Greedy Algorithms

- Introduction to Greedy algorithms
- Importance of Proof of Correctness for Greedy algorithms
- How do we write proof of correctness?
- Land Allocation Problem.
- Introduction to Activity Selection Problem.

Instructor:

Dr. Arpita Biswas

Important Dates

- **Friday 10/10/2025, 2:00-3:20 pm**: Mid-Term Exam
- **Wednesday 12/17/2025, 8:00-11:00 am**: Final Exam (according to University Schedule)

Mark these dates. NO change to the schedule is possible.

In case of absence due to unforeseen emergency, please reach out to the **Dean of Students**. Read https://studentsupport.rutgers.edu/services/absence-and-verification-notices.

Check the **course webpage** for Grade policy, Course Policy, Syllabus, and Course Materials. https://sites.google.com/view/cs344/home.

Please post **your queries only on CANVAS** and not emails.

Announcements

- Practice Problem Set 1 was released on Friday (September 5)
- Office hours are listed on the course webpage
- Recitation session starts this week – practice Problem Set 1 for the quiz

Recap

- Efficiency Analysis: Time and Space Complexity
- Comparison between common functions – polynomial and exponential functions
- Asymptotic upper and lower bounds
- Computing time and space complexity of an algorithm

Credit: David Wu's lecture slides

Complexity Analysis of Algorithms

Time Complexity

- The amount of time an algorithm takes to complete as a function of the length of the input.
- Big O notation: upper bound of the algorithm's running time.

Space Complexity

- the amount of memory an algorithm uses as a function of the length of the input.
- the memory needed for the input data + any additional memory allocated during execution.
- Big O notation: upper bound of the algorithm’s memory usage.

Asymptotic Upper Bound

Big-O notation

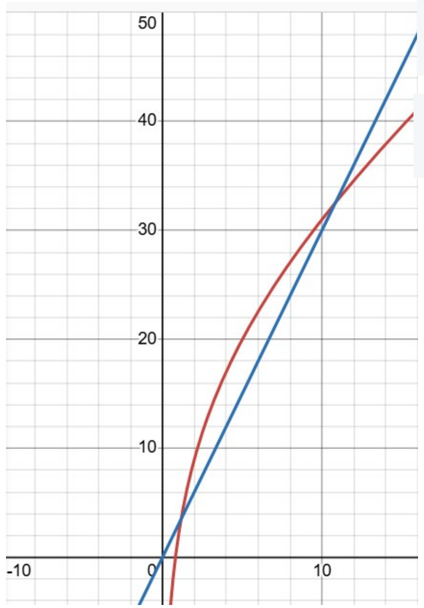
Let T(n) be the worstcase running time of an algorithm

$\mathbf{T(n)}$ is said to be $\mathbf{O(f(n))}$ if there exist $\mathbf{c} > 0$ and $n_0 \ge 0$ such that $T(n) \le c * f(n)$ for all $n \ge n_0$.

$$\boxed{T(n) \le \underbrace{\boxed{3}}_{\mathbf{c}} n \text{ for } n \ge \underbrace{\boxed{10.8}}_{n_0}}$$

Simple example

$$T(n) = n + 20 \log n + 1$$



$3n$

$T(n) = \mathbf{O(n)}$

Properties of Big-O

Constant Factor

$c > 0$

$f(n) = O(g(n))$,
then $\mathbf{c\,f(n)} = O(g(n))$

Sum Rule

$+$

$f(n) = O(g(n))$ and $h(n) = O(k(n))$,
then $\mathbf{f(n) + h(n)} = O(\max(g(n), k(n)))$

Product Rule

$*$

$f(n) = O(g(n))$ and $h(n) = O(k(n))$,
then $\mathbf{f(n) * h(n)} = O(g(n) * k(n))$

Revisiting Examples for Explaining Big-O Properties

```
for (i=1; i<=n; i++){
        print(i)
}
for (j=1; j<=n; j++){
        for (k=1; k<=n; k++){
                print(j)
        }
}
```

$\mathbf{O(n)}$

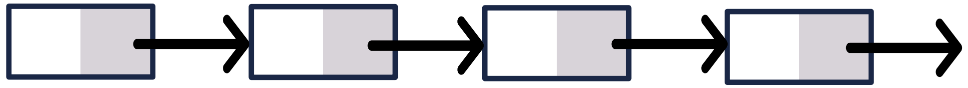
$\mathbf{O(n) * O(n)}$
$\mathbf{= O(n^2)}$

Product Rule

$= \mathbf{O(n + n^2)}$
$= \mathbf{O(max(n, n^2))}$
$= \mathbf{O(n^2)}$

Sum Rule

Time and Space Complexity with Common Data Structures

- Arrays:  Space O(n)
  - Access to i-th element: O(1)
  - Find x in an unsorted array: O(n)
  - Find x in a sorted array: O(log n)
- Linked Lists:  null Space: dynamic
  - Access to i-th element: O(i)
  - Find x in an unsorted list: O(n)
  - Find x in a sorted list: O(n)

Comparison of Two Algorithms for the Same Problem

Given a number a and a positive integer n, suppose we want to compute a^n

<u>Algorithm</u>: Expo(a, n)

```
result = a

for i=2 to n
        result = result * a

return result
```

Time Complexity = O(n)

<u>Algorithm</u>: Expo2(a, n)

```
If n==1, return a
else
        x = Expo2(a, ⌊n/2⌋)
        If x is even,  return x*x
        else,          return x*x*a
```

Time Complexity = O(log n)

Comparison of Two Algorithms for the Same Problem

Given a number a and a positive integer n, suppose we want to compute a^n

$\cancel{T(n) = 0 \text{ when } n==1}$ $T(1)=1$

$$
\begin{aligned}
T(n) &= T(n/2) + 1 \\
&= T(n/4) + 1 + 1 \\
&= T(n/8) + 1 + 1 + 1 \\
&= T(n/2^3) + 1*3 \\
&= 1 + 1*\log_2 n
\end{aligned}
$$

$n/2^k = 1$ implies $k = \log_2 n$

Time Complexity = O(log n)

Algorithm: Expo2(a, n)

If n==1, return a

else

x = Expo2(a, ⌊n/2⌋)

If x is even, return x*x

else, return x*x*a

Polynomial Time Algorithm

A running time $T(n)$ is called polynomial time if $T(n) = O(n^c)$ for a constant c.

Polynomial time is an important concept and efficiency requirement for algorithms.
The word "**efficient**" is often used to refer to polynomial time.

