The NP-hardness of finding a directed acyclic graph for regular resolution

Samuel R. Buss[*]
Department of Mathematics
University of California, San Diego
La Jolla, CA 92093-0112, USA
sbuss@math.ucsd.edu

Jan Hoffmann[†]
Department of Mathematics
University of California, San Diego
La Jolla, CA 92093-0112, USA
hoffmann@cip.ifi.lmu.de

January 13, 2008

Abstract

Let R be a resolution refutation, given as a sequence of clauses without explicit description of the underlying dag. Then, it is NP-complete to decide whether R is a regular resolution refutation.

1 Introduction

A resolution refutation of a set Γ of clauses is usually defined as a sequence of clauses; each clause must either be a member of Γ (an *initial clause*), or be derived from two earlier clauses by a resolution inference. Alternatively, the refutation may be described as a directed acyclic graph (dag). These two representations are p-equivalent in the usual sense that a proof specified by either representation may be easily transformed into a proof in the other representation by a polynomial time algorithm. However, in general, there can be more than one way to convert a sequence of clauses into a dag.

Many widely studied properties of refutations, including the notions of regular resolution, unit resolution, input resolution, linear resolution, and ordered resolution, depend on the choice of the dag structure for the refutation. In particular, these notions are not immediately applicable to

[*]Supported in part by NSF grants DMS-0400848 and DMS-0700533.

[†]Supported in part by the Studienstiftung des deutschen Volkes (German National Merit Foundation) while visiting UCSD. Permanent address: Institut for Informatik, Ludwig-Maximilians Universität, D-80538 München, Germany.

sequence-like proofs unless a dag structure is imposed. The present paper considers primarily regular resolution and the hardness of deciding whether a sequence-like resolution is regular. If a refutation is specified as a sequence rather than a dag, then the definition of regularity is not well-specified. In fact, it is easy to find a sequence-like refutation that admits multiple dag's, some of which are regular and some of which are not. In this paper, we prove a stronger fact. Namely, it is NP-complete to decide whether a sequence-like refutation admits a dag representation which makes it regular.

Lower bounds for the complexity of regular refutations have been extensively studied already by [13, 6, 7, 4, 10, 2]. Several conditional lower bounds for the difficulty of *finding* resolution proofs have been given by Alekhnovich et al. [1], Iwama [8], and Alekhnovich and Razborov [3]. In a different direction, Iwama and Miyano [9] and Szeider [12] gave NP-hardness results on the difficulty of deciding whether formulas have resolution refutations in the read-once and literal-once fragments of resolution. The present paper, however, studies a different type of problem. Rather than considering the hardness of finding a refutation or of determining if a refutation exists, we assume that a refutation is already given and give a lower bound on the hardness of determining if it is regular.

2 Regular refutations

Resolution is a propositional refutation system that uses variables $a, b, c, \ldots$ intended to range over the values *True* and *False*. The negation of a variable a is denoted $\overline{a}$. A *literal* ℓ is either a variable or a negated variable. A *clause* $C = \{\ell_1, \ldots, \ell_k\}$ is a a finite set of literals and denotes the disjunction $\ell_1 \vee \ell_2 \vee \cdots \vee \ell_k$. A set Γ of clauses is interpreted as the conjunction of its clauses, that is, as a formula in conjunctive normal form.

A *resolution refutation* can be used to prove a set Γ to be unsatisfiable. A (sequence-like) resolution refutation consists of a sequence of clauses $C_1, \ldots, C_m$ such that C_m is the empty clause and such that each C_i is either an initial clause from Γ or is derived by a resolution inference from two clauses C_j and C_k that appear earlier in the refutation. A *resolution inference* is any inference of the form

$$\frac{D \cup \{x\} \qquad E \cup \{\overline{x}\}}{D \cup E}.$$

For sequence-like refutations, we always assume that the set Γ of initial clauses is explicitly specified, so that there is a polynomial time procedure to decide which clauses are valid initial clauses.

