Proof Forests with Cut Based on Herbrand's Theorem

Willem Heijltjes

Laboratory for the Foundations of Computer Science
University of Edinburgh
w.b.heijltjes@sms.ed.ac.uk

Abstract. A major obstacle towards an elegant categorical model of classical logic is the absence of a canonical proof system with a confluent notion of cut-elimination. In this note we describe a graphical representation of proofs that naturally arises from Herbrand's Theorem, game semantics and sequent calculus. The representation suggested subtle cut-reduction steps that carried the promise to fill this gap, until an example was found for which termination failed.

1 Introduction

In this note we will work with a representation of classical proofs in the form of forests of formula-trees, strengthened with a partial order. Although not the primary inspiration, cut-free forests are essentially Miller's Expansion Tree Proofs (see [7]), restricted to the (prenex) first-order fragment. These proofs are structurally very economical; propositional content is ignored, and meaning resides in the witnessing terms for existentially quantified variables. The structural minimality reflects back in the relative ease of relating forests to proofs in other systems, as Miller demonstrated; here, we will concentrate on the interpretation as one-sided sequent proofs, vertically divided into trees of inferences.

These proof forests can also be thought of as pictorial representations of Herbrand Expansion Proofs, recently introduced by McKinley [5], based on Buss' Herbrand Proofs (see [1]). Both McKinley and the author have been inspired by Herbrand's Theorem, in the search for a decent cut-elimination procedure as the basis for a non-trivial categorical semantics for classical logic. Another inspiration for the present work has been Coquand's game-theoretical approach to classical arithmetic (see [2]).

The partial order on the forests provides a natural basis for defining cut-reduction steps, exploiting the forests' structure sharing abilities to reduce duplications to a minimum. The sparse structure of the forests gave rise to the hope that the problems facing cut-elimination in the sequent calculus could be avoided; instead, it turned out that the reduction steps fail to terminate. The primary result presented here is an example exhibiting this; we believe the problem is new, and that it may be significant to classical logic as such.

After briefly exposing the influence of Herbrand's Theorem and game semantics, we will show how proof forests relate to sequent proofs and explain the

cut-reduction steps from this perspective, leaving the non-terminating example to the end. The work presented here is still very much in progress, and findings regarding weak normalization, as well as the translation from proof forests with cuts to sequent calculus, are unfortunately not ready for publication yet. Because of this, we will put less emphasis on theorems and definitions, and instead focus on those points required to present the counterexample.

We see this work as fitting in well with other graph-based approaches to classical proof, an area that has seen a spur of activity over the past few years. Recent publications include [4] on geometric proofs, [6] on proof nets, and [3] on graphs for propositional classical logic, using deep inference. However, in contrast to this work we ignore the propositional connectives, and instead focus entirely on quantifiers.

2 Herbrand's Theorem

In his treatment of Herbrand's theorem, [1], Buss presents a proof system based on the axiomatization originally used by Herbrand. Proofs in the system, called Herbrand proofs, consist of the following steps:

- 'Expanding' the formula: repeating the operation of replacing a subformula $\exists.A$ by $\exists.A \vee \ldots \vee \exists A$ an arbitrary number of times.
- Casting it into prenex-normal form, by moving quantifiers from inside the formula to the front (and renaming variables when necessary).
- Giving a substitution map for the existentially quantified variables in the prenex formula, replacing each with a term. The term replaced for a variable x in a formula $Q_1 x_1 \ldots Q_n x_n . \exists x.A$ (where each Q_i is $\forall$ or $\exists$) may use no other bound variables than those of $x_1 \ldots x_n$ that are universally quantified.

A combination of the above three steps would constitute a proof if the quantifier-free part of the resulting formula, which is now only universally quantified, is a tautology. The expansion of the formula essentially constitutes an arbitrary number of choices of instantiating each existentially quantified formula. This suggest a tree-notation in which universal quantifiers have unique successors, and existential quantifiers arbitrarily many. The prenexification gives a tangible restriction, in the form of a linear order, on the variables that may be used in the witnessing terms of the existentially quantified variables.

Similar trees are presented by Coquand in [2], where a game is played by $\forall$belard and $\exists$loise who take turns choosing a branch representing a witness for an arithmetical statement. The game does not force $\exists$loise to respond to $\forall$belard's latest move, but allows her to revert to a previous position and choose another branch.

3 Proof Forests

We will informally introduce proof forests as depicting strategies for a game, in which $\forall$belard and $\exists$loise take turns assigning values, from a certain domain, to

the quantified variables in a sequent of prenex formulae. To represent ∀belard's strategy his choices are generalised to fresh variables on each move; a strategy for ∃loise consists of assigning terms that may range over the variables of ∀belard's previous choices. Here, too, ∃loise is allowed to return to a previous position; when a strategy suggests such a move this is represented by an additional branch at that position. The occurrence of ∀belard's variables in ∃loise's terms restricts the order of the moves in any game played according to that strategy; this imposes a partial order called the 'dependency relation'.

The choices suggested by a strategy are recorded in 'assignment trees', built up from the pieces shown in Figure 1. The first and second items are nodes with their outgoing edges displaying both players' moves, and the third is a leaf, to be decorated with the formula resulting from applying their choices.

