

Greedy Methods

EASY KNAPSACK

Instance: n items where item i has weight w_i and cost c_i and an integer W .

Problem: Determine a fraction f_i of item i , so that

$$\sum_{i=1}^n f_i \cdot w_i \leq W \text{ and } \sum_{i=1}^n f_i \cdot c_i \text{ is maximum.}$$

Greedy Solution: Take as much of each item as possible in order of decreasing cost/weight ratio.

1

Assume $w_i > 0$ otherwise item should always be included if $c_i > 0$.

Order the items by non-increasing $\frac{c_i}{w_i}$.

```

C ← 0; R ← W; i ← 1
while (i ≤ n and R ≥ w[i])
    f[i] ← 1
    R ← R - w[i]
    C ← C + c[i]
    i ← i + 1
endwhile
If i ≤ n and R > 0 then f[i] ← R/w[i]
    C ← C + f[i] · c[i]
    
```

2

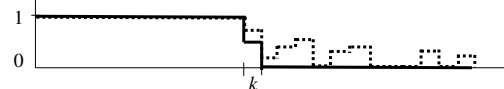
The greedy solution has the form:

$$(1, 1, 1, 1, \dots, 1, 1, g_k, 0, 0, 0, \dots, 0, 0) \text{ where } 0 < g_k \leq 1.$$

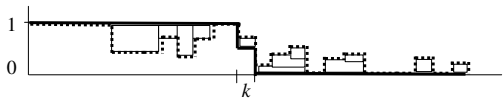
Claim: The greedy solution is always optimal.

Proof: Let (f_1, f_2, \dots, f_n) be an optimal solution.

Case 1: $\forall 1 \leq i \leq n, g_i \leq f_i$



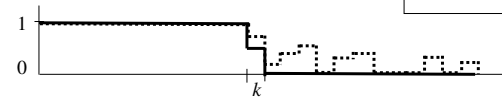
Case 2: $\exists 1 \leq i \leq k, f_i < g_i$



3

Case 1: $\forall 1 \leq i \leq n, g_i \leq f_i$

Assume $W < \sum_{i=1}^n w_i$.

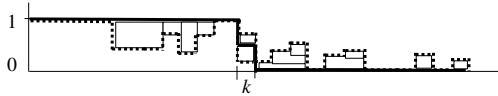


$$\begin{aligned}
 W &\geq \sum_{i=1}^n f_i \cdot w_i = \sum_{i=1}^k f_i \cdot w_i + \sum_{i=k+1}^n f_i \cdot w_i \\
 &= \sum_{i=1}^k g_i \cdot w_i + (f_k - g_k) \cdot w_k + \sum_{i=k+1}^n f_i \cdot w_i \\
 &= \sum_{i=1}^k g_i \cdot w_i + (f_k - g_k) \cdot w_k + \sum_{i=k+1}^n f_i \cdot w_i \\
 &= W + (f_k - g_k) \cdot w_k + \sum_{i=k+1}^n f_i \cdot w_i
 \end{aligned}$$

$$\Rightarrow f_k = g_k, \text{ and } \forall k+1 \leq i \leq n, f_i = 0 \Rightarrow \forall 1 \leq i \leq n, f_i = g_i$$

4

Case 2: $\exists 1 \leq i \leq k, f_i < g_i$

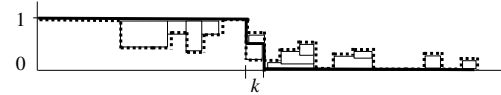


Let j be the first index where $f_j \neq g_j$, ($j \leq k$).