A dag-like refutation is a sequence-like refutation in which each clause C_i either is labelled as an initial clause or is labeled with the indices j and k of the clauses from which it is inferred: a dag can be defined by taking the clauses in the refutation as nodes and letting there be directed edges from any non-initial clause to each of the two clauses from which it is inferred. The clauses labelled as initial clauses must be in Γ. Note, however, that it is permitted that a clause in Γ appear in the dag as a non-initial clause. That is, a clause in Γ can be (re)derived in the refutation. A dag-like refutation is *rooted* provided that, for each C_i, there is a directed path from $C_m = \emptyset$ to C_i.

A dag-like refutation is *regular* if there does not exist any directed path π in the refutation with a variable x used twice as the resolution variable along the path. A sequence-like refutation is defined to be *regular* provided there exists some way to label the clauses with indices so as to make the refutation into a regular dag-like refuation.

Theorem 1 *The set of regular sequence-like refutations is NP-complete.*

An immediate consequence of Theorem 1 is that, assuming $P \neq NP$, the regular sequence-like proofs are not polynomial time recognizable and thus do not form a proof system in the sense of Cook and Reckhow [5].

Theorem 1 is proved by a reduction from the well-known NP-complete Vertex Cover problem. An instance of Vertex Cover is a graph $G = (V, E)$ and an integer k; the problem is to determine whether there is a set of k vertices from V (k pebbles) such that each edge $e \in E$ has at least one of its vertices in the set (that is, has a pebbled vertex).

Given $G = (V, E)$, we construct a sequence-like refutation R. R will be regular if and only if G has a vertex cover of size $\leq k$. The refutation will have two stages: One stage ensures that at most k vertices are pebbled, and the other ensures that each edge is covered by at least one vertex. R uses variables v_i that correspond to vertices in V. R will contain each unit clause $\{v_i\}$, and this is derived by a resolution either with a variable p_j (indicating that the i-th vertex is covered by the j-th pebble) or with a variable q_i (indicating that the i-th vertex is uncovered). In the second stage, variables e_i ensure that the i-th edge has at least one vertex pebbled. We construct the refutation R below; Figure 1 also shows the clauses in R.

The first stage of R consists of the following clauses. As initial clauses, the first clauses in R are the unit clauses $\{\overline{p_j}\}$, $\{\overline{q_i}\}$ and $\{\overline{u_i}\}$, for $i = 1, \ldots, |V|$ and $j = 1, \ldots, k$. There is also an initial clause $\{v_0\}$: here v_0 does not correspond to any vertex of V, but rather helps with the base case. Then, for each $i = 1, 2, \ldots, |V|$, R contains the $k+1$ initial clauses $\{\overline{v_{i-1}}, v_i, q_i, u_i\}$ and $\{\overline{v_{i-1}}, v_i, p_j, u_i\}$ for $1 \leq j \leq k$, followed the $k+3$ non-initial clauses $\{v_i, p_j, u_i\}$, $\{v_i, q_i, u_i\}$, $\{v_i, u_i\}$, and $\{v_i\}$. The clause $\{v_i\}$ can be derived by an inference of the form

$$\cfrac{\cfrac{\cfrac{\{v_{i-1}\} \qquad \{\overline{v_{i-1}}, v_i, x, u_i\}}{\{v_i, x, u_i\}} \qquad \{\overline{x}\}}{\{v_i, u_i\}} \qquad \{\overline{u_i}\}}{\{v_i\}}$$

where x is one of $p_1, \ldots, p_k$ or q_i. Then, for $y \in \{p_1, \ldots, p_k, q_i\} \setminus \{x\}$, the clauses $\{v_i, y, u_i\}$ are derived by resolving $\{v_{i-1}, u_i\}$ and $\{\overline{v_{i-1}}, v_i, y\}$. These clauses $\{v_i, y, u_i\}$ are unused in R (but see the refined construction below).