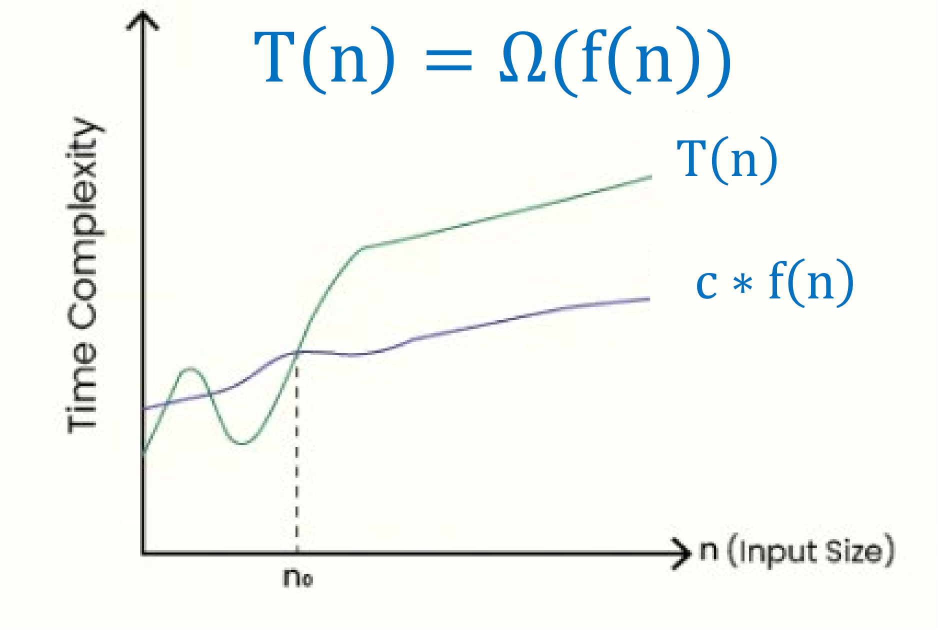
If the input size doubles, a polynomial-time algorithm only becomes a constant (2^c) times slower (in terms of the upper bound $O(n^c)$).

Asymptotic Lower Bound

Big-Omega notation

Let $T(n)$ be the worstcase running time of an algorithm

$\mathbf{T}(\mathbf{n})$ is said to be $\boldsymbol{\Omega}(\mathbf{f}(\mathbf{n}))$ if there exist $\mathbf{c} > 0$ and $n_0 \geq 0$ such that $T(n) \geq c * f(n)$ for all $n \geq n_0$.

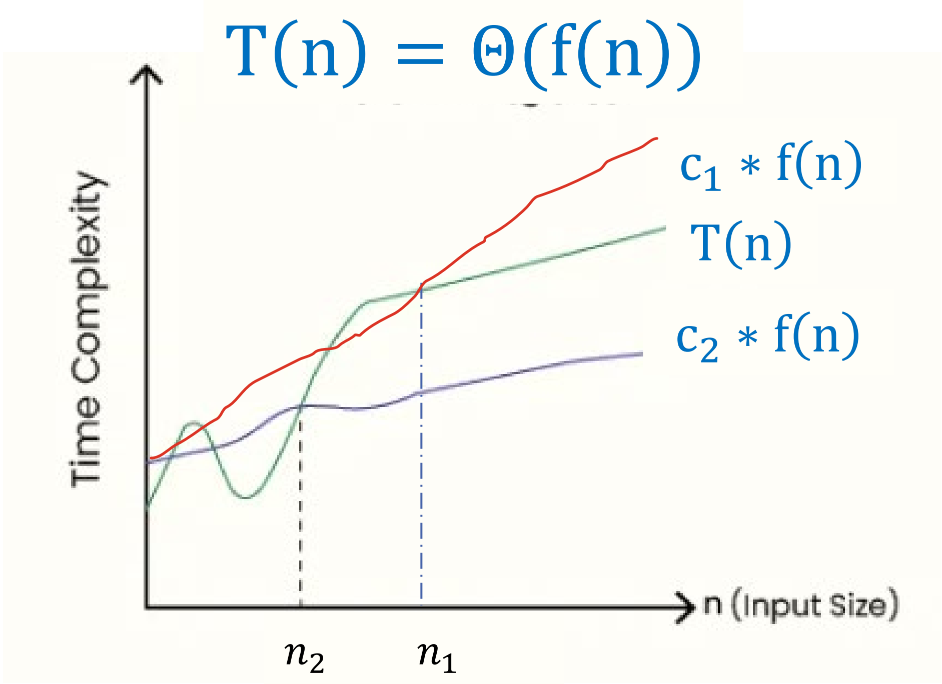


Asymptotic Upper and Lower Bound

Big-Theta notation

Let $T(n)$ be the worstcase running time of an algorithm

$\mathbf{T(n)}$ is said to be $\boldsymbol{\Theta}\big(\mathbf{f(n)}\big)$
iff $T(n) = O\big(f(n)\big)$ and $T(n) = \Omega(f(n))$



Today: Algorithm Design Technique (Greedy)

What are **greedy** algorithms?

- Make the **best local choice** at each step.
- Hopes that local optimum → global optimum.

Greedy Algorithms

What are **greedy** algorithms?

- Make the **best local choice** at each step.
- Hopes that local optimum → global optimum.

Simple Solutions

Fast Runtime

Tricky to Trust

Greedy Algorithms

What are **greedy** algorithms?

- Make the **best local choice** at each step.
- Hopes that local optimum → global optimum.

Simple Solutions

Fast Runtime

Tricky to Trust

Greedy algorithms are incredibly tempting but are rarely correct.

Whenever you write a greedy algorithm, you **must** provide:
a **formal proof of correctness**

Greedy Algorithms

What are **greedy** algorithms?

- Make the **best local choice** at each step.
- Hopes that local optimum → global optimum.

Simple Solutions

Fast Runtime

Tricky to Trust

Greedy algorithms are incredibly tempting but are rarely correct.

You will **NOT** receive **ANY** credit if you do not provide a formal proof for a greedy solution, even if the solution is correct.

Land Allocation Problem

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - lowest cost first
   - highest area first
   - lowest area first
   - ...
2. Repeat picking plot using the same criterion until you exhaust the budget.

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - **lowest cost first**
   - highest area first
   - lowest area first
   - …
2. Repeat picking plot using the same criterion until you exhaust the budget.