$$\begin{aligned} \sum_{i=1}^n f_i \cdot c_i &= \sum_{i=1}^{j-1} f_i \cdot c_i + \sum_{i=j}^{k-1} f_i \cdot c_i + \sum_{i=k}^n f_i \cdot c_i \\ &= \sum_{i=1}^{j-1} g_i \cdot c_i + \sum_{i=j}^{k-1} g_i \cdot c_i + \sum_{i=j}^{k-1} (f_i - g_i) \cdot c_i + \sum_{i=k}^n f_i \cdot c_i \\ &= \sum_{i=1}^{k-1} g_i \cdot c_i + \sum_{i=j}^{k-1} (f_i - g_i) \cdot c_i + \sum_{i=k}^n f_i \cdot c_i \end{aligned}$$

5

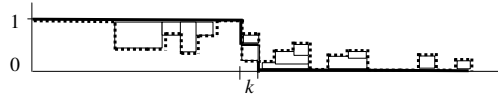
Case 2: $\exists 1 \leq i \leq k, f_i < g_i$ (cont)



$$\begin{aligned} &= \sum_{i=1}^{k-1} g_i \cdot c_i + \sum_{i=j}^{k-1} (f_i - g_i) \cdot \frac{c_i}{w_i} \cdot w_i + \sum_{i=k}^n f_i \cdot \frac{c_i}{w_i} \cdot w_i \\ &\leq \sum_{i=1}^{k-1} g_i \cdot c_i + \frac{c_k}{w_k} \cdot \sum_{i=j}^{k-1} (f_i - g_i) \cdot w_i + \frac{c_k}{w_k} \cdot \sum_{i=k}^n f_i \cdot w_i \\ &= \sum_{i=1}^{k-1} g_i \cdot c_i + \frac{c_k}{w_k} \cdot \left(\sum_{i=j}^n f_i \cdot w_i - \sum_{i=j}^{k-1} g_i \cdot w_i \right) \\ &= \sum_{i=1}^{k-1} g_i \cdot c_i + \frac{c_k}{w_k} \cdot \left(\sum_{i=j}^n f_i \cdot w_i - \sum_{i=j}^{k-1} g_i \cdot w_i + g_k \cdot w_k \right) \end{aligned}$$

6

Case 2: $\exists 1 \leq i \leq k, f_i < g_i$ (cont)



$$\begin{aligned} &= \sum_{i=1}^{k-1} g_i \cdot c_i + \frac{c_k}{w_k} \cdot \left(\sum_{i=1}^n f_i \cdot w_i - \sum_{i=1}^n g_i \cdot w_i \right) + g_k \cdot c_k \\ &\leq \sum_{i=1}^k g_i \cdot c_i + \frac{c_k}{w_k} \cdot \left(\sum_{i=1}^n f_i \cdot w_i - W \right) \\ &\leq \sum_{i=1}^k g_i \cdot c_i = \sum_{i=1}^n g_i \cdot c_i \end{aligned}$$

$$\text{So } \sum_{i=1}^n f_i \cdot c_i \leq \sum_{i=1}^n g_i \cdot c_i$$

QED

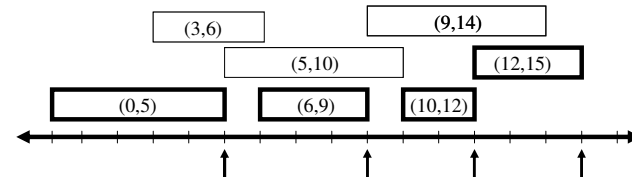
7

ACTIVITY SELECTION

Instance: A set S of n activities with start and end times, (s_i, e_i) .

Problem: Select a largest set of non-overlapping activities.

Greedy Choice: Select the activity with earliest end time and discard any activities that conflict with it.



8

Activities are sorted by non-decreasing end time.

Input: Start $S[1:n]$ and end $E[1:n]$ times.

