CS 217 – Algorithm Design and Analysis

Shanghai Jiaotong University, Spring 2021

Handed out on Monday, 2021-05-24
First submission and questions due on Monday, 2021-05-31
You will receive feedback from the TA.
Final submission due on Monday, 2021-06-07

8 Zero-Sum Games and Satisfiability

8.1 Guess the Cover

Let $G = (V, E)$ be a graph. Bob chooses a vertex cover $B \subseteq V$. Alice tries to "guess" the set B. Namely, she chooses a set $A \subseteq V$, and Bob has to pay her

$$p(A, B) := \left(\frac{1}{2}\right)^{|A \triangle B|} .$$

Here, $A \triangle B = (A \setminus B) \cup (B \setminus) A$ is the *symmetric difference* of A and B. Note that if Alice guesses B correctly, i.e., she guesses $A = B$, her payoff is 1. Otherwise, she gets "punished" with a factor of $1/2$ for every misclassified vertex.

Exercise 1. Determine the value of this game for G being the triangle.

Exercise 2. Show that under an optimal strategy, Alice only chooses sets A that are vertex covers. That is, if $\mathbf{x}$ is an optimal mixed strategy for Alice and $x_A > 0$ then A is a vertex cover.

Algorithm 1 Determine whether a given k-CNF formula is satisfiable

```
procedure lazyBB(F)
    if F is empty, i.e., already satisfied then
        return satisfiable
    else if F contains an empty clause then
        return unsatisfiable
    else
        choose a shortest clause C = (u_1 ∨ ··· ∨ u_ℓ)
        for i = 1, ..., ℓ do
            F_i := F|_{u_1=···=u_{i-1}=0, u_i=1}
            // set u_i to true and all previous literals to false
            if lazyBB(F_i) = 1 then
                return satisfiable
            end if
        end for
        // at this point, we know that F_1, ..., F_ℓ are all unsatisfiable
        return unsatisfiable
    end if
end procedure
```

8.2 SAT algorithms

Define $L_k(n)$ recursively by

$$L_k(n) := \begin{cases} 0 & \text{if } n < 0 \\ 1 & \text{if } n = 1 \\ L_k(n-1) + \cdots + L_k(n-k) & \text{if } n \geq 1 \end{cases}$$

We have proved in the lecture that the recursion tree of $\textsc{lazy}\mathrm{BB}(F)$ has at most $L_k(n)$ leaves, if F is a k-CNF formula with n variables.

Exercise 3. Show that the equation $a^k = 1 + a + a^2 + \cdots + a^{k-1}$ has exactly one positive solution α_k, and this α_k is at least 1.

Exercise 4. Show that $L_k(n) \in \Theta(\alpha_k^n)$.

Note that algorithm $\textsc{lazy}\mathrm{BB}$ always chooses a *shortest* clause in the current formula. Since we start with a k-CNF formula F, the shortest clause will have always at most k literals.

Exercise 5. Let us call $\textsc{lazy}\mathrm{BB}(F)$ for some k-CNF formula F. Suppose that at some point during the execution of the recursive algorithm $\textsc{lazy}\mathrm{BB}(F)$, a recursive call $\textsc{lazy}\mathrm{BB}(F')$ is invoked, where F' is a formula whose shortest clause has size k. Show that F is satisfiable if and only if F' is satisfiable.

Exercise 6. Extend the idea of the previous exercise to get an improved algorithm: a modified $\textsc{lazy}\mathrm{BB}$ whose running time, on k-CNF formulas, is $\Theta\left((\alpha_{k-1}^n \cdot \mathrm{poly}(|F|)\right)$ instead of $\Theta(\alpha_k^n \cdot \mathrm{poly}|F|)$.

Remark. Note that if $k = 2$ then this gives a running time of $\Theta(\alpha_1^n \cdot \mathrm{poly}(|F|)) = \mathrm{poly}(|F|)$. This is simply the polynomial 2-SAT algorithm we saw in class. The previous two exercises ask you to generalize this technique to larger k.

8.3 Extremal results for k-SAT

We call F an *exactly-k-CNF* formula if every clause has exactly k literals.

Exercise 7. Suppose $F = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ is an exactly-k-CNF formula. Show that if $m < 2^k$ then F is satisfiable. **Remark.** This might be difficult if you don't know the "trick". If you cannot solve it, ask me after the first submission!

The *degree* of a variable x in a CNF formula F is the number of clauses containing x or $\bar{x}$. We denote it by $\deg_F(x)$. The *maximum degree* of a formula is $\Delta(F) := \max_x \deg_F(x)$. For example, if F is

$$(\bar{x} \vee y \vee a) \wedge (\bar{y} \vee z \vee a) \wedge (\bar{z} \vee x \vee a) \wedge (x \vee y \vee z) \wedge (\bar{x} \vee \bar{y} \vee \bar{z}) \wedge$$
$$(\bar{u} \vee v \vee \bar{a}) \wedge (\bar{v} \vee w \vee \bar{a}) \wedge (\bar{w} \vee u \vee \bar{a}) \wedge (u \vee v \vee w) \wedge (\bar{u} \vee \bar{v} \vee \bar{w})$$

then $\deg(x, F) = 4$ and $\deg(a, F) = 6$. Note that F is an exactly-3-CNF formula and is F is unsatisfiable.

Let $F = C_1 \wedge \ldots C_m$ be a CNF formula over the variable set $V = \{x_1, \ldots, x_n\}$. The *incidence graph* is a bipartite graph on the vertex set $\{C_1, \ldots, C_m\} \cup \{x_1, \ldots, x_n\}$ in which we connect clause C to variable x if C contains x or $\bar{x}$.

Exercise 8. Suppose F is an exactly-k-CNF formula and $\Delta(F) \leq k$. Show that the incidence graph has a matching of size m.

Exercise 9. Show that F (from the previous exercise) is satisfiable.