

CS513 Design and Analysis of Algorithms

Fall 2008 - HW8 solution

Instructor: S. Muthukrishnan
 TA: André Madeira

Problem 1:

- (a) Let $U = \cup_i S_i$. Denote by OPT as the size of the set of elements covered by the optimum solution and OPT* as the solution given by the greedy algorithm. We are interested in finding the ratio OPT/OPT*. Recall that the greedy algorithm covers at least OPT/k elements at each iteration. Therefore, after k iterations we have that:

$$\text{OPT} - \text{OPT}^* \leq \text{OPT} \cdot (1 - 1/k)^k \approx \text{OPT}/e,$$

and thus

$$\text{OPT}/\text{OPT}^* \leq (1 - 1/e)^{-1} \approx 1.582.$$

Therefore, the greedy algorithm gives a 1.582-approximation.

- (b) Note that after OPT(ln α) iterations, we have that there can be at most

$$|U| \cdot (1 - 1/\text{OPT})^{\text{OPT}(\ln \alpha)} \approx |U|/e^{\ln \alpha} = |U|/\alpha$$

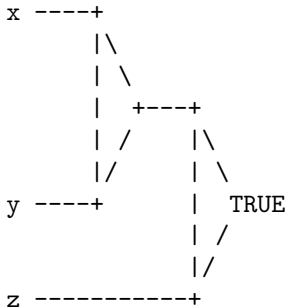
uncovered elements. Since $\alpha \geq |U|/\text{OPT}$ (otherwise it contradicts the definition of OPT), we have that there are at most OPT uncovered elements. These elements need at most another OPT iterations. Therefore the total number of iterations is OPT(ln α) + OPT = OPT(ln α + 1) as desired.

Problem 3:

- (a) Given a formula ϕ of m clauses and n variables x_1, \dots, x_n , we construct a graph $G = (V, E)$ as follows. The set V consists of: a) 2 vertices: one for x_i and one for \bar{x}_i ; b) 5 vertices for each clause (to be defined below); c) and 3 special vertices: TRUE, FALSE, and RED. On the other hand, the set E consists of: a) “literal” edges, 3 per variable forming a triangle between $x_i, \bar{x}_i,$ and RED, $\forall i \in [n]$. Additionally, we add 3 more edges and form a triangle between TRUE, FALSE, and RED; b) “clause” edges, that depend on the clauses as shown below.

We can argue that in any 3 coloring c of G containing these literal edges, exactly one of variable and its negation will be colored with color c(TRUE) and the other colored with color c(FALSE). Indeed, any valid coloring would have to assign two different colors for x_i and \bar{x}_i from c(RED) otherwise it would not be a proper coloring due to the triangle construction achieved by the literal edges. It is also evident that for any truth assignment for ϕ , there is a 3-coloring of G containing just the literal edges (in case there’s a truth assignment with don’t cares, colors for x_i and \bar{x}_i can be picked arbitrarily).

Now, consider the following construction for a clause $x \vee y \vee z$:



We emphasize that each of the 5 vertices “+” is unique for each clause $x \vee y \vee z$. It is not hard to see (check the truth table for example) that if each of x , y , and z is colored $c(\text{TRUE})$ or $c(\text{FALSE})$ (as seen above) then the widget is 3-colorable iff at least one of x , y , or z is colored $c(\text{TRUE})$.

We can conclude that this constructed graph is 3-colorable iff ϕ is satisfiable. Furthermore, note that the instance can be generated in polynomial time as there are only constant numbers of new vertices and edges per variable and clause.

- (b) Perform BFS on G from a starting node s yielding the distances δ_i from s for each node i . Then color all even-distance nodes RED and all odd-distance nodes BLUE (or equivalently set the color for node i : $c_i = \delta_i \bmod 2$). If the graph is not connected, repeat the procedure in all other components. As a final step, check if the coloring is legal (trivial) and return TRUE if so, otherwise return FALSE.

First, observe that if the coloring is legal then the graph is 2-colorable by construction. We then claim that the coloring is only illegal if the graph is not 2-colorable. It is not hard to see that the coloring is only illegal when there's an odd-length cycle in the graph and thus it is not 2-colorable.

Clearly, the total time is polynomial in the input size as BFS and the verification step can be run in linear time (in n and m).