

# CMPT 710 - Complexity Theory: Lecture 23

Valentine Kabanets

November 27, 2007

## 1 Hardness of Approximation

Recall that a Vertex Cover Minimization problem is: Given a graph  $G = (V, E)$  find a smallest-size set  $S \subseteq V$  that covers all the edges of  $G$  (i.e., every edge in  $G$  shares at least one of its endpoints with the set  $S$ ). We know that the decision version of Vertex Cover is NP-complete. Hence, the Minimization version is NP-hard. Assuming that  $P \neq NP$ , this means that there is no polytime algorithm that finds an optimum-size vertex cover in any given graph.

If we can't hope to find a smallest vertex cover, perhaps we can efficiently find a set cover which is "almost" optimal. We say that the Vertex Cover Minimization problem can be efficiently *c-approximated*, for some  $c \geq 1$ , if there is a polytime algorithm that, on any input graph  $G$ , returns a vertex cover  $S$  of  $G$  such that  $|S_{opt}| \leq |S| \leq c|S_{opt}|$ , where  $S_{opt}$  is a smallest-size vertex cover for  $G$ .

For some  $c$ , such an approximation algorithm exists.

**Theorem 1.** *VC is efficiently 2-approximable.*

*Proof.* Let  $G = (V, E)$  be any input graph. Here's the algorithm for finding a vertex cover  $S$  of  $G$ .

Initially set  $S = \emptyset$ . Until there are no edges left, repeat the following: Take any edge  $e = \{u, v\}$ . Add vertices  $u$  and  $v$  to  $S$ . Remove  $e$  from  $G$ , as well as remove all edges that touch  $e$  (at  $u$  or  $v$ ).

Clearly, the described algorithm runs in polytime, and produces a vertex cover  $S$ . Next we argue that  $|S| \leq 2|S_{opt}|$ . To this end, observe that the set  $S$  corresponds to (the endpoints of) a set of mutually disjoint edges of  $G$ . Any vertex cover of  $G$  (including an optimal  $S_{opt}$ ) must cover each of these  $|S|/2$  edges; so every vertex cover must contain at least one vertex for each of these edges. This means that  $|S_{opt}| \geq |S|/2$ , as promised.  $\square$

Now that we have a 2-approximating algorithm for VC, we may ask for a 3/2-approximating algorithm, or more generally, for a  $(1 + \epsilon)$ -approximating algorithm for any  $\epsilon > 0$ . If such an algorithm exists, then it's almost as good as being able to solve VC optimally. Does it exist???

It turns out that the answer is "No, unless  $P = NP$ ". In other words, it is NP-hard to approximate VC to within some constant  $1 < \alpha \leq 2$ . Note that it is hopeless to try to

show *unconditionally* that VC cannot be  $\gamma$ -approximated. This is because if  $P = NP$ , then certainly VC can be 1-approximated. Thus, proving that VC is NP-hard to approximate is the next best thing.

**Remark** This can be viewed as a generalization of NP-hardness from exact optimization problems to approximation problems.

## 2 PCP theorem

The following theorem is equivalent to the PCP Theorem (stated in terms of the PCP verifiers last time).

**Theorem 2** (Arora-Safra, Arora-Lund-Motwani-Sudan-Szegedy). *For some constant  $\gamma < 1$ , there is a polytime reduction  $R$  from SAT to 3SAT such that*

1.  $\phi \in SAT \Rightarrow R(\phi)$  is satisfiable, and
2.  $\phi \notin SAT \Rightarrow$  no assignment satisfies more than the fraction  $\gamma$  of all clauses of the 3-CNF  $R(\phi)$ .

Define MAX3SAT as follows: Given a 3-CNF  $\phi$ , find the maximum value  $0 \leq \alpha \leq 1$  such that at least  $\alpha$  fraction of clauses of  $\phi$  can be satisfied simultaneously. Then the theorem above shows that MAX3SAT is NP-hard to approximate to within any constant factor less than  $\gamma$ .

From the theorem above, we get a PCP system easily: Given  $\phi$ , construct  $\psi = R(\phi)$ . Ask for a satisfying assignment to  $\psi$  (as the PCP proof). Then the verifier picks a random clause of  $\psi$  and queries the given truth assignment in 3 positions corresponding to the variables in the chosen 3-clause. If the clause is satisfied, then the Verifier accepts; otherwise, the Verifier rejects.

It is easy to see that the described PCP verifier is  $(\log n, 1)$ -restricted, as it needs only about  $\log n$  random bits to pick a random clause of  $\psi$  (where  $\psi$  has at most  $n$  clauses), and then reads only 3 bits from the PCP proof.

For the other direction, suppose we have a PCP  $(\log n, 1)$ -restricted verifier for SAT. Given a formula  $\phi$  of size  $n$ , its PCP proof is of size  $m = \text{poly}(n)$ . Think of the bits in the proof as variables  $y_1, \dots, y_m$ . For each choice of randomness of the Verifier, there is a Boolean function (constraint) on constant number of variables that accepts if the Verifier accepts on that randomness after reading the corresponding bits of the proof. Each such constraint can be written as a cnf formula on a constant number of variables. It follows that if  $\phi \in SAT$ , then all constraints are satisfiable simultaneously. On the other hand, if  $\phi \notin SAT$ , then at most a constant fraction of constraints can be simultaneously satisfied. Finally, using the ideas of the Cook-Levin's reduction from SAT to 3SAT, we can rewrite each constraint as a 3cnf, and so get as the final formula  $\psi$  a 3-cnf which is a conjunction of all 3-cnfs for all the constraints.

The above transformation from  $\phi$  to  $\psi$  is the required reduction  $R$  from SAT to 3SAT with the properties as in the Theorem above.