(2), pp. 327–365, 1994.
11. J.S. Hodas. *Logic Programming in Intuitionistic Linear Logic: Theory, Design, and Implementation*. PhD Thesis, University of Pennsylvania, 1994.
12. P. Lincoln. Linear Logic. *SIGACT*, 1992.
13. P. Lopez, and E. Pimentel. Resource Management in Linear Logic Proof Search Revisited. *Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning*, LNAI 1705, pp. 304–319. 1999.
14. L.T. McCarty. A Language for Legal Discourse. I. Basic Features. *Proceedings of the Second International Conference on Artificial Intelligence and Law*, pp. 180–189, 1989.
15. D. Miller. Proof Theory as an Alternative to Model Theory. *Newsletter of the ALP*, 1991.
16. M. Winikoff and J. Harland. Implementing the Linear Logic Programming Language Lygon. In J. Lloyd, editor, *Proceedings of the 1995 International Logic Programming Symposium*, pp. 66–80, 1995.