Output: $A[1:n]$ $A[i]=1$ if activity i is selected else 0

```

i ← 1
while (i ≤ n)
    A[i] ← 1; t ← E[i]
    i ← i + 1
    while (i ≤ n and S[i] < t)
        A[i] ← 0
        i ← i + 1
    endwhile
endwhile

```

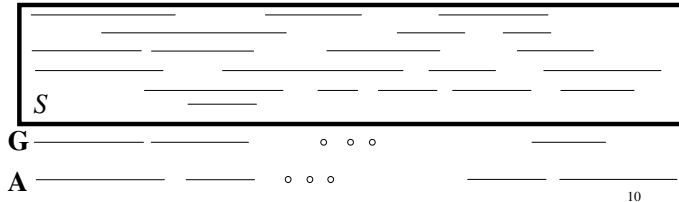
9

Claim: The greedy choice yields an optimal solution to the activity selection problem.

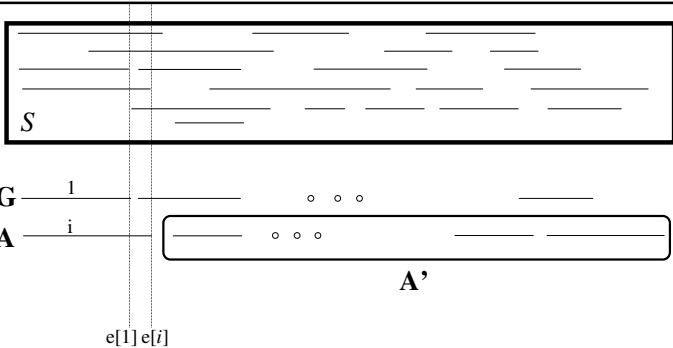
Proof: By induction on $n=|S|$.

Base: $n=0$. Then the only solution is \emptyset . So the greedy solution (\emptyset) is optimal.

Step: $n>0$. Let G be the set of activities selected by the greedy solution and let A be an optimal solution.



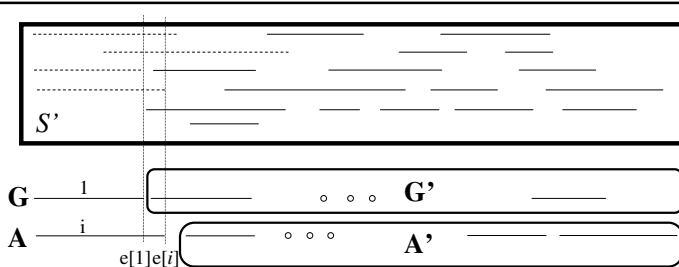
10



Let i be the first activity in A . Let $A' = A - \{i\}$.

Every activity in A' starts after $e[i] \geq e[1]$.

11



Let S' be the subset of activities that start after $e[1]$.

$G' = G - \{1\}$ is an optimal solution to S' by induction.

A' is a solution to S' since every activity in A' starts after $e[i] \geq e[1]$.

$|G'| \geq |A'|$ since G' is an optimal solution to S' .

$|G| = |G'| + 1 \geq |A'| + 1 = |A|$, so G is an optimal solution to S .

QED

12

To show greedy is optimal:

Method 1: Characterize the “Greedy Solution”.

Show any optimal solution is no better than the “greedy solution.”

EASY KNAPSACK

Method 2: Characterize the “Greedy Choice” and remaining subproblem.

Show that there is an optimal solution made up of the greedy choice and an optimal solution to the remaining subproblem.

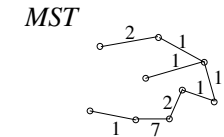
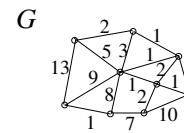
ACTIVITY SELECTION, MINIMUM SPANNING TREE, HUFFMAN CODING

MINIMUM SPANNING TREE

Given a graph, $G = (V, E)$ with edge weights $w(e)$, find a minimum weight spanning tree of G .

$$V = \{v_1, v_2, \dots, v_n\} \quad E = \{e_1, e_2, \dots, e_m\}$$

$T \subseteq E$ induces a *spanning tree* of G , if the graph (V, T) is connected and acyclic.