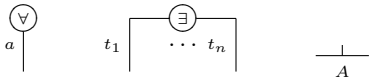


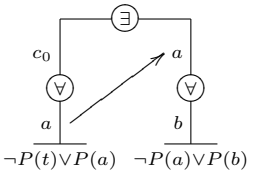
Fig. 1. The elements of assignment trees

Definition 1 (Assignment Trees). *An assignment tree for a prenex formula* $Q_1x_1 \dots Q_nx_nA$*, with each* Q_i *one of* $\exists, \forall$ *and* A *quantifier-free, is an unordered tree built up as follows:*

- *Non-leaf nodes in the tree are either* Ⓥ *or* Ⓔ *and on each path from the root to a leaf match the quantifiers* $Q_1 \dots Q_n$*.*
- *There is one outgoing edge on* Ⓥ*-nodes, labelled with a variable;* Ⓔ*-nodes have arbitrarily many outgoing edges, labelled with terms.*
- *If* $t_1 \dots t_n$ *are the labels on a path from the root to a leaf, then the leaf is labelled with the formula* $A[t_1/x_1, \dots, t_n/x_n]$*.*

A proof forest combines assignment trees with the dependency relation, that shows to which of ∀belard's moves ∃loise is responding. A move assigning a term t is a response, at least, to the moves that introduce the free variables in t; in diagrams we use arrows to display this, where the direction reflects that ∀belard's move must occur before ∃loise's.

$$\exists x \forall y.(\neg P(x) \vee P(y))$$



(1)

The forest proof in (1) shows a strategy for the drinker's formula. A game following this strategy is opened by ∃loise with a constant c_0 from the domain (the

formula is only valid in a non-empty domain). If ∀belard's answer makes $P(a)$ true, then ∃loise wins. Otherwise, she reconsiders her first choice and plays the same value ∀belard did; she wins regardless of his response, since $\neg P(a)$ holds.

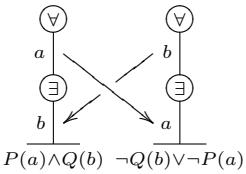
Definition 2 integrates assignment trees, a dependency relation, and a notion of cut that links two trees for dual formulae, for future use. The exact requirements on the dependency are left to Definition 3.

Definition 2 (Proof Forests). *A proof forest $\mathcal{F}$ for a sequent $A_1, \ldots, A_n$ is a multiset of unordered trees, one for each A_i, a multiset of cuts, and a relation $\leq_{\mathcal{F}}$ such that:*

- *a cut is an edge between the root of a tree for A and one for its negation $\overline{A}$;*
- *the edges in a forest are those of its trees plus the cuts;*
- *if i, j are $\textcircled{\forall}$-edges labelled a, b respectively, then $a \neq b$ whenever $i \neq j$;*
- *the dependency $\leq_{\mathcal{F}}$ obeys Def. 3 and is acyclic.*

While the dependency restricts the order of the moves in a game, it does not necessarily define it. An example of this is (2), where ∃loise's moves follow both of ∀belard's, but the order of her two moves is left unspecified, as is the order of his moves.

$$\forall x \exists y. P(x) \wedge Q(y), \forall x \exists y. \neg Q(x) \vee \neg P(y) \tag{2}$$



Definition 3 (Dependency). *A dependency relation $\leq_{\mathcal{F}}$ on the edges in a forest $\mathcal{F}$ is as follows, where i and j are regular edges, and k is a cut between a tree* B *for A and a tree* B′ *for $\overline{A}$:*

- *$i \leq^{\mathrm{D}} j$ iff j is in the subtree of i or $i = j$;*
- *$k \leq^{\mathrm{D}} j$ iff j is in* B *or* B′*, and $k \leq^{\mathrm{D}} k$;*
- *$i \rightarrow^{-} j$ iff i is a $\textcircled{\forall}$-edge labelled a, j is a $\textcircled{\exists}$-edge labelled t, and $a \in \mathrm{FV}(t)$;*
- *$i \rightarrow^{-} k$ iff i is a $\textcircled{\forall}$-edge labelled a and $a \in \mathrm{FV}(A)$;*
- *$\leq_{\mathcal{F}}^{-}$ is the transitive closure of $(\leq^{\mathrm{D}}) \cup (\rightarrow^{-})$;*
- *$\leq_{\mathcal{F}}$ is the transitive closure of $(\leq^{\mathrm{D}}) \cup (\rightarrow^{-}) \cup (\rightarrow^{+})$ for some relation $(\rightarrow^{+})$ such that:*
- *$i \rightarrow^{+} j, k$ only if i is a $\textcircled{\forall}$-edge, and j a $\textcircled{\exists}$-edge.*

A forest constitutes a proof if, regardless of ∀belard's choices, ∃loise can always force one of the formulae decorating the leaves of the forest to be true. This is precisely the case when the disjunction of the leaves forms a propositional tautology.

