Defeasible Inheritance Networks and Linear Logic: Horn Fragments and Proof Nets

Ryo Takemura

Nihon University, Japan.
takemura.ryo@nihon-u.ac.jp

May 6, 2012

Abstract

Based on a remark of Girard (1992) and a formalization given by Fouqueré and Vauzeilles (1994), we investigate a formalization of defeasible inheritance reasoning in terms of linear logic. We show a correspondence between defeasible inheritance networks, a Horn fragment of linear logic, and proof nets for linear logic. In particular, we show a structural correspondence between defeasible inheritance networks and proof nets for our fragment of linear logic.

Contents

1 Introduction **1**

2 Defeasible inheritance networks **3**

3 A Horn fragment of linear logic and defeasible inheritance networks **5**
3.1 Intuitionistic linear logic IMLL and its Horn fragment 6
3.2 Translation of DI-nets into a Horn fragment of IMLL 8

4 Proof nets for linear logic and defeasible inheritance networks **13**
4.1 Proof nets for IMLL (Essential nets) 13
4.2 Transformation of proof nets into DI-nets 15

5 Conclusion and future work **18**

1 Introduction

Linear logic was introduced in [Girard 1987], and its theoretical studies have been much developed in the last couple of decades. Recently, it has been further applied to studies not only in computer science but also in various areas such as quantum physics and molecular biology. Linear logical structure is considered a refined basic logical structure for traditional logic such as classical logic and intuitionistic logic. In particular, it is well known that the duality principle of classical logic and the constructive nature of intuitionistic logic, which are considered to be incompatible in the traditional framework, are compatible in the basic logical structure that is revealed by linear logic. Thus, linear logic has provided new insight into various issues discussed in the framework of traditional logic such as relationships between intuitionistic logic and classical logic, and between syntax and semantics.

In linear logic, the traditional logical structure of propositions and connectives is divided into two parts: "linear" and "stable," cf. [Okada 2008]. Traditional logic, such as classical and intuitionistic logic, is captured in the stable part with the use of exponential modality operators "!" and "?." In contrast, the implication $-\circ$, called linear implication, in the linear part naturally captures the notion of state transition, which is known to be difficult to formalize in traditional logic. Thus, the basic part of linear logic has been applied in studying state transition systems, AI planning, etc. See, for example, [Kanovich-Vauzeilles 2001, Masseron-Tollu-Vauzeilles 1993, Pfenning 2008]. By decomposing the traditional logical structure, linear logic provides a framework in that various logical systems, studied in terms of traditional logic, are uniformly characterized as its subsystems. Furthermore, for such a variety of subsystems, efficient proof-search/proof-construction strategies and their computational complexities have been extensively studied, and various logic programming languages have been developed. See, e.g., [Lincoln 1995, Miller 1995] for surveys. Also, in view of the semantics, various models have been proposed for linear logic. In particular, not only traditional models that characterize the notion of provability but also models at a profounder level that characterize the notion of proofs and also of computation have been developed in the framework of linear logic.

Among these various characteristics and applications of linear logic, we focus here on its wealth of expressive power. By decomposing the traditional logical structure, even the most basic part of linear logic, there are almost twice as many logical connectives as those in traditional logic. This means that we are able to express more subtle differences in meaning of sentences by linear logic than by traditional logic. In particular, we are able to formalize various negations in linear logic without adding nor modifying inference rules. For example, the most basic linear negation, expressed by the symbol $(\ )^{\perp}$, is not necessarily characterized in terms of the contradiction, but is characterized in terms of the De Morgan duality. The intuitionistic negation $\neg A$, which means "the contradiction is derived from A", can be expressed as $!A \multimap \mathbf{0}$ in linear logic. The negation in classical logic, which is characterized by the boolean complement, also can be expressed as $?(A^{\perp})$. Cf. [Girard 1987]. Furthermore, we are able to express a negation meaning the lack of a property A as $A \multimap \mathbf{1}$, in which $\mathbf{1}$ is a propositional constant of linear logic that is neutral with respect to the (multiplicative) conjunction, i.e., "$\mathbf{1}$ 'and' A" is equivalent to "A." This negation expressing the lack of a property is remarked on in [Girard 1992], and formalized in [Fouqueré-Vauzeilles 1993, Fouqueré-Vauzeilles 1994] to apply linear logic to the study of defeasible inheritance reasoning.

In the system of [Fouqueré-Vauzeilles 1994], defeasible inheritance networks, which are called taxonomic networks with exceptions, are characterized in a fragment of intuitionistic multiplicative exponential linear logic called taxonomic linear theories. We observe that in the system, each atomic formula is triplicated with the use of $+$ and $-$ signs: an unsigned usual atom A, a positive-signed atom A^{+}, and a negative signed atom A^{-}. Although the negative-sign can be considered informally to express a certain kind of negation, it is not exactly so as remarked in [Fouqueré-Vauzeilles 1994]. Furthermore, the relationship between an unsigned atom A and a positive-signed atom A^{+} is not clear. Thus, it seems to be difficult to give semantic meaning for these $+, -$ signs, and they seem to be just an *ad hoc* syntactic convention.

In this paper, we characterize the defeasible inheritance networks of [Horty 1994] by a well-established fragment of linear logic, that is, by a "Horn fragment" of linear logic, which is extensively studied in [Kanovich 1992, Kanovich 1994], without changing the language and inference rules but just restricting the language of linear logic. It is, of course, not possible

in traditional classical logic, and linear logic works effectively for that purpose. Furthermore, characterizing defeasible inheritance networks by only restricting the language of linear logic, we are able to apply well-developed studies of linear logic such as semantic frameworks straightforwardly to the study of defeasible inheritance reasoning. Compared with the system of [Fouqueré-Vauzeilles 1994], our system is simple with respect to the following points: (1) our atoms are those of the usual linear logic without introducing any syntactic decorations; (2) our system is a fragment of the most basic "exponential-free" linear logic, i.e., multiplicative linear logic.

Defeasible inheritance reasoning is regarded as a type of nonmonotonic reasoning, which is formalized, for example, by using [Reiter 1980]'s Default logic. The author aims to apply linear logical studies not only to defeasible inheritance reasoning but also to more general nonmonotonic logic. As a first step toward to this end, we investigate a relationship among defeasible inheritance networks, a Horn fragment of linear logic, and proof nets for linear logic (or essential nets of [Lamarche 1994]). In particular, we show a structural correspondence between defeasible inheritance networks and proof nets for our fragment of linear logic. Proof nets, which were introduced in [Girard 1987] as a graph-theoretical representation of logical proofs, are extensively studied and applied to theory of computation, and hence we are able to apply such results to the study of defeasible inheritance reasoning.

The rest of this paper is organized as follows. In Section 2, we review the defeasible inheritance network and define our notion of reachability in a network. In Section 3, we review the most basic Horn fragment of linear logic, and give a translation of defeasible inheritance networks into our Horn fragment. In Section 4, we briefly review proof nets for intuitionistic multiplicative linear logic. We then show that there is a structural correspondence between our proof nets and defeasible inheritance networks, by giving a transformation of the proof nets. In Section 5, we summarize our results, and discuss future work.

2 Defeasible inheritance networks

In this section, we review the defeasible inheritance network of [Horty 1994], and we define our notion of reachability in a network.

Definition 2.1 (Defeasible inheritance network) A **defeasible inheritance network** (or, **DI-net** for short) is a labeled finite directed acyclic graph $D = (D, \rightarrow, \not\rightarrow)$, where:

- D is a nonempty set of labeled nodes,
- $\rightarrow$ is a directed edge called **defeasible edge**,
- $\not\rightarrow$ is a directed edge called **defeasible negative edge**.

We usually do not distinguish a node and its label, and we refer to a node by its label.

We give an informal interpretation of a DI-net. $a \rightarrow b$ is read as "typical a is b" or, "it is most natural to suppose that a is b." $\rightarrow$ is interpreted as a reflexive and transitive relation: we have "typical a is a," and we obtain "typical a is c" when there exists edges $a \rightarrow b$ and $b \rightarrow c$.