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - **lowest cost first**
   - highest area first
   - lowest area first
   - ...
2. Repeat picking plot using the same criterion until you exhaust the budget.

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: ~~$50 M~~ $35 M

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - **lowest cost first**
   - highest area first
   - lowest area first
   - …
2. Repeat picking plot using the same criterion until you exhaust the budget.

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: ~~$50 M~~
~~$35 M~~
$15 M

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - **lowest cost first**
   - highest area first
   - lowest area first
   - …
2. Repeat picking plot using the same criterion until you exhaust the budget.

Land Identifier	Cost (in $M)	Area (sq. ft.)
1/2 A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: ~~$50 M~~
~~$35 M~~
~~$15 M~~ 0

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - **lowest cost first**
   - highest area first
   - lowest area first
   - ...
2. Repeat picking plot using the same criterion until you exhaust the budget.

	Land Identifier	Cost (in $M)	Area (sq. ft.)
1/2	A	30	1050
1	B	15	450
1	C	20	980
0	D	40	2000

Your budget: ~~$50 M~~
~~$35 M~~
~~$15 M~~ 0

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - **lowest cost first**
   - highest area first
   - lowest area first
   - ...
2. Repeat picking plot using the same criterion until you exhaust the budget.

	Land Identifier	Cost (in $M)	Area (sq. ft.)
1/2	A	30	1050
1	B	15	450
1	C	20	980
0	D	40	2000

Your budget: ~~$50 M~~ ~~$35 M~~ ~~$15 M~~ 0

Total = 450 + 980 + 575
= 2005

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - **lowest cost first** **Not optimal**
   - highest area first
   - lowest area first
   - ...
2. Repeat picking plot using the same criterion until you exhaust the budget.

Land Identifier		Cost (in $M)	Area (sq. ft.)
1/2	A	30	1050
1	B	15	450
1	C	20	980
0	D	40	2000

Your budget: ~~$50 M~~
~~$35 M~~
~~$15 M~~ 0

Total = 450 + 980 + 575
= 2005

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - lowest cost first
   - **highest area first**
   - lowest area first
   - ...
2. Repeat picking plot using the same criterion until you exhaust the budget.

Land Identifier		Cost (in $M)	Area (sq. ft.)
?	A	30	1050
?	B	15	450
?	C	20	980
?	D	40	2000

Your budget: $50 M   Total = ?

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - lowest cost first
   - **highest area first**
   - lowest area first
   - ...
2. Repeat picking plot using the same criterion until you exhaust the budget.

	Land Identifier	Cost (in $M)	Area (sq. ft.)
1/3	A	30	1050
0	B	15	450
0	C	20	980
1	D	40	2000

Your budget: ~~$50 M~~ ~~$10 M~~ 0

Total = 2000 + 350
= 2350

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - lowest cost first
   - **highest area first** **Not optimal**
   - lowest area first
   - ...
2. Repeat picking plot using the same criterion until you exhaust the budget.

	Land Identifier	Cost (in $M)	Area (sq. ft.)
1/3	A	30	1050
0	B	15	450
0	C	20	980
1	D	40	2000

Your budget: ~~$50 M~~ ~~$10 M~~ 0

Total = 2000 + 350
= 2350

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - lowest cost first
   - highest area first
   - **lowest area first**
   - highest area/cost first
2. Repeat picking plot using the same criterion until you exhaust the budget.

Exercise:
What is the solution using this approach. Is this optimal?

Land Identifier		Cost (in $M)	Area (sq. ft.)
?	A	30	1050
?	B	15	450
?	C	20	980
?	D	40	2000

Your budget: $50 M Total = ?

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - lowest cost first
   - highest area first
   - lowest area first
   - **highest area/cost first**
2. Repeat picking plot using the same criterion until you exhaust the budget.

Land Identifier		Cost (in $M)	Area (sq. ft.)
?	A	30	1050
?	B	15	450
?	C	20	980
?	D	40	2000

Your budget: $50 M Total = ?

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Greedy algorithm:

1. Fix a criterion

   For example,
   - lowest cost first
   - highest area first
   - lowest area first
   - **highest area/cost first**
2. Repeat picking plot using the same criterion until you exhaust the budget.

Land Identifier		Cost (in $M)	Area (sq. ft.)
0	A	30	1050
0	B	15	450
1/2	C	20	980
1	D	40	2000

Your budget: $50 M ~~$10 M~~ 0

Total = 2000 + 490
= 2490

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Algorithm:

1. Find the area per unit cost of each plot

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B’ = B

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B’ = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B’
      Sol = Sol U {p} and B’=B’-Cost(p)

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B’ = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B’
      Sol = Sol U {p} and B’=B’-Cost(p)
   ii. Else
      f = B’ * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem (Fractional Knapsack Problem)

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem (Fractional Knapsack Problem)

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

What is the output of this algorithm?

Land Allocation Problem (Fractional Knapsack Problem)

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B’ = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B’
      Sol = Sol U {p} and B’=B’-Cost(p)
   ii. Else
      f = B’ * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

Solution: D>C>A>B (Sorted order)
{1, 1/2, 0, 0} Allocate D completely and 1/2 of C.
Total area = 2490

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B’ = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B’
      Sol = Sol U {p} and B’=B’-Cost(p)
   ii. Else
      f = B’ * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B’ = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B’
      Sol = Sol U {p} and B’=B’-Cost(p)
   ii. Else
      f = B’ * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

- **ALG** = Greedy-choice property:
  choose the plot with the **maximum area per unit cost** in every iteration

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B’ = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B’
      Sol = Sol U {p} and B’=B’-Cost(p)
   ii. Else
      f = B’ * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

- **ALG** = Greedy-choice property:
  choose the plot with the **maximum area per unit cost** in every iteration

- Let **OPT** be the optimal fractions $\{y_1, y_2, \dots, y_n\}$ sorted in decreasing (non-increasing) order of $\frac{Area(p_i)}{Cost(p_i)} = \frac{v_i}{c_i}$.
  wlog $\frac{v_1}{c_1} > \frac{v_2}{c_2} > \dots > \frac{v_n}{c_n}$

  Area(OPT) = $\sum_{i=1}^{n} y_i v_i$ is the maximum possible area acquired using a budget of B, i.e.,$\sum_{i=1}^{n} y_i c_i \leq B$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:
Let there be n plots.
Let **ALG** be $\{x_1, x_2, \dots, x_n\}$ and **OPT** be $\{y_1, y_2, \dots, y_n\}$, sorted in decreasing (non–increasing) order of $\frac{v_i}{c_i}$.

