Formal Proofs and their Lengths IV

Refutation systems

Definition 1. We call a poly-time predicate $R(x, y)$ a **refutation system** iff for any propositional formula ϕ there is π_ϕ so that $R(\phi, \pi_\phi)$ accepts iff π is unsatisfiable.

Exercise 2. Show that any refutation system is efficiently transformable into a proof system and vice versa.

Limited extension

Resolution is a refutation system which works with CNFs only. This may seem as a crucial flaw, since there are boolean formulas whose equivalent CNF formulas are of exponential size. The way to overcome this is instead of equivalence consider the *equisatisfiability*.

Definition 3. Two formulas ϕ and ψ are **equisatisfiable** iff ϕ is satisfiable iff ψ is satisfiable.

Exercise 4. How hard is the problem of determining whether two given formulas are equivalent, equisatisfiable?

Theorem 5 (Tseitin)**.** There is a poly-time algorithm L which on input a boolean formula ϕ (in the De Morgan language) produces an equisatisfiable CNF formula $L(\phi)$.

Resolution

From now on we will identify CNF formulas with set of *clauses* C_i, where each C_i is a set of *literals* l_j and a literal is either a variable or its negation.

Definition 6. Given a CNF $\mathcal{C} = \{C_1^{init}, ..., C_m^{init}\}$ a **resolution refutation** of C is a sequence of clauses $C_1, ..., C_k$ so that

- C_k is an empty set,
- each C_j is either an initial clause C_i^{init} or is derived from the previous clauses via the **resolution rule**

$$\frac{C \cup \{p\} \quad C' \cup \{\neg p\}}{C \cup C'}$$

For a general formula ϕ its resolution refutation is defined as a resolution refutation of $L(\phi)$.

Exercise 7. Show that resolution is a sound refutation system, i.e. if ϕ has a resolution refutation, then ϕ is unsatisfiable.

Exercise 8. Express a statement "*there is a linear ordering on 2 elements with no endpoints*" as a CNF. Derive a resolution refutation of this statement.

Exercise 9. Show that resolution is a complete refutation system, i.e. if ϕ is unsatisfiable, then there is a resolution refutation of ϕ.

Exercise 10. Conclude that resolution is a Cook-Reckhow proof system.

Exercise 11. Using resolution show that 2-CNF-SAT is in **P**.

DPLL procedure and $\mathbf{R}^*$

Given a CNF $\mathcal{C}$ we consider the following SAT-solving procedure pictured as labeled binary tree: at the root pick any atom p, label the root by p and consider two arrows leaving it labeled p^1 and p^0, respectively. Going along the arrow p^b, $b \in \{0,1\}$, substitute $p := b$ in all clauses in $\mathcal{C}$. Then pick another variable q to label the node, split the subtree into two and label the arrows q^0 and q^1 as before, etc. Every node v in the tree determines a partial truth assignment α_v by looking at the labels of the arrows leading from the root to v. The procedure stops at node v (i.e. the tree does not branch from v) if α_v falsifies a clause in $\mathcal{C}$.

Exercise 12. Show that the above procedure solves CNF-SAT. What if one applies the procedure to an unsatisfiable CNF?

Exercise 13. Apply the DPLL procedure to the CNF formula from the exercise 8.

We can picture a general resolution refutation as a *directed acyclic graph* (or just DAG) with vertices corresponding to clauses in the refutation and edges connecting a clause with two clauses used to derive it. If mentioned DAG is a tree, then the corresponding refutation is called **tree-like**. A subsystem of resolution with only tree-like refutations is called **tree-like resolution** and is denoted as $\mathbf{R}^*$. A crucial observation is that there is a natural correspondence between $\mathbf{R}^*$ refutations and DPLL computations.

Theorem 14. Let $\mathcal{C}$ be an unsatisfiable CNF and $\tilde{\mathcal{C}}$ contain all clauses which are subclauses of some clauses from $\mathcal{C}$. Then, one can construct an $\mathbf{R}^*$ refutation of $\tilde{\mathcal{C}}$ using a DPLL computation on $\mathcal{C}$. Similarly, an $\mathbf{R}^*$ refutation of $\mathcal{C}$ is transformable into a DPLL computation on $\mathcal{C}$.