Deleting Redundancy in Proof Reconstruction

Stephan Schmitt[1] Christoph Kreitz[2]

[1] Fachgebiet Intellektik, Fachbereich Informatik, Darmstadt University of Technology
Alexanderstr. 10, 64283 Darmstadt, Germany
steph@informatik.tu-darmstadt.de
[2] Department of Computer Science, Cornell University
Ithaca, NY 14853, USA
kreitz@cs.cornell.edu

Abstract. We present a framework for eliminating redundancies during the reconstruction of sequent proofs from matrix proofs. We show that search-free proof reconstruction requires knowledge from the proof search process. We relate different levels of *proof knowledge* to *reconstruction knowledge* and analyze which redundancies can be deleted by using such knowledge. Our framework is uniformly applicable to classical logic and all non-classical logics which have a matrix characterization of validity and enables us to build adequate conversion procedures for each logic.

1 Introduction

Automated theorem proving in non-classical logics has become important in many branches of Artificial Intelligence and Computer Science. As a result, the resolution principle [14] and the connection method [1, 2], which both have led to efficient theorem provers for classical logic [22, 9, 3], have been extended to characterizations of logical validity in modal logics, intuitionistic logic, and fragments of linear logic [20, 10, 21, 19, 7]. These characterizations are the foundation of efficient and uniform proof search procedures for all these logics [12, 13, 7] which are used as inference engines in automatic program development systems [8, 4] and other problem-oriented applications [5].

In many applications of theorem proving it is not sufficient to show *that* a theorem is valid. The need for further processing (e.g. generating programs from proofs) or a deeper understanding of the proof requires that proof details can be presented in a comprehensible form. On the other hand, the efficiency of automated proof methods strongly depends on a compact and machine-oriented characterization of logical validity. This makes it necessary to *reconstruct* a sequent proof, a natural deduction proof, or even a proof in a semi-natural mathematical language from an automatically generated machine proof.

As a complement to existing matrix-based proof search methods we have developed a uniform procedure for transforming classical and non-classical matrix proofs back into sequent style systems [16, 17, 7]. This procedure creates a sequent proof for a given formula by traversing its formula tree in an order which respects a *reduction ordering* induced by the matrix proof. It selects an appropriate sequent rule for each visited node by consulting tables which represent the peculiarities of the different logics. At nodes which cause the sequent proof to branch the reduction ordering has to be divided appropriately and certain redundancies need to be eliminated in order to ensure completeness.

H. de Swaart, ed., *International Conference TABLEAUX-98*, LNAI 1397, pp. 262–277,

Redundancy elimination is the most crucial aspect of proof reconstruction if matrix proofs shall be converted into sequent proofs *without additional search*. We will show that the complexity of eliminating redundancy strongly depends on the amount of *proof knowledge* made available by the proof search method. If the procedure has to rely only on the matrix characterization then additional search becomes necessary, but redundancies can be eliminated in polynomial time in the size of the matrix proof if the history of the proof search is known.

In this paper we shall present a detailed analysis of possible redundancies in a reduction ordering and of the *proof reconstruction knowledge* which is necessary to delete them. We shall study different levels of proof knowledge and their effects on the proof reconstruction process. We will introduce *prefixed connections* as a logic-independent concept which allows us to extract conditions for extending the elimination of redundancies to a maximal level. Our result can be used as a general framework for building efficient and complete proof reconstruction procedures for non-classical logics if the proof search method is known.

In Section 2 we give a brief summary of matrix characterizations and proof reconstruction in non-classical logics. Section 3 classifies redundancies in matrix proofs and the requirements for eliminating them. In Section 4 we discuss proof knowledge available from the *extension procedure* [2, 12] and the resulting redundancy elimination methods. In Section 5 we investigate the complexity, adequate completeness, and correctness of the refined proof reconstruction method.

2 Preliminaries

Matrix characterizations of logical validity were introduced for classical logic [1, 2] and later extended to intuitionistic and modal logics [21] and fragments of linear logic [7]. On this basis an efficient proof search procedure has been elaborated [12, 7] which captures all these logics in a uniform way. A uniform procedure for converting matrix proofs into sequent proofs has been developed in [17, 7].

2.1 Matrix Calculi for Non-classical Logics

In matrix proofs a formula F is represented by its formula tree $\ll$ whose nodes are called *positions*. Each position x of $\ll$ refers to a unique subformula F_x of F. The root w of $\ll$ represents F itself while its leaves (or *atomic positions*) refer to the atoms of F. Because of the corresponding subformula relation $\ll$ is called the *tree ordering* of F. Each position x is associated with a *polarity* $pol(x) \in \{0, 1\}$, a *principal type* $Ptype(x)$, and its operator $op(x)$. The polarity determines whether F_x will appear in the succedent of a sequent proof ($pol(x)$=0) or in its antecedent. $F_x^{pol(x)}$ denotes the *signed formula* at position x. The principal type $Ptype(x)$ is the formula type of F_x according to the tableaux classification in [21, 6]. Principal types are a compact and logic-independent way to express proof-relevant properties of a formula [17]. In the following we will only consider the types α, β, and *atom*. Two atomic positions x and y are *α-related* ($x \sim_\alpha y$) or *β-related* ($x \sim_\beta y$) if their greatest common predecessor in $\ll$ is has principal type α (or β). A *non-normal form matrix* of F is a two-dimensional representation of the

atomic positions of $\ll$ such that β-related positions are placed on top of each other while α-related are written side by side. A *path* p through F is a *maximal* subset of atomic positions which are pairwisely not β-related.

A *connection* is a subset $\{c_1, c_2\}$ of a path p such that the atoms F_{c_1} and F_{c_2} have the same predicate symbol but different polarities. It is *complementary* if F_{c_1} and F_{c_2} can be unified by some *combined substitution* $\sigma = \langle \sigma_Q, \sigma_L \rangle$. σ_Q is the usual quantifier substitution while σ_L, used to analyze non-permutabilities of sequent rules in a non-classical logic $\mathcal{L}$, unifies the prefixes of the connected atoms. The *prefix* of an atom F_x is a string consisting of special positions in $\ll$ between the root w and x. A set of connections $\mathcal{C}$ *spans* a formula F (or is a *spanning mating*) if each path contains a complementary connection $c \in \mathcal{C}$. The substitution σ induces a relation $\sqsubset$ on the positions of $\ll$ such that $(x, a) \in \sqsubset$ iff $\sigma(x) = a$ and a is not a variable. σ is *admissible* if the *reduction ordering* $\lhd = (\ll \cup \sqsubset)^+$ is irreflexive and some additional global conditions hold. Finally, multiple uses of subformulae in a matrix proof are represented by a *combined multiplicity* $\mu = \langle \mu_Q, \mu_L \rangle$ of the positions x in $\ll$. Using these concepts logical validity can be uniformly characterized as follows (see [21, 7] for details).

Theorem 1. *A formula F is* valid *wrt. a logic $\mathcal{L}$ iff there exists a multiplicity μ, an admissible substitution σ, and a set of connections $\mathcal{C}$ which spans F.*

Proof search procedures based on the matrix characterization of logical validity are generalizations of the *extension method* [2] to non-normal form matrices and non-classical logics [12, 7]. They consist of a general path-checking algorithm and a uniform and efficient algorithm for prefix unification [11].

In the following we shall use the reduction ordering $\propto^\star$, a slight technical modification of $\lhd$ (see [17]) as starting point for proof reconstruction and consider only those aspects of σ which are encoded in $\propto^\star$. The following example illustrates the matrix characterization of logical validity in intuitionistic logic.

Example 1. Consider $F_1 \equiv \neg A \vee \neg B \Rightarrow \neg B \vee \neg A$ and its intuitionistic matrix-proof, represented by the reduction ordering $\propto^\star$ on the left hand side of Figure 1. The name α_i, β_i, or a_i for a position x encodes its principal type while its main operator $op(x)$ and the polarity $pol(x)$ are written beside it. There are two paths through F_1, $p_1 = \{a_1, a_3, a_4\}$ and $p_2 = \{a_2, a_3, a_4\}$. The two connections $\{a_1, a_4\}$ and $\{a_2, a_3\}$ (depicted at atomic positions) span F_1 wrt. some intuitionistic admissible substitution σ which induces the relation $\sqsubset = \{(\alpha_6, \alpha_2), (\alpha_5, \alpha_3)\}$ (indicated by curved arrows).

