Exercise IX, Computational Complexity 2024

These exercises are for your own benefit. Feel free to collaborate and share your answers with other students. Solve as many problems as you can and ask for help if you get stuck for too long. Problems marked * are more difficult but also more fun :).

Resolution

1 Prove that tree-like Resolution (and hence Resolution, too) is *complete*: If φ is an unsatisfiable CNF, then there exists some tree-like Resolution refutation of φ.

(*Hint: For the easiest proof, use the equivalence between tree-like Resolution and decision trees solving* $\textsc{Search}(\varphi)$.)

Solution: Because φ is unsatisfiable, there exists for each input $x \in \{0,1\}^n$ a falsified clause $C \in \varphi$. Thus, $\textsc{Search}(\varphi)$ is total and there exists a decision tree solving it (e.g. the one that queries everything). This decision tree combined with the equivalence between tree-like resolution and decision trees shows that φ indeed has a tree-like resolution proof (i.e. simply *flip* the decision tree solving $\textsc{Search}(\phi)$).

2 Prove the lemma for *Tree–Adversary games* from the lecture. Namely, prove that if there exists an Adversary strategy for $\textsc{Search}(\varphi)$ that scores at least r points against any Tree strategy, then any decision tree solving $\textsc{Search}(\varphi)$ has size at least 2^r.

(*Hint: Prove the contrapositive. Given a decision tree, consider the Tree strategy that, when Adversary leaves the choice of value of* x_i *to Tree, it chooses the smaller subtree.*)

Solution: Fix a decision tree t solving $\textsc{Search}(\varphi)$ with $< 2^r$ nodes and let us derive a Tree strategy that makes any adversary score at most $r-1$ points. Tree choose what to query next based on t. Whenever the adversary chooses its answer, Tree follows t and recurses on a smaller tree. When the adversary leaves the choice to Tree, Tree selects the smallest sub-tree, which leaves it with a tree of at most half the size. Since t has size $< 2^r$, Adversary can score at most $r-1$ points before Tree reaches a leaf of t and thus a valid solution.

3 Recall that the n-bit $\textsc{Or}_n \colon \{0,1\}^n \to \{0,1\}$ has a decision tree of size $O(n)$. Let us modify $\textsc{Or}_n$ slightly by replacing each of its input variables with a 2-bit $\textsc{And}_2 \colon \{0,1\}^2 \to \{0,1\}$. Namely, denote by $\textsc{Or}_n \circ \textsc{And}_2$ the function that on a $2n$-bit input $(x,y) \in \{0,1\}^n \times \{0,1\}^n$ outputs

$$(\textsc{Or}_n \circ \textsc{And}_2)(x,y) \;\coloneqq\; \textsc{Or}_n(\textsc{And}_2(x_1,y_1),\ldots,\textsc{And}_2(x_n,y_n)).$$

Show that any decision tree for $\textsc{Or}_n \circ \textsc{And}_2$ requires size 2^n.

Solution: Observe that total Boolean function can be seen as search problems where each instance $x \in \{0,1\}$ has a unique solution within $\{0,1\}$. It is thus enough for our purposes to display an Adversary strategy that scores n points in the Tree-Adversary game for $\textsc{Or}_n \circ \textsc{And}_2$. The adversary strategy is simple: whenever the Tree wants to know the value of some bit $b \in \{x_i, y_i\}$, the adversary lets the Tree choose the value of b if none of $\{x_i, y_i\}$ is known and answers 0 else. In that way, the Tree has to explore all pairs to compute the function and thus the Adversary scores n points.

4 The *width* of a Resolution refutation $\pi = (C_1, \ldots, C_s)$ is the maximum width $|C_i|$ of any clause C_i appearing in the proof.

(a) Show that if a CNF formula φ with n variables admits a width-w refutation, then it also admits one of size $s \leq n^{O(w)}$.

(b) Given a formula φ and a width parameter w, show that one can find a width-w refutation of φ (if one exists) in time $n^{O(w)}$.

(This exercise shows that bounded-width Resolution is polynomial-time *automatable*; that is, short proofs can be found efficiently.)

Solution:

(a) Note that there are $\sum_{i=1}^{w}(2n)^i \leq n^{O(w)}$ possible clause of width at most w on n variables. Since a resolution refutation can re-use previously derived clauses, we get that any width-w refutation can be made into a proof of size $n^{O(w)}$ at most. Note that this would not be true for tree-like resolution.

(b) Let S_0 be the set of clauses in φ – the idea of the algorithm is to grow it by using the resolution rule iteratively. More precisely, the set S_{i+1} is composed of S_i in addition to any clause of width at most w that can be obtained by resolving on two clauses from S_i. The process stops when $S_{i+1} = S_i$. Note that there exists a width-w resolution rule if and only if $\emptyset \in S_i$ and that this process lasts for at most $n^{O(w)}$ step, each taking time $n^{O(w)}$. Finally, a width-w resolution refutation (if one exists) can be computed by storing for each $C \in S_i$ a pointer to the two parent clauses that created it and back-tracking from $\emptyset$.

5 Sometimes (for convenience) one allows an additional *weakening rule* in Resolution: From any clause A this rule allows to derive the clause $A \vee B$ where B is an arbitrary clause. Show that allowing this rule does not add power to Resolution: If a CNF formula φ has a size-s refutation in Resolution-with-weakening, then φ also has a size-s refutation in (usual) Resolution.

Solution: Let φ be an unsatisfiable CNF and fix a resolution refutation of φ that uses weakening. The idea is to see the refutation as a DAG and remove the weakening steps from bottom (i.e. the clauses from φ) to the top (i.e the empty clause). Note that each time a weakening is removed, the resulting clause is a subset of the original clause. When the top of the DAG is reached, the resulting clause is a subset of $\emptyset$, i.e. $\emptyset$ itself: the pruned refutation that does not use weakening is thus still valid.