

Greedy Algs

- make an optimal local decision at each step
- easy to design, but often incorrect

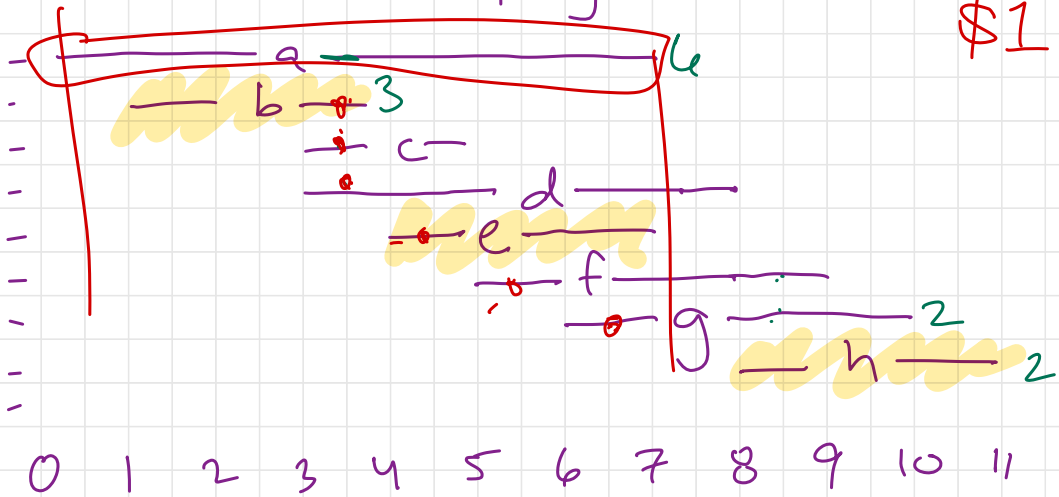
ex Shortest paths

- bad greedy: choosing shortest edge
deleting heaviest edge
- good greedy: Dijkstra's
choose the node
that optimizes shortest total
distance from any explored
node.

Flat Rate

- on your shift, you are given a set of possible jobs to complete.
- jobs have fixed start and end times
- jobs may take diff. amounts of

time, but all pay the same amount \$1



- you can only take jobs that don't overlap (compatible)

how much can you make? \$3

with which set of jobs? {b, e, h}

In general looking for largest set of jobs that are compatible

Greedy Schedule (n jobs, represented by array s for start times f for finish times):

Sort jobs by criterion

$G = \emptyset$

while there is a compatible job w/ G

later in the sorting:
add ^{that} job to G.

Consider 4 criteria:


- ① earliest start time \leftarrow
- ② shortest job length \leftarrow
- ③ # of conflicts
- ④ earliest finish time

for each, come up w/ either:


- a counter example
- an argument that the criterion yields a valid greedy alg.

Minimal counterexamples

earliest start time

 opt = 2
EST chooses 1

shortest job length

 opt = 2
SJL choose 1

conflicts

— 1

— 2

— 2

— 2

— 1