Case 1:
If total cost of all the plots is **within** the budget B

$$\sum_{i=1}^{n} c_i \le B,$$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:
Let there be n plots.
Let **ALG** be $\{x_1, x_2, \dots, x_n\}$ and **OPT** be $\{y_1, y_2, \dots, y_n\}$, sorted in decreasing (non–increasing) order of $\frac{v_i}{c_i}$.

Case 1:
If total cost of all the plots is **within** the budget B

$$\sum_{i=1}^{n} c_i \leq B,$$

then the **OPT** will contain all the n plots, i.e.
OPT = $\{y_1 = 1, y_2 = 1, \dots, y_n = 1\}$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Let there be n plots.

Let **ALG** be $\{x_1, x_2, \ldots, x_n\}$ and **OPT** be $\{y_1, y_2, \ldots, y_n\}$, sorted in decreasing (non−increasing) order of $\frac{v_i}{c_i}$.

Case 1:

If total cost of all the plots is **within** the budget B

$$\sum_{i=1}^{n} c_i \leq B,$$

then the **OPT** will contain all the n plots, i.e.

OPT = $\{y_1 = 1, y_2 = 1, \ldots, y_n = 1\}$

By greedy design, **ALG** will select all the n plots since the budget B is never violated in any iteration.

Hence, **OPT** is same as **ALG** = $\{1,1, \ldots, 1\}$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Let there be n plots.

Let **ALG** be $\{x_1, x_2, \dots, x_n\}$ and **OPT** be $\{y_1, y_2, \dots, y_n\}$, sorted in decreasing (non–increasing) order of $\frac{v_i}{c_i}$.

Case 2:

If total cost of all the plots is **more** the budget B

$$\sum_{i=1}^{n} c_i > B,$$

Then prove **ALG = OPT** by contradiction!

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof by **contradiction**:

- *Proof Sketch:*
  1. Assume ALG ≠ OPT, i.e., Area(ALG) < Area(OPT).
  2. Show that OPT can then be modified to give a new solution ALT satisfying Area(ALT)>Area(OPT)
  3. Implying that OPT could not have been the optimal solution to start with. Thus, the initial assumption Area(ALG) < Area(OPT) is incorrect (contradiction!)
  4. Hence, Area(ALG) ≥ Area(OPT).
  5. Therefore, ALG is optimal (= OPT).

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Assume ALG ≠ OPT , i.e., Area(ALG)<Area(OPT)

$$\sum_{i=1}^{n} x_i v_i < \sum_{i=1}^{n} y_i v_i$$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B’ = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B’
      Sol = Sol U {p} and B’=B’-Cost(p)
   ii. Else
      f = B’ * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Assume ALG ≠ OPT , i.e., Area(ALG)<Area(OPT)

$$\sum_{i=1}^{n} x_i v_i < \sum_{i=1}^{n} y_i v_i$$

Let i^* be the first index at which $x_{i*} \neq y_{i*}$

By the greedy design, we know that each p_i is added as much as it possible, so $x_{i*} > y_{i*}$. Hence, $\sum_{i=1}^{i*} x_i v_i > \sum_{i=1}^{i*} y_i v_i$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B’ = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B’
      Sol = Sol U {p} and B’=B’-Cost(p)
   ii. Else
      f = B’ * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Assume ALG ≠ OPT , i.e., Area(ALG)<Area(OPT)

$$\sum_{i=1}^{n} x_i v_i < \sum_{i=1}^{n} y_i v_i$$

Let i^* be the first index at which $x_{i*} \neq y_{i*}$

By the greedy design, we know that each p_i is added as much as it possible, so $x_{i*} > y_{i*}$. Hence, $\sum_{i=1}^{i*} x_i v_i > \sum_{i=1}^{i*} y_i v_i$

Since $\sum_{i=1}^{n} x_i v_i < \sum_{i=1}^{n} y_i v_i$, there exists an index j $> i*$ such that, $x_j < y_j$.

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Let ALT be a new solution $\{z_1, z_2, \ldots, z_n\}$ where,

- $z_k = y_k \; for \; all \; k \neq \{i^*, j\}$
- $z_{i*} = y_{i*} + (x_{i*} - y_{i*})$
- $z_j = y_j - (x_{i*} - y_{i*})\frac{c_{i*}}{c_j}$

Total cost of ALT remains the same as OPT.

$$z_{i*}c_{i*} + z_j c_j$$
$$= (y_{i*} + (x_{i*} - y_{i*}))c_{i*} + (y_j - (x_{i*} - y_{i*})\frac{c_{i*}}{c_j})c_j$$
$$= y_{i*}c_{i*} + y_j c_j$$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Let ALT be a new solution $\{z_1, z_2, \ldots, z_n\}$ where,

- $z_k = y_k \; for \; all \; k \neq \{i^*, j\}$
- $z_{i*} = y_{i*} + (x_{i*} - y_{i*})$
- $z_j = y_j - (x_{i*} - y_{i*}) \frac{c_{i*}}{c_j}$

Total cost of ALT remains the same as OPT.

$z_{i*} c_{i*} + z_j c_j$

$= (y_{i*} + (x_{i*} - y_{i*})) c_{i*} + (y_j - (x_{i*} - y_{i*}) \frac{c_{i*}}{c_j}) c_j$

$= y_{i*} c_{i*} + y_j c_j$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Let ALT be a new solution $\{z_1, z_2, \ldots, z_n\}$ where,

- $z_k = y_k \; for \; all \; k \neq \{i^*, j\}$
- $z_{i*} = y_{i*} + (x_{i*} - y_{i*})$
- $z_j = y_j - (x_{i*} - y_{i*})\frac{c_{i*}}{c_j}$

Total cost of ALT remains the same as OPT.

$$z_{i*}c_{i*} + z_j c_j$$
$$= (y_{i*} + (x_{i*} - y_{i*}))c_{i*} + (y_j - (x_{i*} - y_{i*})\frac{c_{i*}}{c_j})c_j$$
$$= y_{i*}c_{i*} + y_j c_j$$

What about the total value of ALT?