$a \not\rightarrow b$ is read as "typical a is not b" or, "it is most natural to suppose that a is not b." In contrast to $\rightarrow$ above, $\not\rightarrow$ is not transitive in general: we do not generally have "typical a is not c," from edges $a \not\rightarrow b$ and $b \not\rightarrow c$. Note also that we have "typical a is not c" from

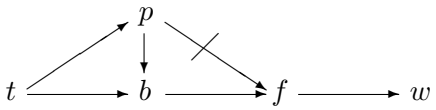
$a \to b$ and $b \not\to c$, but we do not generally have it from $a \not\to b$ and $b \to c$. For example, from "penguins do not fly" and "flying things have wings," we do not have "penguins do not have wings."

Definition 2.2 (Path) We define a **path** in a DI-net as a sequence of different (labels of) nodes connected by $\to$-edges and $\not\to$-edges. We denote a path consisting only of $\to$-edges by $a \twoheadrightarrow b$.
We define the **length** of a path as the number of nodes that constitute the path.

To define the notion of reachability, let us consider the following well-known example of Tweety.

Example 2.3 (Tweety) Assume first that we know "Tweety (t) is a bird (b)," "Birds (b) fly (f)," and "Flying things have wings (w)." Those facts are represented by the following DI-net.

Then, further assume that we obtain the following facts: "Tweety is a penguin (p)", "Penguins are birds," and "Penguins do not fly."



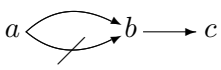
Observe that, in the above DI-net D, we have "Tweety flies" and "Tweety does not fly." In such a case, we say that the node f is conflicting with respect to t, and we assume that nothing can be implied about t (Tweety) with respect to f (flying things), i.e., we cannot say anything about whether Tweety can fly or not. In other words, Tweety lacks certitude with property of "fly," i.e., Tweety has neither the property "fly" nor its negation "not fly."

Definition 2.4 (Conflicting node) When we have $a \twoheadrightarrow b$, and we have $a \not\to b$ or $a \twoheadrightarrow c \not\to b$ for some c in a DI-net, we say that b is a **conflicting node** with respect to a.

How about the property of "has wings (w)" in the above DI-net D? For our characterization of DI-nets in linear logic, we assume that "Tweety has wings" holds. This is because we are able to consider that the property "has wings" is inherited from the property "is a bird" via the property "fly", even if the property "fly" turns out to be in conflict.

Although our assumption may obtain some understanding in the above example, it may be more appropriate to assume that if Tweety does not have the property "fly" then so does any property that is inherited from "fly." For example, if we replace the above property "has wings" by "takes flight," it is natural to conclude "Tweety does not take flight." In fact, [Fouqueré-Vauzeilles 1994] adopted such an interpretation in their basic system, and hence, Tweety does not have properties "fly" or "has wings." We may call such interpretations of DI-nets "strict" interpretation, and our interpretation "weak."

In general, when we have the following form of DI-net,



it seems to be difficult, without considering each concrete situation, to decide between a implies c holds (weak) and not (strict). This difficulty is avoided in the study of defeasible inheritance reasoning or default logic with the use of the essentially model-theoretic notion of *extension*. However, instead of using such a model-theoretic notion, we can assume, when a does not imply c (i.e., $a \not\to c$) actually holds, that it is given in the set of assumptions from the beginning. On the other hand, when a implies c (i.e., $a \to c$) actually holds, although we need to add it to the assumptions under the strict interpretation, we do not need such an arrangement under the weak interpretation. Thus, the weak interpretation is more economical than the strict interpretation, and hence, we study the weak interpretation in this paper. In Section 5, we briefly discuss how our characterization of DI-nets may be extended to those DI-nets with strict interpretation.

Based on the above example, we formally define our notion of reachability in a DI-net as follows.

Definition 2.5 (Reachability) Let $D = (D, \to, \not\to)$ be a DI-net. Let $\overline{D}$ be the set $\{\overline{a} \mid a \in D\}$ of overlined labels of nodes of D. For each node $a \in D$, we define a set $reach(a) \subseteq D \cup \overline{D}$ of labels and overlined labels those are **reachable from** a as follows:

1. $a \in reach(a)$
2. If $a \twoheadrightarrow b$ in D, and if there is no path such that $a \twoheadrightarrow c \not\to b$ for some c or $a \not\to b$ in D, then $b \in reach(a)$.
3. If $a \twoheadrightarrow c \not\to b$ for some c or $a \not\to b$ in D, and if there is no path such that $a \twoheadrightarrow b$ in D, then $\overline{b} \in reach(a)$.

When $reach(a) = \{b_1, \ldots, b_n, \overline{c_1}, \ldots, \overline{c_m}\}$, it is interpreted as "$a$ has properties b_1 and ... and b_n, and not c_1 and ... and not c_m."

Example 2.6 (Reachability) For example, in Example 2.3 of Tweety, $reach(t) = \{t, p, b, w\}$, and hence "Tweety is a penguin, is a bird, and has wings."
In the following Fig. 1, we have $reach(a) = \{a, c\}$. In Fig. 2, g is not reachable from d and $reach(d) = \{d, e, \overline{f}\}$.

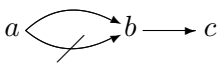


Fig. 1

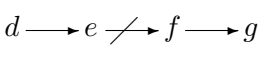


Fig. 2

3 A Horn fragment of linear logic and defeasible inheritance networks

Linear logical structure is considered a refined basic logical structure for traditional logic such as classical logic and intuitionistic logic. In linear logic, the traditional logical structure of propositions and connectives is divided into a linear part and a stable part, cf. [Okada 2008]. Although a formula in linear logic has various informal interpretations, we may interpret

it to denote a changeable state. Then, a traditional proposition may be regarded as an unchangeable state, and it is characterized in the stable part of linear logic with the use of exponential modality operators "!" and "?." With this decomposition of propositions in linear logic, logical connectives are also decomposed into two parts. See [Okada 2008] for a philosophical explanation. In particular, the implication $A \multimap B$, called linear implication, in the linear part can be naturally read as "the state A changes the state B" (and after B is realized, A disappears and does not hold any more). Thus, the linear implication naturally captures the notion of state transition, which is known to be difficult to formalize in traditional logic. [Girard 1987] is the main reference for linear logic, but introductions may be found, for example, in [Girard 1995], and for more comprehensive introductions, see [Curien 2005, Troelstra 1992]. For philosophical aspects of linear logic, see [Okada 2008].

In Section 3.1, we briefly review the syntax of intuitionistic multiplicative linear logic IMLL, in which formulas are constructed by $\otimes$ (the multiplicative conjunction), $\multimap$ (the linear implication), and $\mathbf{1}$ (a propositional constant). IMLL is a fragment of classical multiplicative linear logic MLL (consisting of the above connectives and $\bot$), that is, inference rules of IMLL is obtained by those of MLL by restricting each sequent to have at most one formula on the conclusion side of the sequent. Based on the cut-elimination theorem, IMLL is equivalent to MLL in our fragment: a Horn sequent is provable in IMLL if and only if it is provable in MLL. This is because each cut-free proof of MLL for a given Horn sequent is exactly that of IMLL, since our fragment does not contain the usual negation $(\ )^{\bot}$ (nor the constant $\mathbf{0}$). (See, for example, [Kanovich 1994], whose discussion is naturally extended to our fragment.) Hence we identify them, and instead of classical MLL, we here introduce the simpler intuitionistic IMLL. We further review the most basic Horn fragment of linear logic, for which properties, computational complexity, and applications are extensively studied in [Kanovich 1992, Kanovich 1994]. In Section 3.2, we give a translation of DI-nets into our Horn fragment.

