Using Tables to Construct Non-Redundant Proofs

Vivek Nigam

INRIA & LIX/École Polytechnique, Palaiseau, France
nigam at lix.inria.fr

Abstract. Proofs containing more than one subproof for a common subgoal are less preferred in frameworks such as Proof Carrying Code, where proofs are stored and communicated, than proofs that don't contain such redundancies. In this paper, we show how (cut-free) proofs can be transformed into non-redundant cut-proofs. Two main questions arise when trying to construct these non-redundant proofs: First, which cut-formulas should be used; Second, where to perform cut rules. Some advances in proof theory, namely, our better understanding of focused proofs, allows us to propose the following answers: We use only atomic subgoals of the original proof; and we place cut rules only at the end of the *asynchronous phases*. The backbone of a non-redundant proof is a tree, called *tree of multicut derivations* (**tmcd**), where a node is a derivation containing only multicut rules, and an edge represents the provability dependency between a subgoal introduced by a node's multicut rule and another (tree of) multicut derivation. We show how to obtain a **tmcd** from an existing proof, and later show how to complete the obtained **tmcd** and construct a non-redundant proof. Finally, we argue about the plausibility of **tmcd**s to be used as proof objects.

1 Introduction

Frameworks such as Proof Carrying Code [Nec97,App01], where mobile codes are sent with proofs that assure that these codes satisfy certain properties, provide "real world" concerns, not only for *provability*, but also for the *shape* and *format* of proofs. In these frameworks, since proofs need to be stored and communicated, proofs have to attend certain engineering aspects; for instance, the size of proofs is relevant; more precisely, smaller proofs are preferred.

Proofs that contain redundant sub-proofs, (identical) proofs of a same subgoal, are clearly bigger objects than proofs that are not redundant (from now on called as *non-redundant proofs*); simply because non-redundant proofs contain one and only one (non trivial[1]) proof for each subgoal, whereas redundant proofs may contain several proofs for a common subgoal. Hence, in the frameworks previously mentioned, non-redundant proofs should be preferred to the redundant

[1] We consider a trivial proof for a goal, a proof that contains only one inference rule, namely the initial rule.

proofs. For example, consider the proofs for the goal that the 12th Fibonacci number is 144, denoted by (fib 12 144), and obtained from the following logic program $\{\mathit{fib}\ 1\ 1, \mathit{fib}\ 2\ 1, \forall XYZ.[\mathit{fib}\ X\ Y \wedge \mathit{fib}\ (X+1)\ Z \supset \mathit{fib}\ (X+2)\ (Y+Z)]\}$. Two types of proofs can be distinguished: one where a *forward chaining* behavior is adopted, and another where a *backward chaining* behavior is adopted. The former proof is a linear size non-redundant proof; the later proof is an exponential size redundant proof.

We propose a procedure to construct a non-redundant sequent calculus proof from a redundant sequent calculus proof. This is done by collecting (or *tabling*) from an existing proof a set of atomic lemmas, called *table*. Theses lemmas are then used as cut formulas to construct a non-redundant proof containing cuts. The fact that the non-redundant proof contains cuts is not surprising; it is well known that, when compared with cut proofs, cut-free proofs are potentially bigger (also known as the *cut-elimination blow-up*).

There are two main problems to be addressed: (1) which lemmas should be used; and (2) when to use these lemmas, that is, while constructing the non-redundant proof, when should a cut be used.

Some recent advances in proof theory, namely, our better understanding of focused proofs in classical and intuitionistic logics [LM07], and of the effect of *atomic polarities* in the shape of proofs [CPP06,MN07], provides us with the machinery necessary to propose answers for these questions:

1. We collect (or table), in the existing proof, all the atomic subgoals. This restriction allows us to construct non-redundant proofs with a polarized cut-rule [MN07], where an atomic cut-formula has negative polarity in one branch of the proof tree and positive polarity on the other branch of the proof tree. As we investigate elsewhere [MN07], by using this polarized cut-rule, it is possible to mix a forward chaining behavior with a backward chaining behavior, what enforces some subgoals not to be re-proven;
2. The idea is to use the lemmas as close as possible to the root of the tree, so that if a lemma is to be proved again then it would be available in the set of hypothesis of the sequent, what would allow to finish the proof with an initial rule, that is, a trivial proof. However, it may happen that a lemma can't be proved right from the bottom of the tree, and should be introduced into the proof only when there is an increment in the set of hypothesis of a sequent (by, for example, a right implication rule). We show that focusing provides the discipline necessary to identify the places in a proof tree where new lemmas should be introduced, namely, at the end of the *asynchronous* phases.

This paper is structured as follows: we introduce in Section 2 some key concepts related to *focusing* and introduce the intuitionistic system LJF^t. In Section 3, we define *tables*, explain how tables specify *multicut derivations*, and review some relevant earlier results. The core of the paper lies in Section 4 that is divided in three parts: First, we specify *tree of multicut derivations* (**tmcd**), a derivation that extends of the notion of tables; Second, we describe how **tmcd**s can be obtained from existing uniform cut-free proofs [MNPS91] and show that

tmcds can be completed with simple derivations to obtain non-redundant proofs; Third, we extend the results before to a richer class of *LJF* proofs, where atoms can have arbitrary polarity. In Section 5, we briefly comment on the plausibility of **tmcd**s to be used as proof objects, and finally in Section 6, we finish with some conclusions and point out future work.

2 Focusing and LJF^t

We start by classifying formulas into two distinct classes: A formula has *positive* polarity if it is the formula *true*, $\bot$, a positive atom, or its main connective is either $\wedge$, $\vee$, or $\exists$. Otherwise, a formula has *negative* polarity.

