

Due on November 10
In class, at the beginning of the lecture

The work you submit must be your own. You may discuss problems with each other; however, you should prepare written solutions alone. In particular, you should not leave with any written notes from such discussions. The style and clarity of your answers will be an important factor in the grade.

1. We prove (a) by induction on $s = \text{size of the resolution proof (i.e., number of clauses)}$. If $s = 1$, $\mathcal{C} = \{\{\}\}$, so \mathcal{C} is itself a resolution refutation of width $0 = \lceil \log 1 \rceil + w(\mathcal{C})$.

Now consider a tree-like resolution refutation P of \mathcal{C} of size $s > 1$. Suppose that x is the last variable to be resolved, i.e., the last step of P is

$$\frac{\begin{array}{c} \vdots \\ \{x\} \end{array} \quad \begin{array}{c} \vdots \\ \{\bar{x}\} \end{array}}{\{\}} \tag{1}$$

The dots denote the left and right subtrees, respectively. We want to show that we can transform P to have width $w = \lceil \log(s) \rceil + w(\mathcal{C})$.

Note that one of the subtrees of (1) has to be of size $< s/2$. Assume, wlog, that the subtree rooted at \bar{x} is of size $< s/2$, and the other subtree, the one rooted at x is of size $< s$.

Let $\mathcal{C}|_{x=t}$ be the set of clauses \mathcal{C} , with x set to $t \in \{0, 1\}$, and the natural simplifications done (if the assignment makes a literal false, eliminate it from all the clauses that have it, and if it makes a literal true, eliminate all the clauses that have it).

Thus, $\mathcal{C}|_{x=1}$ has the refutation $P|_{x=1}$ given by the right subtree of (1). By the induction hypothesis, $P|_{x=1}$ can be transformed to P' of width

$$\lceil \log(s/2) \rceil + w(\mathcal{C}|_{x=1}) \leq \underbrace{\overbrace{\lceil \log(s) \rceil + w(\mathcal{C}) - 1}^{= (*)}}_{=w} \tag{2}$$

We now add \bar{x} to all the clauses of P' , and we obtain a derivation of $\{\bar{x}\}$ of width $1 + (*) = w$.

Using the left subtree of (1), we obtain a refutation P'' of $\{\}$ from $\mathcal{C}|_{x=0}$ of width at most w (by induction hypothesis; recall that the left subtree of (1) has size $< s$). But we can obtain $\mathcal{C}|_{x=0}$ from \mathcal{C} using P' to eliminate all the x 's from the the clauses of \mathcal{C} which have x , and simply ignoring the clauses that have \bar{x} . This way, we get a refutation of \mathcal{C} of width w .

Part (b) follows immediately from the observation that $sw(\mathcal{C}) \leq \lceil \log(s) \rceil + w(\mathcal{C})$.

2. If $w(\mathcal{C}) = k$, i.e., it is constant, we know from the previous exercise that a tree-like resolution refutation of size s can be transformed to be of width $O(\log(s))$. Given input size n , there can be at most n -many variables, and hence we can form at most $2^{O(\log(s))} \binom{n}{O(\log(s))} = n^{O(\log(s))} = s^{O(\log(n))}$ -many clauses of width at most $O(\log(s))$. Now, being careless with the space, we use breadth-first search to generate all tree-like resolution refutations with leaves labeled by clauses in \mathcal{C} and with clauses of width at most l . When we can no longer generate new clauses, we increase l by 1.

3. Let L be a language in $\mathbf{NP} \cap \mathbf{co-NP}$. For inputs of a given length n , let $\alpha(\vec{p}, \vec{q})$ code the statement “ \vec{q} is a witness that \vec{p} is in L ”, and let $\beta(\vec{p}, \vec{r})$ code the statement “ \vec{r} is a witness that \vec{p} is not in L ”. Note that $\alpha \wedge \beta$ is *unsatisfiable*. Such formulas exist since $\mathbf{NP} \cap \mathbf{co-NP} = \Sigma_1^P \cap \Pi_1^P$.

Since for any pair (α, β) we have a polytime interpolant C , consider the following polysize circuit family $S = \langle S_i \rangle$ deciding L : in input w , $|w| = n$, compute $C(w)$, and accept iff $C(w) = 1$, i.e., iff $\alpha(w, \vec{r})$ is satisfiable. (A different interpolant exists for different input lengths; we don’t know how to compute the interpolant, but we know it is a polytime algorithm, and hence can be simulated with a polysize circuit.)