There are $k+1$ possible ways of deriving $\{v_i, u_i\}$, but the intent is that $\{v_i, u_i\}$ is derived with the aid of resolving on the variable p_j as x, if the j-th pebble is placed on the i-th vertex. If, however, resolution with q_i is used to derive $\{v_i, u_i\}$, then the i-th vertex is unpebbled. Since the $\{v_i, u_i\}$'s are derived sequentially, any dag-like regular refutation formed from R can use resolution with each fixed p_j only once: this corresponds to the fact that the j-th pebble can be placed on at most one vertex.

Since the clause $\{v_i\}$ must be inferred from $\{v_i, u_i\}$, each clause $\{v_i\}$ will have been derived either with the aid of resolving on p_j (corresponding to the j-th pebble being on vertex i), or with the aid of resolving on q_i (corresponding to the j-th vertex being unpebbled).

The second stage of R is designed so that it can be regular if and only if all edges have a pebbled vertex. Let the s-th edge in E join the i-th and i'-th vertices. Then, R includes two initial clauses $\{e_s, \overline{v_i}, q_i\}$ and $\{e_s, \overline{v_{i'}}, q_{i'}\}$, and the three non-initial clauses $\{e_s, q_i\}$, $\{e_s, q_{i'}\}$, and $\{e_s\}$. The intent is that, when the i-th vertex is pebbled, the inference structure for these five clauses is as follows (using the clauses $\{v_i\}$ and $\{v_{i'}\}$ derived in the first stage of R):

$$\cfrac{\cfrac{\{e_s, \overline{v_i}, q_i\} \qquad \{v_i\}}{\{e_s, q_i\}} \qquad \{\overline{q_i}\}}{\{e_s\}} \qquad\qquad \cfrac{\{e_s, \overline{v_{i'}}, q_{i'}\} \qquad \{v_{i'}\}}{\{e_s, q_{i'}\}}$$

Note that this leaves the clause $\{e_s, q_{i'}\}$ unused in R. If the i-th vertex is pebbled then the resolution on q_i that is used to derive $\{e_s\}$ is a regular inference, but otherwise it is not. If only the i'-th vertex is pebbled then the intent is the inference structure in R should be as above but with the roles of i and i' interchanged.

For $j = 1, \ldots, k$,	
$\quad \{\overline{p_j}\}$.	- initial clause.
For $i = 1, \ldots, \|V\|$,	
$\quad \{\overline{q_i}\}$.	- initial clause.
$\quad \{\overline{u_i}\}$.	- initial clause.
$\{v_0\}$.	- initial clause.
For each $i = 1, \ldots, \|V\|$,	
$\quad \{\overline{v_{i-1}}, v_i, q_i, u_i\}$.	- initial clause.
$\quad \{\overline{v_{i-1}}, v_i, p_j, u_i\}$, for $j = 1, \ldots, k$	- initial clauses.
$\quad \{v_i, q_i, u_i\}$.	
$\quad \{v_i, p_j, u_i\}$, for $j = 1, \ldots, k$.	
$\quad \{v_i, u_i\}$.	
$\quad \{v_i\}$.	
For each edge $s = \{i, i'\}$,	
$\quad \{e_s, \overline{v_i}, q_i\}$.	- initial clause
$\quad \{e_s, \overline{v_{i'}}, q_{i'}\}$.	- initial clause
$\quad \{e_s, q_i\}$.	
$\quad \{e_s, q_{i'}\}$.	
$\quad \{e_s\}$.	
$\{\overline{e_1}, \overline{e_2}, \ldots, \overline{e_{\|E\|}}\}$.	- initial clause.
For $m = 2, \ldots, \|E\|$,	
$\quad \{\overline{e_m}, \overline{e_{m+1}}, \ldots, \overline{e_{\|E\|}}\}$.	
$\emptyset$.	

Figure 1: The refutation constructed in the proof of Theorem 1.

The refutation R ends with one further initial clause

$$\{\overline{e_1}, \overline{e_2}, \ldots, \overline{e_{|E|}}\}$$

and then concludes with resolution inferences using the clauses $\{e_i\}$, $i = 1, \ldots, |E|$, to obtain the empty clause.