Definition 4 (Validity for cut-free forests). *A cut-free assignment forest for a sequent Γ is valid if the disjunction of all formulae decorating the leaves of the trees in the forest, is a tautology.*

Omitted from the definitions above are the validity conditions for forests with cuts. The reason behind the omission is that it would take significant effort to define, and show to be correct and complete, while at the same time being inessential to the arguments we would like to put forward. Nonetheless, we would like to give the two minimal requirements for such a definition:

- combining two valid forests with a cut should give a valid forest;
- validity is preserved under cut-reduction steps.

Although it is clear from the above that the design of the validity conditions might influence that of the cut-reduction algorithm, we are presently concerned with the reduction steps that arise naturally from the structure of these forests.

Finally, composing two forests with a cut is straightforward: a forest for a sequent Γ, A and one for $\Gamma', \overline{A}$ may be composed by combining all the trees into one forest, and adding a cut between the trees for A and $\overline{A}$. Note that without a validity condition for forests with cuts, and without a terminating reduction algorithm, this may not be called a forest proof of Γ, Γ'.

4 Sequent Calculus

There is a simple intuition linking proof forests to sequent proofs: edges correspond to inferences, variable and term labels correspond to eigenvariables and instantiated terms in ∀R- and ∃R-inferences, branching corresponds to contraction, and the propositional part of a proof is discarded. The dependency corresponds more or less to the order of inferences in a proof; this will be made more precise later.

Formalising this requires the sequent proof to have the right shape. Since we are working with prenex formulae, we will use a one-sided system. The propositional rules are omitted, replacing the axiom-rule with a tautology-rule, resulting in the set of rules shown in Figure 2. Since ∀belard always plays a fresh variable, we impose the common constraint that all eigenvariables should be distinct, referred to as the 'eigenvariable condition'.

$$\frac{}{\vdash A_1, \ldots, A_n}\,\mathrm{Taut}^{*} \qquad \frac{\vdash \Gamma}{\vdash \Gamma, A}\,\mathrm{WR} \qquad \frac{\vdash \Gamma, A[t/x]}{\vdash \Gamma, \exists x.A}\,\exists\mathrm{R}$$

$$\frac{\vdash \Gamma, A \quad \vdash \overline{A}, \Gamma'}{\vdash \Gamma, \Gamma'}\,\mathrm{Cut} \qquad \frac{\vdash \Gamma, A, A}{\vdash \Gamma, A}\,\mathrm{CR} \qquad \frac{\vdash \Gamma, A[a/x]}{\vdash \Gamma, \forall x.A}\,\forall\mathrm{R}^{**}$$

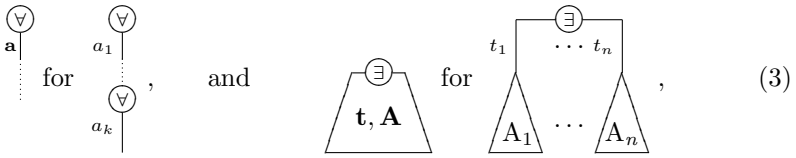
* $\bigvee_{i=1}^{n} A_i$ is a propositional tautology $\qquad$ ** $a \notin \mathrm{FV}(\Gamma)$

Fig. 2. The fragment of sequent calculus used

The one issue complicating the translation is the occurrence of contractions, and weakenings, on universally quantified formulae in a sequent proof, while

branching only occurs on existential nodes in a proof forest. Rather than adding nodes and edges to a forest like the translation of logical rules, translating a contraction on a universal formula consists of performing a kind of 'surgery' on a forest, unifying the topmost ∀-edges of the trees involved—the existential case is covered by the vector $\forall \mathbf{x}$, used to abbreviate $\forall x_1 \dots \forall x_k$, being of zero length.

The fleshed out translation procedure, $[\cdot]$, is shown in Figure 3. The translation step for the cut-rule is omitted, since it is exactly composition via cut as described earlier. The diagrams use the following abbreviations:



(3)

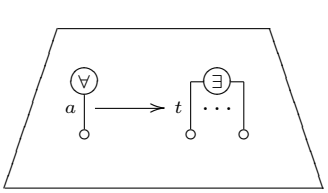
where $\mathbf{a}$, $\mathbf{t}$ and $\mathbf{A}$ are vectors $a_1 \dots a_k$, $t_1 \dots t_n$, and $\mathrm{A}_1 \dots \mathrm{A}_n$ respectively. Trees are drawn as triangles, and in other diagrams also as tiny circles, and forests as trapezoids, with a forest context encompassing a highlighted tree, rather than positioned aside it.

Dependency

The dependency for the forest translation $[\Delta]$ of a sequent proof Δ is the minimal dependency. In the sequent proof, it corresponds to a relation on inferences perhaps best described as one being 'necessarily in the subproof' of the other; 'necessarily', in the sense that the two inferences cannot be permuted.

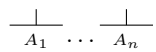
A tree in the forest $[\Delta]$ corresponds to the inferences that address what is, essentially, the same formula throughout the sequent proof. Clearly, permuting inferences that operate on the same formula is impossible, precisely in the way that edges in the corresponding tree are related in the dependency.