The focusing discipline provides a set of constraints to the structure of proofs. Focused proofs are structured into two alternating phase, the *asynchronous* phase and the *synchronous* phase. In the former phase, all the invertible rules are applied until exhaustion, and therefore no backtracking during this phase of search is needed; and in the later phase a positive formula is selected (or *focused* on) and all the rules are applied only to its positive subformulas. When a negative subformula is encountered the focus is lost, and the asynchronous begins. These constraints in the structure of proofs will become more clear when we explain the *LJF* system from where LJF^t is derived.

LJF, proposed by Liang and Miller [LM07], is a focused intuitionistic system, that, differently from other focused intuitionistic systems [CH00,JNS05], allows, as in Andreoli's original focused system in Linear Logic [And92], atoms to be assigned arbitrary polarities.

The Figure 1 depicts the inference rules in *LJF* . Four different types of sequents can be identified:

1. The sequent $[\Gamma] -_A \rightarrow$ is a *right-focusing* sequent (the focus is A);
2. The sequent $[\Gamma] \xrightarrow{A} [R]$: is a *left-focusing* sequent (with focus on A);
3. The sequent $[\Gamma], \Theta \longrightarrow \mathcal{R}$ is an *unfocused sequent*. Here, Γ contains negative formulas and positive atoms, and $\mathcal{R}$ is either in brackets, written as $[R]$, or without brackets;
4. The sequent $[\Gamma] \longrightarrow [R]$ is an instance of the previous sequent where Θ is empty.

The asynchronous phase uses the third type of sequent above (the unfocused sequents): in that case, Θ contains positive or negative formulas. If Θ contains positive formulas, then an introduction rule (either $\wedge_l, \exists_l$ or false_l) is used to decompose it; if it is negative, then the formula is moved to the Γ context (by using the $[]_l$ rule). The end of the asynchronous phase is represented by the fourth type of sequent. Such a sequent is then established by using one of the decide rules, D_r or D_l. The application of one of these decide rules then selects a formula for focusing and switches proof search to the *synchronous phase* or *focused phase*. This focused phase then proceeds by applying sequences of inference rules on focused formulas: in general, backtracking may be necessary in this phase of

$$\frac{[N,\Gamma] \xrightarrow{N} [R]}{[N,\Gamma] \longrightarrow [R]}\ D_l \qquad \frac{[\Gamma] -_P\!\rightarrow}{[\Gamma] \longrightarrow [P]}\ D_r \qquad \frac{[\Gamma], P \longrightarrow [R]}{[\Gamma] \xrightarrow{P} [R]}\ R_l \qquad \frac{[\Gamma] \longrightarrow N}{[\Gamma] -_N\!\rightarrow}\ R_r$$

$$\frac{[\Gamma, N_a], \Theta \longrightarrow \mathcal{R}}{[\Gamma], \Theta, N_a \longrightarrow \mathcal{R}}\ []_l \qquad \frac{[\Gamma], \Theta \longrightarrow [P_a]}{[\Gamma], \Theta \longrightarrow P_a}\ []_r$$

$$\frac{}{[\Gamma] \xrightarrow{A_n} [A_n]}\ I_l \qquad \frac{}{[\Gamma, A_p] -_{A_p}\!\rightarrow}\ I_r$$

$$\frac{}{[\Gamma], \Theta, \bot \longrightarrow \mathcal{R}}\ \mathit{false}_l \qquad \frac{[\Gamma], \Theta \longrightarrow \mathcal{R}}{[\Gamma], \Theta, \mathit{true} \longrightarrow \mathcal{R}}\ \mathit{true}_l \qquad \frac{}{[\Gamma] -_{\mathit{true}}\!\rightarrow}\ \mathit{true}_r$$

$$\frac{[\Gamma], \Theta, A, B \longrightarrow \mathcal{R}}{[\Gamma], \Theta, A \wedge B \longrightarrow \mathcal{R}}\ \wedge_l \qquad \frac{[\Gamma] -_A\!\rightarrow \quad [\Gamma] -_B\!\rightarrow}{[\Gamma] -_{A \wedge B}\!\rightarrow}\ \wedge_r$$

$$\frac{[\Gamma] \xrightarrow{A_i} [R]}{[\Gamma] \xrightarrow{A_1 \wedge^- A_2} [R]}\ \wedge_l^-, i \in \{1,2\} \qquad \frac{[\Gamma], \Theta \longrightarrow A \quad [\Gamma], \Theta \longrightarrow B}{[\Gamma], \Theta \longrightarrow A \wedge^- B}\ \wedge_r^-$$

$$\frac{[\Gamma] -_A\!\rightarrow \quad [\Gamma] \xrightarrow{B} [R]}{[\Gamma] \xrightarrow{A \supset B} [R]}\ \supset_l \qquad \frac{[\Gamma], \Theta, A \longrightarrow B}{[\Gamma], \Theta \longrightarrow A \supset B}\ \supset_r$$

$$\frac{[\Gamma], \Theta, A \longrightarrow \mathcal{R}}{[\Gamma], \Theta, \exists y A \longrightarrow \mathcal{R}}\ \exists_l \qquad \frac{[\Gamma] -_{A[t/x]}\!\rightarrow}{[\Gamma] -_{\exists x A}\!\rightarrow}\ \exists_r \qquad \frac{[\Gamma] \xrightarrow{A[t/x]} [R]}{[\Gamma] \xrightarrow{\forall x A} [R]}\ \forall_l \qquad \frac{[\Gamma], \Theta \longrightarrow A}{[\Gamma], \Theta \longrightarrow \forall y A}\ \forall_r$$

Fig. 1. *LJF* : Here A_n denotes a negative atom, A_p a positive atom, and P a positive formula, N a negative formula, N_a a negative formula or an atom, and P_a a positive formula or an atom. All other formulas are arbitrary and y is not free in Γ, Θ or R.

search. We call the rules $\wedge_l, \exists_l, \mathit{false}_l, \supset_r, \forall_r, []_l, []_r, \wedge_r^-$ as *asynchronous rules*, and the remaining rules as *synchronous rules*.

