CSC373— Algorithm Design, Analysis, and Complexity — Spring 2016

Tutorial Exercise 7: Approximation Algorithms

---

1. **Q6 on the CSC373 Final Exam, April 2012.** Given N items with weights $W = [w_1, w_2, \ldots, w_N]$, we wish to place each of these items into K bins. Multiple items can be placed in the same bin, however each bin has a maximum weight capacity of $C > 0$. So the sum of weights of the items in any single bin cannot exceed C. You can assume that all the quantities above are positive integers, and that $w_n \leq C$ for each n. Define $T = \sum_{n=1}^{N} w_n$ to be the total weight of all the items.

   We wish to minimize the number bins we use, K. Consider the greedy FirstFit algorithm which takes each item from the list $[w_1, w_2, \ldots, w_N]$ in turn (i.e., for n increasing from 1 to N), and places it into the first bin that can fit it, that is, without exceeding that bin's weight capacity.

   For example, consider $C = 10$, and $W = [2, 2, 7, 3, 8, 5, 2, 6, 2]$. Then FirstFit places the first two items in the first bin, the third item is placed in the second bin. The fourth item, with weight 3, is then added to the first bin, since that does not cause its capacity to be exceeded. This process continues, ending up with the bins 1 to 5 each containing a total weight of $[B_1, B_2, \ldots, B_5] = [2+2+3+2, 7+2, 8, 5, 6]$, respectively. Therefore, this FirstFit algorithm uses $K = 5$ bins for this case. (Note an optimal solution requires only $K^* = 4$ bins, e.g., with the first three bins each having a weight of 10.)

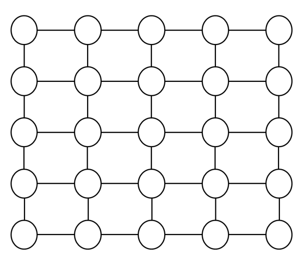
   (a) Define K^* to be the minimum number of bins required. Explain why $K^* \geq \text{ceil}(T/C)$. (Here ceil$(x)$ denotes the minimum integer k such that $k \geq x$.)

   (b) Prove that the FirstFit algorithm leaves at most one bin either half full or less. That is, with B_n denoting the total weight inside the n^{th} bin, after the FirstFit has run, then there is at most one j such that $B_j \in (0, C/2]$. For all other bins k, $1 \leq k \leq K$ and $k \neq j$, we must have $B_k > C/2$. (See the example above.)

   (c) Prove that the number of bins used by the FirstFit algorithm is never more than ceil$(2T/C)$.

   (d) Prove that FirstFit is a 2-approximation.

2. **Q10, Chp 11, Kleinberg and Tardos.** Suppose you are given a weighted graph $G = (V, E, w)$ where G has the form of an $n \times n$ grid graph (see figure below). Assume the weights $w(v)$ are non-negative integers.



   Prof. Jot proposes the following greedy algorithm for obtaining an approximate solution to the maximally weighted independent set problem for this type of graph:

   [S] = wIndSet(V, E, w)
   Initialize $F \leftarrow (V, E)$ and $S \leftarrow \{\ \}$.
   While the graph F is not empty:
       Find a vertex u in F with the largest weight $w(u)$.
       $S \leftarrow S \cup \{u\}$
       Update F by deleting the vertex u and all its neighbouring vertices v (i.e., all vertices v with an edge (u, v) still in F), and delete all the edges ending at any of these deleted vertices.
   End while
   return S

(a) Show that the set S returned by `wIndSet` is an independent set for the graph G.

(b) Show that $w(S) = \sum_{v \in S} w(v)$ is at least $(1/4)w(S^*)$, where S^* is an independent set of G with the maximum possible weight $w(S^*)$.

3. **Modified Q3 Chp 11 Kleiberg and Tardos.** Suppose you are given a list of N integers $L = [a_1, a_2, \ldots, a_N]$, and a positive integer C. The problem is to find a subset $S \subseteq \{1, 2, \ldots, N\}$ such that

$$T(S) = \sum_{i \in S} a_i \leq C, \tag{1}$$

and $T(S)$ is as large as possible.

(a) Consider the decision version of this problem:

Input: A list of positive integers $L = [a_1, a_2, \ldots, a_N]$, and two positive integers $c \leq C$.
Input Size: $|s| = N + b_{max}$, where b_{max} is the maximum number of bits needed to represent c, C or any of the integers a_i (i.e., you can assume the number of bits needed to represent an integer k is bits$(k) = \text{ceil}(\log_2(k)) + 2$).
Problem rPSS (range positive subset sum): Does there exist a subset $S \subseteq \{1, 2, \ldots, N\}$ such that $c \leq T(S) \leq C$ (where $T(S) = \sum_{i \in S} a_i$)?

Sketch a proof that **rPSS** is NP-Complete by making use of a reduction with the NP-complete problem **subsetSum**, described below.

Input: A list of N integers $L = [a_1, a_2, \ldots, a_N]$.
Input Size: $|s| = N + b_{max}$, where b_{max} is the maximum number of bits needed to represent any of the integers a_i.
Problem subsetSum: Does there exist a non-empty subset $S \subseteq \{1, 2, \ldots, N\}$ such that $T(S) = 0$?

(b) Prof. Jot proposes the following greedy algorithm for obtaining an approximate solution to the rPSS problem:

$[S]$ = maxBoundedSetSum($[a_1, \ldots, a_N]$, B)
Initialize $S \leftarrow \{\ \}$, $T = 0$
For $i = 1, 2, \ldots, N$:
    If $T + a_i \leq B$:
        $S \leftarrow S \cup \{i\}$
        $T \leftarrow T + a_i$
End for
return S

Show that Prof. Jot's algorithm is not a ρ-appoximation algorithm for any fixed value ρ.

Note that, since this is a maximization problem, there are two conventions for representing the approximation ratio. Either you can show, for any $\rho > 1$, there exists an example such that $T(S) < (1/\rho)T(S^*)$, or you can switch notation (effectively using $\rho' = 1/\rho$), in which case you need to show that, for any $\rho' \in (0, 1]$, there exists an example such that $T(S) < \rho' T(S^*)$.

(c) Describe a 2-approximation algorithm for this problem (i.e., $\rho = 2 = 1/\rho'$) that runs in $O(N \log(N))$ time.