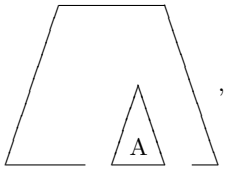
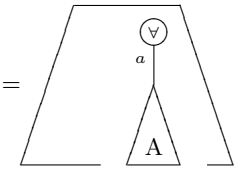
The other thing preventing permutations is the eigenvariable restriction on ∀R-inferences in sequent proofs (the note ** in Figure 2). This corresponds to the 'arrows' in the dependency, illustrated by (4), where two edges in a forest are laid out against two corresponding inferences in a sequent proof.

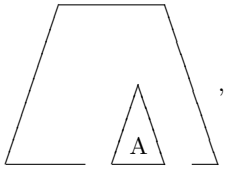
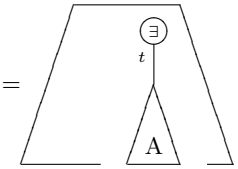


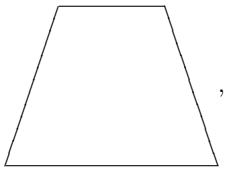
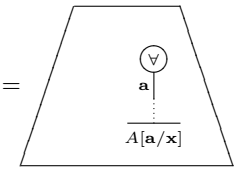
$$\dfrac{\dfrac{\vdots}{\vdash \Gamma', B[t/y]}}{\vdash \Gamma', \exists y.B}\exists\mathrm{R} \qquad \dfrac{\dfrac{\vdots}{\vdash \Gamma, A[a/x]}}{\vdash \Gamma, \forall x.A}\forall\mathrm{R} \qquad \text{where } a \in \mathrm{FV}(t) \tag{4}$$

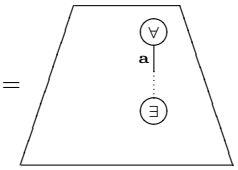
An edge assigning a term t corresponds to an ∃R-inference introducing t. Any variable a free in t will be free in the premise of that inference; permuting it below the ∀R-inference introducing a enters it into the context of that rule's conclusion, violating the eigenvariable restriction.

$\left[\dfrac{}{\vdash A_1, \ldots, A_n}\,\mathrm{Taut}\right] =$  ;

if $\left[\vdash \Gamma, \overset{\Delta}{A[a/x]}\right] =$  , then $\left[\dfrac{\vdash \Gamma, \overset{\Delta}{A[a/x]}}{\vdash \Gamma, \forall x.A}\,\forall\mathrm{R}\right] =$  ;

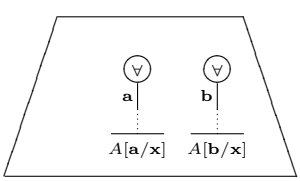
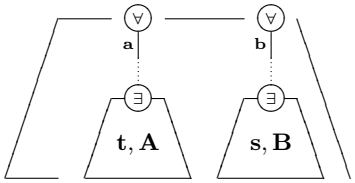
if $\left[\vdash \Gamma, \overset{\Delta}{A[t/x]}\right] =$  , then $\left[\dfrac{\vdash \Gamma, \overset{\Delta}{A[t/x]}}{\vdash \Gamma, \exists x.A}\,\exists\mathrm{R}\right] =$  ;

if $\left[\overset{\Delta}{\vdash \Gamma}\right] =$  , then $\left[\dfrac{\overset{\Delta}{\vdash \Gamma}}{\vdash \Gamma, \forall \mathbf{x}.A}\,\mathrm{WR}\right] =$ 

if A is quantifier-free, and $\left[\dfrac{\overset{\Delta}{\vdash \Gamma}}{\vdash \Gamma, \forall \mathbf{x} \exists y.B}\,\mathrm{WR}\right] =$  ,

where $|\mathbf{a}| = |\mathbf{x}| = k$ and each of $a_1 \ldots a_k$ is fresh;

if $\left[\vdash \Gamma, \overset{\Delta}{\forall \mathbf{x}.A}, \forall \mathbf{x}.A\right]$ is one of (depending on whether A is quantifier-free)

 ,
 ,

then $\left[\dfrac{\vdash \Gamma, \overset{\Delta}{\forall \mathbf{x}.A}, \forall \mathbf{x}.A}{\vdash \Gamma, \forall \mathbf{x}.A}\,\mathrm{CR}\right]$ is respectively one of

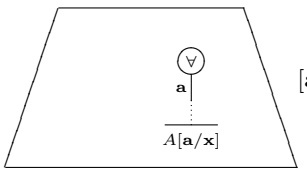
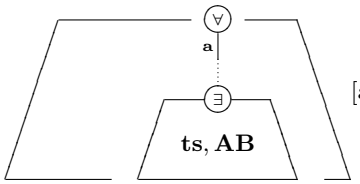
 $[\mathbf{a}/\mathbf{b}]$,
 $[\mathbf{a}/\mathbf{b}]$.

Fig. 3. The translation $[\cdot]$ from sequent to forest proofs.

The relation between inferences of being 'necessarily' in the other's subproof is clearly transitive and acyclic. Hence, it precisely matches the dependency relation, and ensures that the latter is acyclic in a translated forest. Furthermore, the only rules that can introduce an instance of some other inference's eigenvariable are ∃R and Cut; consequently, the arrows in the dependency relation only point from a Ⓐ-edge to an Ⓔ-edge or a cut—as required in Definition 4.

