Redirecting Resolution Proofs

Jasmin Christian Blanchette
Technische Universität München, Germany
blanchette@in.tum.de

1 Introduction

The proofs returned by automatic theorem provers (ATPs) are notoriously difficult to read. This is an issue if an ATP solves an open mathematical problem (such as the Robbins conjecture [?]), because users then certainly want to study the proof closely. But even in the context of program verification, where users are normally satisfied with a "proved" or "disproved," they might still want to look at the proofs—for example, if they suspect that their axioms are inconsistent.

Our interest in readable ATP proofs has a different origin. The tool Sledgehammer [?, ?] integrates ATPs with the interactive theorem prover Isabelle/HOL [?]. Given an Isabelle/HOL conjecture, Sledgehammer heuristically selects a few hundred relevant lemmas from Isabelle's libraries, translates them along with the conjecture, and sends the resulting problem to E [?], SPASS [?], and Vampire [?]. Proofs are reconstructed in Isabelle either using a single invocation of the built-in resolution prover Metis [?] or as structured Isar [?] proofs following the ATP proofs [?]. This latter option is useful for larger proofs, which Metis fails to re-find within a reasonable time. But most users find the proofs hideous and are little inclined to insert them in their formalizations.

As illustration, consider the conjecture "$\mathit{length}\ (\mathit{tl}\ \mathit{xs}) \le \mathit{length}\ \mathit{xs}$", which states that the tail of a list (the list from which we remove its first element, or the empty list if the list is empty) is shorter than or of equal length as the original list. The proof produced by Vampire, translated to Isabelle's structured Isar format, looks as follows:

proof *neg_clausify*
  assume "$\neg\ \mathit{length}\ (\mathit{tl}\ \mathit{xs}) \le \mathit{length}\ \mathit{xs}$"
  hence "$\mathit{drop}\ (\mathit{length}\ \mathit{xs})\ (\mathit{tl}\ \mathit{xs}) \ne []$" **by** (*metis drop_eq_Nil*)
  hence "$\mathit{tl}\ (\mathit{drop}\ (\mathit{length}\ \mathit{xs})\ \mathit{xs}) \ne []$" **by** (*metis drop_tl*)
  hence "$\forall u.\ \mathit{xs}\ @\ u \ne \mathit{xs} \lor \mathit{tl}\ u \ne []$" **by** (*metis append_eq_conv_conj*)
  hence "$\mathit{tl}\ [] \ne []$" **by** (*metis append_Nil2*)
  thus "*False*" **by** (*metis tl.simps*(1))
qed

The *neg_clausify* method transforms the Isabelle conjecture into negated clause form, ensuring that it has the same shape as the corresponding ATP conjecture. The negation of the clause is introduced by the **assume** keyword, and a series of intermediate facts introduced by **hence** lead to a contradiction.

The first obstacle for readability is that the Isar proof, like the underlying ATP proof, is by contradiction. Such proofs can be turned around by applying contraposition repeatedly. For example, the above proof can be transformed into

proof –
  have "$\mathit{tl}\ [] = []$" **by** (*metis tl.simps*(1))

hence "$\exists u.\ xs\ @\ u = xs \wedge tl\ u = []$" **by** ($metis\ \ append\_Nil2$)
hence "$tl\ (drop\ (length\ xs)\ xs) = []$" **by** ($metis\ \ append\_eq\_conv\_conj$)
hence "$drop\ (length\ xs)\ (tl\ xs) = []$" **by** ($metis\ \ drop\_tl$)
thus "$length\ (tl\ xs) \leq length\ xs$" **by** ($metis\ \ drop\_eq\_Nil$)
qed

Most Isabelle users find the direct proof much easier to understand and maintain. The proof is still somewhat difficult to read because lemmas are referred to by name, but it becomes clear once we reveal them:

$$\begin{array}{rl}
tl.simps(1)\text{:} & tl\ [] = [] \\
append_Nil2\text{:} & xs\ @\ [] = xs \\
append_eq_conv_conj\text{:} & xs\ @\ ys = zs \longleftrightarrow xs = take\ (length\ xs)\ zs \wedge ys = drop\ (length\ xs)\ zs \\
drop_tl\text{:} & drop\ n\ (tl\ xs) = tl\ (drop\ n\ xs) \\
drop_eq_Nil\text{:} & drop\ n\ xs = [] \longleftrightarrow length\ xs \leq n
\end{array}$$

The direct proof also forms a good basis for further development: The user can clean it up further to increase readability and maintainability.