It is not hard to verify that R can be given a dag-structure that makes it regular if and only if the graph G has a vertex cover of size k. First, as we remarked earlier, each variable p_j can be used only once to derive a clause $\{v_i, u_i\}$. Thus, all but at most k of the clauses $\{v_i\}$ were derived with the use of resolution on q_i. Second, an edge clause $\{e_s\}$ cannot be derived if both of its endpoint clauses $\{v_i\}$ and $\{v_{i'}\}$ were derived using resolution on q_i and $q_{i'}$ (respectively), that is to say, it cannot be derived unless one of its endpoints was pebbled. Thus, the refutation can be made regular if and only if the graph has a vertex cover of size k. This completes the proof of Theorem 1.

A refined reduction. The above proof of Theorem 1 is formally correct, but has the disconcerting feature that the refuation R admitted only non-rooted dags; that is to say, some clauses were not needed for the refutation. In particular, if the v_i-vertex is not pebbled then the clauses $\{v_i, p_j, u_i\}$ and $\{e_s, q_i\}$ are not used in R. On the other hand, if the i-th vertex is pebbled by the j-th pebble, then the clauses $\{v_i, q_i, u_i\}$, $\{v_i, p_{j'}, u_i\}$ for $j' \neq j$, and possibly $\{e_s, q_i\}$ and $\{v_i\}$ are not used in R. A slightly more complicated construction, outlined below, can overcome this. For this, we describe a refutation R^* that does not have any unused clauses.

First, we make a small modification to R that eliminates some problems with handling the clauses $\{e_s, q_i\}$ in the case where the i-th vertex is not pebbled. The problem is that in this case the clause $\{e_s, q_i\}$ is derived with the aid of resolution with respect to the variable q_i, as that was used to derive $\{v_i\}$ in this situation. This would make it impossible to remove the variable q_i from the clause with regular resolution since it is not allowed to resolve on the variable q_i again. (This is not a problem for the clauses $\{v_i\}$, $\{v_i, q_i, u_i\}$, and $\{v_i, p_j, u_i\}$, since these are not derived using resolution on any variable appearing in the clause.) For each s and each i, R is modified by inserting two new initial clauses $\{e_s, q_i, z\}$ and $\{e_s, q_i, \overline{z}\}$. The purpose is to add an alternate method of deriving $\{e_s, q_i\}$. We also modify R by replacing the clause $\{\overline{e_1}, \ldots, \overline{e_{|E|}}\}$ with the clause

$$\{\overline{e_1}, \overline{e_2}, \ldots, \overline{e_{|E|}}, z\}.$$

The subsequent clauses at the end of R are modified by adding the variable z. This transforms the final (empty) clause into the unit clause $\{z\}$. Finally, two clauses are added to the end of R: a new initial clause $\{\overline{z}\}$ and the empty clause.

The point of adding the extra clauses at the end of the refutation is that, by resolving with respect to the variable z to obtain the empty clause, we prevent the use of a clause $\{e_s, q_i\}$ that was earlier obtained by resolving with respect to the variable z. Thus, adding the new initial clauses $\{e_s, q_i, z\}$ and $\{e_s, q_i, \overline{z}\}$ will not create an unwanted dag-like proof structure.

Let R' be the refutation just constructed; the next claim is easily checked.

Claim G has a vertex cover of size k if and only if R' has a dag structure which makes it a regular refutation such that, for every clause C in R' and every literal $\ell \in C$, ℓ is not used as a resolution literal on any directed path starting from C in R'.

We now construct a sequence-like R^* from R' such that the empty clause is reachable from each clause in R^*. First add a new variable w to all the clauses of R' and let the resulting clauses be $C_1, \ldots, C_m$, so that $C_m = \{w\}$. For $t = 1, \ldots, m$, let y_t be a new variable and let $\ell_{t,i}$ denote the i-th literal in C_t. For each clause C_t, add to the end of R^* new initial clauses $\{y_t, \overline{\ell_{t,i}}\}$ and, then the non-initial clauses that arise when repeatedly resolving with clause C_t to derive the clause $\{y_t\}$. Then add, as a new initial clause at the end of R^*, the clause

$$\{\overline{y_1}, \overline{y_2}, \ldots, \overline{y_{m-1}}\}.$$

and then add, as non-initial clauses, the clauses that are obtained by resolving repeatedly with the clauses $\{y_t\}$ to yield the empty clause. This completes the construction of R^*.