As pointed out elsewhere [LM07,CPP06,MN07], the atomic polarities play an important role in the shape of the proofs, without affecting in no way provability. For instance, if all atoms have positive polarity, only proofs with a forward chaining behavior are possible, and on the other hand, if all atoms have negative polarity, only proofs with a backward chaining behavior are possible, for example *uniform proofs* [MNPS91].

LJF^t capitalizes on the observation that atomic polarities can be arbitrarily assigned, and extends *LJF* in two ways. (1) Extends the *LJF* sequents with a polarity context, $\mathcal{P}$, which specifies all the positive atoms in a sequent. An atom in a sequent, $\mathcal{P}; [\Gamma] \longrightarrow [R]$, is positive if and only if $A \in \mathcal{P}$; (2) extends *LJF* with the following polarized multicut rule:

$$\frac{\mathcal{P}; [\Gamma] \longrightarrow [A_1] \qquad \cdots \qquad \mathcal{P}; [\Gamma] \longrightarrow [A_n] \qquad \mathcal{P} \cup \Delta; [\Gamma \cup \Delta] \longrightarrow [R]}{\mathcal{P}; [\Gamma] \longrightarrow [R]}\ mc.$$

Where $\Delta = \{A_1, \ldots, A_n\}$ is a set of atoms. The multicut rule is the only rule that can change the polarity context in a proof. Since, from this point on, we

only use LJF^t, we name its rules with the same names used for the LJF rules. We often omit sequent's polarity context, when it is easy to infer it from the context.

Proposition 1. *[MN07] LJF^t is sound and complete with respect to LJF .*

Remark: The results in this paper could be easily applied to (focused) classic logics, such as LKF [LM07].

In the next section, we are going to use this polarized multicut rule to construct multicut derivations that are the principal object used to construct non-redundant proofs.

3 Tables as Multicut Derivations

We consider a table as a partially ordered finite set of atoms.

Definition 1. *A* table *is a tuple $\mathcal{T} = \langle \mathcal{A}, \prec \rangle$, where $\mathcal{A}$ is some finite set of atoms, and $\prec$ is a partial order relation over the elements of $\mathcal{A}$.*

In a table, each atom represents, intuitively, a provable sub-goal necessary in the proof of a sequent (say $\Gamma \longrightarrow G$), and the order relation the provability dependency between the atoms, that is, if $A \prec B$ then A is a subgoal used to prove the goal B.

By using the information contained in a table, we can construct a (multicut) derivation that introduces each element of this table into a context, say Γ, respecting the provability relation between the subgoals in the table. For instance, if $A \prec B$, we introduce first (from bottom to top) A in Γ, by using it as a cut formula, and later introduce B in Γ, with another cut. As specifies the following definition.

Definition 2. *Let $\mathcal{T} = \langle \mathcal{A}, \prec \rangle$ be a table. The* multicut derivation *for $\mathcal{T}$ and the sequent $\mathcal{S} = \Gamma \longrightarrow G$, written as $mcd(\mathcal{T}, \mathcal{S})$, is defined inductively as follows: if $\mathcal{A}$ is empty, then $mcd(\mathcal{T}, \mathcal{S})$ is the derivation containing just the sequent $\Gamma \longrightarrow G$. Otherwise, if $\{A_1, \ldots, A_n\}$ is the collection of $\prec$-minimal elements in $\mathcal{A}$ and if Π is the multicut derivation for the smaller table $\langle \mathcal{A} \setminus \{A_1, \ldots, A_n\}, \prec \rangle$ and the sequent $\Gamma, A_1, \ldots, A_n \longrightarrow G$, then $mcd(\mathcal{T}, \mathcal{S})$ is the derivation*

$$\frac{\Gamma \longrightarrow A_1 \quad \cdots \quad \Gamma \longrightarrow A_n \qquad \begin{matrix} \Pi \\ \Gamma, A_1, \ldots, A_n \longrightarrow G \end{matrix}}{\Gamma \longrightarrow G}\ mc$$

Multicut derivations are always open *derivations (that is, they contain leaves that are not proved). A* proof of a multicut derivation *is any (closed) proof that extends this open derivation.*

Elsewhere [MN07], we investigated the use of tables for obtaining non-redundant proofs in the Horn fragment. We used the following observation:

Proposition 2. *[MN07] Let Γ be a set of Horn clauses, $A \in \mathcal{P} \cap \Gamma$, and Ξ be an arbitrary LJF^t proof tree for $\mathcal{P}; [\Gamma] -_G\!\!\rightarrow$. Then every occurrence of a sequent with right-hand side the atom A is the conclusion of an I_r rule.*

If in a proof of Ξ, there are several non-trivial proofs for the subgoal A, one could table A and construct the corresponding multicut derivation for this table. By the property above, it must be the case that in all the possible proofs for this multicut derivation, the subgoal A has only one non-trivial proof. For example, when comparing the two derivation below, the left derivation could have several non-trivial subproofs for A, while the right derivation must have only one non trivial proof for A: the proof of the cut's left branch.

$$\frac{\Gamma \longrightarrow A \qquad \Gamma \longrightarrow G}{\Gamma \longrightarrow A \wedge G} \Longrightarrow \frac{\mathcal{P}; [\Gamma] \longrightarrow [A] \qquad \mathcal{P} \cup \{A\}; [\Gamma, A] \longrightarrow [A \wedge G]}{\mathcal{P}; [\Gamma] \longrightarrow [A \wedge G]}\ mc.$$

Furthermore, in Horn theory, we also showed how to obtain a table from an existing cut-free proof Ξ. The table consists of all atoms that are on the right-hand side of some sequent in Ξ. The occurrences of proved atoms in Ξ can be ordered using postorder traversal (i.e., process a nodes premises before processing the node). The final order used for the table (which is on atomic formulas and not their occurrences) is then obtained from this postorder traversal by retaining only the first occurrence of any repeated atomic formula.