3.1 Intuitionistic linear logic IMLL and its Horn fragment

Definition 3.1 (Formulas and inference rules of IMLL) Formulas of IMLL, i.e., the $\{\otimes, \multimap, \mathbf{1}\}$-fragment of LL and the sequent calculus inference rules of IMLL are defined as in Table 1. We assume that a sequence of formulas Γ is a multiset, and hence, the exchange rule is assumed implicit in the above inference rules. We usually will omit the overline above the ax- and $\mathbf{1}$-rules.

$$A, B \quad ::= \quad \mathbf{1} \mid a \mid A \otimes B \mid A \multimap B$$

$$\frac{}{A \vdash A}\ ax \qquad \frac{\Gamma \vdash A \quad A, \Delta \vdash C}{\Gamma, \Delta \vdash C}\ cut \qquad \frac{}{\vdash \mathbf{1}}\ \mathbf{1} \qquad \frac{\Gamma \vdash C}{\mathbf{1}, \Gamma \vdash C}\ \mathbf{1}L$$

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

Table 1 Formulas and inference rules of IMLL

We extend slightly the most basic Horn fragment of [Kanovich 1994], including only the above connectives, by introducing negative atoms of the form $\overline{a} := a \multimap \mathbf{1}$ (cf. [Girard 1992]). It is well known that the usual intuitionistic negation $\neg a$ is defined as $a \to \bot$, which means

"the contradiction is derived from a." Hence, with this negation, we have:

$$a \wedge (a \to \bot) \wedge b \implies \bot \wedge b \Leftrightarrow \bot$$

On the other hand, with our negation $a \multimap \mathbf{1}$, we have:

$$a \otimes (a \multimap \mathbf{1}) \otimes b \implies \mathbf{1} \otimes b \Leftrightarrow b$$

since $\mathbf{1}$ is neutral with respect to the multiplicative conjunction $\otimes$. Thus, $a \multimap \mathbf{1}$ cancels a without leading to the contradiction, and we may say that the negation $a \multimap \mathbf{1}$ expresses "the lack of the property a."

Definition 3.2 (Literal) Our positive and negative atoms, and literals are defined as follows.

- **Positive atoms** are $a, b, c, \ldots$.
- A **negative atom** is defined as $\overline{a} := a \multimap \mathbf{1}$,

  where $\mathbf{1}$ is a constant of linear logic which is the unit of the $\otimes$ connective, i.e., $A \otimes \mathbf{1} = \mathbf{1} \otimes A = A$.

  We assume that the constant $\mathbf{1}$ appears only in the above form $a \multimap \mathbf{1}$.
- Positive atoms and negative atoms are collectively called **literals**, and a literal is denoted by $L, M, N, \ldots$. Multisets of literals, such as $L_1 \cdots L_n$ (which is also written as $L_1, \ldots, L_n$ particularly when it appears on the left-hand side of a sequent), are denoted by $\vec{L}, \vec{N}, \vec{M}, \ldots$.

  In particular, a multiset $L \cdots L$ of n literals formed from the same literal L is denoted by L^n.

Under associativity and commutativity of the $\otimes$ connective, we identify the multiset of literals $L_1 \cdots L_n$ (or $L_1, \ldots, L_n$) and the tensor product $L_1 \otimes \cdots \otimes L_n$. Thus we do not explicitly treat the $\otimes$ connective in what follows.

Definition 3.3 (Program formula and Horn sequent) A formula of the form $\vec{L} \multimap \vec{M}$ is called a **program formula**, and a set of program formulas, called a **program set**, is denoted by $\mathcal{P}, \mathcal{Q}, \mathcal{P}_1, \mathcal{P}_2, \ldots$.
For a program set $\mathcal{P}$, and multiset of literals $\vec{L}$ and $\vec{M}$, a sequent of the form $\mathcal{P}, \vec{L} \vdash \vec{M}$ is called a **Horn sequent**.

Note that our negative atom $\overline{a}$ may occur in any position in a Horn sequent generally. However, as we will find later in Definition 3.5 and Theorem 4.3, occurrences of $\overline{a}$ are restricted to the succedent of a program formula, and to the conclusion side of a Horn sequent.

Example 3.4 (Sequent calculus proof) We have the following sequent calculus proof of the Horn sequent $t, t \multimap tp, p \multimap pb, b \multimap bf, f \multimap fw, p \multimap p\overline{f} \vdash tpbw$, which corresponds to the DI-net D in Example 2.3 of Tweety. The sequent expresses that under the knowledge of the

program set, we know that "Tweety is a penguin, is a bird, and has wings."

$$\dfrac{t \vdash t \quad \dfrac{p \vdash p \quad \dfrac{b \vdash b \quad \dfrac{f \vdash f \quad \dfrac{p \vdash p \quad \dfrac{f \vdash f \quad \dfrac{t, b, w, p \vdash tpbw}{t, b, w, p, \mathbf{1} \vdash tpbw}\ (\text{Here, } \overline{f} := f \multimap \mathbf{1})}{t, b, fw, p, \overline{f} \vdash tpbw}}{t, p, b, fw, p \multimap p\overline{f} \vdash tpbw}}{t, p, bf, f \multimap fw, p \multimap p\overline{f} \vdash tpbw}}{t, pb, b \multimap bf, f \multimap fw, p \multimap pf \vdash tpbw}}{tp, p \multimap pb, b \multimap bf, f \multimap fw, p \multimap p\overline{f} \vdash tpbw}}{t, t \multimap tp, p \multimap pb, b \multimap bf, f \multimap fw, p \multimap p\overline{f} \vdash tpbw}$$

Note that since we identify the multiset t, b, w, p and the formula $tpbw$ ($\equiv t \otimes p \otimes b \otimes w$), the top-sequent is obtained by the ax-rule.

3.2 Translation of DI-nets into a Horn fragment of IMLL

In this section, we give a translation of DI-nets into our Horn fragment of linear logic. We first give our definition of the translation of DI-nets. Then, the soundness of the translation (Theorem 3.9) will be demonstrated later, after we introduce natural deduction-style inference rules for DI-nets.

Definition 3.5 (Translation of DI-nets) When D is a DI-net, we define a program set D^* as follows:

- Each node a of the DI-net D is translated into an atom a of IMLL.
- $a \multimap ab \in D^*$ if and only if there is an edge $a \to b$ in D.
- $a \multimap a\overline{b} \in D^*$ if and only if there is an edge $a \not\to b$ in D.
- For each node $a \in D$,

  $\underbrace{a \cdots a}_{in_{\to}(a)} \multimap a \in D^*$, if the indegree $in_{\to}(a) \geq 2$, and

  $\underbrace{\overline{a} \cdots \overline{a}}_{in_{\not\to}(a)} \multimap \overline{a} \in D^*$, if the indegree $in_{\not\to}(a) \geq 2$.

  Here, the indegree $in_{\to}(a)$ (resp. $in_{\not\to}(a)$) of a is the number of edges of the form $c \to a$ (resp. $c \not\to a$).

We call the above formula of the form $L^n \multimap L$ a **contraction formula**.

To prove the soundness of our translation of DI-nets, we introduce natural deduction-style inference rules. Traditional natural deduction-style rules are not very often applied to linear logic, since the description of structural rules of sequent calculus becomes somewhat complicated. However, as seen in Section 2, natural reading of DI-nets corresponds to transitive (or inheritance) reasoning on properties, and such reasoning is more intuitively characterized by inference rules of natural deduction than those of sequent calculus. Thus, we here introduce our natural deduction as an intermediary between DI-nets and sequent calculus. These natural deduction-style inference rules make our translation of DI-nets considerably clear and intuitive.

Based on the above translation of DI-nets, we observe that program formulas are restricted to either form $a \multimap aL$ (with $a \not\equiv L$) or $L^n \multimap L$. Thus, instead of introducing natural deduction-style inference rules for the full Horn-fragment, we introduce it only sufficient for the translation of DI-nets.