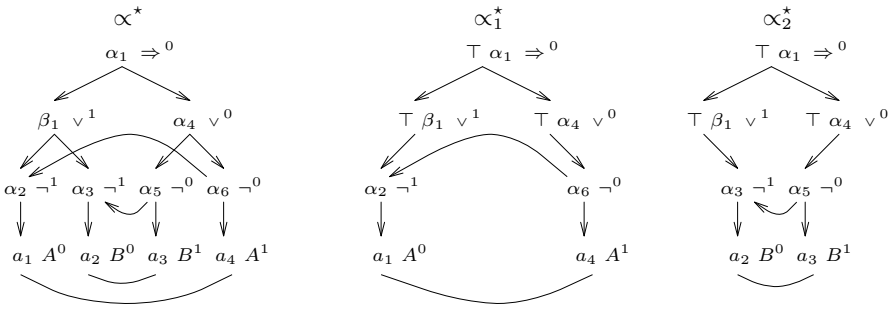


Fig. 1. Reduction ordering $\propto^\star$ for $\neg A \vee \neg B \Rightarrow \neg B \vee \neg A$

2.2 Proof Reconstruction in Non-classical Logics

An algorithm for converting a matrix proof of a formula F into a sequent proof essentially has to traverse the reduction ordering $\propto^\star$ while constructing a sequent rule at each visited position x. We focus on conversion into multiple conclusion sequent calculi (cf. [6]) where a sequent $\Gamma \vdash \Delta$ is described by *associated sets* of signed formulae: $\mathcal{S}_\Delta = \{F_x^0 \mid F_x \in \Delta\}$, $\mathcal{S}_\Gamma = \{F_x^1 \mid F_x \in \Gamma\}$, and $\mathcal{S} = \mathcal{S}_\Delta \cup \mathcal{S}_\Gamma$. The main operator $op(x)$ and polarity $pol(x)$ uniquely describe the sequent rule necessary to reduce the sequent formula $F_x^{pol(x)} \in \mathcal{S}$. The subformulae resulting from applying this rule to $F_x^{pol(x)}$ are determined by the set $succ(x)$ of immediate successors of x in $\ll$. The induced relation $\sqsubset$ encodes the non-permutabilities of sequent rules in a logic $\mathcal{L}$ and "blocks" certain positions x: rule construction for x will be delayed until all its predecessors wrt. $\sqsubset$ have been visited first.

At a β-position x a sequent proof branches into two independent subproofs. Accordingly, the reduction ordering $\propto^\star$ must be split into suborderings $\propto_1^\star, \propto_2^\star$ and conversion continues separately on each of them. The operation $split(\propto^\star, x)$, developed in [17] and illustrated in Figure 1, first divides $\propto^\star$ and then eliminates components of each $\propto_i^\star$ which are no longer relevant for the corresponding sequent subproof. Proof reconstruction terminates when all branches of the sequent proof have been closed by converting a connection from the matrix proof. Because of the uniformity of the conversion procedure the technical details are subtle. In the following we give a rather informal account of traversal and splitting and refer to [17, 18, 7] for a complete and algorithmic presentation.

Traversal of $\propto^\star$. Each position in $\ll$ has to be visited and marked as *solved* if it is not blocked. A position x is *open* (i.e. elegible to be visited next) if its immediate predecessor $pred(x)$ is already solved but x is not. After x has been solved the set P_o of all open positions is updated to $P_o' = (P_o \setminus \{x\}) \cup succ(x)$. If x is a β-position, then $split(\propto^\star, x)$ (see below) divides $\propto^\star$ into $\propto_1^\star, \propto_2^\star$, recomputes the corresponding sets P_o^1, P_o^2 and each $\propto_i^\star$ is traversed recursively. If two solved positions form a complementary connection then the conversion of $\propto_i^\star$ terminates. P_o is initialized as $P_o = \{w\}$ where w is the root of $\propto^\star$.

Example 2. Consider the formula F_1 from Example 1 and its matrix proof represented by the reduction ordering in Figure 1. We begin by initializing $P_o = \{\alpha_1\}$. We solve α_1 and construct the sequent rule $\Rightarrow r$ obtained from the main operator $\Rightarrow$ and polarity 0. Updating P_o yields P_o=$\{\beta_1, \alpha_4\}$. Solving α_4 next leads to applying $\vee r$ to $F_{\alpha_4}^0$. At β_1 we create $\vee l$ and the sequent proof branches: $split(\propto^\star, \beta_1)$ divides $\propto^\star$ into the suborderings $\propto_1^\star$ and $\propto_2^\star$ depicted in Figure 1 (where a $\top$ marks the already solved positions). Recomputing the sets of open positions results in $P_o^1 = \{\alpha_2, \alpha_6\}$ and $P_o^2 = \{\alpha_3, \alpha_5\}$.

We continue by traversing $\propto_1^\star$. The position α_2 is blocked by α_6 since the corresponding sequent rule $\neg r$ has to be applied before $\neg l$ which belongs to α_2. We must solve α_6 to unblock α_2. The atomic position a_4 is next but no sequent rule will be generated since its connection partner a_1 has not been solved yet. We solve α_2 and complete the subproof by applying the axiom rule based on the connection $\{a_1$,$a_4\}$. Converting the subordering $\propto_2^\star$ works as before. The resulting sequent proof is shown to the right.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\, ax.(a_1,a_4)}{\neg A, A \vdash}\, \neg l(\alpha_2) \quad \dfrac{\dfrac{}{B \vdash B}\, ax.(a_2,a_3)}{\neg B, B \vdash}\, \neg l(\alpha_3)}{\neg A \vdash \neg B, \neg A}\, \neg r(\alpha_6) \quad \dfrac{\dfrac{}{\neg B \vdash \neg B, \neg A}\, \neg r(\alpha_5)}{\neg A \vee \neg B \vdash \neg B, \neg A}\, \vee l(\beta_1)}{\dfrac{\neg A \vee \neg B \vdash \neg B \vee \neg A}{\vdash \neg A \vee \neg B \Rightarrow \neg B \vee \neg A}\, \Rightarrow r(\alpha_1)}\, \vee r(\alpha_4)$$

Splitting at β-positions. The main modification of the reduction ordering during proof reconstruction occurs when traversal has reached a β-position x and $\propto^\star$ has to be divided into $\propto_1^\star$ and $\propto_2^\star$. If $\{x_1, x_2\}$ are the successors of x then $F_{x_1}^{pol(x_1)}$ will move to the left subproof of the corresponding sequent proof and $F_{x_2}^{pol(x_2)}$ to the right one. Since the set of open positions P_o encodes the actual sequent, only one of the two successors of x will be added to each P_o^i, i.e. $P_o^i = (P_o \setminus \{x\}) \cup \{x_i\}$. Formally splitting is based on the following definitions.

Definition 1. *Let x be a position of $\propto^\star$ and $\ll^x$ the subtree ordering with root x and position set $pos(x)$, including the pair $(pred(x), x) \in \ll^x$. The* restriction *of $\propto^\star$ involving positions from $pos(x)$ is defined as $t^x := \ll^x \cup \sqsubset^x$, where $\sqsubset^x := \{(x_1, x_2) \in \sqsubset \mid x_1 \in pos(x) \vee x_2 \in pos(x)\}$. If $\mathcal{C}$ is the connection set of $\propto^\star$ then $\mathcal{C}^x := \{\{c_1, c_2\} \in \mathcal{C} \mid c_1 \in pos(x)\}$.*

Definition 2. *Let x be of type β and $succ(x) = \{x_1, x_2\}$. The β-split of $\propto^\star$ at x is defined by β–split$(\propto^\star, x) := [\propto_1^\star, \propto_2^\star]$, where $\propto_1^\star = \propto^\star \setminus t^{x_2}$ and $\propto_2^\star = \propto^\star \setminus t^{x_1}$. For the subrelations and connections we have $\sqsubset_i := \sqsubset \setminus \sqsubset^{x_j}$, $\ll_i := \ll \setminus \ll^{x_j}$, and $\mathcal{C}_i := \mathcal{C} \setminus \mathcal{C}^{x_j}$ where $i \neq j \in \{1, 2\}$.*

After a *β–split* certain redundancies need to be deleted from each $\propto_i^\star$. This improves the efficieny of the reconstruction process and is necessary for ensuring its completeness when dealing with non-classical logics. We will discuss this now.