4 Tree of Multicut Derivations - tmcd

In the previous section, we show how to obtain a table from an existing proof and how to construct a non-redundant proof from this table. However, it is only possible to construct such proofs for the Horn fragment. This is because, in this fragment, it is always possible to construct a proof where the context of the sequent doesn't change; Prolog constructs roughly these type of proofs. As the context doesn't change, all subgoals in a table are provable from the initial context, and hence, it is possible to construct a non-redundant proof by placing the multicut derivation at the root of the proof tree. When we move to more expressive logics, it might happen that the context changes in a proof, and therefore, a subgoal might not be provable from the initial context, but only from a context that is incremented with some formula.

We now propose a more complex derivation, where multicuts derivations may appear not only at the root of a tree, but anywhere in a proof. Instead of a single multicut derivation, specified by a table and a root sequent, we use a *tree of multicut derivations* (**tmcd**). Each node of this tree is a multicut derivation introducing into the context all the subgoals that are provable from its base context. An edge between two multicut derivations connects an open branch of the ancestor multicut derivation and the root sequent of its descendent multicut derivation; and it represents the provability dependency between the goal of the open derivation and the child **tmcd** connected by this edge. As done with tables, we can specify a derivation from a **tmcd**. The architecture of this new

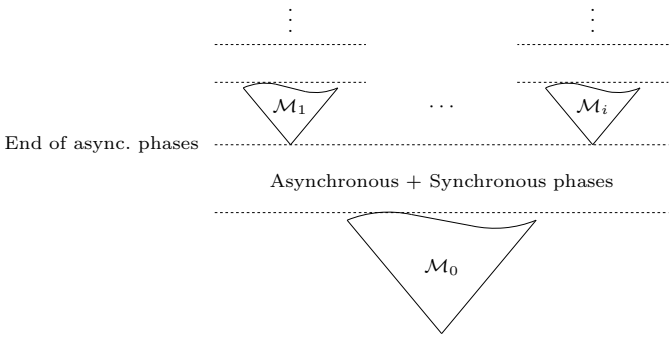

Fig. 2. Illustration of a derivation obtained from a tree of multicut derivations. The triangles are multicut derivations, the parallel dashed lines represents the end of an asynchronous phase, and the spaces between these lines are sequences of asynchronous and synchronous phases.

derivation is depicted in Figure 2. The idea is that each multicut derivation is only used when there is a change in the context, for example, by $\supset_r$ rule. Since the context can only be modified by the asynchronous rules $\forall_r, \exists_l$, and $\supset_r$[2], focusing provides a natural way to identify where to place a multicut derivation, namely, whenever an asynchronous phase ends. While focused cut-free proofs are structured in two alternating phases, namely, asynchronous phases and synchronous phases, focused cut proofs are structured with one more phase, called *cut phase*, appearing always between an asynchronous phase and the following synchronous phase. Moreover, the tree structure of a **tmcd** reveals in which branch of a multicut derivation a child multicut derivation is to be placed.

4.1 Extracting a tmcd from a Uniform Proof

We now specify how to extract a tree of multicut derivations from a finite (goal-directed) proof tree, that is, *LJF* proofs where all atoms are assigned negative polarity.

We only use sequents of the form $[\Gamma] \longrightarrow [G]$, that is, sequents where no more asynchronous rules can be applied. Given a tree path of multicut derivations $\mathcal{M}_1 \rightarrow \ldots \rightarrow \mathcal{M}_n$ where $n \geq 0$ and $\mathcal{M}_i = mcd(\langle \mathcal{A}_i, \prec \rangle, \mathcal{S}_i)$; a *LJF* proof tree, Ξ, with root $[\Gamma] \longrightarrow [G]$ and all atoms with negative polarity (that is a goal directed proof); and an atom $A_o \in \mathcal{A}_n$[3]; we proceed as follows: (1) extract the subtree Ξ_Γ of Ξ that contains all nodes in Ξ with context $[\Gamma]$. (2) compute the table, $\mathcal{T} = \langle \mathcal{A}, \prec \rangle$, obtained by applying the postorder traversal algorithm to Ξ_Γ, described in Section 3. (3) if $n > 0$, then there are two cases: (3.1) if $\mathcal{A}_n = \emptyset$, then skip to step (4) with $\mathcal{P} = \emptyset$; otherwise check in $\mathcal{M}_n$ the polarity context, $\mathcal{P}$, of the sequent in the derivation $\mathcal{M}_n$ with right hand side A_o. (4) increment the

[2] The first two rules augment the context with a new eigenvariable, and the last rule augments the context with some formula.

[3] If $\mathcal{A}_n = \emptyset$, the inputed atom is not important.

path tree with the new multicut derivation $\mathcal{M}_{n+1} = mcd(\mathcal{T}, \mathcal{P}; [\Gamma \cup \mathcal{P}] \longrightarrow [G])$, obtaining the path $\mathbf{Path} = \mathcal{M}_1 \rightarrow \ldots \rightarrow \mathcal{M}_n \rightarrow \mathcal{M}_{n+1}$, where the incremented edge either, connects the subgoal A_o in $\mathcal{M}_n$ to the root sequent of $\mathcal{M}_{n+1}$, if $\mathcal{A}_n \neq \emptyset$; or connects the roots of the multicut derivations, if $\mathcal{A}_n = \emptyset$. (5) if $\mathcal{A} \neq \emptyset$, for each child tree, Ξ^i_Γ, of Ξ_Γ determine, by inspection on Ξ, the immediate ancestor sequent in Ξ_Γ of Ξ^i_Γ, which contains an atomic formula, A^i, in the right hand side. Since, in uniform proofs, goals are first decomposed, it must be the case that the immediate ancestor sequent is of the form $[\Gamma] \longrightarrow [A^i]$, where A^i is an atom[4]. For all the children trees, Ξ^i_Γ, of Ξ_Γ, repeat the steps (1-5) inputing the child tree Ξ^i_Γ, the path of multicut derivations **Path**, and the atom A^i, until there are no more subtrees to be processed. The result of this process is a collection of multicut derivation paths. With these paths, one can easily build a tree by inspecting the original *LJF* proof to determine the correct order of a node's different branches. This algorithm terminates because the initial tree is finite, and the recursion is done with smaller trees. We illustrate part of this procedure in Figure 3 with an example.