Definition 3.6 (Natural deduction-style inference rules for DI-nets)

$$\frac{}{a \multimap a}\ ax \qquad \frac{\begin{array}{c}\vdots\\ a \multimap \vec{L}c \quad c \multimap cM\end{array}}{a \multimap \vec{L}cM}\ trans \qquad \frac{\begin{array}{c}\vdots\\ a \multimap \vec{L}M^n \quad M^n \multimap M\end{array}}{a \multimap \vec{L}M}\ contr \qquad \frac{\begin{array}{c}\vdots\\ a \multimap \vec{L}c\overline{c}\end{array}}{a \multimap \vec{L}}\ cancel$$

Note that one of the upper formulas in the rules of $trans, contr$ is restricted to be an open assumption, which is not derived from other formulas, and hence, our natural deduction proofs are just chains of inference rules.

We say that $a \multimap \vec{L}$ is provable from a program set $\mathcal{P}$, when there exists a natural deduction proof of $a \multimap \vec{L}$ from open assumptions $\mathcal{P}$, for which each program is used exactly once.

We show that the above natural deduction-style inference rules are simulated by IMLL sequent calculus rules.

Proposition 3.7 *If a formula $a \multimap \vec{L}$ is provable from a program set $\mathcal{P}$ with natural deduction-style inference rules, then the Horn sequent $a, \mathcal{P} \vdash \vec{L}$ is provable in* IMLL.

Proof. By the induction on the construction of natural deduction proof of $a \multimap \vec{L}$ from $\mathcal{P}$.

1. When $a \multimap a$ is provable by the ax-rule, $a \vdash a$ is also obtained by the ax-rule in IMLL.
2. When $\mathcal{P} = \{a \multimap \vec{L}\}$ and it forms a proof, the Horn sequent $a, a \multimap \vec{L} \vdash \vec{L}$ is provable in IMLL as follows:
$$\frac{a \vdash a \quad \vec{L} \vdash \vec{L}}{a, a \multimap \vec{L} \vdash \vec{L}} \multimap L$$
Although the same applies to the case of $\mathcal{P} = \{L^n \multimap L\}$, it does not appear in our translation of DI-nets.
3. When $a \multimap \vec{L}cM$ is provable by the following application of the $trans$-rule:
$$\frac{\begin{array}{c}\mathcal{Q}\\ \vdots\\ a \multimap \vec{L}c \quad c \multimap cM\end{array}}{a \multimap \vec{L}cM}\ trans$$
This proof is transformed into an IMLL-proof as follows:
$$\frac{\begin{array}{c}\vdots\ IH \quad c \vdash c \quad \vec{L}, cM \vdash \vec{L}cM\\ a, \mathcal{Q} \vdash \vec{L}c \quad \vec{L}, c, c \multimap cM \vdash \vec{L}cM\end{array}}{a, \mathcal{Q}, c \multimap cM \vdash \vec{L}cM}\ \begin{array}{l}\multimap L\\ cut\end{array}$$

4. When $a \multimap \vec{L}M$ is provable by the following application of the *contr*-rule:

$$\dfrac{\begin{array}{c}\mathcal{Q}\\ \vdots\\ a \multimap \vec{L}M^n \quad M^n \multimap M\end{array}}{a \multimap \vec{L}M}\ contr$$

This proof is transformed into an IMLL-proof as follows:

$$\dfrac{\begin{array}{cc}\vdots\ IH & \dfrac{M^n \vdash M^n \quad \vec{L}, M \vdash \vec{L}M}{\vec{L}, M^n, M^n \multimap M \vdash \vec{L}M}\multimap L\\ a, \mathcal{Q} \vdash \vec{L}M^n & \end{array}}{a, \mathcal{Q}, M^n \multimap M \vdash \vec{L}M}\ cut$$

5. When $a \multimap \vec{L}$ is obtained by the following application of the *cancel*-rule:

$$\dfrac{\begin{array}{c}\mathcal{P}\\ \vdots\\ a \multimap \vec{L}c\overline{c}\end{array}}{a \multimap \vec{L}}\ cancel$$

This proof is transformed into an IMLL-proof as follows:

$$\dfrac{\begin{array}{cc}\vdots\ IH & \dfrac{c \vdash c \quad \dfrac{\vec{L} \vdash \vec{L}}{\mathbf{1}, \vec{L} \vdash \vec{L}}\mathbf{1}L}{\vec{L}, c, \overline{c} \vdash \vec{L}}\multimap L\\ a, \mathcal{P} \vdash \vec{L}c\overline{c} & \end{array}}{a, \mathcal{P} \vdash \vec{L}}\ cut$$

■

Before we prove the soundness of our translation, we explain how to construct a natural deduction proof from a given DI-net by the following example.

Example 3.8 (Translation of DI-nets) Let us consider the DI-net D of Example 2.3 of Tweety. For simplicity, we omit the node w.

We first introduce a weaker notion of *semi-reachability* than reachability that is obtained by regarding conflicting nodes as also reachable. (See the definition given in the proof of Theorem 3.9 below.) We denote by $semireach(a)$ the set of nodes semi-reachable from a node a. For example, in Example 2.3 of Tweety, we have $semireach(t) = \{t, p, b, f, \overline{f}, w\}$. For Figs. 1 and 2 of Example 2.6, we have $semireach(a) = \{a, b, \overline{b}, c\}$ and $semireach(d) = \{d, e, \overline{f}\}$, respectively. Thus, we have $reach(a) \subseteq semireach(a)$, and $reach(a)$ is obtained from $semireach(a)$ by taking all conflicting nodes away.

Then, the DI-net D is translated as follows. For nodes p and b that are reachable in one step from t, we have $t \multimap tb, t \multimap tp \in D^*$ by the definition of translation of DI-nets. Thus by combining these formulas with the use of the *trans*-rule, we obtain the following proof:

$$\dfrac{t \multimap tb \quad t \multimap tp}{t \multimap tpb}\ trans$$

Next, for each node $b, f, \overline{f}$ that is reachable in two steps from t, there exists an edge from a node that is reachable in one step from t. In this case, we have $b \to f$, $p \to b$, and $p \not\to f$ in D, and hence we have $b \multimap bf, p \multimap pb, p \multimap p\overline{f} \in D^*$ by our translation. Thus, by combining these formulas one by one with the above proof, we obtain the following proof:

$$\dfrac{\dfrac{\dfrac{\vdots}{t \multimap tpb} \quad b \multimap bf}{t \multimap tpbf}\ trans \quad p \multimap pb}{t \multimap tpbbf}\ trans \quad p \multimap p\overline{f}}{t \multimap tp\overline{f}bbf}\ trans$$

For the node f that is reachable in three steps from t, we have $b \to f$ in D. However, the fact that there exists an edge from b to f is already reflected in the above proof. That is, the *trans*-rule is already applied to the formula $b \multimap bf$, and hence, we do not repeat this.

In the above proof, the last formula $t \multimap tp\overline{f}bbf$ contains a repetition of b. For this node b with $in_{\to}(b) = 2$, we have $b^2 \multimap b \in D^*$ by definition, and hence, by applying the *contr*-rule, we obtain $t \multimap tp\overline{f}bf$.

Thus resultant formula $t \multimap tp\overline{f}bf$ still contains conflicting f and $\overline{f}$. Hence, by applying the *cancel*-rule at the end, we obtain the following proof of $t \multimap tpb$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{t \multimap tb \quad t \multimap tp}{t \multimap tpb}\ trans \quad b \multimap bf}{t \multimap tpbf}\ trans \quad p \multimap pb}{t \multimap tpbbf}\ trans \quad p \multimap p\overline{f}}{t \multimap tp\overline{f}bbf}\ trans \quad bb \multimap b}{t \multimap tp\overline{f}bf}\ contr}{t \multimap tpb}\ cancel$$

As we have already seen, the above proof is translated into a sequent calculus proof of IMLL. Then, by applying the cut-elimination theorem of IMLL, we obtain the proof of Example 3.4 (without the atom w).

