CS612 Assignment 8

Institute of Computing Technology,
Chinese Academy of Sciences, Beijing, China

December 18, 2009

Notice:

1. Due Dec. 31, 2009.
2. Please send your answer to wangchao1987@ict.ac.cn, shaomingfu@gmail.com, yuanxiongying@ict.ac.cn
3. You can arbitrarily choose two problems from Problems 1-5.

1 Approximation Algorithm(10 marks)

Consider the following algorithm for (unweighted) **Vertex Cover**: In each connected component of the input graph execute a depth first search (DFS). Output the nodes that are not the leaves of the DFS tree. Show that the output is indeed a vertex cover, and that it approximates the minimum vertex cover within a factor of 2.

2 Apptoximation Algorithm(10 marks)

Given a graph $G = V, E$ with edge costs and set $T \subseteq V$ of terminal vertices, the $SteinerTreeProblem$ is to find a minimum cost tree in G containing every vertex in T (vertices in $V - T$ may or may not be used in T).

(a)Give a 2-approximation algorithm if the edge costs satisfy the triangle inequality.

(b)Give a 2-approximation algorithm for general edge costs (The graph also need not be complete).

3 Approximation Algorithm(10 marks)

Consider the following maximization version of the 3-Dimensional Matching Problem. Given disjoint sets X, Y, Z, and given a set $T \subseteq X \times Y \times Z$

of ordered triples, a subset $M \subseteq T$ is a $3-dimensional\ matching$ if each element of $X \cup Y \cup Z$ is contained in at most one of these triples. The $Maximum\ 3-Dimensional\ Matching\ Problem$ is to find a 3-dimensional matching M of maximum size. (You may assume $|X| = |Y| = |Z|$ if you want.)

Give a polynomial-time algorithm that finds a 3-dimensional matching of size at least $\frac{1}{3}$ times the maximum possible size.

4 Approximation Algorithm(10 marks)

Consider an optimization version of the Hitting Set Problem defined as follows. We are given a set $A = a_1,\ a_2, ..., a_n$ and a collection $B_1,\ B_2, ..., B_m$ or subsets of A. Also, each element $a_i \in A$ has a *weight* $w_i \geq 0$. The problem is to find a hitting set $H \subseteq A$ such that the total weight of the elements in H, that is, $\Sigma_{a_i \in H} w_i$, is as small as possible. (H is a hitting set if $H \cap B_i$ is not empty for each i.)Let $b = max_i |B_i|$ denote the maximum size of any of the sets $B_1,\ B_2, ..., B_m$. Give a polynomial-time approximation algorithm for this problem that finds a hitting set whose total weight is at most b times the minimum possible.

5 Approximation Algorithm(10 marks)

Recall that in the Knapsack Problem, we have n items, each with a weight w_i and a value v_i. We also have a weight bound W, and the problem is to select a set of items S of highest possible value subject to the condition that the total weight does not exceed W, that is, $\Sigma_{i \in S} w_i \leq W$. Here's one way to look at the approximation algorithm that we designed in this chapter. If we are told there exists a subset ϑ whose total weight is $\Sigma_{i \in \vartheta} w_i \leq W$ and whose total value is $\Sigma_{i \in \vartheta} v_i = V$ for some V, then our approximation algorithm can find a set A with total weight $\Sigma_{i \in A} w_i \leq W$ and total value at least $\Sigma_{i \in A} v_i \geq V/(1+\epsilon)$. Thus the algorithm approximates the best value, while keeping the weights strictly under W.

Now, as is well known, you can always pack a little bit more for a trip just by "sitting on your suitcase", in other words, by slightly overflowing the allowed weight limit. This too suggests a way of formalizing the approximation question for the Knapsack Problem, but it's the following, different, formalization.

Suppose that you are given n items with weights and values, as well as parameters W and V; and you are told that there is a subset ϑ whose total weight is $\Sigma_{i \in \vartheta} w_i \leq W$ and whose total value is $\Sigma_{i \in \vartheta} v_i = V$ for some V. For a given fixed $\epsilon > 0$, design a polynomial-time algorithm that finds a subset of items A such that $\Sigma_{i \in A} w_i \leq (i + \epsilon)W$ and $\Sigma_{i \in A} v_i \geq V$. In other words, you want A to achieve at least as high a total value as the given bound V, but you are allowed to exceed the weight limit W by a factor of $1 + \epsilon$.