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Total value of ALT is
$z_{i*}v_{i*} + z_jv_j$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Total value of ALT is

$$z_{i*}v_{i*} + z_j v_j$$
$$= (y_{i*} + (x_{i*} - y_{i*}))v_{i*} + (y_j - (x_{i*} - y_{i*})\frac{c_{i*}}{c_j})v_j$$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Total value of ALT is

$$z_{i*}v_{i*} + z_j v_j$$
$$= (y_{i*} + (x_{i*} - y_{i*}))v_{i*} + (y_j - (x_{i*} - y_{i*})\frac{c_{i*}}{c_j})v_j$$
$$= y_{i*}v_{i*} + y_j v_j + (x_{i*} - y_{i*})(v_{i*} - \frac{c_{i*}}{c_j} v_j)$$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B’ = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B’
      Sol = Sol U {p} and B’=B’-Cost(p)
   ii. Else
      f = B’ * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Total value of ALT is

$$z_{i*}v_{i*} + z_j v_j$$
$$= (y_{i*} + (x_{i*} - y_{i*}))v_{i*} + (y_j - (x_{i*} - y_{i*})\frac{c_{i*}}{c_j})v_j$$
$$= y_{i*}v_{i*} + y_j v_j + \boxed{(x_{i*} - y_{i*})(v_{i*} - \frac{c_{i*}}{c_j} v_j)} > 0$$

Greedy algorithm

Land Allocation Problem

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Total value of ALT is

$$z_{i*}v_{i*} + z_j v_j$$
$$= (y_{i*} + (x_{i*} - y_{i*}))v_{i*} + (y_j - (x_{i*} - y_{i*})\frac{c_{i*}}{c_j})v_j$$
$$= y_{i*}v_{i*} + y_j v_j + \boxed{(x_{i*} - y_{i*})(v_{i*} - \frac{c_{i*}}{c_j} v_j)} > 0$$

- $x_{i*} > y_{i*}$ (as claimed before)
- By greedy design, $\frac{v_{i*}}{c_{i*}} > \frac{v_j}{c_j}$ since index $i^* > j$.

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Total value of ALT is

$$z_{i*}v_{i*} + z_j v_j$$
$$= (y_{i*}+(x_{i*} - y_{i*}))v_{i*} + (y_j-(x_{i*} - y_{i*})\frac{c_{i*}}{c_j})v_j$$
$$= y_{i*}v_{i*} + y_j v_j + \boxed{(x_{i*} - y_{i*})(v_{i*} - \frac{c_{i*}}{c_j} v_j)} > 0$$

- $x_{i*} > y_{i*}$ (as claimed before)
- By greedy design, $\frac{v_{i*}}{c_{i*}} > \frac{v_j}{c_j}$ since index $i^* > j$.

$$> y_{i*}v_{i*} + y_j v_j$$

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Total value of ALT is

$$z_{i*}v_{i*} + z_j v_j$$

$$= (y_{i*} + (x_{i*} - y_{i*}))v_{i*} + (y_j - (x_{i*} - y_{i*})\frac{c_{i*}}{c_j})v_j$$

$$= y_{i*}v_{i*} + y_j v_j + \boxed{(x_{i*} - y_{i*})(v_{i*} - \frac{c_{i*}}{c_j} v_j)} > 0$$

- $x_{i*} > y_{i*}$ (as claimed before)
- By greedy design, $\frac{v_{i*}}{c_{i*}} > \frac{v_j}{c_j}$ since index $i^* > j$.

$$> y_{i*}v_{i*} + y_j v_j$$

The total value of ALT is greater than OPT.

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

Total value of ALT is

$$z_{i*}v_{i*} + z_j v_j$$
$$= (y_{i*}+(x_{i*} - y_{i*}))v_{i*} + (y_j-(x_{i*} - y_{i*})\frac{c_{i*}}{c_j})v_j$$
$$= y_{i*}v_{i*} + y_j v_j + \boxed{(x_{i*} - y_{i*})(v_{i*} - \frac{c_{i*}}{c_j} v_j)} > 0$$

- $x_{i*} > y_{i*}$ (as claimed before)
- By greedy design, $\frac{v_{i*}}{c_{i*}} > \frac{v_j}{c_j}$ since index $i^* > j$.

$$> y_{i*}v_{i*} + y_j v_j$$

The total value of ALT is greater than OPT. Contradiction!

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Proof of Correctness?

Proof:

The total value of ALT is greater than OPT.
Contradiction to the optimality of OPT!

Therefore, initial assumption ALG ≠ OPT is incorrect.

In other words, ALG = OPT.

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Time Complexity?

O(n)

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Time Complexity?

O(n)

O(n log n)

Land Allocation Problem

Greedy algorithm

Algorithm:

1. Find the area per unit cost of each plot
2. Sort the plots according to decreasing area per unit cost
3. Remaining budget B' = B

4. For each plot p in the sorted order,
   i. If Cost(p) ≤ B'
      Sol = Sol U {p} and B'=B'-Cost(p)
   ii. Else
      f = B' * Area(p) / Cost(p)
      Sol = Sol U {fraction f of p}

Time Complexity?

O(n)

O(n log n)

O(1)

Land Allocation Problem

Greedy algorithm

Algorithm:

Time Complexity?

1. Find the area per unit cost of each plot — O(n)
2. Sort the plots according to decreasing area per unit cost — O(n log n)
3. Remaining budget B' = B — O(1)
4. For each plot p in the sorted order, — O(n)
   i. If Cost(p) ≤ B' — * O(1)
      Sol = Sol U {p} and B'=B'-Cost(p) — O(1)
   ii. Else
      f = B' * Area(p) / Cost(p) — O(1)
      Sol = Sol U {fraction f of p} — O(1)

Land Allocation Problem

Greedy algorithm

Algorithm:

Time Complexity = O(n log n)

1. Find the area per unit cost of each plot — O(n)
2. Sort the plots according to decreasing area per unit cost — O(n log n)
3. Remaining budget B' = B — O(1)
4. For each plot p in the sorted order, — O(n)
   i. If Cost(p) ≤ B' — * O(1)
      Sol = Sol U {p} and B'=B'-Cost(p) — O(1)
   ii. Else
      f = B' * Area(p) / Cost(p) — O(1)
      Sol = Sol U {fraction f of p} — O(1)

Land Allocation Problem (0-1 Knapsack Problem)

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

No fractional allocation allowed!

Question: How would you use **your budget** to buy **as much land as possible?**

Land Allocation Problem (0-1 Knapsack Problem)

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

No fractional allocation allowed!

Question: How would you use **your budget** to buy **as much land as possible?**

Would greedy work?

