

**Exercise 1.** CNF and satisfiability

The kSAT problem is a *satisfiability* problem. Given a formula  $B$  in conjunctive normal form (CNF), the formula is satisfiable iff there is a truth assignment that makes the formula true.

For this problem, we want to generate CNF forms. One method proposed by J.Random, went as follows.

- Using the proof rules from slide 16 of the lecture on November 13, generate a proof tree.
- The leaves will have the form  $A_1, \dots, A_n \vdash B_1, \dots, B_m$ .
- This sequent is satisfiable iff one of  $A_1, \dots, A_n$  is false, or one of  $B_1, \dots, B_m$  is true. In other words, the sequent is satisfiable iff formula  $(\neg A_1 \vee \dots \vee \neg A_n \vee B_1 \vee \dots \vee B_m)$  is satisfiable.
- Take the conjunction of the formulas for all the leaves.

For example, suppose we want a CNF for  $(A \vee B) \Rightarrow (\neg A \wedge C)$ . Here is a proof tree.

$$\frac{\frac{\frac{A \vdash A, C \quad B \vdash A, C}{A \vee B \vdash A, C}}{A \vee B, \neg A \vdash C}}{A \vee B \vdash \neg A \Rightarrow C}}{\vdash (A \vee B) \Rightarrow (\neg A \Rightarrow C)}$$

We can read off the CNF from the leaves as the formula

$$(\neg A \vee A \vee C) \wedge (\neg B \vee A \vee C)$$

Using this method, give CNF formulas for the following:

1.  $(A \vee \neg A) \Rightarrow B$
2.  $(\neg A \wedge B) \Rightarrow (C \Rightarrow (\neg D \vee E))$
3.  $((A \Rightarrow B) \Rightarrow A) \Rightarrow A$
4.  $(A \Rightarrow (\neg B \wedge \neg C)) \wedge (B \Rightarrow (\neg A \wedge \neg C)) \wedge (C \Rightarrow (\neg A \wedge \neg B)) \wedge (A \vee B \vee C)$ 
  - a. What is the complexity of this method?
  - b. Does it always work?

**Exercise 2.** NP problems

Consider an undirected graph  $G = (V, E)$ , and a finite set  $C$ . A *coloring* is a map  $\varphi : V \rightarrow C$  such that  $\varphi(u) \neq \varphi(v)$  for  $(u, v) \in E$ . Given  $G$  and a constant  $k$ , the *k-coloring problem* is to determine whether there exists a coloring using no more than  $k$  colors.

1. Give an algorithm in NP for the k-coloring problem.
2. Argue informally that a deterministic program would be unable to solve the problem in polynomial time.

**Exercise 3.** Laboratory

The laboratory will be posted separately.