3 Classifying Redundancy in Matrix Proofs

Usually, the order in which a reduction ordering $\propto^\star$ can be traversed while respecting the 'blocks' induced by $\sqsubset$ is not unique. In Example 2 we could have visited α_1, α_4, and then α_5 instead of β_1. This, however, would not lead to a successful sequent proof since applying the $\neg r$ rule corresponding to α_5 causes the deletion of the formula $\neg A$ which is relevant for completing the proof. Thus the reduction ordering $\propto^\star$ is *not complete* wrt. rule non-permutabilities of the (non-standard) sequent calculus. In [17] we have introduced the concept of *wait*-labels which are dynamically assigned to special positions of $\propto^\star$ during conversion and make $\propto^\star$ complete wrt. all non-classical logics under consideration.

In intuitionistic logic an open position $x \in P_o$ is blocked by such a *wait*-label (denoted by $wait[x] = \top$) iff applying the corresponding sequent rule to F_x^0 would delete proof-relevant formulae. In Example 2 *wait*-labels must be assigned to α_5 and α_6 after solving α_1, α_4 since reducing $F_{\alpha_5}^0$ would delete $F_{\alpha_6}^0$ and vice versa. Hence β_1 must be solved next by applying β–split$(\propto^\star, \beta_1)$. But the resulting suborderings $\propto_i^\star$ contain redundancies which would create a deadlock. In $\propto_1^\star = \propto^\star \setminus t^{\alpha_3}$, for instance, the open positions are $P_o^1 = \{\alpha_2, \alpha_5, \alpha_6\}$ where α_5, α_6 are blocked by *wait*-labels and α_2 is blocked because of $(\alpha_6, \alpha_2) \in \sqsubset$. $F_{\alpha_5} = \neg B$ is not relevant for the subproof represented by $\propto_1^\star$ and can safely be deleted by applying $\neg r$ to $F_{\alpha_6} = \neg A$. $wait[\alpha_6] = \top$ should not longer hold as well.

Since *wait*-labels shall prevent only the deletion of *proof-relevant* sequent formulae, we have to remove outdated *wait*-labels in order to guarantee completeness. We will solve this problem by *redundancy deletion* after *β–splits*, the

identification of proof-relevant positions and elimination of redundant ones from the $\propto_i^\star$. This procedure strongly depends on the amount of *proof knowledge* made available by the proof search process, which leads to *reconstruction knowledge* about relevant and redundant positions.

3.1 Literal Purity

The minimal proof knowledge available after proof search is the set of connections $\mathcal{C}$ and the substitution σ which induces the relation $\sqsubset$. It leads to a generalized *purity reduction* (cf. [2]): an atomic position x of $\propto^\star$ is called *pure* if it is not connected. Complementarity of paths will not depend on x or any literal in the same "clause" and the whole tree containing these literals is redundant.

Definition 3. *A position k with $|succ(k)|\geq 2$ is a* β-node *if $Ptype(k)=\beta$ and otherwise a* Θ-node. *The greatest predecessor k of a position x in $\ll$ with $|succ(k)|\geq 2$ is called the* associated node *of x. We write $k \ll^\beta x$ if k is a β-node and $k \ll^\Theta x$ otherwise.*

If x is pure after a *β–split* and $k \ll^\beta x$ then the whole subtree with root k can be eliminated from $\propto^\star$ and the predecessor position of k inherits the purity property. If $k \ll^\Theta x$ then only the branch containing x can be deleted whereas k and all other branches have to remain in $\propto^\star$ (usually k is no longer a Θ-node afterwards). Combining these two reductions yields the function *(β,Θ)–purity* which will be applied to each subrelation $\propto_i^\star$ after *β–split* .

Definition 4. *Let $P_r = \{\, b \mid succ(b)=\emptyset \wedge \forall c \in \mathcal{C}.\, b \notin c \,\}$ be the set of pure leaf positions in $\propto^\star$. Let $succ_j^+(x) := \{succ_j(x)\} \cup succ^+(succ_j(x))$ where $succ^+(x)$ is the set of* all *successors of x in $\ll$ and $succ_j(x)$ is a selection function with $succ_j(x) = x_j$ if $succ(x)=\{x_1,..,x_n\}$. The (β,Θ)–purity reduction is defined as:*

function *(β,Θ)–purity*$(\propto^\star, \mathcal{C})$: *reduction–ordering*
  while $P_r \neq \emptyset$ **do**
    select $b \in P_r$; $P_r := P_r \setminus \{b\}$; let k be the associated node of b
    if $k \ll^\beta b$ **then** $\propto^\star := \propto^\star \setminus t^k$; % *$\beta$–purity*
      $\boxed{\mathcal{C} := \mathcal{C} \setminus \mathcal{C}^k;}$
      $P_r := \{b \mid succ(b)=\emptyset \wedge \forall c \in \mathcal{C}.\, b \notin c\}$
    else compute s **where** $b \in succ_s^+(k)$; $\propto^\star := \propto^\star \setminus t^{succ_s(k)}$ % *Θ–purity*

To ensure completeness *(β,Θ)–purity* must be integrated into *β–split* as follows.

Definition 5. *The* split-operation *at a β-node x is defined by $split(\propto^\star, x) = [\propto_{1'}^\star, \propto_{2'}^\star]$, where $\propto_{i'}^\star = (\beta,\Theta)$–purity$(\propto_i^\star, \mathcal{C}_i)$ and $[\propto_1^\star, \propto_2^\star] = \beta$–split$(\propto^\star, x)$.*

Consider again Example 2. After applying *β–split*$(\propto^\star, \beta_1)$ the position a_3 (a_4) becomes pure in $\propto_1^\star$ ($\propto_2^\star$). Since $\alpha_4 \ll_1^\Theta a_3$ ($\alpha_4 \ll_2^\Theta a_4$) applying *$\Theta$–purity* deletes t^{α_5} in $\propto_1^\star$ (t^{α_6} in $\propto_2^\star$). Thus all *wait*-labels can be removed and traversal can proceed with each of the resulting suborderings $\propto_{i'}^\star$ (see Figure 1).

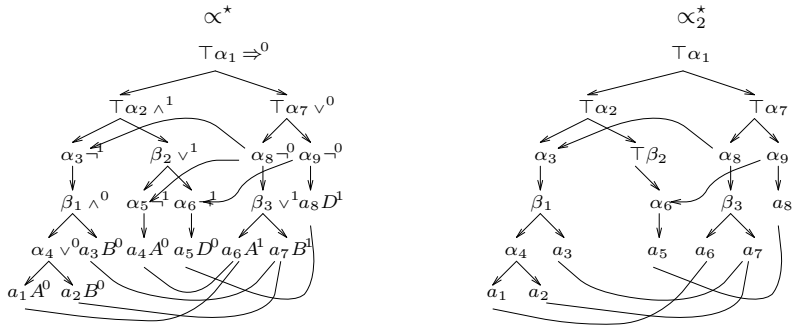


Fig. 2. Reduction orderings for $\neg((A \vee B) \wedge B) \wedge (\neg A \vee \neg D) \Rightarrow \neg(A \vee B) \vee \neg D$

3.2 The Decomposition Problem in $\propto^\star$

The (β, Θ)*–purity* reduction is suitable for dealing with *first level redundancy* where proof relevance is determined by being connected. Pure positions, however, are not the only redundancies that may occur during proof reconstruction. After β*–split* and (β, Θ)*–purity* one of the resulting suborderings may consist of several "isolated" subrelations which do not have connections between each other. In this case, only *one* of these subrelations is sufficient for making all paths complementary. Nevertheless the purity principle may not apply if all leaves in the othere subrelations are connected and hence assumed to be proof-relevant.