Theorem 2 *If $G = (V, E)$ has a vertex cover of size $\leq k$, then R^* has a regular, rooted dag-structure. If G does not have a vertex cover of size k, then R^* is not regular.*

To prove this, note that since the first m clauses of R^* are the same as in R' except for the addition of the variable w, it is clear that the dag structures that can be put on the first m clauses R^* are exactly the same as the dag structures that can be put on R'. Furthermore, there is a simple way to put a dag structure on the remaining part of R^* that preserves regularity since the property of the claim holds. It follows that R' admits a dag that makes R' a regular proof if and only there is a dag-like structure that makes R^* both regular and rooted. This proves Theorem 2.

Corollary 3 *The set of sequence-like refutations that admit a regular, rooted dag structure is NP-complete.*

One consequence of our theorems is that there is no good method for defining regular refutations without referring to the underlying dag. It is interesting to ask how this compares to other common forms of resolution, such as unit resolution, input resolution, linear resolution, and tree resolution. A *unit* clause is a clause that contains a single literal; a *unit* resolution refutation is one in which every inference has a unit clause as one of its hypotheses. An *input* refutation is one in which every inference has an initial clause as one of its hypotheses. A *linear* refutation is a refutation $C_1, \ldots, C_m$ that has a subsequence $C_{t_1}, \ldots, C_{t_n}$ such that (a) each clause C_j is an initial clause if and only if $j \notin \{t_1, \ldots, t_n\}$, (b) C_{t_1} is inferred from two initial clauses, (c) for each $i > 1$, C_{t_i} is inferred from $C_{t_{i-1}}$ and some other earlier clause, and (d) $t_n = m$ (thus C_{t_n} is the empty clause).

Theorem 4 *There is a polynomial time algorithm which, given a sequence-like refutation R which is either unit, input, or linear, produces a dag-like refutation G which is (respectively) unit, input, or linear. Furthermore, G is obtained by finding a dag-structure on R and letting G be the dag obtained from the component of the dag containing the empty clause.*

Theorem 4 is easy to prove for unit and input refutations; namely, one builds the dag by arbitrarily choosing for each non-initial clause C, two earlier clauses D and E such that C is the resolvent of D and E and such that E is a unit clause (respectively, an initial clause). The proof is similar for linear resolutions: first, the clauses C_{t_i} are identified in polynomial time (which is possible by our assumption that initial clauses are polynomial time recognizable), then one picks, for each $C_{t_{i+1}}$ a clause E earlier in the refutation such that $C_{t_{i+1}}$ is the resolvent of C_{t_i} and E.

Note the proof of Theorem 1 shows that Theorem 4 fails for regular resolution, even for the case where the algorithm's runtime is bounded by a polynomial of the size of the tree-like version of the regular refutation.

Unfortunately, the algorithms of Theorem 4 do not determine whether the sequence-like proofs can be converted into *rooted* dags. A similar issue arose in our proof of Theorem 1, but this was handled by the "refined" construction. For unit, input, and tree refutations, we have not been able to adequately determine the computational complexity of recognizing rooted dag refutations. For linear refutations, Van Gelder [private communication] has given a polynomial time algorithm for recognizing linear refutations which admit a rooted dag linear refutation.

We conclude with some conjectures about the tree case. A dag-like refutation is a *tree* provided it is rooted and all clauses other than the root are used exactly once as a hypothesis (that is, have in-degree one). Thus, in a tree, if any clause (including any initial clause) is used more than once, it must appear in the refutation more than once.