Now, let us prove the following theorem by formalizing the above example of construction of a natural deduction proof from a given DI-net.

Theorem 3.9 (Translation of DI-nets) *Let D be a DI-net. For each node a of D, if $reach(a) = \{L_1, \ldots, L_n\}$, then the Horn sequent $a, \mathcal{P} \vdash L_1 \cdots L_n$ is provable in* IMLL *for some $\mathcal{P} \subseteq D^*$.*

Proof. We show the theorem by translating a given DI-net D into a natural deduction proof. Then, by translating it into a sequent calculus proof of IMLL, we obtain the theorem.

We first introduce the notion of semi-reachability as follows:
Definition (Semi-reachability): For each node a of D, the set of nodes $semireach(a)$ is defined as follows:

1. $a \in semireach(a)$.
2. If $a \twoheadrightarrow b$ in D, then $b \in semireach(a)$.

3. If $a \twoheadrightarrow c \not\to b$ for some c or $a \not\to b$ in D, then $\overline{b} \in semireach(a)$.

When $L \in semireach(a)$, we say that L is **semi-reachable** from a.

By definition, we have $reach(a) \subseteq semireach(a)$, and $reach(a)$ is obtained from $semireach(a)$ by taking all conflicting nodes away.

We show the following slightly weaker lemma than the theorem, in which $reach(a)$ of the theorem is replaced by $semireach(a)$, and from which the theorem is easily obtained.

For a set S, we denote by $[S]$ a multiset consists of all elements of S by allowing some repetitions.

Lemma: *For each node a of D, a formula $a \multimap [semireach(a)]$ is provable from D^* with a natural deduction proof.*

We observe that the set $semireach(a)$ is expressed by the following union of not necessarily disjoint subsets:

$$semireach(a) = semireach_1(a) \cup semireach_2(a) \cup \cdots \cup semireach_k(a),$$

where $semireach_i(a)$ is the set of nodes to which a is semireachable by a path of length i, in other words, the set of nodes that are semireachable from a in $i-1$ steps. Then, we show the above lemma by induction on k.

(Base step) When $k = 1$, $semireach_1(a) = \{a\}$, and $a \multimap a$ is an axiom.

(Induction step) When $k > 1$, let $semireach_{k+1}(a) = \{L_1, \ldots, L_l\}$. By the induction hypothesis, we have $a \multimap [\bigcup_{1 \le i \le k} semireach_i(a)]$ is provable from D^*, and we show that $a \multimap [\bigcup_{1 \le i \le k} semireach_i(a)] \cdot L_1 \cdots L_l$ is provable from D^*.

Note that for each $c \in semireach_{k+1}(a)$ (resp. $\overline{c} \in semireach_{k+1}(a)$), we have $b \to c$ (resp. $b \not\to c$) in D for some $b \in semireach_k(a)$. Thus, by the definition of translation, we have $b \multimap bc \in D^*$ (resp. $b \multimap b\overline{c} \in D^*$). If it has already been applied in the former construction of the proof, there is no more to be done. Otherwise, by applying the *trans*-rule to $b \multimap bc$ (resp. $b \multimap b\overline{c}$), we have the following proof (the case of $b \multimap b\overline{c}$ is similar):

$$\dfrac{\begin{array}{c}\vdots\\ a \multimap [\bigcup_{1 \le i \le k} semireach_i(a)]\end{array} \qquad b \multimap bc}{a \multimap [\bigcup_{1 \le i \le k} semireach_i(a)] \cdot c}\ trans$$

In this way, we have $a \multimap [\bigcup_{1 \le i \le k} semireach_i(a)] \cdot L_1 \cdots L_l$ from D^*. □

By the above lemma, the theorem is obtained by applying the appropriate *contr* and *cancel*-rules as follows.

Note that $[semireach(a)]$ of the above lemma may contain some repetitions of literals. When L is such a repeated literal, we have $in_{\to}(L) \ge 2$ or $in_{\not\to}(L) \ge 2$ in D. For example, when we have $b \to L \leftarrow c$ in D for some $b, c \in semireach(a)$ (i.e., a case $in_{\to}(L) = 2$), we obtain the following inference:

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ a \multimap [semireach(a)]\end{array} \quad b \multimap bL}{a \multimap [semireach(a)] \cdot L} \qquad c \multimap cL}{a \multimap [semireach(a)] \cdot LL}$$

In such a case, by the definition of translation, we have $L^j \multimap L \in D^*$ with $j = in_{\to}(L)$ or $= in_{\not\to}(L)$, and hence, we are able to apply the *contr*-rule and to eliminate repetitions of

literals. Thus, when $semireach(a) = \{L_1, \ldots, L_m\}$, the formula $a \multimap L_1 \cdots L_m$ is provable from D^*.

As we have already seen, $reach(a)$ is obtained by taking away all conflicting nodes of $semireach(a)$. Let $L_i, L_j \in semireach(a)$ be conflicting nodes. Then we are able to apply the following form of the *cancel*-rule:

$$\frac{a \multimap L_1 \cdots L_l \cdot L_i L_j}{a \multimap L_1 \cdots L_l}\ cancel$$

In this way, we are able to eliminate all conflicting literals, and to obtain $a \multimap L_1 \cdots L_n$ from D^*. ■

It may be worth pointing that, as it is seen in the above proof, the order of applications of natural deduction style inference rules is important: we first apply the *trans*-rule, then the *contr*-rule, and then the *cancel*-rule.

4 Proof nets for linear logic and defeasible inheritance networks

Proof nets, a graph-theoretical representation of logical proofs, were introduced in [Girard 1987] as natural deduction for classical linear logic, as contrasted with sequent calculus that is sensitive to usually inessential details such as irrelevant orders of applications of inference rules. By ignoring such inessential details, proof nets reveal the essence of structures of proofs. For example, the computational nature of proofs such as normalization and confluency are made clear by proof nets, and a criterion of equivalence between proofs is given in terms of proof nets.

Proof nets are defined as a correct subclass of undirected graphs called proof-structures, in which nodes are formulas and the way of linking those nodes by edges is defined as links (see Fig. 3 below). Proof nets are defined graph-theoretically, and hence, we are able to apply graph-theoretical methods and techniques to the study of proofs. In particular, the non-sequential nature of graphical proof nets is applied to the analysis of the concurrent computation, for example.

Proof nets for intuitionistic IMLL is obtained from those for classical MLL under the identification between $A \multimap B$ and $A^{\perp} \parr B$, where $\parr$ is the multiplicative disjunction of MLL. While well-known correctness conditions of proof nets for classical MLL is given in [Danos-Regnier 1989], for intuitionistic IMLL, appropriate conditions are given in [Lamarche 1994] with the use of directed graphs. [Lamarche 1994] called his proof nets for intuitionistic IMLL *essential nets*. See, e.g., [Lamarche 1994, Moot 2004] for proof nets for IMLL.

In Section 4.1, we briefly review proof nets for IMLL. Our definition mainly follows [Moot 2004]. In Section 4.2, we show that there is a structural correspondence between our proof nets and DI-nets, by giving a transformation of the proof nets.

4.1 Proof nets for IMLL (Essential nets)

Each formula of IMLL is assigned with input (i) and output (o) polarities, and these are written as $A^i, B^i, C^i, \ldots$ and $A^o, B^o, C^o, \ldots$, respectively. Proof nets for IMLL consist of the following (polarized) **links** of Fig. 3 that connect (polarized) formulas.

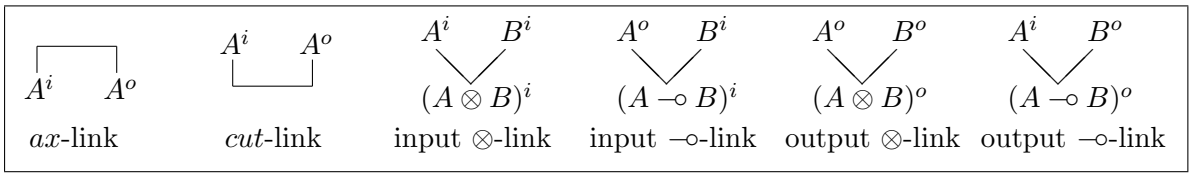


Fig. 3 (Polarized) links

A link has its premises at the top and its conclusions at the bottom. ax-link has no premises and two conclusions. cut-link has two premises and no conclusion. Each of the input/output $\otimes/\multimap$-link has two premises and one conclusion.