Example 3. Consider $F_2 \equiv \neg((A \vee B) \wedge B) \wedge (\neg A \vee \neg D) \Rightarrow \neg(A \vee B) \vee \neg D$ and the reduction ordering $\propto^\star$ resulting from its intuitionistic matrix proof in Figure 2 (left hand side). For proof reconstruction we solve the positions $\alpha_1, \alpha_2, \alpha_7$ and split at β_2 which corresponds in the following proof fragment:

$$\dfrac{\dfrac{\neg((A \vee B) \wedge B), \neg A \vdash \neg(A \vee B), \neg D \quad \neg((A \vee B) \wedge B), \neg D \vdash \neg(A \vee B), \neg D}{\neg((A \vee B) \wedge B), \neg A \vee \neg D \vdash \neg(A \vee B), \neg D}\ \vee l}{\vdash \neg((A \vee B) \wedge B) \wedge (\neg A \vee \neg D) \Rightarrow \neg(A \vee B) \vee \neg D}\ \Rightarrow r,\ \wedge l,\ \vee r$$

In the subrelation $\propto_2^\star$ resulting from splitting (Figure 2, right hand side), which corresponds to the right sequent after $\vee l$, the set of open positions is $P_o^2 = \{\alpha_3, \alpha_6, \alpha_8, \alpha_9\}$. The positions α_3 and α_6 are blocked by $\sqsubset$ whereas α_8 and α_9 are blocked by *wait*-labels since reducing $F^0_{\alpha_8} = \neg(A \vee B)^0$ would delete $F^0_{\alpha_9} = \neg D^0$ and vice versa. Furthermore, all atomic positions are connected and (β, Θ)*–purity* is not applicable. Thus proof reconstruction would run into a deadlock. But $\propto_2^\star$ has been *decomposed* into two "isolated" subrelations $\{t^{\alpha_3}, t^{\alpha_8}\}$ and $\{t^{\alpha_6}, t^{\alpha_9}\}$ and the *wait*-labels could be removed if we could determine which subrelation suffices for constructing the proof.

Deadlocks of above kind occur in intuitionistic logic and in all modal logics considered in [17] where additional *wait*-labels are required for proof reconstruction. In linear logic, *wait*-labels do not cause deadlocks and proof reconstruction has not to deal with this kind of redundancy [7]. Finding the appropriate isolated subrelation is the *decomposition problem in* $\propto^\star$ which we will formalize now.

Definition 6. *Let P_o be the set of open positions, P_a the set of atomic positions which are* solved *but connected, $P_u = P_o \cup P_a = \{x_1, .., x_n\}$ the set of* usable *positions, and $T_u = \{t^{x_1}, .., t^{x_n}\}$. The* connection relation $R_{\mathcal{C}} \subseteq T_u \times T_u$ *is defined by* $R_{\mathcal{C}} = \{\ (t^{x_i}, t^{x_j}) \mid 1 \leq i, j \leq n \wedge \exists\{c_i, c_j\} \in \mathcal{C}.\ c_i \in pos(x_i) \wedge c_j \in pos(x_j)\ \}$.
By $R^*_{\mathcal{C}}$ we denote the transitive closure of $R_{\mathcal{C}}$.

Let P_r be the set of pure positions It is easy to see that $R^*_{\mathcal{C}}$ defines an equivalence relation on T_u if $P_r = \emptyset$. In this case we will write $\sim_{\mathcal{C}}$ instead of $R^*_{\mathcal{C}}$. The equivalence class $[t^x] \in T_u/\sim_{\mathcal{C}}$ is defined by $[t^x] := \{t^y \mid t^y \sim_{\mathcal{C}} t^x\}$.

Definition 7. *Let $P_r = \emptyset$ and $T_u/\sim_{\mathcal{C}} = \{[t^{x_1}], \ldots, [t^{x_n}]\}$. The* decomposition problem *in $\propto^\star$ is the problem of selecting the proof-relevant $[t^{x_i}] \in T_u/\sim_{\mathcal{C}}$.*

Definition 8. *A reduction ordering $\propto^\star$ is called a* deadlock *iff $P_r = \emptyset$ and for all $x \in P_o$ either $(y, x) \in \sqsubset$ for some unsolved position y, or wait$[x] = \top$.*

In Example 3 we have a decomposition problem in $\propto^\star_2$ after *split*$(\propto^\star, \beta_2)$ since $T_u/\sim_{\mathcal{C}} = \{[t^{\alpha_8}], [t^{\alpha_9}]\} = \{\{t^{\alpha_3}, t^{\alpha_8}\}, \{t^{\alpha_6}, t^{\alpha_9}\}\}$. In addition, $\propto^\star_2$ is a deadlock. The same situation is caused by *α-reduction* when solving α_7 after β_2.

In general, proof reconstruction will require a solution for a decomposition problem if $\propto^\star$ is also a deadlock. This may only occur in non-classical logics and is characterized by the following lemma (see [18] for a proof).

Lemma 1. *Let $P_r = \emptyset$. If $\propto^\star$ is a deadlock then there exists $\{w_1, w_2\} \subseteq P_o$ such that wait$[w_1]$ = wait$[w_2] = \top$ and $[t^{w_1}] \neq [t^{w_2}]$.*

Thus deadlocks can only occur if there is also a decomposition problem $T/\sim_{\mathcal{C}} = \{[t^{w_1}], [t^{w_2}]\}$. Completeness of proof reconstruction can only be guaranteed if the decomposition problem can be either avoided or solved since otherwise proof-relevant formulae might be deleted. A complete solution of the decomposition problem consists of establishing a selection function $f_{\sim_{\mathcal{C}}}$ which chooses the only relevant class $[t^{x_i}]$ from $\propto^\star$. A solution is called *adequate* if $f_{\sim_{\mathcal{C}}}$ can be realized without any additional search. In Example 3 there is a deadlock in $\propto^\star_2$ and $[t^{\alpha_8}] \neq [t^{\alpha_9}]$ holds for the two *wait*-labeled positions α_8, α_9. $f_{\sim_{\mathcal{C}}}$ should select $[t^{\alpha_9}]$ since $[t^{\alpha_8}]$ does not lead to a proof.

Decomposition problems cannot be avoided during proof reconstruction. Nor can selection functions which are both complete and adequate be characterized for their solution. Completeness can be achieved only at the expense of adequateness, by searching all selections $[t^{x_i}]$ until proof reconstruction succeeds. We call this kind of redundancy *second level redundancy* since they cannot be solved adequately without the use of additional proof knowledge.

4 Integrating Proof Knowledge into Proof Reconstruction

Constructing adequate solutions for a decomposition problem requires additional knowledge about the proof search method and the proof knowledge it has gathered while developing the matrix-proof. In the following we characterize the knowledge that must be provided by the proof search method and assume this method to be based on the usual *extension procedure* [1, 2, 12]. We will encode the proof history of this procedure as *reconstruction knowledge* in the form of *prefixed connections* and derive a refinement of *β–split* and *(β, Θ)–purity* such that $T_u/\sim_{\mathcal{C}}$ will always consist of a single class. This makes it possible to avoid decomposition problems and deadlocks during the reconstruction process.

4.1 Prefixed Connections

The extension procedure checks the complementarity of sets of paths by following connections. It keeps track of the order in which connections have been followed and uses *active paths* to denote the sets of paths with the same initial subpath which have not yet been proven complementary. The history of constructing a matrix-proof can be expressed by *directed connections*, i.e. pairs $(a,b) \in \mathcal{C}$ of atomic positions (instead of sets $\{a,b\}$), together with their *active path context*.

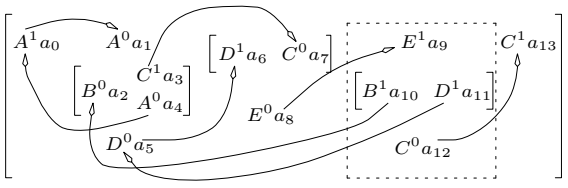
Definition 9. *An active path from a matrix proof $\propto^\star$ is a a sequence $P_i = (a_i^1, b_i^1) \circ \ldots \circ (a_i^{m_i}, b_i^{m_i})$ of pairs of atomic positions such that each a_i^j is β-related to b_i^{j-1} in $\propto^\star$, The set $\{P_1, \ldots, P_n\}$ of all active paths from a matrix proof $\propto^\star$ will be denoted by $\mathcal{P}$. $at(P_i) = a_i^1 \cdots a_i^{m_i}$ is called the* atom string *of path P_i.*

Each connection $c \in \mathcal{C}$ can be related to the set of active subpaths in which it occurs. These subpaths will be represented by a set of *prefixes* $Pre\,(c)$ such that the cardinality of $Pre\,(c)$ encodes the multiple use of c in the matrix proof.