There is a large body of research about making resolution proofs readable. Earlier work focused on translating detailed resolution proofs into natural deduction or sequent calculi. Although they are arguably more readable, these calculi still operate at the logical level, whereas humans reason mostly at the assertion level, invoking definitions and lemmas without providing the full logical details. A line of research focused on transforming natural deduction proofs into assertion-level proofs, culminating with the systems TRAMP [?] and Otterfier [?].

We would have liked to try out TRAMP and Otterfier, but these are large pieces of unmaintained software that are hardly installable on modern machines and that only support older ATPs. Regardless, if we look at Sledgehammer proofs, the problem looks somewhat different. Modern ATPs produce proofs with fairly large steps. Because Sledgehammer supplies the ATPs with hundreds of lemmas, they tend to find short proofs, typically involving a handful of lemmas. Moreover, Sledgehammer merges consecutive logical steps into larger steps and can be further instructed to keep only each nth larger step, if short proofs are desired. Replaying is also not such an important issue for us, since we can rely on the fairly powerful Metis prover.

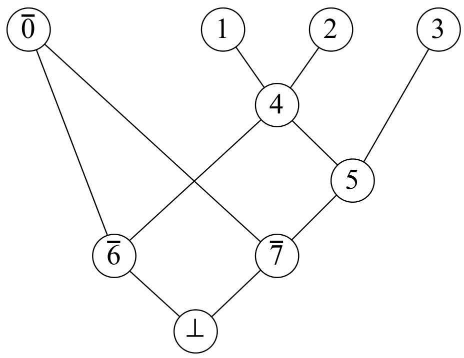
As a first step toward more intelligible proofs, I looked for a method to turn contradiction proofs around. It seemed obvious the method should not be tied to any logic as long as it is classical, or to any calculus. In particular, it should work on the Isar proofs generated by Sledgehammer, or directly on first-order TSTP proofs. Finally, the direct proof should be expressible comfortably in a block-structured Isar-like language, with case splits and nested subproofs.

In my PAAR 2010 and IWIL 2010 talks [?, ?], I sketched a method for transforming proofs by contradiction into direct proofs. The method was very simple and simply exploited the contrapositive, but in the worst case the resulting proof could explode exponentially in size. In this paper, I propose a new method transform resolution proofs into direct proofs expressed in a simple Isar-like syntax (Section 2), based on a simple set of rules (Sections 3 and 4). The new approach is sound and complete and has the advantage that each deduction in the ATP proof gives rise to exactly one deduction in the direct proof. A linear number of additional steps are introduced in the direct proof, but these are simple logical rules, such as modus ponens, and do not require the full power of Metis.

2 Proof Notations

First, we need to define the format we want to use for proofs. We need several formats.

Proof graph. A proof graph is a directed acyclic graph where an arrow $a \rightarrow b$ indicates that a is used to derive b. By ensuring that derived nodes appear lower than their parent nodes in the graph, we can omit the arrowheads:



ATP proofs usually identify formulas (typically clauses) by numbers: for example, the conjecture might be called 0, the axioms might be numbered 1, 2, ..., 250, and new derivations might be called 251 and above. In this paper, we abstract the ATP proofs by ignoring the actual formulas and just keeping the numbers. Also, for our own convenience, we put a bar on top of the negated conjecture and all the formulas derived directly or indirectly from it, denoting negation. To unnegate the conjecture, or negate any of the steps that are *tainted* by the negated conjecture, we simply remove the bar. For the last step, we write $\bot$ rather than $\overline{\top}$.