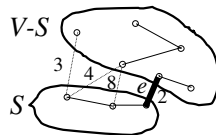


MST LEMMA

If $A \subseteq E$ and \exists a minimum spanning tree T with $A \subseteq T$,

V can be partitioned into S and $V-S$ such that no edge in A spans S and $V-S$,

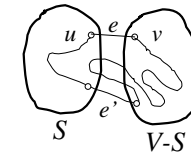
and e is an edge of minimum weight between S and $V-S$, then $A \cup \{e\}$ is also a subset of a minimum spanning tree.



Proof:

Case 1: $e \in T$. Then $A \cup \{e\} \subseteq T$.

Case 2: $e \notin T$. $e = (u, v)$



Since e spans S and $V-S$, $u \in S$ and $v \in V-S$ (or vice versa).

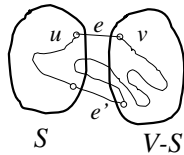
Then $T \cup \{e\}$ has a cycle since u and v are connected in T .

This cycle must have another edge e' that spans S and $V-S$. $e' \notin A$ since it spans S and $V-S$ and $w(e) \leq w(e')$.

Let $T' = (T \cup \{e\}) - \{e'\}$.

$T' = (T \cup \{e\}) - \{e'\}$ is acyclic and connected, ($T \cup \{e\}$ was connected and had only one cycle containing e').

Proof: (cont)



So T' is a spanning tree.

$\text{cost}(T') = \text{cost}(T) - w(e') + w(e) \leq \text{cost}(T)$ since $w(e) \leq w(e')$.

So T' is a minimum spanning tree.

Since $e' \notin A$, $A \subseteq T - \{e'\}$.

So $A \cup \{e\} \subseteq (T \cup \{e\}) - \{e'\} = T'$, a minimum spanning tree.

QED

17

```

A ← ∅
while ((V,A) is not connected)
  (S,V-S) ← a cut that respects A
  e ← a minimum weight edge spanning (S,V-S)
  A ← A ∪ {e}
endwhile

```

	Kruskal's	Prim's
How to pick $(S, V-S)$	Use UNION-FIND to maintain connected components	Let S be the component of a fixed vertex, all others are singletons
How to pick e	Order all edges by increasing cost, and test endpoints to identify their components	For each vertex not in S keep track of cheapest edge to S

18

Kruskal MST

```

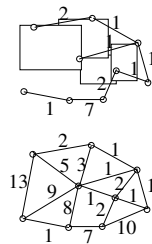
A ← ∅
for each vertex u
  MakeSet(u)
endfor
H ← BuildHeap(E)
while (|A| < |V| - 1)
  (u,v) ← ExtractMin(H)
  If FindSet(u) ≠ FindSet(v)
    then A ← A ∪ {(u,v)}
    Union(u,v)
endwhile

```

```

BuildHeap  $O(m)$ 
m ExtractMin's  $O(m \lg m)$ 
 $O(m)$  UNION-FIND ops.  $O(m \lg^* n)$ 
Total =  $O(m \lg m)$ 

```



19

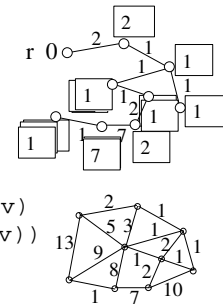
Prim MST

```

A ← ∅; H ← MakeHeap()
for each vertex u
  key(u) ← +∞;
  n[u] ← nil
  Insert(H,u)
endfor
DecreaseKey(r,0)
while (|A| < |V| - 1)
  u ← ExtractMin(H)
  A ← A ∪ {(u,n[u])}
  for each edge (u,v)
    if v ∈ H and key[v] > w(u,v)
      then DecreaseKey(v, w(u,v))
      n[v] ← u
  endfor
endwhile

```

	Binomial	Fibonacci
n Insert's	$O(\lg n)$	$O(1)$
m DecreaseKey's	$O(\lg n)$	$O(1)$
n ExtractMin's	$O(\lg n)$	$O(\lg n)$
Total	$O(m \lg n)$	$O(m + n \lg n)$



20