Definition 10. *Let $c = (a,b) \in \mathcal{C}$ be a connection, $\{P_1, \ldots, P_m\} \subseteq \mathcal{P}$ be the set of active paths in which c occurs. The set of* prefixes *assigned to c is defined by $Pre\,(c) = \bigcup_{i=1}^{m} \{q_i a \mid q_i a \preceq at(P_i)\}$.*[1] *$Pre\,(c)$ is also called the* paths context *of c. For a set of connections $\mathcal{C}' \subseteq \mathcal{C}$ we define $Pref(\mathcal{C}') = \bigcup_{c \in \mathcal{C}'} Pre\,(c)$.*

The basic property of the extension procedure is that *all* paths through an established connection are known to be complementary. Since active paths are explored depth-first we know that two connections used in the matrix proof cannot have common prefixes, i.e. $Pre\,(c_1) \cap Pre\,(c_2) = \emptyset$ if $c_1 \neq c_2$. From now on we shall illustrate matrix proofs and prefixed connections using the two-dimensional matrix representation of a formula (see section 2.1), although the reduction ordering $\propto^\star$ remains to be the basic data structure for proof reconstruction.

Example 4. Consider the following (classical) non-normal form matrix which has been proven valid using the extension procedure with start clause $\{A^1\}$.



c	$Pre\,(c)$
(a_0, a_1)	$\{a_0\}$
(a_3, a_7)	$\{a_0a_3\}$
(a_8, a_9)	$\{a_0a_3a_8, a_0a_5a_8\}$
(a_{10}, a_2)	$\{a_0a_3a_8a_{10}\}$
(a_{12}, a_{13})	$\{a_0a_3a_8a_{12}, a_0a_5a_8a_{12}\}$
(a_4, a_0)	$\{a_0a_4\}$
(a_5, a_6)	$\{a_0a_5\}$
(a_{11}, a_5)	$\{a_0a_5a_8a_{11}\}$

To obtain a unique indexing we have written atomic positions x from $\propto^\star$ besides the atoms F_x. There are five active paths $\mathcal{P} = \{P_1, P_2, P_3, P_4, P_5\}$ with

$P_1 = (a_0, a_1) \circ (a_3, a_7) \circ (a_8, a_9) \circ (a_{10}, a_2)$ $\quad$ $P_4 = (a_0, a_1) \circ (a_5, a_6) \circ (a_8, a_9) \circ (a_{11}, a_5)$
$P_2 = (a_0, a_1) \circ (a_3, a_7) \circ (a_8, a_9) \circ (a_{12}, a_{13})$ $\quad$ $P_5 = (a_0, a_1) \circ (a_5, a_6) \circ (a_8, a_9) \circ (a_{12}, a_{13})$
$P_3 = (a_0, a_1) \circ (a_4, a_0)$

The atom strings $at(P_i) = p_i$ are $p_1 = a_0a_3a_8a_{10}$, $p_2 = a_0a_3a_8a_{12}$, $p_3 = a_0a_4$, $p_4 = a_0a_5a_8a_{11}$, and $p_5 = a_0a_5a_8a_{12}$. Each connection $c \in \mathcal{C}$ receives a set of prefixes $Pre\,(c)$ (see next to the matrix). The connections (a_{12}, a_{13}) and (a_8, a_9) have two prefixes due to their multiple occurrence in the path contexts P_2, P_5 and P_1, P_2, P_4, P_5 respectively.

[1] $q \preceq p$ denotes that q is an initial substring of p. ϵ denotes the empty string.

4.2 Splitting with Prefixed Connections

Prefixed connections encode multiple occurrences of active subpaths in a matrix proof. The active paths encode possible split structures of sequent proofs wrt. the axioms determined by the connections. Splitting with prefixed connections basically derives a classification of the active paths (i.e. prefixes) which have been interrupted during *β–split* . From this, the operation *β–split* can be refined to extend redundancy deletion along these "interrupted subpaths".

Definition 11. *Let y be a position in $\propto^\star$, $\ll^y$ its subtree, At^y the corresponding set of atomic positions, and $\mathcal{C}^y$ the set of directed connections in $\ll^y$. We divide $\mathcal{C}^y$ into the set of* entry connections $\mathcal{C}^y_{en} = \{(c,d) \in \mathcal{C}^y \mid d \in At^y\}$ *and the set of* extension connections $\mathcal{C}^y_{ex} = \{(c,d) \in \mathcal{C}^y \mid c \in At^y\}$*. Accordingly, $Pref(\mathcal{C}^y)$ is divided into the set of* entry– *and* extension prefixes*: $Pref(\mathcal{C}^y_{en})$ and $Pref(\mathcal{C}^y_{ex})$.*

$\mathcal{C}^y_{en} \cap \mathcal{C}^y_{ex}$ is nonempty iff $(c,d) \in \mathcal{C}^y$ is a connection and $\{c,d\} \subseteq At^y$. Splitting at a β-position causes, at a first level, the elimination of a submatrix At^y with connections C^y and, at a second level, *interruptions* of active subpaths. By separating entry- and extension connections after splitting we can identify prefixes of connections which depend on deleted connections and determine the interrupted active subpaths. Prefixes depending on extension connections cannot contribute to a subproof any longer and are *forward redundant*. Prefixes depending on entry connections are *backward redundant* towards a minimal subprefix.

In order to eliminate this kind of redundancy, we must respect these dependencies while deleting connections during *β–split* and *(β,Θ)–purity* . In a first step we will eliminate prefixes from $\mathcal{C}$ which are redundant wrt. the deleted connection set $\mathcal{C}^y$. For this we need a copy of the *original* matrix proof, i.e. the reduction ordering and connection set. Moreover, we define a concept of α-related subproofs in a matrix in order to capture non-normal form reduction steps.

Definition 12. *Let $\propto^\star_o, \mathcal{C}_o$ denote the original matrix proof. For a prefix qx_q let At^z_o be the* minimal *submatrix in $\propto^\star_o$ such that $q \in Pref(\mathcal{C}^z_o)$ denotes the entry connection and $qx_q \in Pref(\mathcal{C}^z_o)$ the extension connection in At^z_o. The set of* α-related subproofs *of qx_q is determined by the prefix set $T^{qx_q}_o = \{t \in Pref(\mathcal{C}^z_o) \mid qx_q \prec t\}$. The set of atomic positions which are involved by $T^{qx_q}_o$ is defined as:*

$$P^{qx_q}_o = \{a \in At^z_o \mid \exists c{=}(c_1,c_2) \in \mathcal{C}^z_o.\, \exists t \in T^{qx_q}_o.\, a{=}c_1 \vee a{=}c_2 \wedge t \in Pre\,(c)\}$$

During the conversion process, the set of α-related subproofs of qx_q is denoted by $T^{qx_q} \subseteq T^{qx_q}_o$. The corresponding atomic positions are given by $P^{qx_q} \subseteq P^{qx_q}_o$. Backward redundancy starts the deletion process at an entry prefix p and terminates at a minimal subprefix q of p which is determined either by a non-normal form reduction step in $\propto^\star$ or by the original matrix proof $\propto^\star_o$.

Definition 13. *Let $D \subseteq Pref(\mathcal{C}_o)$, $p \in D$, and $M_{qx_q} \subseteq T^{qx_q}_o$. q is called the* minimal subprefix *of p wrt. D, M_{qx_q} iff $q \in Pref(\mathcal{C}_o) \cup \{\epsilon\}$, $q \prec p$, and either*

1. *q is the* maximal *initial substring of p for which there are $t, qx_q \in D$ such that $qx_q \preceq p \preceq t$ and $t \in M_{qx_q}$, or*
2. *q is the* maximal *initial substring of p for which $q \notin D$.*

For abbreviation, we write $q \prec_{min} p$ wrt. D, M_{qx_q}.