Land Allocation Problem (0-1 Knapsack Problem)

Land Identifier	Cost (in $M)	Area (sq. ft.)
A	30	1050
B	15	450
C	20	980
D	40	2000

Your budget: $50 M

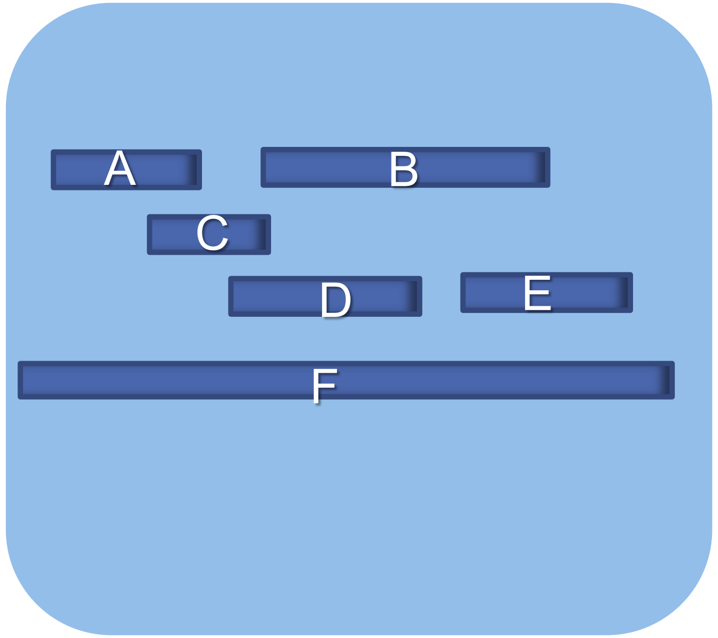
No fractional allocation allowed!

Question: How would you use **your budget** to buy **as much land as possible?**

Would greedy work?

We will cover this later in the course

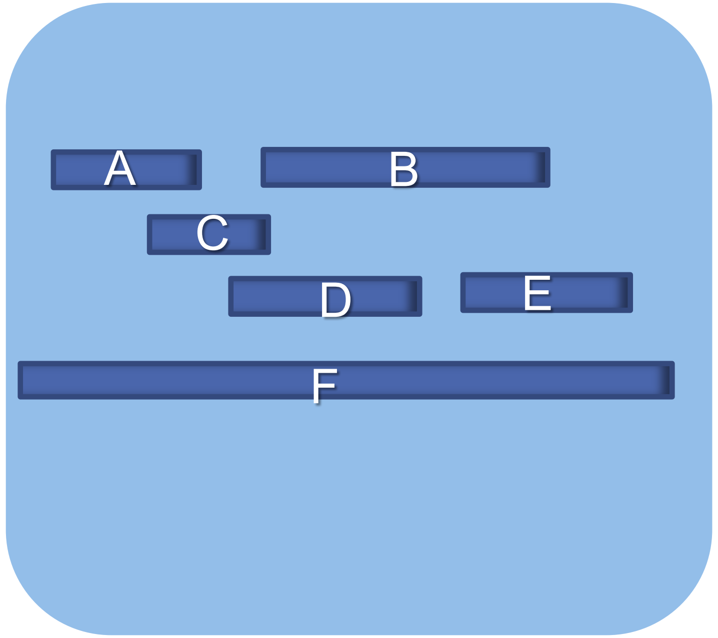
Activity Selection Problem



Given: Set of activities (or, intervals), each having a start time and finish time

Problem: Select the maximum numbers of non-overlapping (or, compatible) activities

Activity Selection Problem

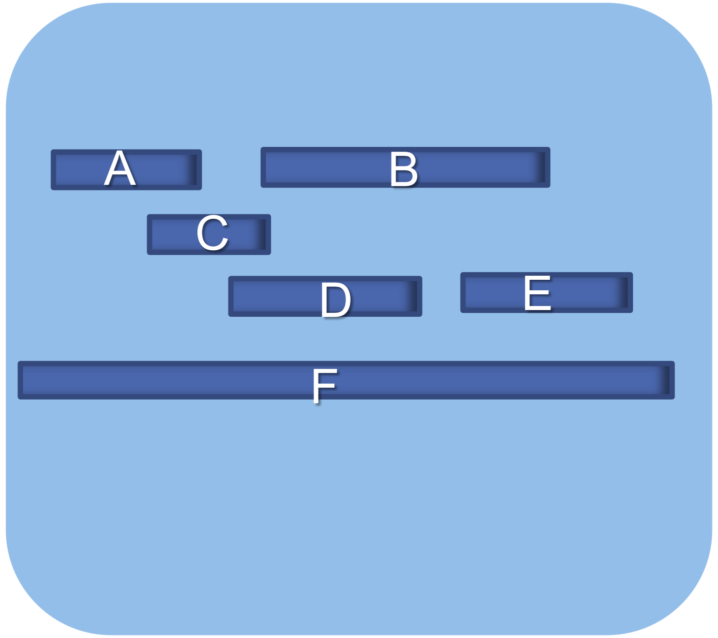


Possible **greedy** approaches:

1. Choose the available activity that start earliest

Earliest start: F<A<C<D<B<E

Activity Selection Problem



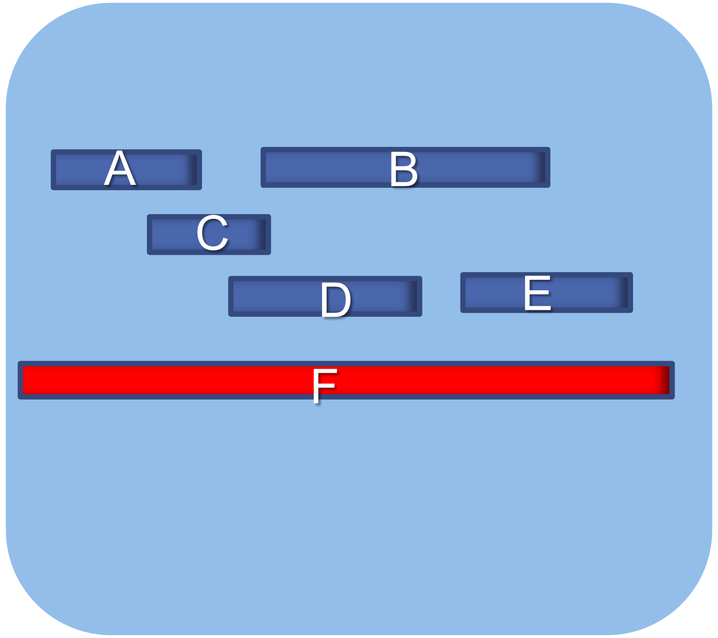
Possible **greedy** approaches:

1. Choose the available activity that start earliest

Earliest start: F<A<C<D<B<E

Total number of activities selected?