The **tmcd** obtained up to now is still not the final result we are interested in because it might still happen that an atom that appears in an ancestor multicut derivation appears also in a descendent multicut derivation. Similarly as done after the post-order traversal in the algorithm specified in Section 3, we eliminate repeated appearances as follows: let $\mathcal{M}_a$ be a multicut derivation containing the atom A, and let $\mathcal{M}_d$ be a descendent multicut derivation of $\mathcal{M}_a$. If $\mathcal{M}_d$ has the base sequent $\mathcal{P}; [\Gamma] \longrightarrow [A]$ then we obtain a new **tmcd**, by removing the **tmcd**'s subtree with root $\mathcal{M}_d$; or if A is in $\mathcal{M}_d$, then we obtain a new **tmcd** by removing A from $\mathcal{M}_d$ and its possible descendent subtrees from the tree of multicut derivations.

To complete a **tmcd** to a proof, it is only necessary to add derivations containing at most one decide left rule, as shows the following proposition. The proof of this proposition also shows how to construct proofs such that if a subgoal, A, appears in a multicut derivation, then all its child trees contain only trivial proofs for A.

Lemma 1. *In any sequent, $\mathcal{P}; [\Gamma] \longrightarrow [G]$, of a **tmcd**, the polarity context $\mathcal{P} \subseteq \Gamma$.*

Proposition 3. *Let Ξ be a uniform LJF proof and let τ be the tree of multicut derivations obtained from Ξ as described before. The derivation obtained from τ can be completed to a proof by adding derivations containing only one D_l rule.*

Proof (*Sketch*) We proceed by induction on the depth of the tree of multicut derivations τ.

Base Case: Suppose that the depth of τ is 0, that is there is only one multicut derivation, $\mathcal{M}$, in τ. We prove the base case, by another induction on

[4] As we are dealing with all proofs that have only negative polarity atoms, a larger class of proofs, it may happen that there is no such atom, and hence we need the condition $\mathcal{A} \neq \emptyset$, before proceeding with the 5th step

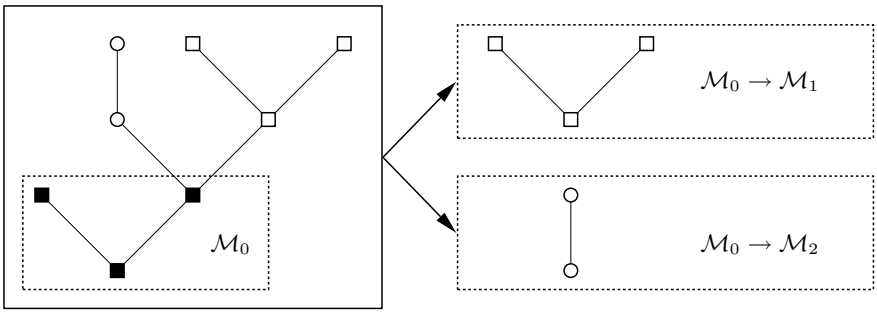

Fig. 3. Illustration of the procedure to obtain a collection of tree paths of multicut derivations from an existing (goal directed) proof. The three different kinds of nodes (filled squares, ellipses, and blank squares) represent sequents with different contexts. First, the subtree with the filled squares is extracted, and, from it, the multicut derivation $\mathcal{M}_0$ is constructed with the table obtained by using a postorder traversal in this tree and the tree's root. Later, this procedure is repeated with the two remaining subtrees, namely the one with elliptical nodes and the one with blank squares, obtaining, respectively, the multicut derivations $\mathcal{M}_1$ and $\mathcal{M}_2$. Finally, these data structures are connected to the inputed path, which contains only the multicut derivation $\mathcal{M}_0$, and outputs the two tree paths depicted in the figure.

the height of $\mathcal{M}$. The base case is trivial, since it would mean that the multicut $\mathcal{M}$ contains only one cut, and this cut would have to use an atom that is already present in the contex of the proof. And therefore, it would require only one decision rule to complete the derivation to a proof.

Now the inductive step: Consider that the sequent $\mathcal{P}; [\Gamma \cup \mathcal{P}] \longrightarrow [A_i]$ is a sequent branch in the multicut derivation $\mathcal{M}$. We can find a proof for this sequent with one decision left rule, by proceeding as follows: We check in Ξ the formula F that was decided on to prove A_i. After this decision is made, it suffices to perform decide right rules, since this would decompose the possible subgoal originated by performing the previous decide left rule (for instance if $F = G \supset A$). Because of how the tables are constructed, when this subgoal is completely decomposed it must be the case that it encounters a previously proved atom, that is, a positive atom which, by Lemma 1, is already present in the context, and therefore, another decision right finishes the proof.

Inductive Step: We now have to show that there is a derivation, between a branch sequent of a multicut derivation and its direct descendent multicut derivation, that contains only one decision left rule. This is similar to the base case; it suffices to decide in the same formula as in Ξ; and, after performing a synchronous phase and later a possible asynchronous phase, there are two possible outcomes: 1) The goal is not a positive atom and hence there must be a descendent multicut for this goal 2) It is a positive atom and , by Lemma 1, a right decision rule is enough to finish the proof for this branch of the multicut derivation. In both cases, there is only one decision left rule. □

The following proposition states that the **tmcd**, obtained from a Horn uniform proof, is composed of a single multicut derivation. As we would expect, the procedure described in this subsection is a generalization of the procedure described in Section 3.

