Satisfiability of Boolean Formulas - Combinatorics and Algorithms
Autumn 2008

Special Exercises

Regulations:

- There will be a total of four special exercise sets during this semester.
- You are expected to solve them carefully and then write a nice and complete exposition of your solutions using LaTeX.
- You are welcome to discuss the tasks with your colleagues, but we expect each of you to hand in your own, individual writeup.
- Your solutions will be graded. The three highest out of your four achieved grades will account for 10% of your final grade for the course each (so 30% of the grade in total).

Special Exercise Set 1

Due date: Friday, October 17, 2008 (at the beginning of the 10 o'clock lecture)

Problem 1 Only double conflicts

Let F be a CNF formula such that every pair of clauses $C, D \in F$ either have no complementary literals, or at least two pairs of complementary literals, but never exactly one, i.e. using the notation

$$\bar{C} := \{ \bar{u} \mid u \in C \}$$

we have

$$\forall C, D \in F : |C \cap \bar{D}| \neq 1.$$

a. Prove that F is satisfiable.

b. Exhibit a polynomial-time algorithm that finds a satisfying assignment for such a formula.

Problem 2 Check-free Chess Board

You are given a list of chess pieces, e.g. [k kings, q queens, r rooks, b bishops, g knights], with $k, q, r, b, g \in \mathbb{N}$. For simplicity, we do not allow any pawns in the list (due to the special capturing rules that apply for them). You would like to position all the pieces on a board of size $n \times n$ in such a way that

(i) no two pieces are on the same field and

(ii) no piece can capture another piece.

Note that we disregard the colours of the pieces in this problem. Finding a solution is not an easy task, but perhaps you can create a CNF formula out of it which is satisfiable if and only if your task has a solution.

a. Give a (high-level) description of how an algorithm could construct a CNF formula for the problem in polynomial time.

b. Argue why your solution is correct and how a satisfying assignment reveals a solution for the original task.

c. Describe the 'metrics' of the formulas your algorithm produces, e.g. the number of variables, the number of clauses and the sizes of the clauses.

Problem 3 NAE-Satisfiability

A CNF formula F is said to be *Not-All-Equal satisfiable*, or *NAE-satisfiable* for short, if there exists an assignment for it such that in every clause, at least one literal evaluates to true and at least one literal evaluates to false.

a. Give a 2-CNF with 2 clauses that is <u>not</u> NAE-satisfiable
(and demonstrate that it really isn't!).

b. Give a 3-CNF with 4 clauses that is <u>not</u> NAE-satisfiable
(and demonstrate that it really isn't!).

c. Show that every k-CNF with <u>less</u> than 2^{k-1} clauses <u>is</u> NAE-satisfiable.

d. Show that for every k, there exists a k-CNF formula with 2^{k-1} clauses which is <u>not</u> NAE-satisfiable.

Problem 4 Derandomizing the Local Lemma

Let F be a k-CNF formula of the form we require in Theorem 2*.1 and $V := \mathrm{vbl}(F)$ its variable set. Let $m := |F|$. Recall the various definitions concerning witness trees in Chapter 2*.

a. Show: for every witness tree T for F we can exhibit a clause C_T over V such that T is consistent with an assignment $\alpha \in \{0,1\}^V$ if and only if C_T is violated. What is the size of such a clause?

b. Let $u > k$ be any fixed number. Prove: if there exists a witness tree T of size <u>at least</u> u for F which expands and is consistent with a given $\alpha \in \{0,1\}^V$, then there exists also a witness tree T' of a size in the range $[u, (k+1)u]$ that expands as well and is equally consistent with α.

HINT: As you would expect, T' is a subtree of T. Use reductio ad absurdum: assume the claim is wrong for some fixed value u and then assume that T is the smallest counterexample to the claim, so the smallest tree larger than u but such that no (expanding, consistent,...) subtree of a size in the range between u and $(k+1)u$ exists. Then derive a contradiction.

c. Prove: there exists a list containing polynomially (in the size of F) many witness trees, each of them being of polynomial size, such that if for a given assignment α, all the witness trees in the list are <u>non-consistent</u> with α, then $F^{[\alpha_p]}$ consists of small components exclusively, where 'small' is defined along the same lines as in the script, i.e. $\mathrm{lcs}(F^{[\alpha_p]}) < c\log(m)$ for some constant c.

HINT: Use b.

d. Prove: the randomized algorithm presented in Chapter 2* can be derandomized, i.e. there is a deterministic polynomial-time algorithm that finds a satisfying assignment to any F which has

$$\forall C \in F : |\Gamma_F(C)| \le 2^{k/2-6}.$$

HINT: Use a, c and at some point Theorem 2.2.