For a position set P we abbreviate $\forall y \in P.\, x \sim_\alpha y$ by $x \sim_\alpha P$ (assume $x \sim_\alpha \emptyset, x \sim_\alpha x$).

Definition 14. *Let p be an entry prefix and q its minimal subprefix. The operations for deleting backward and forward redundancy are defined by:*

$$del_{en}(p,q) = \{s \mid qx'_q \preceq s \wedge p \not\preceq s \wedge x'_q \sim_\alpha P^{qx_q}\} \quad \text{and} \quad del_{ex}(p) = \{s \mid p \preceq s\}.$$

$del_{en}(p,q)$ covers all active subpaths s which have an entry connection with a prefix p as a *direct* or as an *indirect* subgoal. Suppose that y is the β-position for splitting. A connection with prefix p uses an atom from At^y as entry point and will become *open ended* or "interrupted" after deletion of At^y. Hence, no split ordering in a sequent proof can close the subpaths s between the α-related subproofs of qx_q and all s with $p \not\preceq s$, using the remaining connections from $\propto^\star$.

$del_{ex}(p)$ describes all active subpaths s which depend on an extension connection with prefix p. Each such connection uses an atom from At^y as extension point and will receive an *open beginning* after the deletion of At^y. Again, the subpaths s cannot be closed with the given connections.

In the following we extend these deletion operations to all entry / extension prefixes $Pref(\mathcal{C}^y_{en})$ / $Pref(\mathcal{C}^y_{ex})$ of a submatrix At^y which has to be deleted next. This extension is defined stepwisely since the result of one deletion step may influence the remaining candidate sets for the next step. The resulting prefix set depends on the order of prefix selection during this iteration and does not lead to *unique* subproofs after splitting.

Definition 15. *Let $\mathcal{C}$ be the connection set of $\propto^\star$ and y be a position in $\propto^\star$. Let $Pref(\mathcal{C}^y_{en})$ the set of entry prefixes and $Pref(\mathcal{C}^y_{ex})$ be the set of extension prefixes corresponding to $\mathcal{C}^y$. We set $D^0 = Pref(\mathcal{C})$, $P^0_{en} = Pref(\mathcal{C}^y_{en})$, $P^0_{ex} = Pref(\mathcal{C}^y_{ex})$, $M^0_r = T^r$ for all $r \in Pref(\mathcal{C})$, and $Q^0 = \emptyset$. Then we define the following iteration:*

$$D^i = D^{i-1} \setminus S^i, \qquad P^i_{en} = D^i \cap P^{i-1}_{en}, \qquad M^i_r = D^i \cap M^{i-1}_r \text{ for all } r \in D^i,$$
$$P^i_{ex} = D^i \cap P^{i-1}_{ex}, \qquad Q^i = (Q^{i-1} \cup U^i) \setminus S^i$$

until $i = n$ such that $P^n_{en} = P^n_{ex} = \emptyset$. During iteration we have:

$$S^i = \begin{cases} del_{en}(p,q) & \text{for a } p \in P^{i-1}_{en} \text{ with } q \prec_{min} p \text{ wrt. } D^{i-1}_{en}, M^{i-1}_{qx_q} \\ del_{ex}(p) & \text{for a } p \in P^{i-1}_{ex} \end{cases}$$

$$U^i = \{qx_q \mid qx_q \notin D^i \wedge y \not\sim_\beta x_q \wedge M^{i-1}_{qx_q} \neq \emptyset \wedge M^i_{qx_q} = \emptyset\}$$

The redundant prefixes are given by the sets S^i, depending if deletion of backward redundancy $p \in P^{i-1}_{en}$ or forward redundancy $p \in P^{i-1}_{ex}$ has been performed. The sets U^i denote *unblocked redundancies* caused during deletion of S^i. They consists of already deleted prefixes qx_q whose α-related subproofs M_{qx_q} have blocked deletion of backward redundancy in former steps. If step i results in a complete deletion of $M^i_{qx_q}$, the blocked backward deletion wrt. qx_q has to be continued. If $y \sim_\beta x_q$ (where y is the splitting node) continuation of backward deletion has to be performed by *β–split* in order to retain soundness. The sets U^i will be collected in a set Q^j and regularly updated modulo the actual deletion set S^j. This operation forces the sets S^j to deal with prefixes which have not necessarily been deleted from D^{j-1} in the *actual* step. The concept ensures a robust treatment of ordering dependencies in which prefixes will be selected from the candidate sets P^i_{en} and P^i_{ex}. In order to characterize a complete prefix deletion we integrate the elimination of unblocked redundancies as follows:

Definition 16. *We start with the sets D^n, Q^n, and M^n_r for all $r \in D^n$. Then*
$D^{n+j} = D^{n+(j-1)} \setminus S^j$ $\quad M^{n+j}_r = D^{n+j} \cap M^{n+(j-1)}_r$ for all $r \in D^{n+j}$,
$Q^{n+j} = (Q^{n+(j-1)} \cup U^j) \setminus S^j$

until $j = m$ such that $Q^{n+m} = \emptyset$. Again, we use during iteration:
$S^j = del_{en}(p, q)$ for a $p \in Q^{n+(j-1)}$ with $q \prec_{min} p$ wrt. $D^{n+(j-1)} \cup \{p\}, M^{n+(j-1)}_{qx_q}$
$U^j = \{qx_q \mid qx_q \notin D^{n+j} \wedge y \not\succ_\beta x_q \wedge M^{n+(j-1)}_{qx_q} \neq \emptyset \wedge M^{n+j}_{qx_q} = \emptyset\}$

pref_del$(\mathcal{C}, \mathcal{C}^y)$ denotes the complete operation prefix deletion *in $\mathcal{C}$ wrt. a connection set $\mathcal{C}^y$ and yields a set $\mathcal{C}'$ with $Pref(\mathcal{C}') = D^{n+m}$.*

Obviously, $Pref(\mathcal{C}^y) = \emptyset$ after applying *pref_del$(\mathcal{C}, \mathcal{C}^y)$* since $s = p$ satisfies the conditions of Definition 14, for all $p \in Pref(\mathcal{C}^y)$. A connection $c \in \mathcal{C}$ is called *redundant* if it does not occur within at least one active subpath, i.e. if $Pre(c) = \emptyset$.

Definition 17. *Let y be a position in $\propto^\star$ and $\mathcal{C}^y \subseteq \mathcal{C}$ be the connection set corresponding to At^y. Then* connection deletion *in $\mathcal{C}$ wrt. $\mathcal{C}^y$ is defined by*
con_del$(\mathcal{C}, \mathcal{C}^y)$ $:=$ $\mathcal{C}' \setminus \{c \mid Pre(c) = \emptyset\}$, where $\mathcal{C}' =$ pref_del$(\mathcal{C}, \mathcal{C}^y)$.

Redefining the split operation. An extended elimination of redundancies during proof reconstruction is realized by redefining the *split* operation at β-positions. Connection deletion during *β–split* and *(β, Θ)–purity* is determined by $\mathcal{C} \setminus \mathcal{C}^k$ wrt. the deleted submatrix At^k. The refined connection deletion will remove all interrupted subpaths from $\mathcal{C}$ such that $\mathcal{C}^k \subseteq \{c \in \mathcal{C}' \mid Prec(c) = \emptyset\}$ will hold. This incremental process guarantees redundancy deletion on the "low" connection level and on the "higher" level of paths contexts. As a consequence, redundancy will be deleted already when it *occurs* and not when it becomes visible in form of a decomposition problem in $\propto^\star$.

The refined connection deletion requires us to modify some definitions since $\mathcal{C}$ is now a set of directed connections. We redefine $\mathcal{C}^x$ in Definition 1 as $\{(c_1, c_2) \in \mathcal{C} \mid c_1 \in pos(x) \vee c_2 \in pos(x)\}$. In Definition 2 we replace $\mathcal{C}_i := \mathcal{C} \setminus \mathcal{C}^{x_j}$ by $\mathcal{C}_i :=$ *con_del$(\mathcal{C}, \mathcal{C}^{x_j})$*. Definition 4 is modified by extending $b \in c$ to directed connections $c = (c_1, c_2)$ when defining the set P_r and removing the boxed part in *(β, Θ)–purity* since the refinements of *β–split* avoid further prefix and connection deletions. Finally, the *split* operation from Definition 5 uses the new *β–split* and *(β, Θ)–purity* . The following example illustrates the refined operations.

