CS:3330 Homework 5 Solution, Spring 2017

1(a) Consider an input with three intervals, A, B, and C. Suppose that B is a short interval (say, 1 unit long) and A and C are long intervals (say, each 2 units long). Also, suppose that A starts first, then B starts, then A ends, then C starts, then B ends, and finally C ends. Thus B overlaps with A and C, but A and C are non-overlapping with each other. The "shortest interval first" algorithm outputs $\{B\}$, whereas the optimal solution is $\{A, C\}$.

1(b) Suppose that $|O| = t$ and the intervals in O are labeled $x_1, x_2, \ldots, x_t$ in left-to-right order. To obtain a contradiction, we suppose that there is an interval $y \in A$ such that y overlaps with 3 or more intervals in O. Call the intervals that y overlaps: $x_i, x_{i+1}, \ldots, x_{i+p}$, where $p \geq 2$. Since y overlaps x_i and x_{i+2}, the interval x_{i+1} starts after the start time of y and ends before the end time of y. Thus x_{i+1} is strictly shorter than y. The question then is why did the "shortest Interval first" algorithm not pick x_{i+1} instead of y. The only reason for not picking x_{i+1} is that the algorithm picked an interval x' even shorter than x_{i+1} and x' overlapped with x_{i+1} and eliminated it. But, any interval x' that overlaps with x_{i+1} will also overlap with y and eliminate it. Thus y cannot be in A – a contradiction.

1(c) (i) Each interval in A is charged at most 2 dollars. (ii) Thus the total number of dollars charged is *at most* $2|A|$. We already know that the total number of dollars charged is $|O|$. Therefore, $|O| \leq 2|A|$ and equivalently $|A| \geq 1/2 \cdot |O|$. (iii) This tells us that the "shortest interval first" algorithm always produces a solution whose size is at least 1/2 the size of an optimal solution. Therefore, this algorithm is a 1/2-approximation.

2(a) Suppose that are two bins B_i and B_j, $j > i$, that are both more than half empty. Then B_j contains an item of size strictly less than 0.5. When this item was processed, B_i had enough space for it and the item would have been placed in B_i. Hence, it cannot be the case that both B_i and B_j are more than half empty.

Suppose that the First Fit algorithm uses t bins. We know from the above argument that at least $t - 1$ of these are at least half full and therefore the total size of the all items in the input is at least $(t - 1)/2$.

2(b) Suppose that an optimal bin packing uses b^* bins and suppose that the First Fit algorithm uses t bins. By the argument in (a) we know that the total input size is at least $(t - 1)/2$ and since eash bin has size 1 unit, $b^* \geq (t - 1)/2$. Hence, $t \leq 2b^* + 1$, implying that the First Fit algorithm uses at most $2b^* + 1$ bins.

3(a) For this problem $m = C$, $p = 1/4$ and so the expected value of the variable *count* is $C/4$. Since $n = 10^6$, the algorithm is expected to return $10^6 \cdot (C/4) \cdot (1/C) = 10^6/4$.

3(b) We would like the algorithm to return a value in the range

$$[n/4 - (1/10) \cdot n/4, n/4 + (1/10) \cdot n/4] = [(9/10) \cdot n/4, (11/10) \cdot n/4].$$

For this to happen, the variable *count* needs to be in the range $[(9/10) \cdot (C/4), (11/10) \cdot (C/4)]$. Let L denote $(9/10) \cdot (C/4)$ and let U denote $(11/10) \cdot (C/4)$. Using the expression for the binomial distribution, we see that the probability that *count* is in this range is

$$\sum_{k=L}^{U} \binom{C}{k} \cdot (1/4)^k \cdot (3/4)^{C-k}.$$

I did not implement this formula and produce results for $C = 100, 200, \ldots, 1000$.
