Interval Scheduling

Most basic version:

Given a set of $N$ intervals:

Each defined by a start time $s_i$ and finish time $f_i$.

(think: people that want to rent your house).

What is the maximum number of disjoint intervals you can pick?

That is, find

$$S \subseteq \{1, \ldots, n\}$$

maximizing $|S|$ such that

$$[s_i, f_i) \cap [s_j, f_j) = \emptyset \quad \forall i, j \in S, i \neq j$$

The example has 4.
Many plausible greedy algorithms.

Shortest first?

Interval w/ fewest overlaps first?

First come first serve?

Earliest finish time?

Yes

but needs a proof to show it's different from the above fallacious arguments.
GreedySchedule (I): \[ I = \mathbb{E}(S, f)_{i \in [n]} \]

Sort \( I \) by increasing \( f_i \).
\[ S = [] \]
\[ t = -\infty \]

For \((S, f)\) in \( I \):
   if \( S \leq t \): continue
   \[ t = f \]
   \[ S.\text{append}((S, f)) \]

Return \( S \).

**Theorem.** GreedySchedule returns a valid set of intervals of maximum size.

**Proof.** Because \( S \leq f \) for every interval, \( t \) never decreases. Therefore \( t \) is always the largest \( f \) in \( S \). This ensures each new interval added to \( S \) does not overlap with any interval already in \( S \), so the returned \( S \) is valid.

We now prove by induction on \( n \) that the returned \( S \) has maximal size.

The base case of \( n = 0 \) is trivial. Now consider some \( n > 1 \), and assume the theorem holds for all \( I \) with \( |I| \leq n-1 \).

For any time \( t \), define \( I_t = \mathbb{E}(S, f) \in I \mid S \leq t \).
Notice that, if \((S_i, f_i)\) has the earliest finish time in \( I \), then

\[ \text{GreedySchedule (I)} = [(S, f)] + \text{GreedySchedule (I_t)} \]
Let $S$ be a maximum disjoint subset of $I_f$ which we can write out in order as

$$S = (s_1, f_1), (s_2, f_2), \ldots, (s_k, f_k)$$

For $k = |S|$. Similarly, Greedy Schedule returns a $S$ with

$$S = (s_1, f_1), (s_2, f_2), \ldots, (s_k, f_k)$$

For $k = |S|$. We claim $k = k^*$. Since $f_1$ is the earliest possible finish time,

$$f_1 \leq f_i \leq s_i^*$$

Therefore, $(s_2, f_2), \ldots, (s_k, f_k)$ all lie in $I_{f_1}$. Since these are disjoint, the optimal solution for $I_{f_1}$ has at least $k_i - 1$ intervals. By the inductive hypothesis, this means

$$|\text{Greedy Schedule}(I_{f_1})| \geq k^*_i - 1$$

So $k = |\text{Greedy Schedule}(I)| \geq k^*$.

Since $k^*$ is the maximum possible, $k = k^*$ as desired.

So: $O(n \log n)$ sort + $O(n)$ time.
**Weighted interval Scheduling**

Each interval now has a weight \( w_i \).

Maximize \( \sum_{i \in S} w_i \)

over sets of disjoint intervals.

**Naive recursion:**

\[
\text{Sched}(I) = \begin{cases} 
\text{first elt of } I & \text{not chosen:} \\
\text{min of} & \text{chosen:} \\
\text{Sched}(I \setminus i) + w_i & \text{Sched}(I \setminus \frac{i}{2}) \text{ or anything conflicting with } i \frac{1}{3} + w_i
\end{cases}
\]

**Problem:** \( 2^n \) possible inputs.

**Solution:** If I sorted by \( f_3 \), then only \( n+1 \) inputs ever happen.
(suffix of \( I \) sorted by \( f_1 \))

\( \Rightarrow \) memoized time is \( n \cdot (\text{time per input}) \)

\( n^2 \) naively

\( n \) more carefully

\( \text{Sched (index in } I, f \text{ or last chosen) } \)