Example 5. Consider the proof history from the extension proof of Example 4. During proof reconstruction we split at the clause $\{E^1, \{B^1, D^1\}, C^0\}$ (dashed boxed) by solving a β-position x in $\propto^\star$ with $succ(x) = \{x_1, a_{12}\}$. We have $At^{x_1} = \{a_9, a_{10}, a_{11}\}$, $\mathcal{C}^{x_1}_{en} = \{(a_8, a_9)\}$ with $Pref(\mathcal{C}^{x_1}_{en}) = \{a_0a_3a_8, a_0a_5a_8\}$, and $\mathcal{C}^{x_1}_{ex} = \{(a_{10}, a_2), (a_{11}, a_5)\}$ with $Pref(\mathcal{C}^{x_1}_{ex}) = \{a_0a_3a_8a_{10}, a_0a_5a_8a_{11}\}$. Similarly, $At^{a_{12}} = \{a_{12}\}$, $\mathcal{C}^{a_{12}}_{ex} = \{(a_{12}, a_{13})\}$, and $Pref(\mathcal{C}^{a_{12}}_{ex}) = \{a_0a_3a_8a_{12}, a_0a_5a_8a_{12}\}$. Applying *$\beta$–split$(\propto^\star, x)$* results in

$\propto^\star_1 = \propto^\star \setminus t^{x_1}$ (i.e. deleting literals E^1, B^1, D^1) and $\mathcal{C}_1 =$ *con_del$(\mathcal{C}, \mathcal{C}^{x_1})$*,
$\propto^\star_2 = \propto^\star \setminus t^{a_{12}}$ (i.e. deleting literal C^0) and $\mathcal{C}_2 =$ *con_del$(\mathcal{C}, \mathcal{C}^{a_{12}})$*.

We illustrate the connection deletion wrt. $\propto^\star_1$. The operation *pref_del$(\mathcal{C}, \mathcal{C}^{x_1})$* starts with an initialization according to the first row of the table below. We begin with the entry prefix $a_0a_3a_8$ and its minimal subprefix $q{=}a_0$, $x_q{=}a_3$. The first deletion set is given by S^1 where the α-related subproof $M^0_{a_0a_3}$ prevents deletion of a_0a_5 (see

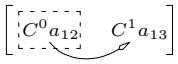
Definition 14). The second step selects $a_0a_5a_8$ with minimal subprefix a_0 and x_q=a_5. The whole iteration yields $D^4 = \{a_0, a_0a_3a_8a_{12}, a_0a_5a_8a_{12}\}$, $Q^4 = \{a_0a_3, a_0a_5\}$ and $M^4_r = \emptyset$ for all $r \in D^4$. Prefix deletion in $\propto^\star_1$ has to be completed by eliminating the unblocked redundancies Q^4 according to Definition 16. (see the right hand side of the table). We select a_0a_3 with its minimal subprefix $q = \varepsilon$ and $x_q = a_0$. Recall, that S^5 also contains all prefixes from Q^4 which were not contained in D^4. Thus, after one step, we obtain $Q^5 = \emptyset$ and terminate with $\mathit{pref\_del}(\mathcal{C}, \mathcal{C}^{x_1}) = \mathcal{C}'$ where $\mathit{Pref}(\mathcal{C}') = D^5 = \{a_0a_3a_8a_{12}\}$.

i	select	D^i	P^i_{en}	P^i_{ex}	S^i
0	–	$\mathit{Pref}(\mathcal{C})$	$\{a_0a_3a_8, a_0a_5a_8\}$	$\{a_0a_3a_8a_{10}, a_0a_5a_8a_{11}\}$	$\emptyset$
1	$a_0a_3a_8$	$D^0 \setminus S^1$	$\{a_0a_5a_8\}$	$\{a_0a_3a_8a_{10}, a_0a_5a_8a_{11}\}$	$\{a_0a_3a_8, a_0a_3, a_0a_4\}$
2	$a_0a_5a_8$	$D^1 \setminus S^2$	$\emptyset$	$\{a_0a_3a_8a_{10}, a_0a_5a_8a_{11}\}$	$\{a_0a_5a_8, a_0a_5\}$
3	$a_0a_3a_8a_{10}$	$D^2 \setminus S^3$	$\emptyset$	$\{a_0a_5a_8a_{11}\}$	$\{a_0a_3a_8a_{10}\}$
4	$a_0a_5a_8a_{11}$	$D^3 \setminus S^4$	$\emptyset$	$\emptyset$	$\{a_0a_5a_8a_{11}\}$

i	$M^i_{a_0a_3}$	$M^i_{a_0a_5}$	U^i	Q^i
0	$\{a_0a_3a_8a_{10}\}$	$\{a_0a_5a_8a_{11}\}$	$\emptyset$	$\emptyset$
1	$\{a_0a_3a_8a_{10}\}$	$\{a_0a_5a_8a_{11}\}$	$\emptyset$	$\emptyset$
2	$\{a_0a_3a_8a_{10}\}$	$\{a_0a_5a_8a_{11}\}$	$\emptyset$	$\emptyset$
3	$\emptyset$	$\{a_0a_5a_8a_{11}\}$	$\{a_0a_3\}$	U^3
4	$\emptyset$	$\emptyset$	$\{a_0a_5\}$	$Q^3 \cup U^4$

i	select	D^i	Q^i	U^i	S^i
4	–	D^4	Q^4	U^4	$\emptyset$
5	a_0a_3	$D^4 \setminus S^5$	$\emptyset$	$\emptyset$	$\{a_0, a_0a_5a_8a_{12}\} \cup Q_4$

Finally, *β–split* performs connection deletion wrt. $\propto^\star_1$: $\mathit{con\_del}(\mathcal{C}, \mathcal{C}^{x_1}) = \{(a_{12}, a_{13})\}$ with $\mathit{Pre}((a_{12}, a_{13})) = \{a_0a_3a_8a_{12}\}$. The refined *split*-operation terminates on $\propto^\star_1$ with applications of the refined (β, Θ)*–purity* . The set P_r is initialized with all atomic positions of $\propto^\star_1$, except a_{12} and a_{13}. Additional connection deletions are not required and all subrelations depending on iterative updating of P_r will be deleted from $\propto^\star_1$ (see Definition 4). The resulting relevant submatrix corresponding to $\propto^\star_{1'}$ is shown below:



The example can easily be extended such that the "conventional" application of *β–split* would cause a decomposition problem in $\propto^\star_{1'}$.

5 Complexity, Correctness, and Completeness

We will now show that the refinements of the *β–split* operation which we introduced in sections 3 and 4 do in fact lead to an adequate and complete proof reconstruction procedure. Since a traversal of $\propto^\star$ can be completed in polynomial time if no deadlocks are present, it suffices to prove that the refined *split* operation has a polynomial complexity in the size of the matrix proof (*adequateness*) and that splitting with prefixed connections deletes all redundancies from a matrix proof (*completeness*). The proofs of the following lemmata and theorems can be found in the first author's technical report [15].

Complexity. The *size of a matrix proof* is usually defined as the number of inference steps for testing a mating $\mathcal{C}$ to be spanning. This measure is equivalent to the number of active paths $|\mathcal{P}|$ in a matrix proof which may increase exponentially in the size of $\propto^\star$, i.e. the number of positions.

Lemma 2. *Let $|\mathcal{P}|$ be the size of a matrix proof, n the maximal length of paths $P_i \in \mathcal{P}$, and k the maximal length of (non-normal form) clauses in $\propto^\star$. Then $|\mathcal{P}| \in \mathcal{O}(k^n)$ and n is polynomial wrt. the size of $\propto^\star$.*

The refinements of the operation *β–split* are based on the deletion of prefixes wrt. a prefix set $Pref(\mathcal{C}^y)$. Testing the basic elimination conditions (Definition 14) as well as computing the relevant sets during the iteration process (Definition 15 and 16) can be done in polynomial time in the size of $|Pref(\mathcal{C})|$. But this this complexity is polynomial wrt. the size of the matrix proof $|\mathcal{P}|$.