Proof trees. Proof trees are the standard notation for natural deduction proofs. For example, the proof graph above is captured by the following proof tree:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\overline{0}]}{\overline{0}} \quad \dfrac{\dfrac{1 \quad 2}{1\wedge 2} \quad 1\wedge 2 \longrightarrow 4}{4}}{\overline{0}\wedge 4} \quad \overline{0}\wedge 4 \longrightarrow \overline{6}}{\overline{6}} \quad \dfrac{\dfrac{\dfrac{[\overline{0}]}{\overline{0}} \quad \dfrac{\dfrac{3 \quad \dfrac{\dfrac{1 \quad 2}{1\wedge 2} \quad 1\wedge 2 \longrightarrow 4}{4}}{3\wedge 4} \quad 3\wedge 4 \longrightarrow 5}{5}}{\overline{0}\wedge 5} \quad \overline{0}\wedge 5 \longrightarrow \overline{7}}{\overline{7}}}{\overline{6}\wedge\overline{7}} \quad \overline{6}\wedge\overline{7} \longrightarrow \bot}{\bot}$$

The proof graph notation is more compact, not least because it enables sharing of subproofs. For that reason, we will prefer them to proof trees.

Isar proofs. Isar proofs are a linearization of natural deduction proofs, but unlike proof trees they enable sharing:

proof *neg_clausify*
  assume $\overline{0}$

have 4 **by** (*metis* $1\ 2$)
have 5 **by** (*metis* $3\ 4$)
have $\overline{6}$ **by** (*metis* $\overline{0}\ 4$)
have $\overline{7}$ **by** (*metis* $\overline{0}\ 5$)
show $\bot$ **by** (*metis* $\overline{6}\ \overline{7}$)
qed

The above proof is by contradiction. A direct proof of the above in Isar would be

proof –
have 4 **by** (*metis* $1\ 2$)
have 5 **by** (*metis* $3\ 4$)
have $6 \vee 7$ **by** *metis*
{ **assume** 6
have 0 **by** (*metis* $4\ 6$) }
moreover
{ **assume** 7
have 0 **by** (*metis* $5\ 7$) }
ultimately show 0 **by** (*metis* ‘$6 \vee 7$’)
qed

We refer to the Isar tutorial [**?**] for more information on the Isar syntax.

Shorthand proofs. The last proof format we need to review is a shorthand notation for a subset of Isar proofs. In their simplest form, these shorthand proofs are simply a list of derivations, where ‘$a_1, \ldots, a_n \ \triangleright\ c$’ means “from a_1 and … and a_n, we conclude c.” If among the derivation’s assumptions a_j we have the previous derivation’s conclusion, we can omit the assumption and write ‘$a_1, \ldots, a_{j-1}, a_{j+1}, \ldots, a_n \ \blacktriangleright\ c$’ (corresponding to Isar’s **hence** and **thus** keywords). Depending on whether we use the abbreviated format, our running example becomes

$1,2 \ \triangleright\ 4$	$1,2 \ \triangleright\ 4$
$3,4 \ \triangleright\ 5$	$3 \ \blacktriangleright\ 5$
$\overline{0},4 \ \triangleright\ \overline{6}$	$\overline{0},4 \ \triangleright\ \overline{6}$
$\overline{0},5 \ \triangleright\ \overline{7}$	$\overline{0},5 \ \triangleright\ \overline{7}$
$\overline{6},\overline{7} \ \triangleright\ \bot$	$\overline{6} \ \blacktriangleright\ \bot$

Each step is essentially a sequent $\Gamma \vdash \varphi$. The assumptions are either the negated conjecture ($\overline{0}$), facts that were proved elsewhere (1, 2, and 3), or formulas that were proved in preceding sequents (4, 5, $\overline{6}$, and $\overline{7}$). The succedent of the last sequent is empty ($\bot$).