Proposition 4. *Let Ξ be a uniform LJF proof in Horn theory. Then the tree of multicut derivations extracted from Ξ, as described before, has depth 0.*

The following example illustrates how a tree of multicut derivations can be extracted from a proof.

Example 1. Consider the following uniform proof, where $\Gamma = \{A, A \supset B, B \supset C\}$ is the initial context, A, B, C are negative atoms, and some simple derivations are not shown.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{[\Gamma, A] \longrightarrow [A]}\, D_l, I_l}{[\Gamma, A] \longrightarrow [B]}\, D_l, \supset_l, R_r, []_r}{[\Gamma, A] \longrightarrow [C]}\, D_l, \supset_l, R_r, []_r}{[\Gamma] \text{-}_{A \supset C} \!\!\longrightarrow}\, R_r, \supset_r, []_l, []_r \qquad \dfrac{\dfrac{\dfrac{}{[\Gamma] \longrightarrow [A]}\, D_l, I_l}{[\Gamma] \longrightarrow [B]}\, D_l, \supset_l, R_r, []_r}{[\Gamma] \text{-}_{B} \!\!\longrightarrow}\, R_r, []_r}{[\Gamma] \longrightarrow [(A \supset C) \wedge B]}\, D_r, \wedge_r$$

From the proof's subtree, containing only the context Γ, we obtain the table $\mathcal{T}_1 = (a \prec b)$[5], and from its remaining child tree, we obtain the table $\mathcal{T}_2 = (a \prec b \prec c)$. However, since the subgoal a, b already appear in $\mathcal{T}_2$'s ancestor, $\mathcal{T}_1$, these subgoals are removed from it, and we obtain the refined table $\mathcal{T}_2' = (c)$. The tree of multicut derivation derived from the proof above is as follows:

$$\dfrac{[\Gamma] \longrightarrow [A] \qquad \dfrac{[\Gamma, A] \longrightarrow [B] \qquad \dfrac{[\Gamma, A, B] \longrightarrow [C] \quad [\Gamma, A, B, C] \longrightarrow [C]}{[\Gamma, A, B] \longrightarrow [C]} \;\vdots\; [\Gamma, A, B] \longrightarrow [(A \supset C) \wedge B]}{[\Gamma, A] \longrightarrow [(A \supset C) \wedge B]}}{[\Gamma] \longrightarrow [(A \supset C) \wedge B]}$$

The ellipses denote an edge in the tree of multicut derivations and, for simplicity, we omit the polarity context of the sequents.

It is important to notice that, even though with the procedure above it is possible to construct non-redundant proofs for several proofs, there are cases where it is not possible to completely eliminate all redundancies. For instance, the tree of multicut derivations obtained from the following proof would still require two non-trivial proofs for A.

$$\dfrac{\dfrac{\dfrac{\Xi}{[\Gamma], G \longrightarrow A}}{[\Gamma] \longrightarrow (G \supset A)}\, \supset_r \qquad \dfrac{\dfrac{\Xi}{[\Gamma], G \longrightarrow A}}{[\Gamma] \longrightarrow (G \supset A)}\, \supset_r}{[\Gamma] \longrightarrow (G \supset A) \wedge (G \supset A)}\, D_r, \wedge_r, R_r$$

[5] Strictly, a table is a tuple with a set of atom and a partial order. We represent the tables in this example as an ordered sequence of atoms to ease readability.

The problem is that A would appear in multicuts situated in two different branches of the **tmcd**, and therefore, it is not possible to use the proof of A appearing in one branch to immediately prove (with a trivial proof) A in the other branch. With the procedure above, it is only possible to assure that if a subgoal appears in a multicut, then all of this multicut's descendents have trivial proofs for A. One could try to solve this problem by tabling formulas more complex than atoms, but then, as we investigate elsewhere [Nig07], to complete **tmcd**s constructed with more complex lemmas can be much harder; or one could check if A is provable from the context $[\Gamma]$, and if so, include A in the multicut derivation obtained for the subtree Ξ_Γ, containing sequents with context Γ, but it is easy to find examples where this approach would not work.

4.2 Extracting a tmcd from a *LJF* proof

In the previous subsection, we considered only uniform *LJF* proofs, that is, proofs where all atoms have negative polarity. We now extend the results obtained before to a more richer class of *LJF* proofs where there can also be atoms with positive polarity. When considering this more general class of proofs, there are two main differences with respect to uniform *LJF* proofs:

1. **Reaction Left with Atomic Formula** - By allowing atoms with positive polarity, proofs can perform forward chaining steps. For instance, consider the following derivation in which the atom A has positive polarity:

$$\dfrac{[\Gamma]-_G\!\!\rightarrow \quad \dfrac{[\Gamma], A \longrightarrow [G']}{[\Gamma] \xrightarrow{A} [G']}\, R_r}{[\Gamma] \xrightarrow{G \supset A} [G']}\, \supset_l$$

   This type of derivation is not possible to occur in a uniform proof, since the only rule that introduces a focused negative atom in the left is the initial left rule;
2. **Initial Right Rule** - Initial right rules can end the proof; in uniform proofs this would never happen since all atoms are of negative polarity, and hence, when such atoms are focused in the right the focus has to be lost.

Definition 3. *A derivation, Ξ, is purely synchronous (reps. asynchronous) if Ξ contains only synchronous (resp. asynchronous) rules.*

The following lemmas are proved by induction on the derivation Ξ. And the theorem follows immediately from the lemmas.