A **proof-structure** is a collection of links that satisfies the following conditions:

1. Every formula is the conclusion of exactly one link;
2. Every formula is the premise of at most one link;
3. Formulas that are not the premise of a link are called the conclusions of the proof-structure. A proof-structure has exactly one 'output' conclusion.

Note that there are several conclusions in a proof-structure in general. Based on the above condition (3), we may call 'input' conclusions "premises" of the proof-structure.

A proof-structure is called a proof net when it satisfies certain correctness conditions. For the proof nets for classical MLL, several correctness conditions are proposed such as Girard's long trip condition ([Girard 1987]) and Danos-Regnier condition ([Danos-Regnier 1989]). For the proof nets for intuitionistic IMLL, there is the following correctness condition (cf. [Lamarche 1994, Moot 2004]).

We first introduce the following directions for edges of links as in Fig. 4, and we call such links **directed links**.

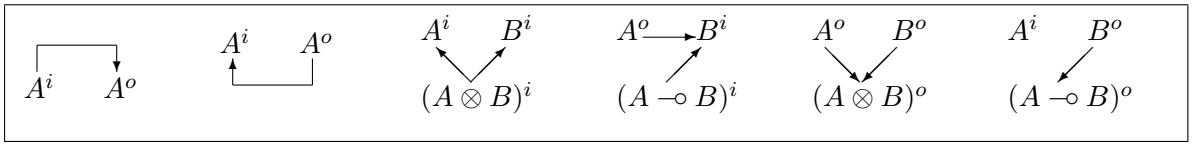


Fig. 4 Directed links

We call a proof-structure for which all links are directed a **correction graph**.

A proof-structure is correct, i.e., a **proof net**, if its correction graph satisfies the following **correctness conditions**:

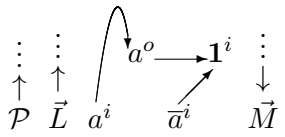
1. It is acyclic.
2. Every path from the input conclusions (i.e., premises) of the graph reaches the (unique) output conclusion of the graph.
3. Every path from the input premise of an output $\multimap$-link passes through the output conclusion of the link.

The above condition (3) does not play an essential role in our Horn fragment.

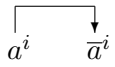
For our DI-net, we have negative atoms of the form $\overline{a} := a \multimap \mathbf{1}$, and hence it seems that we need a link for the constant $\mathbf{1}$. However in our fragment, based on the definition of translation

of DI-nets, occurrences of **1** are restricted in a given Horn sequent, and hence, in a cut-free sequent calculus proof of the Horn sequent, inference rules for **1** appear only in the following form on the left. (We omit the trivial case of axiom.)

$$\dfrac{a \vdash a \quad \dfrac{\vec{L}, \mathcal{P} \vdash \vec{M}}{\mathbf{1}, \vec{L}, \mathcal{P} \vdash \vec{M}}\,\mathbf{1}L}{\mathcal{P}, \vec{L}, a, \overline{a} \vdash \vec{M}}\,{\multimap}L$$



The sequent calculus proof corresponds to the correction graph on the right above. Hence, in our fragment, by introducing the following form of axiom link by abbreviating the above part of proof nets, we are able to regard $\overline{a}$ as a kind of atom without decomposing it to a and **1**. Thus, with the help of the new axiom link, we are able to regard our proof nets as the usual most basic nets without a link for **1**, and the results for them such as sequentialization (cf. the following Proposition 4.1) are immediately applied to our proof nets.



Let $\Gamma \vdash C$ be a sequent of IMLL. When π is a proof net for which input conclusions (i.e., premises) are just formulas of Γ, and for which the unique output conclusion is C, we say that π is a proof net for $\Gamma \vdash C$.

As an immediate consequence of the theorem given in [Lamarche 1994], we have the following proposition.

Proposition 4.1 *If a Horn sequent $\mathcal{P}, \vec{L} \vdash \vec{M}$ is provable in* IMLL*, then there is a proof net for the sequent.*

Example 4.2 (Correction graph) We have a correction graph in Fig. 5 for Example 2.3 of Tweety.

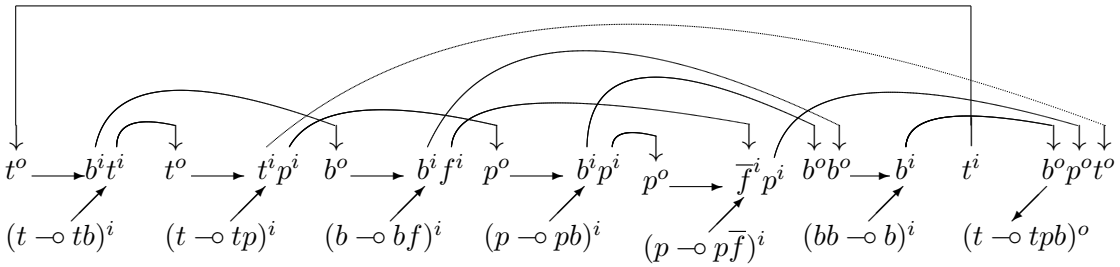


Fig. 5 A correction graph

4.2 Transformation of proof nets into DI-nets

In this section, we give a transformation of our proof nets into DI-nets.

Theorem 4.3 (Transformation of proof nets) *Let $\mathcal{P}$ be a program set consisting of formulas of the forms $b \multimap bL$ with $b \not\equiv L$ and $L^n \multimap L$. Let $a, M_1, \ldots, M_m$ be different literals. A proof net for $\mathcal{P} \vdash a \multimap aM_1 \cdots M_m$ is transformed into a DI-net $D = (D, \rightarrow, \not\rightarrow)$ such that:*

- *The domain is* $D = \{c \mid \ c \text{ or } \overline{c} \text{ appears in the sequent } \mathcal{P} \vdash a \multimap aM_1 \cdots M_m\}$
- *The translation* $D^* = \mathcal{P}$*, that is,*
  - $b \to c$ *in* D *if and only if* $b \multimap bc \in \mathcal{P}$
  - $b \not\to c$ *in* D *if and only if* $b \multimap b\overline{c} \in \mathcal{P}$
- $reach(a) = \{a, M_1, \ldots, M_m\}$

Proof. Let π be a proof net of $\mathcal{P} \vdash a \multimap a\vec{M}$. Then, its correction graph is a directed acyclic graph, in which a^i is an initial node, and $(a \multimap a\vec{M})^o$ and conflicting nodes of the form $\overline{c}$ are the terminals. We transform the correction graph into the required DI-net.

We observe that our correction graph for the above sequent in general consists of the following parts (cf. Example 4.2):

- The terminal part that consists of the unique output conclusion $(a \multimap a\vec{M})^o$ of the proof net and its related formulas a^i and $a^o\vec{M}^o$.
- A transition part that consists of a program formula of the form $(c \multimap cL)^i$ and its related formulas c^o and c^iL^i.
- A contraction part that consists of a program formula of the form $(L^n \multimap L)^i$ and its related formulas $L^o \cdots L^o$ and L^i.
- A cancel part in which there is a link between conflicting nodes c^i and $\overline{c}^i$.

We transform each part as follows.

(1) Terminal part: This part takes the form appearing on the left of the following diagram in Fig. 6. Here, the leftmost a^i is an initial node of the given correction graph. This terminal part is transformed into the form on the right as follows: By ignoring the i, o-polarities, and by deleting the conclusion formula $a \multimap M_1 \cdots M_m$ as well as its incident edge, we obtain a graph, in which each M_i is a terminal of the graph.