Lemma 3. *Let $\mathcal{C}$ be the spanning mating from a matrix proof, $Pref(\mathcal{C})$ its prefix set, and k, n as in Lemma 2. Then $|Pref(\mathcal{C})| \in \mathcal{O}(k^n)$.*

The number of purity applications within (β, Θ)*–purity* is determined by the possible updates of the set P_r, which is polynomial wrt. the size of $\propto^\star$. Lemma 3 also states that integration of proof knowledge $Pref(\mathcal{C})$ into the proof reconstruction process requires polynomial (space) complexity wrt. $|\mathcal{P}|$. From this we conclude the following adequateness theorem:

Theorem 2. *The refined split operation $split(\propto^\star, x)$ at a β-node x using prefixed connections is polynomial wrt. the size of the matrix proof $|\mathcal{P}|$.*

In special cases the size $|\mathcal{P}|$ of a matrix proof and the size $|\mathcal{C}|$ of its spanning mating may differ exponentially. If k, n are defined as above and all active paths $P_i \in \mathcal{P}$ share the same connections from $\mathcal{C}$, we find examples such that $|\mathcal{C}| < n \cdot k$ and $|\mathcal{P}| \in \mathcal{O}(k^n)$. Hence, the additional search complexity on a decomposition problem when using spanning matings $\mathcal{C}$ may be transformed into an exponential representation requirement for $|\mathcal{P}|$ when integrating prefixed connections. However, the complexity of a matrix proof is reflected more adequately when taking its size $|\mathcal{P}|$ into account. Furthermore, conversion with prefixed connections avoids redundant steps in the resulting sequent proof which cannot be guaranteed when using matings together with additional search.

Correctness and Adequate Completeness. The correctness of the conversion procedure using prefixed connections is obvious. The split operation with refined connection deletion cuts subrelations from $\propto^\star$ without violating the relation $\sqsubset$ in the *remaining* part of $\propto^\star$. Incorrect applications of sequent rules wrt. rule non-permutabilities are impossible. Thus, reconstructing a sequent proof from $\propto^\star$ implies that the input formula is valid wrt. the selected logic.

In order to prove completeness we show that no redundant connections will survive in the $\propto_i^\star$ after applications of the refined split operation.

Lemma 4. *Let $\propto^\star$ represent an extension proof with connection set $\mathcal{C}$. Let x be a β-node and $\mathcal{C}'_i \subseteq \mathcal{C}$ be the connection sets of $\propto_i^\star$, after executing the refined split operation at x. Then a connection $c \in \mathcal{C}$ is proof-relevant wrt. $\propto_i^\star$ iff $c \in \mathcal{C}'_i$.*

The occurrence of a decomposition problem $T_u/\sim_{\mathcal{C}} = \{[t^{a_1}], .., [t^{a_n}]\}$ is always based on redundant connections since only a single $[t^{x_i}]$ is proof-relevant. From Lemma 4 it also follows that $T_u/\sim_{\mathcal{C}} = \{[t^a]\}$ for $a \in P_u$. According to Lemma 1 this implies that $\propto^\star$ will be deadlock free during the whole proof reconstruction process which leads to the concluding theorem.

Theorem 3. *Let x be a β-node. The extended split operation $split(\propto^\star, x)$ is correct and adequately complete for redundancy deletion in $\propto^\star$.*

6 Conclusion and Future Work

We have presented a method for eliminating redundancies during a conversion of matrix proofs into sequent proofs. Our approach refines the proof reconstruction procedure presented in [17, 7] and covers classical and intuitionistic logic, the modal logics $K, K4, D, D4, T, S4, S5$, and fragments of linear logic. For obtaining *adequate* (search-free) *completeness* of proof reconstruction, we have classified two levels of redundancy. We have shown that adequate solutions require additional knowledge from proof search in the matrix calculus. Assuming the usual extension proof search strategy we have introduced *prefixed connections* as a means for representing a proof history. We have integrated this concept into the proof reconstruction procedure and shown that the refined procedure will always generate a sequent proof in polynomial time wrt. the size of the matrix proof.

In the future we will make use of the uniformity of our approach and combine it with existing proof procedures for non-classical logics in order to guide derivations in interactive proof development systems. We will also be able to extend our approach to additional logics, such as larger fragments of linear logic, as soon as matrix characterizations and proof procedures have been developed for them. Apart from this we will generalize the concept of prefixed connections to other proof strategies. For example proof histories from tableau based proof procedures [13] may be expressed in terms of extension proofs in order to combine tableau provers with our proof reconstruction procedure as well.

References

1. W. Bibel. On matrices with connections. *Journal of the ACM*, 28, p. 633–645, 1981.
2. W. Bibel. *Automated theorem proving.* Vieweg, 1987.
3. W. Bibel, S. Brüning, U. Egly, T. Rath. Komet. *CADE–12*, LNAI 814, pp. 783–787. 1994.
4. W. Bibel, D. Korn, C. Kreitz, F. Kurucz, J. Otten, S. Schmitt, G. Stolpmann. A Multi-Level Approach to Program Synthesis. 7^{th} *LOPSTR Workshop*, LNCS , 1998.
5. W. Bibel, D. Korn, C. Kreitz, S. Schmitt. Problem-Oriented Applications of Automated Theorem Proving. *DISCO-96*, LNCS 1128, pp. 1–21, 1996.
6. M. C. Fitting. *Proof Methods for Modal and Intuitionistic Logic.* D. Reidel, 1983.
7. C. Kreitz, H. Mantel, J. Otten, S. Schmitt. Connection-based proof construction in Linear Logic. *CADE–14*, 1997.
8. C. Kreitz, J. Otten, and S. Schmitt. Guiding Program Development Systems by a Connection Based Proof Strategy. 5^{th} *LOPSTR Workshop*, LNCS 1048, pp. 137–151. 1996.
9. R. Letz, J. Schumann, S. Bayerl, W. Bibel. Setheo: A high-performance theorem prover. *Journal of Automated Reasoning*, 8:183–212, 1992.
10. H. J. Ohlbach. A resolution calculus for modal logics. PhD Thesis, Univ. Kaiserslautern, 1988.
11. J. Otten, C. Kreitz. T-string-unification: unifying prefixes in non-classical proof methods. 5^{th} *TABLEAUX Workshop*, LNAI 1071, pp. 244–260, 1996.
12. J. Otten, C. Kreitz. A uniform proof procedure for classical and non-classical logics. *KI-96: Advances in Artificial Intelligence*, LNAI 1137, pp. 307–319, 1996.
13. J. Otten. ileanTAP: An intuitionistic theorem prover. 6^{th} *TABLEAUX Workshop*, 1997.
14. J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, 1965.
15. S. Schmitt. Avoiding Redundnacy for Proof Reconstruction in Classical and Non-Classical Logics. Technical Report, TU-Darmstadt, 1997.
16. S. Schmitt, C. Kreitz. On transforming intuitionistic matrix proofs into standard-sequent proofs. 4^{th} *TABLEAUX Workshop*, LNAI 918, pp. 106–121, 1995.
17. S. Schmitt, C. Kreitz. Converting non-classical matrix proofs into sequent-style systems. *CADE–13*, LNAI 1104, pp. 418–432, 1996.
18. S. Schmitt, C. Kreitz. A uniform procedure for converting non-classical matrix proofs into sequent-style systems. Technical Report AIDA-96-01, TU-Darmstadt, 1996.
19. T. Tammet A Resolution Theorem Prover for Intuitionistic Logic *CADE–13*, LNAI 1104, pp. 2–16, 1996.
20. L. Wallen. Matrix proof methods for modal logics. *IJCAI-87*, p. 917–923. 1987.
21. L. Wallen. *Automated deduction in non-classical logics.* MIT Press, 1990.
22. L. Wos et. al. Automated reasoning contributes to mathematics and logic. *CADE–10*, LNCS 449, p. 485–499. 1990.