Interval Scheduling Recap

“sharing a single resource”

Input:
- $n$ jobs
- one machine
- requests: job $i$ needs machine between times $s(i)$ and $f(i)$

Goal: schedule to maximize # of jobs scheduled.

Algorithm: Greedy by Min. Finish Time

1. $S = \emptyset$
2. Sort jobs by increasing finish time.
3. For each job $j$ (in sorted order):
   - if $j$ if compatible with $S$
     schedule $j$: $S \leftarrow S \cup \{j\}$
   - else discard $j$

Correctness

“schedule is compatible and optimal”

Lemma 1: schedule of algorithm is compatible

Proof: (by induction, straightforward)
• let $i_1, \ldots, i_k$ be jobs scheduled by greedy
• let $j_1, \ldots, j_m$ be jobs scheduled by OPT

Goal: show $k = m$.

Approach: “Greedy Stays Ahead”

Lemma 2: for $r \leq k$, $f(i_r) \leq f(j_r)$

Proof: (induction on $r$)

base case: $r = 0$

• add dummy job 0 with $s(0) = f(0) = -\infty$
• only change: OPT and GREEDY schedule dummy
• so $f(i_0) = f(j_0)$

inductive hypothesis: $f(i_r) \leq f(j_r)$

inductive step:

• Let $I = \{\text{jobs starting after } f(i_r)\}$
  $J = \{\text{jobs starting after } f(j_r)\}$
• IH $\Rightarrow J \subseteq I$
• Alg $\Rightarrow f(i_{r+1}) = \min_{j \in I} f(j)$
  $\leq \min_{j \in J} f(j)$
  $\leq f(j_{r+1})$.

Theorem: Greedy alg. is optimal

Proof: (by contradiction)

• OPT has job $j_{k+1}$ but greedy terminates at $k$.
• lemma 2 (with $r = k$)
  $\Rightarrow f(i_k) \leq f(j_k)$ (1)
• $j_{k+1}$ is compatible with $j_k$
Greedy by Value

“to pick a feasible set with maximum total value”

**Example 1**: weighted interval scheduling

“if jobs have values”

**input:**
- $n$ jobs $J = \{1, \ldots, n\}$
- $s_i = \text{start time of job } i$
- $f_i = \text{finish time of job } i$
- $v_i = \text{value of job } i$

**output**: Schedule $S \subseteq J$ of compatible jobs with maximum total value.

**Question**: does greedy by finish time work?

**Answer**: no

```
1 1 1 1
----- ---- ---- ----
2
```

**Algorithm**: Greedy-by-Value

1. $S = \emptyset$
2. Sort els by decreasing value.
3. For each elt $e$ (in sorted order):
   - if $\{e\} \cup S$ is feasible
     - add $e$ to $S$
   - else discard $e$.

**Question**: does greedy by value work?

**Answer**: no

```
   1
  ----
     2
```

**Example 2**: minimum spanning tree

“maintaining minimal connectivity in a network, e.g., for broadcast”

**input:**
- graph $G = (V, E)$
- costs $c(e)$ on edges $e \in E$

**output**: spanning tree with minimum total cost.

**Def**: a spanning tree of a graph $G = (V, E)$ is $T \subseteq E$ s.t.

(a) $(V, T)$ is connected.
(b) $(V, T)$ is acyclic.

**Note**: Greedy-by-Value = Kruskal’s Alg

**Example**:

```
1 --- 5 2 --- 6 --- 3
   
4
```

**Runtime**

$\Theta(m \log n)$

- $\Theta(m \log n)$ to sort.
- check connectivity with union-find data structure
  amortized $O(\log^* n)$ runtime per operation.
  (recall $\ell = \log^* n \Leftrightarrow n = \frac{2^{2^{2^{\cdots}}}}{\ell \text{ times}}$)
  total $O(m \log^* n)$ runtime.
Correctness

“output is tree and has minimum cost”

**Lemma 1:** Greedy outputs a forest.

**Proof:** Induction.

**Lemma 2:** if $G$ is connected, Greedy outputs a tree.

**Proof:** (by contradiction)

**Theorem:** Greedy-by-Value is optimal for MSTs

**Proof:** (by contradiction)

- Greedy and OPT have $n - 1$ edges (Fact 1)
- Let $I = \{i_1, \ldots, i_{n-1}\}$ be elt’s of Greedy. (in order)
- Let $J = \{j_1, \ldots, j_{n-1}\}$ be elt’s of OPT. (in order)
- Assume for contradiction: $c(I) > c(J)$
- Let $r$ be first index with $c(j_r) < c(i_r)$
- Let $I_{r-1} = \{i_1, \ldots, i_{r-1}\}$
- Let $J_r = \{j_1, \ldots, j_r\}$
- $|I_{r-1}| < |J_r|$ & Augmentation Lemma
  \[\Rightarrow\text{ exists } j \in J_r \setminus I_{r-1}\]
  such that $I_{r-1} \cup \{j\}$ is acyclic.
- Suppose $j$ considered after $i_k$ ($k \leq r - 1$)

$\Rightarrow I_k \subseteq I_{r-1}$

$\Rightarrow I_k \cup \{j\} \subseteq I_{r-1} \cup \{j\}$

$I_{r-1} \cup \{j\}$ acyclic & Fact 2

$\Rightarrow$ all subsets are acyclic

$\Rightarrow I_k \cup \{j\}$ acyclic

$\Rightarrow j$ should have been added.
**Structural Observations about MSTs**

**Def:** $G' = (V, E')$ is a **subgraph** of $G = (V, E)$ if $E' \subseteq E$.

**Def:** An acyclic undirected graph is a **forest**

**Def:** $A, B \subseteq V$ is a **cut** if $A \cup B = \emptyset$ and $A \cap B = E$. Edge $e = (u, v)$ **crosses cut** if $u \in A$ and $v \in B$ (or vice versa).

**Fact 1:** an MST on $n$ vertices has $n - 1$ edges.

**Lemma 1:** If $G = (V, F)$ is a forest with $m$ edges then it has $n - m$ connected components.

**Proof:** Induction (on number of edges)

base case: 0 edges, $n$ CCs.

IH: assume true for $m$.

IS: show true for $m + 1$

- IH $\Rightarrow n - m$ CCs
- add new edge.
- must not create cycle

$\Rightarrow$ connects two connected components.

$\Rightarrow$ these 2 CCs become 1 CC.

$\Rightarrow$ $n - m - 1$ CCs.

QED

**Lemma 2:** (Augmentation Lemma) If $I, J \subset E$ are forests and $|I| < |J|$ then exists $e \in J \setminus I$ such that $I \cup \{e\}$ is a forest.

**Proof:**

Lemma 1