A final observation before turning to the cut-reduction steps, is that translating back from a proof forest to a sequent proof is trivial after topologically sorting the edges in the forest. From the translations in both directions it follows immediately that proof forests are sound and complete for (prenex) first-order classical logic.

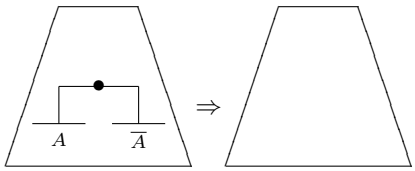
5 Cut-Reduction

The cut-reduction steps we present here are rather intricate, but we will try to show that they arise naturally from the structure of the forests and the analogy with sequent calculus. The lack of branching on Ⓐ-nodes eliminates the analogy of a cut on two contractions or two weakenings in the sequent calculus-setting, notorious sources for non-confluence and even non-termination. This leaves four cases to be considered:

I. a cut on propositional formulae A and $\overline{A}$,
II. a cut on a Ⓐ-node and a dead-end Ⓔ-node, corresponding to a ∀R- and a weakening inference in sequent calculus,
III. a cut on a Ⓐ-node and a single-branching Ⓔ-node, corresponding to a ∀R- and a ∃R-inference in sequent calculus,
IV. a cut on a Ⓐ-node and a Ⓔ-node with more than one branch, corresponding to a ∀R-inference and a contraction.

The four reduction steps will be presented as rewrite diagrams. Except for the first case, which is straightforward, the diagrams will be explained from the perspective of their counterparts in sequent calculus.

I. In the first case, both leaves will simply be removed, illustrated in (5). When presenting a reduction step we will indicate by a small black token which cut will be reduced.

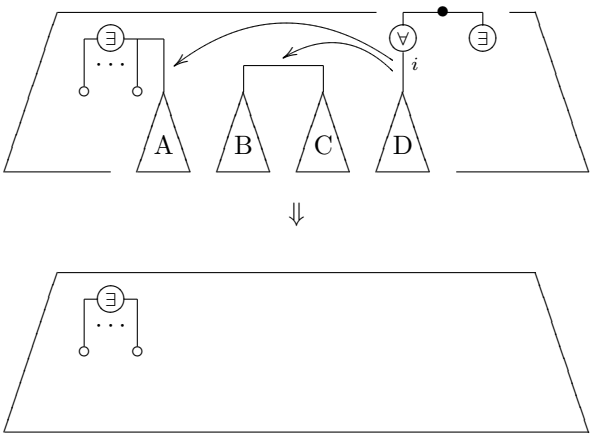


(5)

II. In sequents, a cut on a formula that is introduced by weakening will result in the removal of the entire subproof—Δ in (6)—of the formula on the other side of the cut, replacing it with a series of weakenings to introduce the remainder of the sequent.

$$\dfrac{\dfrac{\vdash \Gamma}{\vdash \Gamma, A}\mathrm{WR} \qquad \dfrac{\Delta}{\vdash \overline{A}, \Gamma'}}{\vdash \Gamma, \Gamma'}\mathrm{Cut} \quad \Rightarrow \quad \dfrac{\vdash \Gamma}{\vdash \Gamma, \Gamma'}\mathrm{WR} \tag{6}$$

The equivalent of a sequent subproof in forests is the set of dependants, the edges smaller in the dependency. In (7) the trees A, B, C and D represent the dependants of the edge i. At the top-left $\exists$-node only the dependent branches are removed, possibly leaving the node as a dead end.



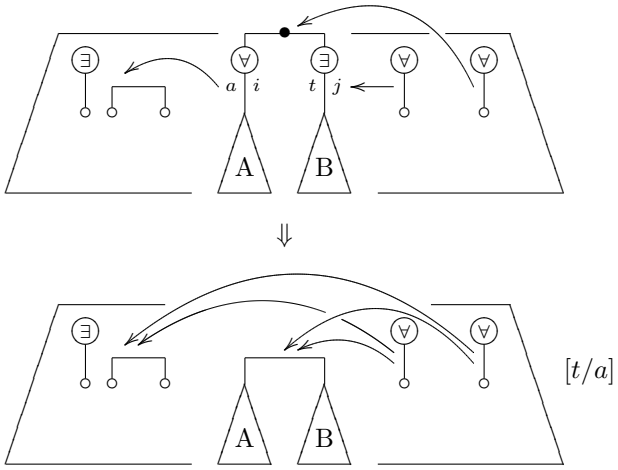
(7)

III. Next, reducing a cut on a ∀R- and a ∃R-inference in sequent calculus replaces all instances of the eigenvariable of the former by the term substituted by the latter, shown in (8).

$$\dfrac{\dfrac{\dfrac{\Delta}{\vdash \Gamma, A[a/x]}}{\vdash \Gamma, \forall x.A}\forall\mathrm{R} \quad \dfrac{\dfrac{\Delta'}{\vdash \overline{A}[t/x], \Gamma'}}{\vdash \exists x.\overline{A}, \Gamma'}\exists\mathrm{R}}{\vdash \Gamma, \Gamma'}\mathrm{Cut} \quad \Rightarrow \quad \dfrac{\dfrac{\Delta[t/a]}{\vdash \Gamma, A[t/x]} \quad \dfrac{\Delta'}{\vdash \overline{A}[t/x], \Gamma'}}{\vdash \Gamma, \Gamma'}\mathrm{Cut} \tag{8}$$

At this point there is a choice to be made regarding the dependency relation. In principle, after the substitution and the removal of the $\forall$-edge and $\exists$-edge, the dependency for the resulting forest could simply be the minimal one. We adopt the alternative of updating the dependency during the reduction steps, according not to the actual, but to the possible migration of the variables. In this setting, given an initial dependency, we can then abstract away from the variables, which allows a sleeker presentation of the counterexample in Section 6—the choice does not affect its non-termination.

