Logics Exercise

TU München
Institut für Informatik

Prof. Tobias Nipkow
Lars Hupel

SS 2018 | Exercise Sheet 1 | 10.04.2018

Submission of homework: Before tutorial on 17.04.2018. Until further notice, homework has to be submitted in groups of two students.
You have to achieve a minimum combined score of 50% on your homeworks to be granted a grade bonus on the final exam, provided that the exam is passed.

Exercise 1.1. [**Short Questions**]
Let M be a set of formulas, and let F and G be formulas. Which of the following assertions hold?

1. If F satisfiable then $M \models F$
2. F is valid iff $\top \models F$
3. If $\models F$ then $M \models F$
4. If $M \models F$ then $M \cup \{G\} \models F$
5. $M \models F$ and $M \models \neg F$ cannot hold simultaneously
6. If $M \models G \to F$ and $M \models G$ then $M \models F$

Exercise 1.2. [**Coincidence Lemma**]
Assume that for all atomic formulas A_i in F, $\mathcal{A}(A_i) = \mathcal{A}'(A_i)$. Show that

$$\mathcal{A} \models F \text{ iff } \mathcal{A}' \models F$$

Exercise 1.3. [**Semantic Proof**]
Let $\models F \to G$ where F and G do not share any atoms. Show that then F is unsatisfiable or G is a tautology (or both). *Hint:* you may want to use the previous result.

Homework 1.1. **[CNF and DNF]** (6 points)
Use the rewriting-based procedure from the lecture to convert the following formulas F and G first to NNF, and then to CNF and DNF. Document each rewriting step.

$$F = \neg\neg(\neg A_1 \wedge \neg\neg(A_2 \vee A_3)) \qquad G = (A_1 \vee A_2 \vee A_3) \wedge (\neg A_1 \vee \neg A_2)$$

Homework 1.2. **[Basic equivalences]** (8 points)
Let F and G be formulas. Are the following statements equivalent? Proof or counterexample!

1. $\models F \leftrightarrow G$
2. $F \equiv G$

How about these two statements?

1. F is valid
2. $F \equiv \top$

Homework 1.3. **[Efficient CNF satisfiability check]** (6 points)
In general, solving satisfiability for CNF formula is a hard problem. Consider the special case where clauses may only contain up to two literals. Give an efficient algorithm to check satisfiability.