Activity Selection Problem



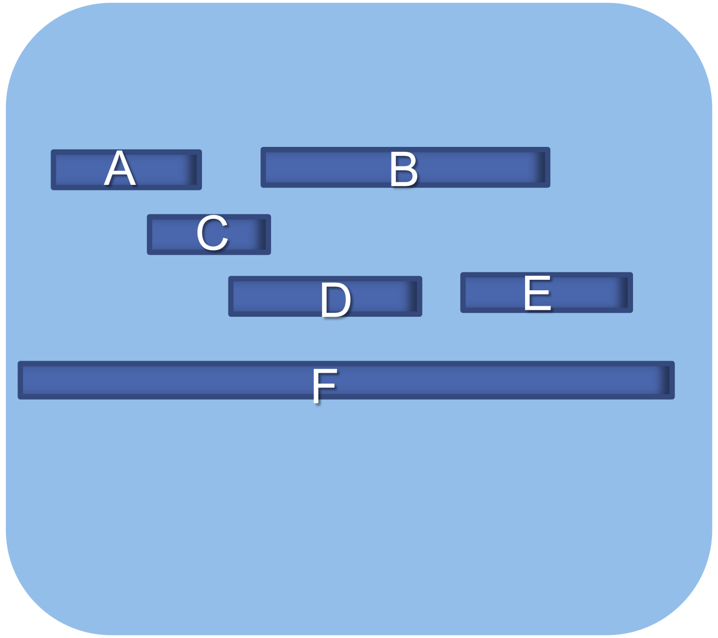
Possible **greedy** approaches:

1. Choose the available activity that start earliest

Earliest start: F<A<C<D<B<E

Total number of activities selected? **1**

Activity Selection Problem



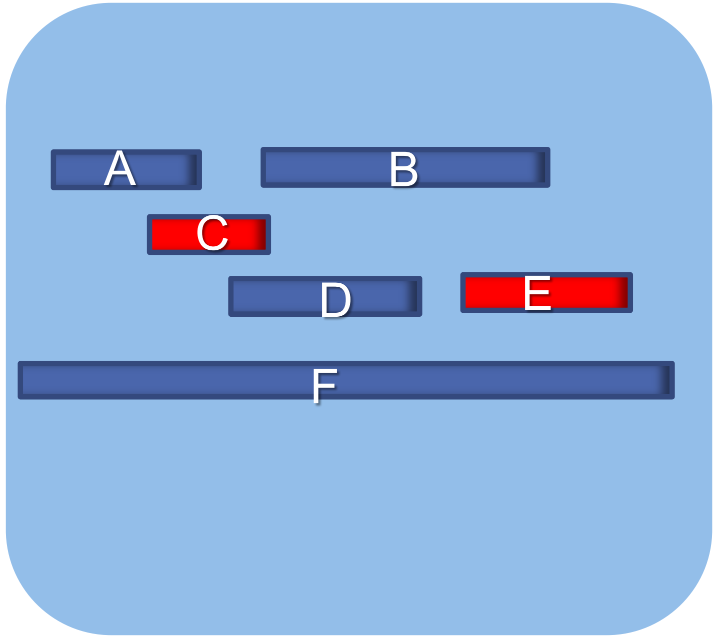
Possible **greedy** approaches:

1. Choose the available activity that start earliest
2. Choose the available activity with the smallest time interval

Interval size: C<A<E<D<B<F

Total number of activities selected?

Activity Selection Problem



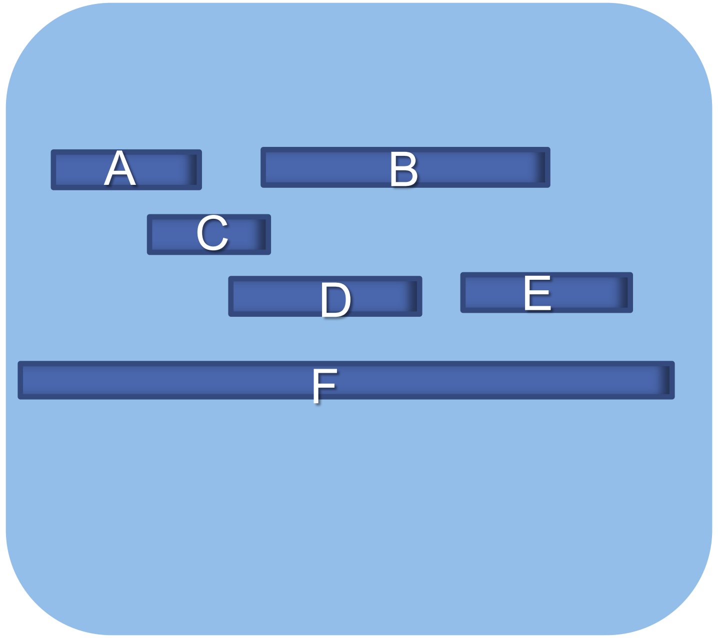
Possible **greedy** approaches:

1. Choose the available activity that start earliest
2. Choose the available activity with the smallest time interval

Interval size: C<A<E<D<B<F

Total number of activities selected? **2**

Activity Selection Problem

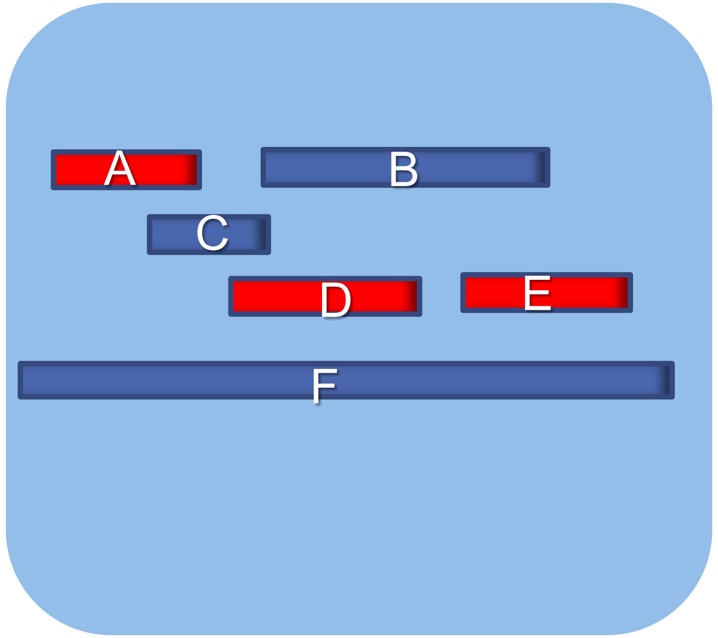


Possible **greedy** approaches:

1. Choose the available activity that start earliest **1**
2. Choose the available activity with the smallest time interval **2**
3. For each activity, count the number of incompatible activities, and choose the available activity with the fewest number of incompatible activities

Total number of activities selected?