The two remaining reduction steps will then not only change the trees in the forest, but explicitly address the dependency relation as well. Specific to the present one, in the right diagram of (8) the variables in t may occur in the resulting cut, and wherever a occurred before. In the forest, then, the edges that first had j as dependant will receive all the former dependants of i, and the new cut, in their new dependency set.



(9)

IV. The last case, a cut on a contraction, is technically the most involved. In the sequent setting, shown in (10), the subproof Δ of the ∀R-inference is duplicated, and each copy is cut against one of the two formulae of the contraction. The diagram is incomplete in the sense that the eigenvariable condition may not be respected in the result; one of the copies of Δ should have all its eigenvariables replaced by fresh ones.

$$\dfrac{\dfrac{\overset{\Delta}{\vdash \Gamma, P[a/x]}}{\vdash \Gamma, \forall x.P}\forall\mathrm{R} \quad \dfrac{\overset{\Delta'}{\vdash \exists x.\neg P, \exists x.\neg P, \Gamma'}}{\vdash \exists x.\neg P, \Gamma'}\mathrm{CR}}{\vdash \Gamma, \Gamma'}\mathrm{Cut}$$

$$\Downarrow$$

$$\dfrac{\dfrac{\overset{\Delta[a'/a]}{\vdash \Gamma, P[a'/x]}}{\vdash \Gamma, \forall x.P}\forall\mathrm{R} \quad \dfrac{\dfrac{\overset{\Delta}{\vdash \Gamma, P[a/x]}}{\vdash \Gamma, \forall x.P}\forall\mathrm{R} \quad \overset{\Delta'}{\vdash \exists x.\neg P, \exists x.\neg P, \Gamma'}}{\vdash \exists x.\neg P, \Gamma, \Gamma'}\mathrm{Cut}}{\vdash \Gamma, \Gamma, \Gamma'}\mathrm{Cut}$$

$$\dfrac{\vdash \Gamma, \Gamma, \Gamma'}{\vdash \Gamma, \Gamma'}\mathrm{CR}$$

(10)

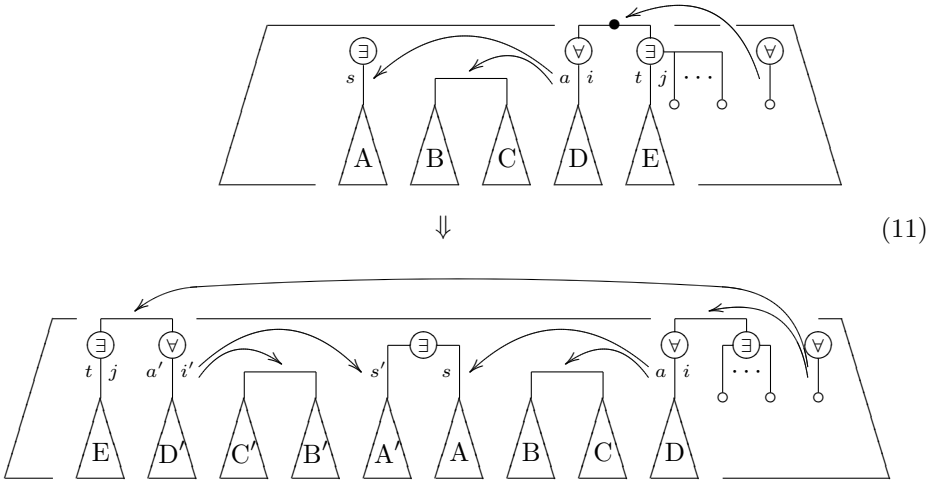
The duplication of Δ in (10) translates in (11) to the duplication of the dependants of i: the tree A with the edge carrying the term s, the trees B and C with their connecting cut, and D. Besides the duplication there is some mopping up to do: switching to fresh eigenvariables in one copy, and preserving the structure of the dependency within each copy. The following three steps complete the definition in (11):

– Let H be the set of dependants of i in the upper diagram, and let σ be a substitution map from the eigenvariables of the ⓥ-edges in H onto a set of

fresh variables such that $\sigma(a) = a'$; then for each tree X the tree X' is X under the substitutions of σ, and s' is the term s under σ.

– Let m and n be edges in trees X and Y respectively, such that $i \leq m \leq n$, and let m' and n' be the corresponding edges in the duplicated trees X' and Y'. Then in the updated dependency we have $i \leq m \leq n$ and $i' \leq m' \leq n'$.

