Logic Exercises

Technical University of Munich
Chair for Logic and Verification

Prof. Tobias Nipkow
Kevin Kappelmann

SS 2021 | Exercise Sheet 2 | 21.04.2021

Exercise 2.1. [**¡Viva La Resolutión!**]

1. We learnt that resolution is a decision procedure for the unsatisfiability problem of CNF formulas. Explain: what does it mean for an algorithm $\mathcal{A} : U \to \{0, 1\}$ to be a "decision procedure" for a problem class $\mathcal{P} \subseteq U$?
2. Let S be a set of clauses and C be a clause. Does $S \models C$ imply $S \vdash_{\text{Res}} C$? Proof or counterexample!
3. Can you prove $S \models C$ by resolution?

Exercise 2.2. [**Resolution of Horn-Clauses**]
Can the resolvent of two Horn-clauses be a non-Horn clause?

Exercise 2.3. [**The clause is trivial and left as an exercise**]
We call a clause C *trivially true* if $A_i \in C$ and $\neg A_i \in C$ for some atom A_i. Show that the resolution algorithm remains complete if it does not consider trivially true clauses for resolution.

Exercise 2.4. [**Finite Axiomatisation**]
Let S_0 and S be sets of formulas. S_0 is called an *axiom schema* for S if for all assignments $\mathcal{A}$, $\mathcal{A} \models S_0$ iff $\mathcal{A} \models S$.

A set S is called *finitely axiomatisable* iff there is a finite axiom schema for S.

1. Are all sets of formulas finitely axiomatisable? Proof or disprove!
2. Let $S = \{F_i \mid i \in \mathbb{N}\}$ be a set of formulas such that for all i, $F_{i+1} \models F_i$ and $F_i \not\models F_{i+1}$. Is S finitely axiomatisable?

Exercise 2.5. [**What's Semantics Anyway?**]
Discuss: Can you think of other ways to give a semantic interpretation of propositional formulas than the one introduced in the lecture? What makes for a good semantic interpretation? What makes for a good model of a set of axioms?

Homework 2.1. [by auto] (+)
Use the resolution procedure to decide if the following formulas are satisfiable. Show your work (by giving the corresponding DAG or linear derivation)!

1. $(A_1 \vee A_2 \vee \neg A_3) \wedge \neg A_1 \wedge (A_1 \vee A_2 \vee A_3) \wedge (A_1 \vee \neg A_2)$
2. $(\neg A_1 \vee A_2) \wedge (\neg A_2 \vee A_3) \wedge (A_1 \vee \neg A_3) \wedge (A_1 \vee A_2 \vee A_3)$

Homework 2.2. [**Model Extraction**] (+++)
In the lecture, you proved completeness of propositional resolution (if $F \not\vdash_{\text{Res}} \square$ then F is satisfiable) in a way that does not directly give raise to a model of F. In practice, however, it is of course very useful to obtain such a model.

On slide 15 of the Resolution lecture slides, the professor gave an algorithm that iteratively adds new clauses to F until no new clause can be added; in other words, it computes the least fixed point of the resolution rule starting on F. We say that the resulting set of this process is *saturated under resolution*.

Give a constructive method that builds a model $\mathcal{M}$ for F from the saturated set of clauses created by the resolution process. Proof the correctness of your construction.

If you need a hint: you can find the construction without a proof here. Only slides 4, 11–14 and 16 are relevant.

Homework 2.3. [by blast] (+)
Check the following formulas for satisfiability using one of the algorithms seen in the lecture:

1. $(A \vee \neg B \vee \neg D \vee \neg E) \wedge (\neg B \vee C) \wedge B \wedge (\neg C \vee D) \wedge (\neg D \vee E)$
2. $\neg(((A \to B) \wedge (B \to A)) \to (A \leftrightarrow B))$
3. $(A \to E) \wedge (B \to \bot) \wedge (C \to B) \wedge (\top \to A) \wedge (A \wedge B \to C) \wedge (C \to D)$

Show your work! Remember to give a model for satisfiable formulas.

Homework 2.4. [Kőnig's Lemma] (++)

A finitely branching tree has the following structure:

- There is exactly one root node.
- Every node has a finite number of children.

We assign the root node the *level* 0 and the children of a node at level n the level $n+1$. Let T_n denote the set of all nodes at level n, and T the set of all nodes, i.e. $T = \bigcup_{n \in \mathbb{N}} T_n$. Let P_t for $t \in T$ be the set of parent nodes of a node, i.e. t is a child (or grand-child, ...) of all $t' \in P_t$. A path is a sequence of connected nodes, starting from the root node.

Prove the following lemma using the compactness theorem: Every countably infinite, finitely branching tree has an infinite path.

Hint: Use the following template for the proof.

1. Fix a set of tree nodes T. This set is (countably) infinite. You can assume that the sets T_n and the sets P_t are given.
2. For each node $t \in T$, let A_t be an atom. If an assignment $\mathcal{A}$ makes A_t true, the node t is part of the path.
3. Define a set of propositions S that together guarantee the existence of an infinite path. That set is composed of three subsets:
   (a) For each level $n \in \mathbb{N}$, a node $t \in T_n$ is part of the path.
   (b) If a node t is part of the path, so are all of its parent nodes $t' \in P_t$.
   (c) For each level $n \in \mathbb{N}$, there is at most one node of level n part of the path.
4. Show that any finite subset of $S' \subseteq S$ is satisfiable by constructing an assignment such that $\mathcal{A}_{S'} \models S'$. Consider the largest n for which a proposition from subset (a) is contained in S'.
5. Hence, S is satisfiable. Show that a model $\mathcal{A} \models S$ represents an infinite path in T.

Homework 2.5. [Negative Resolution] (++)

We call a clause C *negative* if it only contains negative literals. Show that resolution remains complete if it only resolves two clauses if one of them is negative.

> If you use a trick in logic, whom can you be tricking other than yourself?
> — Ludwig Wittgenstein