4. Suppose that V has feasible interpolation and it is also polybounded. Consider the following two formulas: $\alpha(\vec{p}, \vec{q})$ code the statement “ \vec{q} is a satisfying truth assignment for the propositional formula encoded by \vec{p} ”, and let $\beta(\vec{p}, \vec{r})$ code the statement “ \vec{r} codes a V -refutation of \vec{p} ”. Obviously $\alpha \wedge \beta$ is not satisfiable. A polysize interpolant for these formulas would give us a polysize circuit for satisfiability, and hence \mathbf{NP} would be contained in $\mathbf{P/poly}$.

5. For part (a), let p_{ij} be the adjacency matrix of a graph, $i, j \in [n]$. We use \vec{q} to describe a clique of size m . Let q_{ij} , for $i \in [m], j \in [n]$ assert that vertex j is the i -th vertex of the clique. We can state that \vec{p} has an m -clique with the following clauses:

- (a) For each $i \in [m]$ we have the clause $\{q_{i1}, \dots, q_{in}\}$ asserting that some vertex is the i -th vertex.
- (b) For each pair $i \neq i'$ in $[m] \times [m]$, and for each $j \in [n]$, we have the clause $\{\bar{q}_{ij}, \bar{q}_{i'j}\}$ asserting that no vertex is placed twice in the clique.
- (c) For each pair $i \neq i'$ in $[m] \times [m]$, and for each $1 \leq j < j' \leq n$, we have $\{p_{jj'}, \bar{q}_{ij}, \bar{q}_{i'j'}\}$, asserting that if two vertices are in the clique, they must be connected by an edge.

All these clauses taken together (i.e., their conjunction) make up α . Note that \vec{p} occurs only positively in α . Now let r_{ij} state that vertex $i \in [n]$ is in the j -th group of the partition, $j \in [m-1]$. So we need clauses of the form $\{r_{i1}, \dots, r_{i(m-1)}\}$, for $i \in [n]$, asserting that every vertex belongs to some group, and clauses $\{\bar{r}_{ij}, \bar{r}_{i'j}, \bar{p}_{ii'}\}$, for $1 \leq i < i' \leq n$ and $1 \leq j \leq (m-1)$ asserting that any two vertices in the same group are not connected by an edge. The conjunction of these clauses makes up β , and note that \vec{p} occurs only negatively in β . Clearly, a feasible monotone interpolant for $\{\alpha \wedge \beta\}_n$ where $m = \frac{n}{10}$ contradicts Razborov’s theorem.

For part (b), suppose that P is a resolution refutation of $\alpha(\vec{p}, \vec{q}) \cup \beta(\vec{p}, \vec{r})$, where the p_i 's occur only positively in α and only negatively in β .

Let σ be a truth value assignment to \vec{p} . We apply a transformation to P to obtain a DAG $P|_\sigma$ and at the same time construct a feasible monotone interpolant $\mathbf{I}(\sigma)$ (note that once we apply the setting σ , the DAG $P|_\sigma$ can be easily turned into an actual resolution refutation). If C was a clause in the original refutation, it will be denoted C' after applying the procedure, and $\mathbf{I}_C(\sigma)$ will be the interpolant associated with clause C .

We say that a clause C' is an α -clause if it contains variables from among \vec{p}, \vec{q} only, and if it only contains variables from \vec{p} , it is an α -clause if all its ancestors are α -clauses. Similarly, we define a β -clause symmetrically, with \vec{r} and negations of \vec{p} .

We let $\mathbf{I}_C(\sigma)$ be 0 if C' is an α -clause, and 1 if it is a β -clause. Initially, every clause C in α and β stays the same, i.e., $C' = C$, and clauses in α are declared to be α -clauses, and all the clauses in β are declared to be β -clauses. We now describe the rest of the procedure.

Suppose a clause C in P is obtained by:

$$\frac{C_1 \cup \{x\} \quad C_2 \cup \{\bar{x}\}}{C} \quad (3)$$

Assume inductively that we have already transformed the premises, and we have obtained $(C_1 \cup \{x\})', (C_2 \cup \{\bar{x}\})'$, and we have also declared their sides.

The task is to define C' and $\mathbf{I}_C(\sigma)$. We do it separately for x being a variable in one of the three groups: p_i, q_i, r_i .