Direct proofs can be done the same way, but they have the constraints that we may not assume the negated conjecture $\overline{0}$ in any of the sequent and that the last sequent has the conjecture 0 as succedent. However, in some of the direct proofs, we will find it useful to introduce case splits, as we had in the Isar proof above. For example:

$$\begin{array}{rcl} 1,2 & \rhd & 4 \\ 3 & \blacktriangleright & 5 \\ & \rhd & 6,7 \end{array}$$

$$\left[\begin{array}{rcl|rcl} & & [6] & & & [7] \\ 4 & \blacktriangleright & 0 & 5 & \blacktriangleright & 0 \end{array}\right]$$

The notation in brace is a case split. In general, a case split has the form

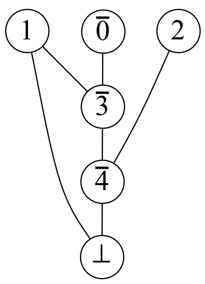
$$\left[\begin{array}{rcl|c|rcl} & & [\Gamma_1] & \cdots & & & [\Gamma_n] \\ \Delta_{11} & \rhd & \varphi_{11} & \cdots & \Delta_{n1} & \rhd & \varphi_{n1} \\ & & \vdots & & & & \vdots \\ \Delta_{1k_1} & \rhd & \varphi_{1k_1} & \cdots & \Delta_{nk_n} & \rhd & \varphi_{nk_n} \end{array}\right]$$

with the requirement that a sequent with the succedent $\Gamma_1, \ldots, \Gamma_n$ has been proved already. Also, each of the branches must be a valid proof. The assumptions $[\Gamma_j]$ may be used to discharge assumptions in the same branch, just as if they had been sequents $\rhd\ \Gamma_j$. Seen from outside, the case split expression stands for a sequent with the succedent $\varphi_{1k_1}, \ldots, \varphi_{nk_n}$.

3 Introductory Examples

3.1 A Linear Proof

We start with a simple proof by contradiction expressed as a proof tree and using our shorthand notation:



$$\begin{array}{rcl} \overline{0}, 1 & \rhd & \overline{3} \\ 2, \overline{3} & \rhd & \overline{4} \\ 1, \overline{4} & \rhd & \bot \end{array}$$

Recall from Sect. 2 that the negated conjecture all all the proof steps that are tainted by it are shown as negated. Next, we turn the sequents around using contraposition at the sequent level to avoid all negations. This gives

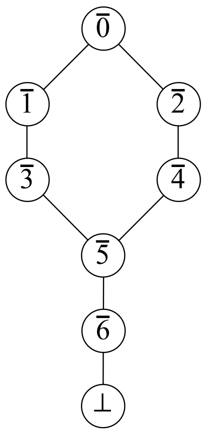
$$\begin{array}{rcl} 1,3 & \rhd & 0 \\ 2,4 & \rhd & 3 \\ 1 & \rhd & 4 \end{array}$$

Finally, if we reorder the sequents and introduce $\blacktriangleright$ wherever possible, we obtain a direct proof:

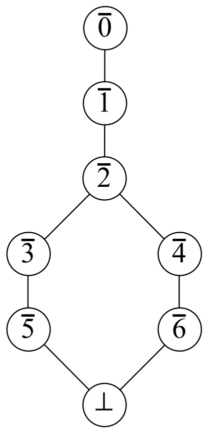
$$
\begin{array}{ll}
\textbf{proof} - & \\
\quad \textbf{have } 4 \textbf{ by } \textit{metis} & \quad\quad \rhd\ 4 \\
\quad \textbf{hence } 3 \textbf{ by } (\textit{metis } 2) & 2\ \blacktriangleright\ 3 \\
\quad \textbf{thus } 0 \textbf{ by } (\textit{metis } 1) & 1\ \blacktriangleright\ 0 \\
\textbf{qed} &
\end{array}
$$

