Want to schedule classes

All classes happen on Monday.

Cannot overlap class durations.

Want to schedule max # classes.

Formally: Given arrays

$S[1..n]$ of start times

$F[1..n]$ of finish times.

$0 \leq S[i] < F[i]$ for all i.

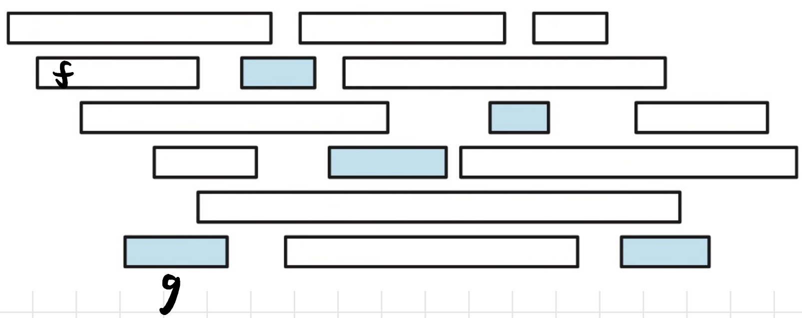
Want maximal conflict-free schedule X of max size.

$X \subseteq \{1, 2, \ldots, n\}$

For each $i, j \in X$ $(i \neq j)$

either $S[i] > F[j]$ or

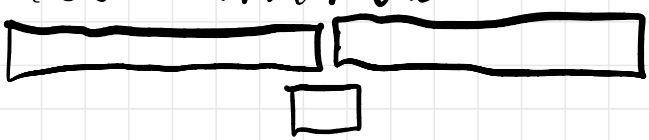
$S[j] > F[i]$.



Dynamic Programming $O(n^3)$ or $O(n^2)$

Greedy alg:

shortest interval?



Lemma: At least one maximal conflict-free schedule includes the class that finishes first.

Proof: Let f finish first.

Let X be some maximal conflict-free schedule.

If $f \in X$, we're done.

Otherwise, let g be the member of X that finishes first.

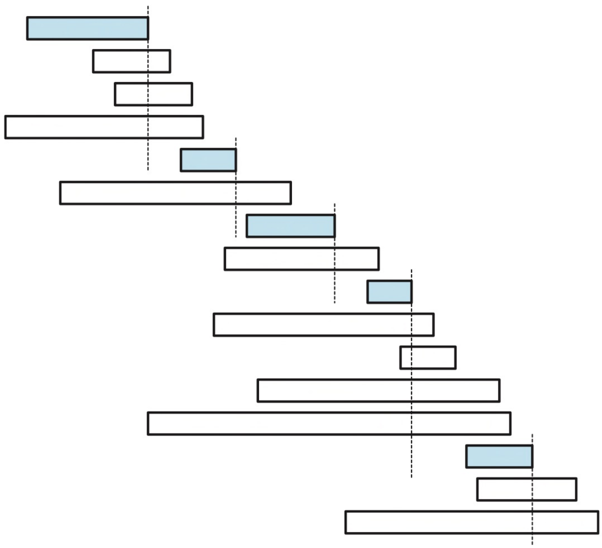
f finishes before g which finishes

before everything else in X, so $X' := X - g + f$ is conflict free.

Also $|X'| = |X|$, so X' is maximal conflict-free. $\square$

Algorithm: Take class that finishes first. Recurse on subset that don't conflict.

```
GREEDYSCHEDULE(S[1..n], F[1..n]):
    sort F and permute S to match
    count ← 1
    X[count] ← 1
    for i ← 2 to n
        if S[i] > F[X[count]]
            count ← count + 1
            X[count] ← i
    return X[1..count]
```



Time: $O(n \log n)$ (we have to sort)

Greedy Algorithm

- backtracking without backtracking

proof by <u>exchange argument</u>

want to argue <u>some</u> optimal solution agrees on your first choice

1) Start with some optimal solution X. If X uses your first choice, great!
2) o.w. do an exchange

so you get another solution x' that **does** use you choice

3) Argue x' is as good as x.

Max Ind Set on Trees

A tree T has ≥ 1 leaves.

Say u is a leaf of T.

If u in max ind set, parent is not, but that is only restricition on $T \setminus \{u\}$.

General problem:

Given rooted tree T with same nodes marked <u>unusable</u>. An independent set is <u>restricted</u> if it contains no unusable nodes. What is the max restricted independent set?

Lemma: Let u be an arbitrary leaf of T. There exists a max usable restricted ind set containing u.

Proof: Let S be any max restricted ind. set of T.

If $u \in S$, we're done.

o.w. if u's parent is not in S, we can add u to get a bigger ind set.

o.w.o.w. let v be u's parent.

$S' := S - v + u$ is a max res.

ind set.

Recursive alg: Take any leaf u. If u is usable, include in output, mark its parent as unusable, & recurse on $T \setminus \{u\}$.

If u is unusable, recurse on $T \setminus \{u\}$.

Full alg for regular max ind set:

Maintain two booleans for each node.

is the node unusable

is the node in our output?

For each node u in postorder

if u is usable

mark it for output

mark parent as unusable

else

do nothing

$O(n)$ time

(use dynamic programming if nodes have weights)