CS 1510: Algorithm Design
Homework 5 - Greedy Problem 10
Zach Sadler, John Hofrichter
zps6@pitt.edu — jmh162@pitt.edu
September 8, 2013

Problem 10a

We will disprove that the SJF algorithm is correct. Consider the input

$$J_1 = (0, 12)$$
$$J_2 = (4, 10)$$

When given this input, the algorithm will run J_1 until $t = 4$, then compare the two jobs. Since $x_1 = 12 > 10 = x_2$, the algorithm will switch and run J_2 until $t = 14$, then finish J_1 at $t = 22$. Thus the total completion time is $14 + 22 = 36$.
However, this is not the minimum completion time. Instead, one could run J_1 from $t = 0$ until $t = 12$, then J_2 from $t = 12$ until $t = 22$. In this case, the total completion time would be $12 + 22 = 34$.
Since $34 < 36$, the algorithm does not produce optimal output.

Problem 10b

We will prove that the algorithm SRPT (which we will call G) is correct.
Suppose towards contradiction that there is an input $I = J_1, J_2, \cdots, J_n$ (with each $J_i = (r_i, x_i)$ specifying the release time and size of the job) on which G produces unacceptable output $G(I)$. Then by definition $G(I)$ is output which does not give the minimum total completion time.
Now define $Opt(I)$ to be an optimal output that agrees with $G(I)$ for more terms than any other optimal output. That is, beginning at time $t = 0$ and until time $t = k$, both $G(I)$ and $Opt(I)$ run the exact same jobs at each time interval. However, from time $(k, k + 1)$, G runs J_k and Opt runs J_a where $J_k \neq J_a$.
We know by definition of G that

$$x_k - y_{k,k} \leq x_a - y_{a,k}$$

that is, that J_k has no more remaining time than J_a.
Now we will construct an output $Opt'(I)$ which agrees with $G(I)$ for one more term than $Opt(I)$ and is still optimal. To do this, we will consider all future instances where job J_k is run in $Opt(I)$ and label these instances $J_{k_1}, J_{k_2}, \cdots, J_{k_p}$. Similarly, we will label all instances where job J_a is run in $Opt(I)$ as $J_{a_1}, J_{a_2}, \cdots, J_{a_q}$. We know that $p \leq q$ since we have shown J_k has no more remaining time than J_a, and so there are less unit intervals where J_a is run than J_k.
Now we will begin swapping interval for interval segments of J_k and J_a. That is, we will replace J_{a_1} with J_{k_1}, J_{a_2} with J_{k_2}, and so on. Since $p \leq q$ we know that we may run out of instances of J_k to replace segments of J_a with. If this happens, we will simply begin copying in the remaining instances of J_a.
Note that we know that both J_k, J_a have been released since both algorithms have been identical to this point (the jobs have not already been completed), and the algorithms were able to select from already-released jobs and chose those two.
In the case where $p = q$, that is, J_k and J_a both had equal remaining time, we have simply swapped their positions in the schedule. Since they both had equal remaining time, doing this swap will not change the total completion time (it will increase the completion time of J_a by some amount of units (say m) and decrease the completion time of job J_k by the same number of units, thus resulting in $\delta = (+m) - m = 0$ no net change).
In the case where $p < q$, we will swap the locations of the first p terms of J_a and J_k. This will complete J_k's runtime, granting it a smaller completion time in Opt' than J_a had in Opt. Then we will copy the remaining $q - p$ terms of J_a into the remaining space left over since J_k does not occupy the same space as J_a. Thus J_a will finish in Opt' at the same time that J_k finished in Opt.
Because J_k completes at an earlier time in Opt' than J_a did in Opt, and J_a completes at the same time in Opt' that J_k did in Opt, and no other jobs are affected since we simply swapped the two in-place, we know that the total completion time of Opt' is less than the total completion time of Opt.
Thus if either $p = q$ or $p < q$, we know that Opt' has a total completion time which is no worse than Opt. so if Opt' is optimal if Opt is optimal. We also know that Opt' agrees with G for at least one more term than Opt, since G ran J_k at time k and Opt ran J_a. Thus we have a contradiction that Opt is an optimal output which agrees with G for more terms than any other optimal output and we are done.