3.2 Two Lasso-Shaped Proofs

The next two examples are lasso-shaped proofs:



$$
\begin{array}{rcl}
\overline{0} & \rhd & \overline{1} \\
\overline{0} & \rhd & \overline{2} \\
\overline{1} & \rhd & \overline{3} \\
\overline{2} & \rhd & \overline{4} \\
\overline{3},\overline{4} & \rhd & \overline{5} \\
\overline{5} & \rhd & \overline{6} \\
\overline{6} & \rhd & \bot
\end{array}
$$



$$
\begin{array}{rcl}
\overline{0} & \rhd & \overline{1} \\
\overline{1} & \rhd & \overline{2} \\
\overline{2} & \rhd & \overline{3} \\
\overline{2} & \rhd & \overline{4} \\
\overline{3} & \rhd & \overline{5} \\
\overline{4} & \rhd & \overline{6} \\
\overline{5},\overline{6} & \rhd & \bot
\end{array}
$$

We start with the contradiction proof on the left-hand side. Starting from $\bot$, it is easy to turn the proof around up to the lasso cycle:

$$
\begin{array}{rcl}
 & \rhd & 6 \\
6 & \rhd & 5 \\
5 & \rhd & 3 \vee 4
\end{array}
$$

When applying the contrapositive to eliminate the negations in $\overline{3},\overline{4} \ \rhd\ \overline{5}$, we obtain a disjunction on the right-hand side of the sequent: $5\ \rhd\ 3 \vee 4$. To continue from there, we need a case split. Then we can finish then branches:

$$
\left[
\begin{array}{rcl|rcl}
 & & [3] & & & [4] \\
3 & \rhd & 1 & 4 & \rhd & 2 \\
1 & \rhd & 0 & 2 & \rhd & 0
\end{array}
\right]
$$

The second lasso example is more tricky, because the cycle occurs near the end of the contradiction proof. Already when redirecting the last inference, we get a disjunction:

$$\rhd\ 5 \vee 6$$

And if we split naively and finish each branch independently of each other, we end up with a fair share of duplication:

$$\left[\begin{array}{r|r} [5] & [6] \\ 5 \rhd 3 & 6 \rhd 4 \\ 3 \rhd 2 & 4 \rhd 2 \\ 2 \rhd 1 & 2 \rhd 1 \\ 1 \rhd 0 & 1 \rhd 0 \end{array}\right]$$

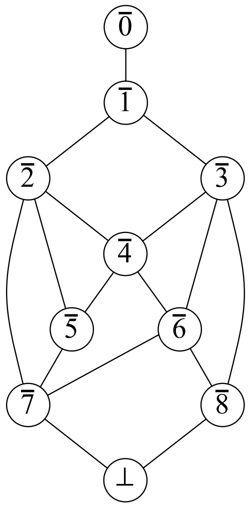
In this case, the key is to observe that it is enough if both branches prove 2, and from there we prove the rest. The complete proof is (without and with $\blacktriangleright$):

$$\begin{array}{c} \rhd\ 5 \vee 6 \\ \left[\begin{array}{r|r} [5] & [6] \\ 5 \rhd 3 & 6 \rhd 4 \\ 3 \rhd 2 & 4 \rhd 2 \end{array}\right] \\ 2 \rhd 1 \\ 1 \rhd 0 \end{array} \qquad \begin{array}{c} \rhd\ 5 \vee 6 \\ \left[\begin{array}{r|r} [5] & [6] \\ \blacktriangleright\ 3 & \blacktriangleright\ 4 \\ \blacktriangleright\ 2 & \blacktriangleright\ 2 \end{array}\right] \\ 2 \blacktriangleright 1 \\ \blacktriangleright\ 0 \end{array}$$

Here we were lucky because we could join both branches the node 2 to avoid all duplication. If we want to stick to a strict no-duplication policy, in general we sometimes need to join on a disjunction $\varphi_1 \vee \cdots \vee \varphi_n$, as the next example will illustrate.

