Jonathan Asprilla Saavedra - Homework 3

Exercise 4.1:

Optimal strategies: c, h, i.

(a) Choose the course x that ends last, discard classes that conflict with x, and recurse.

This doesn't work. The greedy algorithm chooses the single long course, but the optimal schedule contains all the short
courses.

(b) Choose the course x that starts first, discard all classes that conflict with x, and recurse.

This doesn't work. The greedy algorithm chooses the single long course, but the optimal schedule contains all the short
courses.

(c) Choose the course x that starts last, discard all classes that conflict with x, and recurse.

There is an optimal schedule that includes the course that starts last.
Proof: Let x be the course that starts last. Let S be any schedule that does not contain x, and let y be the last course in S. Because x starts last, we have $S[y]<S[x]$ Thus $F[i]<S[y]<S[x]$ for every other class i in S, which implies that $S'=S-y+x$ is still a valid schedule, containing the same number of classes as S. In particular, if S is an optimal schedule, then S' is an optimal schedule containing x.

(d) Choose the course x with shortest duration, discard all classes that conflict with x, and recurse.

This greedy algorithm chooses the single course in the middle, but the optimal schedule contains the other two courses.

(e) Choose a course x that conflicts with the fewest other courses, discard all classes that conflict with x, and recurse.

This greedy algorithm would choose the course in the center, which has only two conflicts, and thus
would return a schedule containing only three courses. But the optimal schedule
contains four courses.

(f) If no classes conflict, choose them all. Otherwise, discard the course with longest duration and recurse.

This algorithm chooses the single interval in the middle, but the optimal schedule contains the other two intervals.

(g) If no classes conflict, choose them all. Otherwise, discard a course that conflicts with the most other courses and recurse.

This one would discard one of the courses in the middle of the bottom row, which has only five conflicts, and thus would return a schedule containing only three courses. But the optimal schedule contains four courses.

(h) Let x be the class with the earliest start time, and let y be the class with the second earliest start time.

Claim 1. If x and y are disjoint, then every optimal schedule contains x.
Proof: If x and y are disjoint, then $F[x]<S[y]$, which implies that $F[x]<S[i]$ for all $i \neq x$. Thus, if S is any valid schedule that does not contain x, then $S+x$ is a larger valid schedule. Thus, no optimal schedule excludes x.

Claim 2. If x contains y, there is an optimal schedule that excludes x
Proof: If x contains y, then every class that conflicts with y also conflicts with x Thus, for any valid schedule S that contains x, there is another valid schedule $S-x+y$ of the same size that excludes x

Claim 3. If x and y overlap, but x does not contain y, there is an optimal schedule that excludes y
Proof: Suppose x and y overlap, but x does not contain y. Then x must end before y ends, and therefore every class that conflicts with x also conflicts with y. Thus, for any valid schedule S that contains y, there is another valid schedule $S-y+x$ of the same size that excludes y.

(i) If any course x completely contains another course, discard x and recurse. Otherwise, choose the course y that ends last, discard all classes that conflict with y, and recurse.

Claim 1. If any course x contains another course y, there is an optimal schedule that does not include x.
Proof: Let S be any valid schedule that contains x. Then $S-x+y$ is another valid schedule of the same size.

Claim 2. Suppose no course contains any other course. Then there is an optimal schedule containing the course that ends last.
Proof: If no course contains any other course, then the class that ends last is also the class that starts last.

Exercise 4.13:

An optimal algorithm is to schedule the jobs in decreasing order of wi/ti. We prove the optimality of this algorithm by an exchange argument.
Thus, consider any other schedule. As is standard in exchange arguments, we observe that this schedule must contain an inversion, a pair of jobs i, j for which i comes before j in the alternate solution, and j comes before i in the greedy solution. There must be an adjacent such pair i, j. Note that for this pair, we have wj/tj >= wi/ti, by the definition of the greedy schedule. If we can show that swapping this pair i, j doesn't increase the weighted sum of completion times, then we can iteratively do this until there are no more inversions, arriving at the greedy schedule without having increased the function we're trying to minimize. It will then follow that the greedy algorithm is optimal.
So consider the effect of swapping i and j. The completion times of all other jobs remain the same. Suppose the completion time of the job before i and j is C.