(2) Transition part: This part takes the form appearing on the left in Fig. 6. By ignoring the i, o-polarities, by deleting the program formula $c \multimap cL$ as well as its incident edge, and by dividing the edge of $c \to cL$ into the form $c \swarrow \searrow \begin{matrix} c \\ L \end{matrix}$, we obtain the intermediate graph. We finally obtain an edge $c \to L$ on the right, by contracting all nodes of the same label.

(3) Contraction part: This part takes the form appearing on the left in Fig. 6. For the simplicity, we consider a contraction formula of the form $c^2 \multimap c$. The same applies to the general case. By ignoring the i, o-polarities, by deleting the program formula $cc \multimap c$ as well as its incident edge, and by identifying the duplicated nodes cc, we obtain the intermediate graph. We finally obtain the node c and its incident edges on the right, by contracting all nodes of the same label.

(4) Cancel part: This part takes the form appearing at the upper left in Fig. 6. By ignoring the i, o-polarities, by deleting the program formulas $c \multimap cL$ and $b \multimap b\overline{L}$ as well as their incident edges, and by dividing the edges of $c \to cL$ and $b \to b\overline{L}$, we obtain the graph at the upper right. Then, by replacing the node $\overline{L}$ and its incident edge of $b \to \overline{L}$ to L

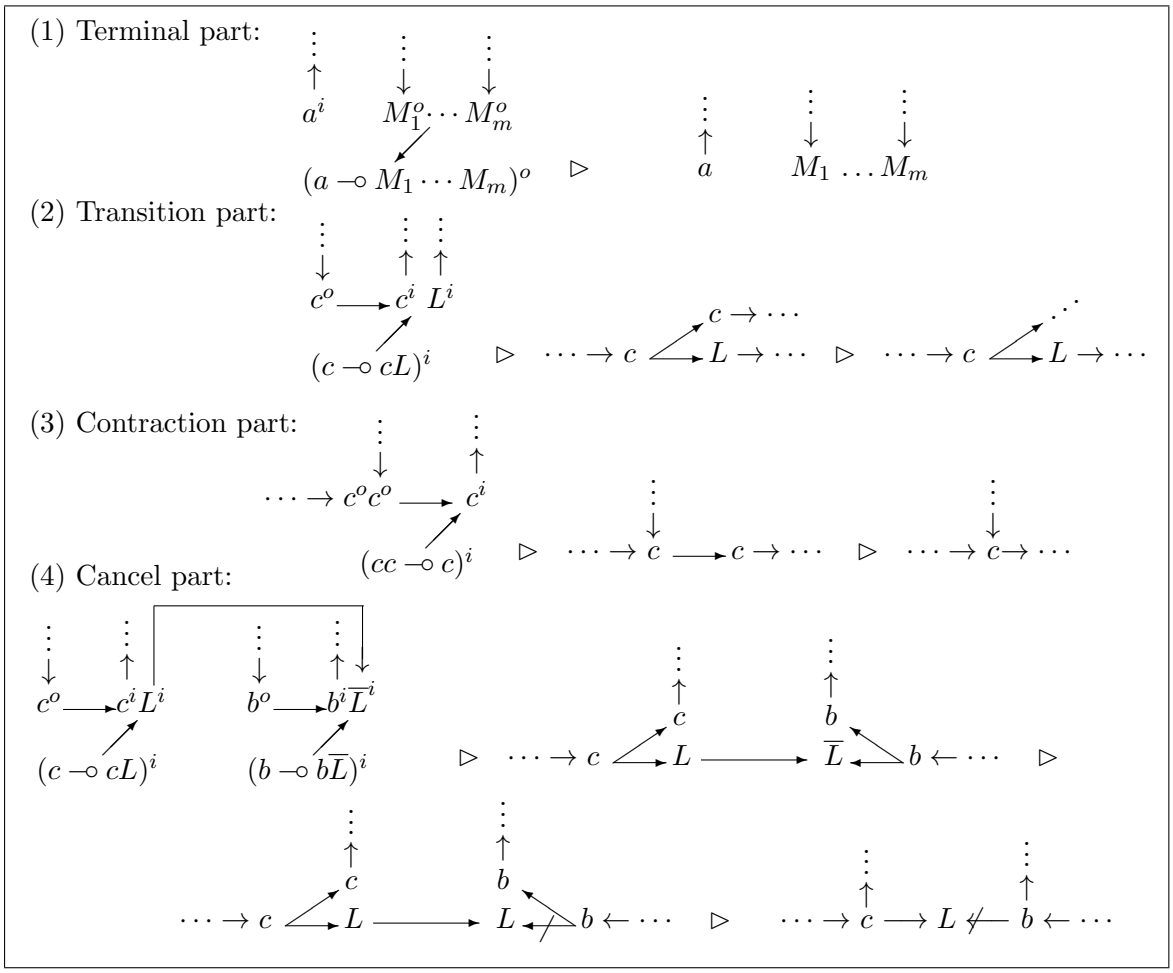


Fig. 6 Transformation of proof nets

and $b \not\rightarrow L$, respectively, we obtain the graph at the lower left. We finally obtain the graph at the lower right that contains a conflicting node L, by contracting all nodes of the same label.

It is clear that the above transformations of a given correction graph maintain the directedness and acyclicity of the graph. Furthermore, it does not change the fact that a is the initial node (it is unique since all input program formulas are deleted) and $a\vec{M}$ as well as some conflicting nodes are the terminals of the graph. Since such conflicting nodes are not reachable by definition, and since the initial node a reaches all of $a, M_1, \ldots, M_m$ by the correctness condition (2) of the graph, we have $reach(a) = \{a, M_1, \ldots, M_m\}$. Therefore, we obtain the required DI-net by the above transformation. ■

Observe that our transformation does not contain any non-trivial rewriting, and hence, it shows that a correction graph of a proof net and a DI-net share essentially the same structure.

Example 4.4 (Transformation) The correction graph given in Example 4.2 is transformed as follows. (For accessibility purposes, we keep the i, o-polarities for the graph on the left.)

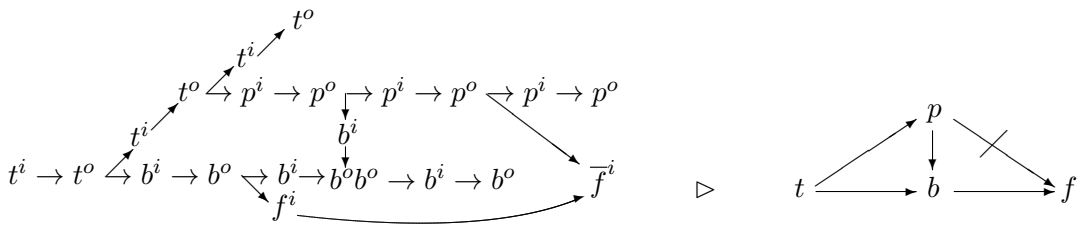


Fig. 7 Transformation of the correction graph of Example 4.2

5 Conclusion and future work

Towards a characterization of defeasible inheritance reasoning and more generally of nonmonotonic reasoning, we investigated in this paper a relationship between defeasible inheritance networks, a Horn fragment of linear logic, and proof nets (essential nets) for linear logic. We obtained the following equivalence:

Corollary 5.1 *Let $\mathcal{P}$ be a program set that consists of formulas of the forms $b \multimap bL$ with $b \not\equiv L$ and $L^n \multimap L$. Let $a, M_1, \ldots, M_m$ be different literals. The following are equivalent:*

1. *There exists a proof net for the Horn sequent $a, \mathcal{P} \vdash aM_1 \cdots M_m$.*
2. *The Horn sequent $a, \mathcal{P} \vdash aM_1 \cdots M_m$ is provable in* IMLL.
3. *There exists a DI-net D such that $D^* = \mathcal{P}$ and $reach(a) = \{a, M_1, \ldots, M_m\}$.*

Proof. (3 $\Rightarrow$ 2) is obtained by Theorem 3.9. (2 $\Rightarrow$ 1) is obtained by [Lamarche 1994]. (1 $\Rightarrow$ 3) is obtained by Theorem 4.3. ■

Thus, we characterized the notion of reachability from a node in a DI-net by the provability of the corresponding Horn sequent in IMLL. Furthermore, we showed that there is a structural correspondence between DI-nets and proof nets. Our result is important in view of implementation of defeasible inheritance reasoning, since various logic programmings are developed for linear logic, and our exponential-free Horn fragment is the most basic one of them.