For example, in the case that $x = p_i$, then you might want to let C' be $(C_1 \cup \{p_i\})'$ if $p_i = 0$, and $(C_2 \cup \{\bar{p}_i\})'$ otherwise. The temptation is now to define

$$\mathbf{I}_C := (p_i \vee \mathbf{I}_{C_1'}) \wedge (\bar{p}_i \vee \mathbf{I}_{C_2'}) \quad (4)$$

This would work if we were not constructing a monotone circuit for the interpolant, but we are, so \bar{p}_i is not possible.

(i) Show how to define C' and \mathbf{I}_C so that we have monotonicity in the case that $x = p_i$. Keep in mind that one of the goals of the definition of C' is to ensure that it is either an α -clause or a β -clause. The second goal is to be able to do question (iii).

To warm up, consider the truth table for \mathbf{I}_C when defined the wrong way as in the paragraph above (by (4)):

p_i	$\mathbf{I}_{C_1 \cup \{p_i\}}$	$\mathbf{I}_{C_2 \cup \{\bar{p}_i\}}$	\mathbf{I}_C
0	0	0	0
1	0	0	0
0	1	0	1
1	1	0	0
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

Note the two “problem” rows, where I_C goes from 1 to 0, despite the fact that p_i changes from 0 to 1. This is the only case where monotonicity is spoiled. But, using the values of \mathbf{I} on the premises, we can make this problem vanish. The point is that we can turn the 1 in the box into a 0, while defining C' consistently. Show how/why. (Note also, that a symmetric answer is possible; turn the 0 below the boxed 1 into a 1.)

Solution: Let $\mathbf{I}_C := (p_i \vee \mathbf{I}_{(C_1 \cup \{p_i\})}) \wedge \mathbf{I}_{(C_2 \cup \{\bar{p}_i\})}$, and we let C' be $(C_1 \cup \{p_i\})'$ or $(C_2 \cup \{\bar{p}_i\})'$ as defined the wrong way, except that if $(C_2 \cup \{\bar{p}_i\})'$ is an α -clause, we let C' be equal to it, regardless of the value of p_i . This works, because if $(C_2 \cup \{\bar{p}_i\})'$ is an α -clause, then by definition it cannot have \bar{p}_i in it (it must have disappeared along the way).

(ii) The cases for $x = q_i$ and $x = r_i$ are simpler than the case for p_i , so you might want to do this question before (i). (Note, if your answer is very complicated, something is not right.)

Solution: For $x = q_i$, let C' be the result of resolving $(C_1 \cup \{q_i\})'$ and $(C_2 \cup \{\bar{q}_i\})'$ on q_i if possible (meaning that one still has q_i and the other \bar{q}_i), and taking C' to be the β -clause, if one of them is (if they both are, say, the left one), and if neither is a β -clause, take the one without the q_i -literal (say, the left one, if they both miss the q_i -literal). Let $\mathbf{I}_C := \mathbf{I}_{C_1 \cup \{q_i\}} \vee \mathbf{I}_{C_2 \cup \{\bar{q}_i\}}$. The case of $x = r_i$ is the dual case to the q_i , but the interpolant is defined with a conjunction.

(iii) Now show that if C' is an α -clause, then $\alpha|_\sigma \models C'|_\sigma$, and same for β -clauses. This can be done with an inductive argument on the depth of the clause.

Solution: This is clearly true for the input clauses. Suppose that we are deriving as in (3), with $x = q_i$. Suppose that $\hat{\sigma}' \supseteq \sigma$ is a truth assignment to all the variables, extending the truth assignment σ to the \bar{p} . Suppose both premises are α -clauses, and $\hat{\sigma}$ satisfies them. If C' is the result of resolution on q_i , then by soundness of the resolution rule it is satisfied by $\hat{\sigma}$. Similar basic arguments show the other cases.

(iv) Define the interpolant for P to be $\mathbf{I}_{\{\}}$, i.e., the value of the interpolant at the final empty clause. Show, using (iii), that it works correctly, it is feasible, and monotone.

Solution: It is monotone because we only use $\{\wedge, \vee, 0, 1\}$ and variables (no negations). It is feasible since the number of connectives is linear in the size of the proof P . It works correctly because it establishes that $\{\}' = \{\}$ is an α - or β -clause, and so by (iii) $\{\}$ follows from α or β clauses, so α or β must be unsatisfiable.