Lemma 2. *If Ξ is a purely asynchronous LJF derivation, then atoms in Ξ can have arbitrary polarities.*

Lemma 3. *If Ξ is a purely synchronous LJF derivation either containing a rule R_l, introducing a focused atom; or containing a rule I_r, then there is at least one atom in Ξ with positive polarity. Otherwise, Ξ can have all atoms with negative polarity.*

Theorem 1. *Only LJF derivations that contain a rule R_l, introducing a focused atom, or/and contain a rule I_r, can occur in a proof with positive atoms and never occur in proofs with no positive atom.*

We modify in two ways the procedure, described in the previous subsection, used to extract **tmcd**s from uniform proofs. The first modification is related to the first difference, namely that reaction left rules can happen with atomic formulas. We distinguish two cases according to the context immediately after the reaction rule is performed: 1) the context is not incremented with a formula. This case occurs, for example, when there is a previous decision left rule in a positive atom and immediately there is a reaction left rule. Clearly, these cycles don't affect the context in any way. 2) (called *forward chaining case*) the context is augmented with the positive atom that was immediately unfocused. We consider only the later case, since the former case is easily dealt by the post-order procedure specified in Section 3. We identify when the second case occurs in a proof, Ξ, by checking the sequents of the form $[\Gamma] \xrightarrow{A} [G']$ in Ξ; if A is of positive polarity and $A \notin \Gamma$, then there is an occurrence of the forward chaining case. When constructing the table for the sub-derivation Ξ_Γ of Ξ, containing only sequents with context Γ, include the (forward chaining) atom A into the table as a maximal element. The idea is to transform a forward chaining steps into a a cut, as illustrates the following transformation:

$$\dfrac{\dfrac{\dfrac{[\Gamma], A \longrightarrow [G']}{[\Gamma]{-_G\!\!\rightarrow}\quad [\Gamma] \xrightarrow{A} [G']}\, R_r}{\dfrac{[\Gamma] \xrightarrow{G \supset A} [G']}{[\Gamma] \longrightarrow [G']}\, D_l}\, \supset_l}{} \quad \Longrightarrow \quad \dfrac{\dfrac{\dfrac{}{\mathcal{P}; [\Gamma]{-_G\!\!\rightarrow}\quad \mathcal{P}; [\Gamma] \xrightarrow{A} [A]}\, I_l}{\dfrac{\mathcal{P}; [\Gamma] \xrightarrow{G \supset A} [A]}{\mathcal{P}; [\Gamma] \longrightarrow [A]}\, D_l}\, \supset_l \qquad \mathcal{P}'; [\Gamma, A] \longrightarrow [G']}{\mathcal{P}; [\Gamma] \longrightarrow [G']}\, mc$$

The second modification is related to the second difference, namely that initial right rules can appear in a proof. In the original proof, Ξ, we change the sequents of the form $[\Gamma]{-_A\!\!\rightarrow}$ to $[\Gamma] \longrightarrow [A]$, what means that the atom A is also included in a table. Furthermore, this atom will be included as a minimal element, and therefore, when the multicut derivation of this table is constructed, the polarity of A is changed from negative to positive by performing a multicut with A as a cut formula already at the beginning of the derivation.

The remaining steps are the same as in the previous subsection.

Proposition 5. *Let Ξ be an LJF proof and let τ be the tree of multicut derivations obtained from Ξ as described before. The derivation obtained from τ can be completed with derivations containing only one D_l rule.*

Proof (*Sketch*) This proposition is proved in a similar way as Proposition 3. The only extra consideration is with respect to the forward chaining steps, which adds an extra step in the inductive proof. When there is a forward chaining step, over an atom A, in Ξ, it means that there is a branch in a multicut derivation of the form $\mathcal{P}; [\Gamma] \longrightarrow [A]$. It suffices to check which formula in Γ was used to make a forward chaining step and decide on the same formula. This branch would need therefore of one decide left rule to reach its descendent multicut derivation(s), or to finish the proof, similarly as depicted in the transformation above. □

Example 2. Consider the following proof where A, B, C are of positive polarity, the initial context $\Gamma = \{A, A \supset B, B \supset C\}$, and some trivial derivations are not shown:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{[\Gamma, B, C] \longrightarrow [C]}\, D_r}{[\Gamma, B] \stackrel{B \supset C}{\longrightarrow} [C]}\, \supset_l, R_l, []_l}{[\Gamma, B] \longrightarrow [C]}\, D_l}{[\Gamma] \stackrel{A \supset B}{\longrightarrow} [C]}\, \supset_l, R_l, []_l}{[\Gamma] \longrightarrow [C]}\, D_l$$

There are two occurrences of the forward chaining case: one over the atom B and another over the atom C. The following tree of multicut derivation is extracted from this proof. For simplicity we omit the polarity context of the sequents.

$$\dfrac{[\Gamma] \longrightarrow [A] \qquad \dfrac{[\Gamma, A] \longrightarrow [B] \qquad \dfrac{[\Gamma, A, B] \longrightarrow [C] \qquad [\Gamma, A, B, C] \longrightarrow [C]}{[\Gamma, A, B] \longrightarrow [C]}}{[\Gamma, A] \longrightarrow [C]}}{[\Gamma] \longrightarrow [C]}$$

Notice that it is very simple to complete the derivation above to a proof. This observation provides another possible use for a tree of multicut derivations: as *proof objects*. As we discuss in the next section, one could send a **tmcd** as a proof object to a consumer, who would need only to check it, by searching for simple derivations to complete it to a proof.

5 Tables as Proofs and Related Work

In the *proof carrying code* setting [Nec97], proof objects are transmitted together with mobile codes to assure that some (safety) properties are satisfied by these programs. Before a client executes the transmitted code, it checks that the proof that that code is carrying proves the program's safety. Thus, proof objects must be engineered so that they are not too large (in order to reduce transmission costs) and not too complex to check (in order to reduce resource requirements on client proof checkers).