3.3 A Diabolical Proof

Our final example is truly diabolical (and sightly unrealistic):



$$\begin{array}{rcl} \overline{0} & \rhd & \overline{1} \\ \overline{1} & \rhd & \overline{2} \\ \overline{1} & \rhd & \overline{3} \\ \overline{2},\overline{3} & \rhd & \overline{4} \\ \overline{2},\overline{4} & \rhd & \overline{5} \\ \overline{3},\overline{4} & \rhd & \overline{6} \\ \overline{2},\overline{5},\overline{6} & \rhd & \overline{7} \\ \overline{3},\overline{6} & \rhd & \overline{8} \\ \overline{7},\overline{8} & \rhd & \perp \end{array}$$

We start with the contrapositive of the last sequent:

$$\rhd\ 7 \vee 8$$

Then we identify the nodes that dominate only one branch in the refutation graph (i.e., that are reachable by navigating upward). Looking at the graph, we see that $\overline{5}$ dominates $\overline{7}$ but not $\overline{8}$, and is the only such node. We perform a case split and allow ourselves to perform inferences requiring 5 and 7 in the left branch and 8 in the right branch. First the right branch:

$$\begin{array}{c}[8]\\ 8 \;\rhd\; 3 \lor 6\end{array}$$

There is nothing more we can do for now: Any further inferences we would do in the right branch would need to be repeated in the left branch. Now we do the left branch:

$$\begin{array}{c}[7]\\ 7 \;\rhd\; 2 \lor 5 \lor 6\end{array}$$

We would now like to perform the inference $5 \;\rhd\; 2 \lor 4$. This requires a case split:

$$\left[\begin{array}{c|c|c} & [5] & \\ [2] & 5 \;\rhd\; 2 \lor 4 & [6]\end{array}\right]$$

The 2 and 6 subbranches are left alone. Since only one branch of the subcase split is nontrivial, we find it more esthetically pleasing to abbreviate it to

$$2 \lor 5 \lor 6 \;\rhd\; 2 \lor 4 \lor 6$$

Hence, the left branch proves $2 \lor 4 \lor 6$, the right branch proves $3 \lor 6$, and both branches together prove $2 \lor 3 \lor 4 \lor 6$.

This gives us a new case split, but notice that 6 is dominated by 2, 3, and 4. We start with it:

$$\left[\begin{array}{c|c|c|c} & & & [6] \\ [2] & [3] & [4] & 6 \;\rhd\; 3 \lor 4\end{array}\right]$$

This proves $2 \lor 3 \lor 4$. Since all but one branches are trivial, we abbreviate it:

$$2 \lor 3 \lor 4 \lor 6 \;\rhd\; 2 \lor 3 \lor 4$$

It might help to think of such steps as "rewriting" steps, where 6 is rewritten into $3 \lor 4$. Now, 4 is dominated by 2 and 3, so we perform another case split:

$$\left[\begin{array}{c|c|c} & & [4] \\ [2] & [3] & 4 \;\rhd\; 2 \lor 3\end{array}\right]$$

And we abbreviate it:

$$2 \lor 3 \lor 4 \;\rhd\; 2 \lor 3$$

We are left with the conclusion $2 \lor 3$. The rest is reminiscent of our second lasso-shaped proof:

$$\left[\begin{array}{r|r} [2] & [3] \\ 2 \;\rhd\; 1 & 3 \;\rhd\; 1\end{array}\right]$$

$$1 \;\rhd\; 0$$

Putting all of this together (and using $\blacktriangleright$), we get

```
proof –
  have 7 ∨ 8 by metis
  moreover
  { assume 7
    hence 2 ∨ 5 ∨ 6 by metis
    hence 2 ∨ 4 ∨ 6 by metis }
  moreover
  { assume 8
    hence 3 ∨ 6 by metis }
  ultimately have 2 ∨ 3 ∨ 4 ∨ 6 by metis
  hence 2 ∨ 3 ∨ 4 by metis
  hence 2 ∨ 3 by metis
  moreover
  { assume 2
    hence 1 by metis }
  moreover
  { assume 3
    hence 1 by metis }
  ultimately have 1 by metis
  thus 0 by metis
qed
```

$$\begin{array}{c} \rhd\ 7 \lor 8 \\ \left[\begin{array}{c|c} [7] & \\ \blacktriangleright\ 2\lor 5\lor 6 & [8] \\ \blacktriangleright\ 2\lor 4\lor 6 & \blacktriangleright\ 3\lor 6 \end{array}\right] \\ \blacktriangleright\ 2\lor 3\lor 4 \\ \blacktriangleright\ 2\lor 3 \\ \left[\begin{array}{c|c} [2] & [3] \\ \blacktriangleright\ 1 & \blacktriangleright\ 1 \end{array}\right] \\ \blacktriangleright\ 0 \end{array}$$

Which is not too bad, considering the spaghetti we started with.

