PRACTICE EXERCISES: BIN-PACKING

THE SOLUTIONS WILL BE AVAILABLE IN 1-3 WEEKS

Hardness and Inapproximability. The objective of this exercise is to prove that the decision version of Bin-Packing is NP-complete and an inapproximabillity result. In the decision version of Bin-Packing we are given a set of items, a positive integer number k, and the goal is to decide wether is possible to pack the items in those k bins of capacity one. The proof comes from a reduction from the NP-complete *Partition problem*,

Partition: Given a sequence $a_1, a_2, \ldots, a_n$ of non-negative integers, decide wether there is a subset $S \subseteq [n]$ such that

$$\sum_{j \in S} a_j = \sum_{j \notin S} a_j.$$

Theorem 1. *The decision version of Bin-Packing is NP-complete.*

Theorem 2. *There is no α-approximation algorithm for Bin-Packing with $\alpha < 3/2$,* **unless** *P=NP.*

(1) Given an instance I_P to Partition, the following instance I_B for Bin-Packing is constructed: there are two bins and for each number a_j there is an item j of size $s_j = 2a_j/A$, where $A = \sum_{j \in [n]} a_j$.

   (a) Prove that if I_P is a YES instance for Partition, then I_B is a YES instance for Bin-Packing.
   (b) Prove that if I_B is a YES partition for Bin-Packing with two bins, then I_P is a YES instance for Partition.
   (c) Conclude Theorem 1.

(2) Suppose that exists $\varepsilon > 0$ and an algorithm that is a $(3/2 - \varepsilon)$-approximation for Bin-Packing. In particular, this algorithm can be run over those instances where the optimal packing uses two bins.

   (a) Given an instance I for the partition problem, construct the same instance for Bin-Packing as before, and use $(3/2 - \varepsilon)$-approximation to decide wether I is a YES instance.
   (b) Conclude Theorem 2 using the fact that the algorithm runs in polynomial time, and that Partition is an NP-complete problem.

FFD algorithm. Given a Bin-Packing instance with n items and sizes $s_1, \ldots, s_n$, we sort them according to non-increasing order. Consider the *First-Fit decreasing* algorithm: a bin $j = 1$ is opened and if item 1 fits in this bin then it is packed in it. We continue with item 2, if it fits into the bin $j = 1$ then it is packed there, and if not then a new bin $j = 2$ is opened. In general, given an item i, it is packed into the first bin where it is possible to pack, and if not then a new bin is opened. Prove that this algorithm returns a packing using at most $3/2 \cdot \mathrm{opt} + 1$ bins.