Conjecture 5 *The following problem is NP-complete: Given a sequence-like resolution proof, does it have a compatible dag structure which is a tree?*

The best we have proved so far is that it is NP-complete to determine whether a sequence-like proof contains a subsequence which is a valid tree refutation.

A more general form of the conjecture, which may be easier to prove, can be formulated by abstracting away from refutations. Consider the finite set $[n] = \{1, \ldots, n\}$ with n odd, and suppose that, for each i, the set P_i is either empty or is some set of unordered pairs $P_i = \{\{c_k, d_k\}\}_k$ with

$1 \leq c_k < d_k < i$. We say this collection $\{P_i\}_i$ admits a tree structure if there is some tree with nodes $[n]$ such that each leaf node i has $P_i = \emptyset$ and such that for each internal node i, the set of its children is a member of P_i.

Conjecture 6 *It is NP-complete to determine if a collection $\{P_i\}_i$ admits a tree structure.*

Acknowledgement. We thank Nicolas Rachinsky for suggesting the problem of converting sequence-like proofs to dag-like proofs. We also thank the two anonymous referees for helpful suggestions, and especially A. Van Gelder for many comments, including a correction to the proof of Theorem 1.

References

[1] M. Alekhnovich, S. Buss, S. Moran, and T. Pitassi, *Minimum propositional proof length is NP-hard to linearly approximate*. To appear in the *Journal of Symbolic Logic*. An extended abstract appeared in *Mathematical Foundations of Computer Science (MFCS'98)*, Springer-Verlag Lecture Notes in Computer Science #1450, 1998, pp. 176-184.

[2] M. Alekhnovich, J. Johannsen, T. Pitassi, and A. Urquhart, *An exponential separation between regular and general resolution*, in Proc. 34th Annual ACM Symposium on Theory of Computing, 2002, pp. 448–456.

[3] M. Alekhnovich and A. A. Razborov, *Resolution is not automatizable unless $W[P]$ is tractable*, in Proc. 42nd IEEE Conf. on Foundations of Computer Science (FOCS), 2001, pp. 210–219.

[4] M. L. Bonet, J. L. Esteban, N. Galesi, and J. Johannsen, *On the relative complexity of resolution refinements and cutting planes systems*, SIAM Journal on Computing, 30 (2000), pp. 1462–1484.

[5] S. A. Cook and R. A. Reckhow, *The relative efficiency of propositional proof systems*, Journal of Symbolic Logic, 44 (1979), pp. 36–50.

[6] Z. Galil, *On the complexity of regular resolution and the David-Putnam procedure*, Theoretical Computer Science, 4 (1977), pp. 23–46.

[7] A. Goerdt, *Regular resolution versus unrestricted resolution*, SIAM Journal on Computing, 22 (1993), pp. 661–683.

[8] K. Iwama, *Complexity of finding short resolution proofs*, in Mathematical Foundations of Computer Science 1997, I. Prívara and P. Ruzicka, eds., Lecture Notes in Computer Science #1295, Springer-Verlag, 1997, pp. 309–318.

[9] K. Iwama and E. Miyano, *Intractibility of read-once resolution*, in Proceedings of the Tenth Annual Conference on Structure in Complexity Theory, Los Alamitos, California, 1995, IEEE Computer Society, pp. 29–36.

[10] T. Pitassi and R. Raz, *Regular resolution lower bounds for the weak pigeonhole principle*, in Proc. 33rd Annual ACM Symposium on Theory of Computing, 2001, pp. 347–355.

[11] J. Siekmann and G. Wrightson, *Automation of Reasoning*, vol. 1&2, Springer-Verlag, Berlin, 1983.

[12] S. Szeider, *NP-completeness of refutability by literal-once resolution*, in Automated Reasoning: First International Joint Conference, (IJCAR), Springer Verlag, 2001, pp. 168–181.

[13] G. S. Tsejtin, *On the complexity of derivation in propositional logic*, Studies in Constructive Mathematics and Mathematical Logic, 2 (1968), pp. 115–125. Reprinted in: [11, vol 2], pp. 466-483.