In some previous work [MN07], we argue about the plausibility of tables to be used as proof objects. We can use the same arguments for a **tmcd**s. First, **tmcd**s only represent declarative information and not procedural information: in particular, **tmcd**s only describe what is provable and does not enter in detail on how things are proved. The task of proof checking a **tmcd** is reduced to the problem of searching for derivations to complete this proof object. The trade-offs between proof size and proof checking time are fairly clear: if the producer tables all the subgoals of a proof, then the proof checking can be straight forward, simply by searching for small derivations (with only one Decide left rule, as stated in Proposition 5). If not all subgoals are tabled, proof checking may reduce to the search of bigger derivations; for example derivations containing more than

one decide left rule, but, on the other hand, the **tmcd** would smaller and less costly to be transmitted.

Necula and Lee [NL98] proposed the logical framework LF_i, strongly based in the Edinburgh Logical Framework [HHP93]. In LF_i, when constructing a proof object, (some) redundant parts of a proof are deleted and later when the proof object is typed checked, the missing parts of the proof are reconstructed. We would like to hightlight some differences between our approaches: (1) Even though the proof object constructed in their approach does not contain redundancies, the proof represented by a LF_i object is still a redundant; whereas in our approach, we give a more fundamental solution, by constructing new non-redundant proof from the redundant one; (2) A consumer of a LF_i proof object needs to implement the reconstruction algorithm to verify the correctness of the proof; in our approach, the consumer can have a variety of search options to complete **tmcd**s to proofs, for instance, top-down (reps. bottom-up) approach, where the consumer completes first the leaves (resp. root) of the **tmcd** and later the root (resp. leaves) of the **tmcd**; (3) The LF_i reconstruction algorithm strongly relies on the sequence of rules that are applied in the proof. Therefore, it is necessary to include in the proof object the trace of many of the inference rules applied throughout a proof, for instance the asynchronous rules; whereas in our approach it is not necessary to send theses traces.

6 Conclusions and Future Works

This paper presents an approach, from a proof theoretic point of view, for reducing size of proofs through redundancy elimination. By a careful study of focused proofs, we propose a new structure, called tree of multicut derivations, to be the backbone for the construction of non-redundant proofs. We investigate, first, how to extract these data structures from uniform proofs, *LJF* proofs that have only negative polarity atoms, and later, we investigate how proofs containing positive polarity atoms differ from uniform proofs, to propose a procedure to extract tree of multicut derivations from this richer type of proofs. Finally, we briefly argue about the plausibility of **tmcd**s to be used as proof objects.

We studied how to obtain non redundant proofs from any *LJF* proof. A future work could be how to remove redundancies in even larger systems containing induction and coinduction, for example $\mu MALL$ [BM07]. Another direction for future work could be to investigate how to use a *multifocusing* system [MS07], where set of formulas can be simultaneously focused on, to reduce even more the size of proofs. When considering **tmcd**s as proof objects, one still has to send together with a **tmcd** the witnesses for the $\exists_r$ and $\forall_l$ rules. A future work could be to use unification to avoid that witnesses have to be also communicated, possibly in a similar way as done in Necula and Rahul's oracle strings [NR01].

Acknowledgments I thank Dale Miller for our fruitful discussions and for his helpful comments. This work has been supported in part by INRIA through the "Equipes Associées" Slimmer and by the Information Society Technologies

programme of the European Commission, Future and Emerging Technologies under the IST-2005-015905 MOBIUS project.

References

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

[App01] Andrew W. Appel. Foundational proof-carrying code. In *16th Symp. on Logic in Computer Science*, 2001.

[BM07] David Baelde and Dale Miller. Least and greatest fixed points in linear logic. Accepted to LPAR07, April 2007.

[CH00] Pierre-Louis Curien and Hugo Herbelin. The duality of computation. In *ICFP '00: Proceedings of the fifth ACM SIGPLAN international conference on Functional programming*, pages 233–243, New York, NY, USA, 2000. ACM Press.

[CPP06] Kaustuv Chaudhuri, Frank Pfenning, and Greg Price. A logical characterization of forward and backward chaining in the inverse method. In U. Furbach and N. Shankar, editors, *Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR'06)*, number 4130 in LNCS, pages 97–111. Springer, August 2006.

[HHP93] Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *Journal of the ACM*, 40(1):143–184, 1993.

[JNS05] Radha Jagadeesan, Gopalan Nadathur, and Vijay Saraswat. Testing concurrent systems: An interpretation of intuitionistic logic. In *Proceedings of FSTTCS*, 2005.

[LM07] Chuck Liang and Dale Miller. Focusing and polarization in intuitionistic logic. In J. Duparc and T. A. Henzinger, editors, *CSL 2007: Computer Science Logic*, volume 4646 of *LNCS*, pages 451–465. Springer-Verlag, 2007.

[MN07] Dale Miller and Vivek Nigam. Incorporating tables into proofs. In J. Duparc and T. A. Henzinger, editors, *CSL 2007: Computer Science Logic*, volume 4646 of *LNCS*, pages 466–480. Springer-Verlag, 2007.

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

[MS07] Dale Miller and Alexis Saurin. From proofs to focused proofs: a modular proof of focalization in linear logic. In J. Duparc and T. A. Henzinger, editors, *CSL 2007: Computer Science Logic*, volume 4646 of *LNCS*, pages 405–419. Springer-Verlag, 2007.

[Nec97] George C. Necula. Proof-carrying code. In *Conference Record of the 24th Symposium on Principles of Programming Languages 97*, pages 106–119, Paris, France, 1997. ACM Press.

[Nig07] Vivek Nigam. Investigating the reuse of lemmas. June 2007.

[NL98] George C. Necula and Peter Lee. Efficient representation and validation of proofs. In *13th Symp. on Logic in Computer Science*, pages 93–104, Los Alamitos, CA, 1998. IEEE Computer Society Press.

[NR01] George C. Necula and Shree Prakash Rahul. Oracle-based checking of untrusted software. In *POPL*, pages 142–154, 2001.