4 The Algorithm

The process we applied to the examples in the previous section can be generalized into an algorithm. The algorithm relies on the proof by contradiction expressed as a set of sequents, which are turned around using the contrapositive so that the formulas are unnegated (as we did in all the examples). It maintains a set of *proved nodes* (initially the empty set) and a set of *target nodes* that it may proceed to prove (initially the set of all unnegated nodes). The constructed proof is expressed using the shorthand notation.

1. Derive as many sequents as possible with their conclusion in the target set based on the proved nodes. Each time a sequent is appended to the proof, its conclusion is added to the set of proved nodes.

2. If all the nodes in the target set are proved, we are done. Otherwise, the last sequent must be of the form $\Gamma \rhd\ c_1 \lor \cdots \lor c_n$ for $n \geq 2$. Perform an n-way case split:[1]

$$\left[\begin{array}{c|c|c} [c_1] & \cdots & [c_n] \\ & & \end{array}\right]$$

[1] A generalization would be to perform an m-way case split, with $m < n$, on disjuncts. For example, we could do a 3-way case split with $c_1 \lor c_2$, c_3, and c_4 as the assumptions instead of breaking all the disjunctions and doing a 4-way split. This could potentially lead to cleaner proofs, if the groupings are chosen carefully. We will not explore this avenue further in this paper.

3. For each of the branches, invoke the procedure recursively, with the assumption added to the proved set and with all the nodes that dominate the branch and no other branch as the target set.

4. After the case split, go back to step 1. The case split itself is seen as a sequent of the form $\Gamma \rhd c_1 \vee \cdots \vee c_n$ as far as step 2 is concerned.

Once this procedure has terminated, we abbreviate case splits in which all but one branch are trivial using "rewriting," transforming

$$\left[\begin{array}{c|c|c|c|c|c|c} & & & [c_i] & & & \\ & & & \Delta_1 \rhd d_1 & & & \\ & & & \vdots & & & \\ [c_1] & \cdots & [c_{i-1}] & \Delta_n \rhd d_n & [c_{i+1}] & \cdots & [c_m] \end{array}\right]$$

into

$$\begin{array}{l} \Delta_1 \rhd c_1 \vee \cdots \vee c_{i-1} \vee d_1 \vee c_{i+1} \vee \cdots c_m \\ \qquad \vdots \\ \Delta_n \rhd c_1 \vee \cdots \vee c_{i-1} \vee d_n \vee c_{i+1} \vee \cdots c_m \end{array}$$

This works even if the c_i branch has case splits. What we are doing effectively is taking the nontrivial branch and adding $c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_m$ as disjuncts in the conclusions.

Finally, we clean up the proof by introducing ► and translate it to Isar.

5 Conclusion

We presented a method to transform proofs by contradiction as returned by resolution provers into direct proofs. This sometimes introduces case splits, but our procedure avoids duplicating proof steps in the different branches of the split by joining again as early as possible. The result is direct Isar proofs that have some of the structure Isabelle users have come to expect. Our approach is fairly simple, it would not surprise us if it were part of folklore or a special case of existing work.

In a second step, we are interested in transformations that increase proof readability, such as those available for Mizar proofs [**?**]. We also need to address other issues not treated satisfactorily in Sledgehammer [**?**], notably Skolemization.

Acknowledgment. I want to thank Tobias Nipkow, Geoff Sutcliffe, and Joseph Urban for motivating me to look into this. I am also thankful to Andrei Popescu, who with his deep theoretical insights helped simplify the approach considerably.