As we pointed out after Example 2.3 of Section 2, our characterization of DI-nets in the Horn fragment of linear logic is based on a weaker interpretation of DI-nets than that of [Fouqueré-Vauzeilles 1994]. To extend our results to include the strict interpretation of DI-nets, when we cancel a conflicting literals, say f and $\overline{f}$, we need to cancel all nodes that inherit from f or $\overline{f}$ at the same time. Although it is generally difficult to capture inheritance or consequence relationships among formulas at the level of formulas, in our simple framework of defeasible inheritance, i.e., only transitive inheritance relationships between atoms are concerned, such transitive relationships seem to be captured with an appropriate use of parentheses. This is illustrated by the following example of a natural deduction proof of Example 2.3 of Tweety. (For simplicity, we skip the node t below.)

$$\dfrac{\dfrac{\dfrac{\dfrac{p \multimap (pb) \quad b \multimap (bf)}{p \multimap (p(bf))}\, trans \qquad f \multimap (fw)}{p \multimap (p(b(fw)))}\, trans \qquad p \multimap (p\overline{f})}{p \multimap ((p\overline{f})(b(fw)))}\, trans}{p \multimap (pb)}\, cancel$$

Note that in the above example, the inheritance relation is naturally captured by parentheses. For example, in the formula $p \multimap ((p\overline{f})(b(fw)))$, the parentheses of (fw) indicates that w inherit from f; the outermost parentheses of $(b(fw))$ indicates that fw inherit from b; and $(p\overline{f})$ indicates that $\overline{f}$ inherit from p. In the above application of the *cancel*-rule, at the same time when we cancel f and $\overline{f}$, we also cancel w that is in the scope of the parentheses of f, i.e., that inherit from f. Thus, in such a system, we need (1) to keep structures of parentheses strictly, and (2) to cancel several atoms at the same time. (1) suggests non-associative linear logic, and (2) suggests a subsystem of affine linear logic, in which the weakening rule is allowed for only atoms, i.e., $(p\overline{f}) \multimap p$ is derivable from $p \multimap p$, and $(b(fw)) \multimap b$ is derivable from $b \multimap b$. We leave a detailed discussion and formalization thereof as future work.

In this paper, we only considered two types of edges in our DI-nets: the defeasible edge ($\rightarrow$) and the defeasible negative edge ($\not\rightarrow$). However, as discussed in [Horty 1994], we are able to introduce another type of edges to DI-nets such as the "strict edge" ($\Rightarrow$), which corresponds to the usual implication admitting no exceptions. If we regard it as an intuitionistic implication, we may be able to characterize it by the formula $!a \multimap b$ in linear logic with the use of the exponential operator. We also leave such an extension as future work.

Mainly for reasons of space, we are not able to discuss semantics for our defeasible inheritance reasoning. Note however that our system is one of the simplest fragments of linear logic, i.e., we have only restricted the language of linear logic. Hence, we are able to straightforwardly apply semantics of linear logic such as phase semantics (cf. [Okada 2008]) to our defeasible inheritance reasoning. We will discuss this on another occasion.

References

[Curien 2005] Pierre-Louis Curien, Introduction to linear logic and ludics, part I, and part II, *Advances in Mathematics (China)* 34 (5), 513-544, 2005, and 35 (1), 1-44, 2006.

[Danos-Regnier 1989] Vincent Danos and Laurent Regnier, The structure of multiplicatives, *Archive for Mathematical Logic*, 28, 181-203, 1989.

[Fouqueré-Vauzeilles 1993] Christophe Fouqueré, Jacqueline Vauzeilles, Taxonomic Linear Theories, *Proceedings of Symbolic and Quantitative Approaches to Reasoning and Uncertainty, European Conference, ECSQARU'93*, M. Clarke, R. Kruse, S. Moral (Eds.), Lecture Notes in Computer Science, 747, 121-128, 1993.

[Fouqueré-Vauzeilles 1994] Christophe Fouqueré, Jacqueline Vauzeilles, Linear Logic and Exceptions, *Journal of Logic and Computation*, 4 (6), 859-876, 1994.

[Girard 1987] Jean-Yves Girard, Linear Logic, *Theoretical Computer Science*, 50, 1-102, 1987.

[Girard 1992] Jean-Yves Girard, Logic and Exceptions: A Few Remarks, *Journal of Logic and Computation*, 2 (2), 111-118, 1992.

[Girard 1995] Jean-Yves Girard, Linear Logic: its syntax and semantics, in *Advances in Linear Logic*, Girard, Lafont, Regnier, eds., Cambridge University Press, 1995.

[Horty 1994] John F. Horty, Some direct theories of nonmonotonic inheritance, D. Gabbay, C. Hogger, J.A. Robinson editors, *Handbook of logic in artificial intelligence and logic programming (vol. 3)*, Oxford University Press, 111-187, 1994.

[Kanovich 1992] Max I. Kanovich, Horn Programming in Linear Logic is NP-complete, *Proceedings of 7th Annual IEEE Symposium on Logic in Computer Science*, Santa Cruz, 200-210, 1992.

[Kanovich 1994] Max I. Kanovich, The Complexity of Horn Fragments of Linear Logic, *Annals of Pure and Applied Logic*, 69 (2-3), 195-241, 1994.

[Kanovich-Vauzeilles 2001] Max I. Kanovich, Jacqueline Vauzeilles, The classical AI planning problems in the mirror of Horn linear logic: semantics, expressibility, complexity, *Mathematical Structures in Computer Science*, 11(6), 689-716, 2001.

[Lamarche 1994] François Lamarche, Proof Nets for Intuitionistic Linear Logic I: Essential Nets, Technical report, Imperial College, 1994.

[Lincoln 1995] Patrick Lincoln, Deciding provability of linear logic formulas, *Proceedings of the Workshop on Advances in Linear Logic*, J.-Y. Girard, Y. Lafont, and L. Regnier (eds.), Cambridge University Press, 197-210, 1995.

[Masseron-Tollu-Vauzeilles 1993] M. Masseron, Christophe Tollu, Jacqueline Vauzeilles, Generating Plans in Linear Logic I. Actions as Proofs, Theoretical Computer Science, 113(2), 349-370, 1993.

[Miller 1995] Dale Miller, A Survey of Linear Logic Programming, *Computational Logic: The Newsletter of the European Network in Computational Logic*, Volume 2, No. 2, December 1995, pp. 63-67.

[Moot 2004] Richard Moot, Graph Algorithms for Improving Type-Logical Proof Search, *Proceedings of Categorial Grammars 2004*, 13-28, 2004.

[Okada 2008] Mitsuhiro Okada, Some remarks on linear logic, in M. van Atten, P. Boldini, M. Bourdeau, and G. Heinzmann, editors, *One Hundred Years of Intuitionism (1907-2007): The Cerisy Conference*, Publications of the Henri Poincaré Archives, 280-300, 2008.

[Pfenning 2008] Frank Pfenning, On Linear Inference, Short expository note, 2008.

[Reiter 1980] Raymond Reiter, A Logic for Default Reasoning, *Artificial Intelligence*, 13(1-2), 81-132, 1980.

[Troelstra 1992] Anne S. Troelstra, *Lectures on Linear Logic*, CSLI Lecture Notes, vol. 29, Stanford, 1992.