– Let m, n, and n' be as above, but this time let $i \not< m$, $i \neq m$ and $i, m < n$ (using $(<)$ for $(\leq) \cap (\neq)$); in the update we have $i, m < n$ and $i', m < n'$. The exceptions to this are the old and new cuts, for which the dependency is updated according to the diagram.

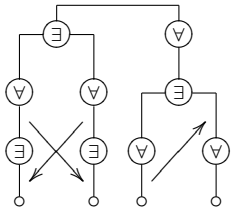


(11)

An alternative way of organising the reduction steps would be to amalgamate the third and fourth steps, immediately applying a logical step to the cut isolated from a contraction. For the case of a cut on a single-branch ∃-node, the duplication by the combined step would leave one reduced copy, and one copy cut against a dead-end ∃-node.

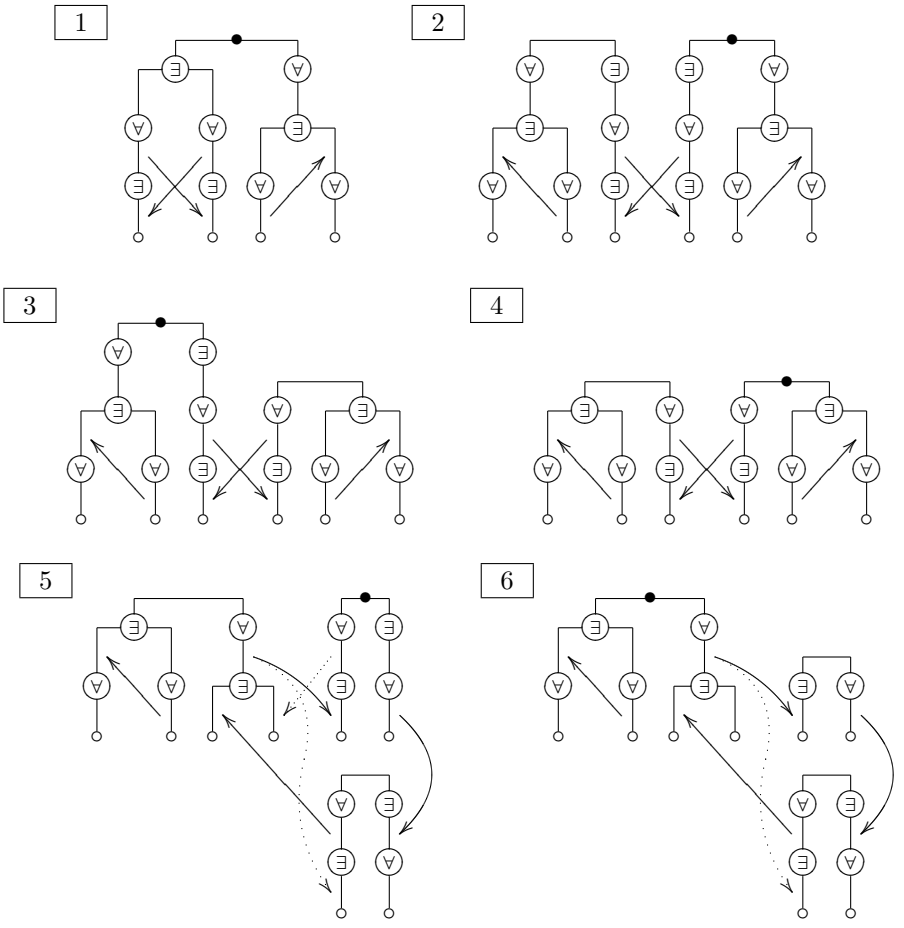
Taking a step back from the technical details, we would like to point out that the algorithm is actually not that complicated, and even rather elegant. The logical step merely applies a substitution and updates the dependency accordingly, while the structural steps remove or duplicate all edges smaller in the dependency than the universal side of the cut, where the technicalities do nothing more than preserve the structure of the original in the copy. Crucially, the duplication concerns precisely what is needed for the substitution by a logical cut, since the (minimal) dependency pinpoints exactly those edges in which it will take place.

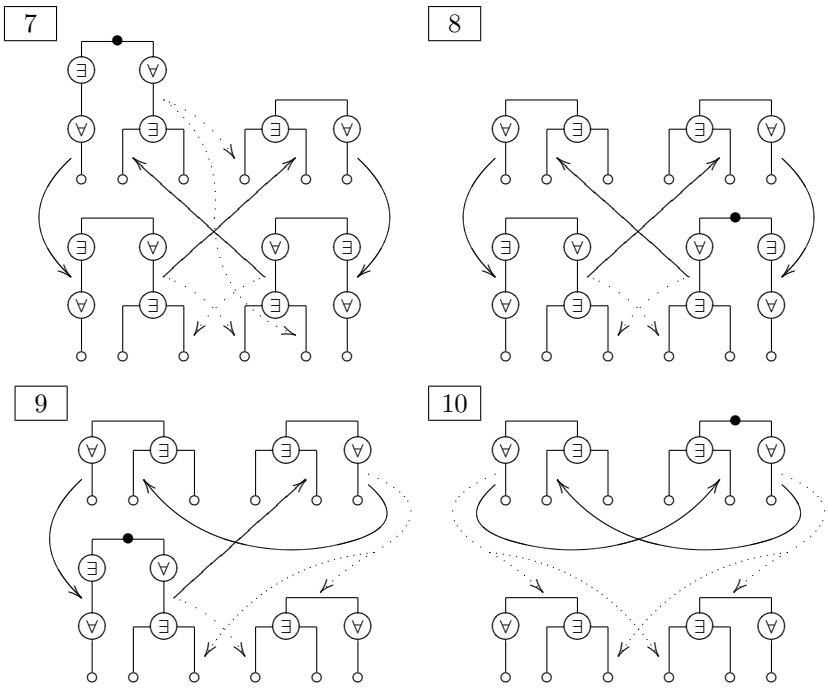
6 The Counterexample to Strong Normalisation



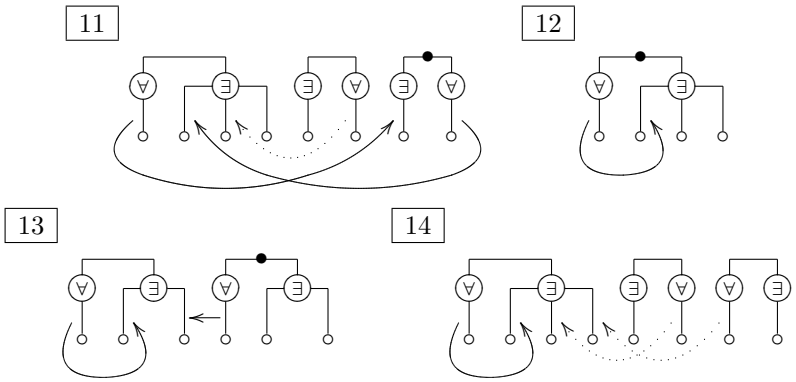
(13)

The example in (13) has a non-terminating reduction in the cut-reduction algorithm given above, which we will shortly exhibit. The structure of the two examples from Section 3 can be recognised on either side of the cut. In the reduction trace below, unfolded until a cycle becomes apparent, the significant part of the dependency relation will be drawn as normal, with the rest drawn as dotted arrows to prevent clutter.





From here on we will omit the trees that are inessential to the example, such as the lower half of step [10] above. These would continue to be duplicated, and as will be clear from the final steps, the cycle does not return the reduction to an earlier configuration, but the forest will grow indefinitely.



7 Significance of the Example

The configuration that most clearly causes non-termination, occurring in the final stages of the reduction trace, is a dependency between the two sides of a single cut. In sequent calculus the corresponding configuration will never occur, since the two sides of a cut are different subproofs; yet the example (13) can

very accurately be replicated as a sequent proof. This suggests the problem may be a new one, different from those known for cut-reduction in sequent calculus.

Unlike the sequent calculus, proof forests have no structural features that require an arbitrary, but significant choice, such as permutable inferences. Structure in these forests is therefore not just sparse, but in a sense even minimal. This property, above all, gave rise to the hope that the natural cut-reduction steps for these forests would be well-behaved. Now that it turns out they are not, the question rises what this means for classical logic in general.

8 Notes and Further Work

We will quickly glance over some relevant observations that have not been investigated thoroughly yet, but certainly deserve to be mentioned. Firstly, there are also reduction paths for (13) that do terminate, but it is not yet clear if weak normalization can be shown in general. Secondly, related to the existence of both terminating and infinite reduction paths, the example (13) also contains a configuration that gives rise to non-confluence. Future investigations will naturally focus on devising a strategy for the algorithm that ensures termination, but may also take the direction of finding out precisely how fundamental the problem exhibited by this example is to classical logic, in particular related to the possibility of confluence in other classical proof systems.

Thanks

Special thanks go to Alex Simpson, who provided the basis of this work and has continued to give invaluable feedback during its development. Thanks also go to Richard McKinley, for a fruitful discussion, and to the anonymous referees for their constructive comments. All diagrams were typeset using XY-pic.

References

1. Buss, S.R.: On Herbrand's Theorem. LNCS, vol. 960, pp. 195–209. Springer-Verlag (1995)
2. Coquand, T.: A Semantics of Evidence for Classical Arithmetic. JSL, vol. 60 (1), pp. 325–337 (1995)
3. Guglielmi, A. and Gundersen, T.: Normalisation Control in Deep Inference via Atomic Flows, Logical Methods in Computer Science, vol 4(1:9), pp. 1–36 (2008)
4. Hughes, D.J.D.: Proofs without syntax. Annals of Mathematics, vol 164(3), pp. 1065–1076 (2006)
5. McKinley, R.I.: Herbrand Expansion Proofs and Proof Identity. Submitted to this workshop (2008)
6. Lamarche, F. and Straßburger, L.: Naming Proofs in Classical Propositional Logic. LNCS, vol. 3461, pp. 246–261. Springer-Verlag (2005)
7. Miller, D.A.: A Compact Representation of Proofs. Studia Logica, vol. 46(4), pp. 347–370 (1987)