Activity Selection Problem

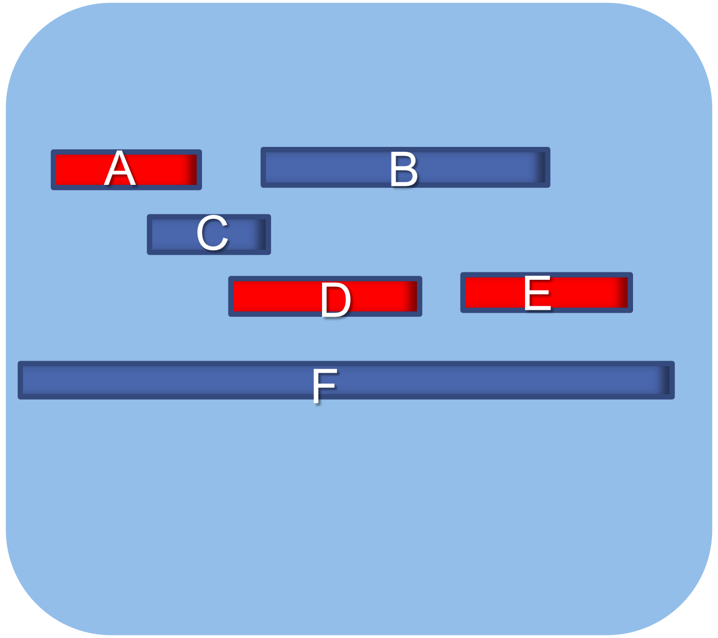


Possible **greedy** approaches:

1. Choose the available activity that start earliest **1**
2. Choose the available activity with the smallest time interval **2**
3. For each activity, count the number of incompatible activities, and choose the available activity with the fewest number of incompatible activities

Total number of activities selected? **3**

Activity Selection Problem



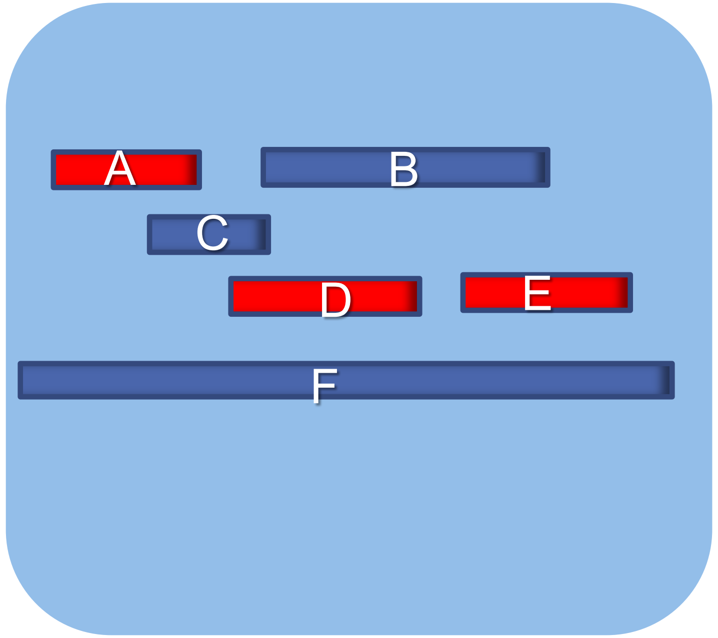
Possible **greedy** approaches:

1. Choose the available activity that start earliest **1**
2. Choose the available activity with the smallest time interval **2**
3. For each activity, count the number of incompatible activities, and choose the available activity with the fewest number of incompatible activities

Total number of activities selected? **3**

Is this greedy strategy optimal?

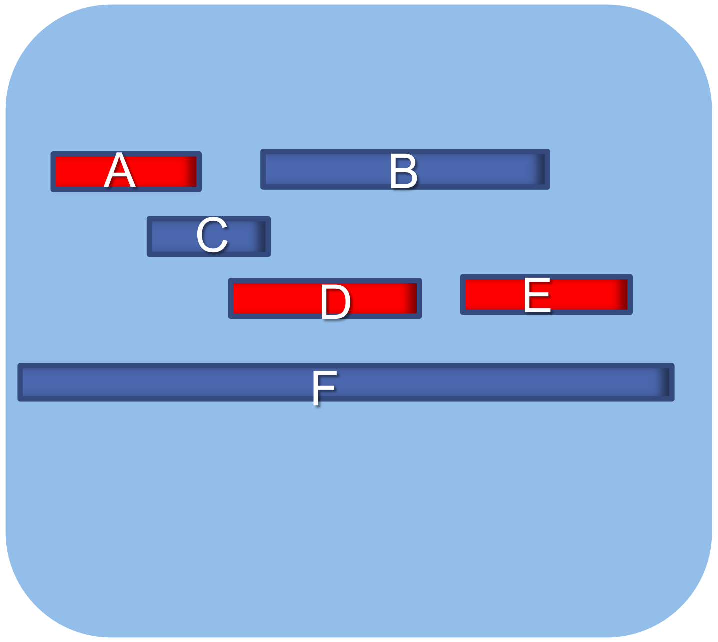
Activity Selection Problem



For each activity, count the number of incompatible activities, and choose the available activity with the fewest number of incompatible activities

Exercise:
Is this greedy approach optimal?
- If not, give an example (counterexample) where approach 3 does not select the highest number of possible activities.
- If yes, prove by contradiction!

Activity Selection Problem



We will revisit activity selection problem during the next lecture

For each activity, count the number of incompatible activities, and choose the available activity with the fewest number of incompatible activities

Exercise:

Is this greedy approach optimal?

- If not, give an example (counterexample) where approach 3 does not select the highest number of possible activities.
- If yes, prove by contradiction!

References

- Algorithms by Jeff Erickson [Chap 4]
- Algorithm Design by Kleinberg and Tardos [Chap 4]
- Demo of interval scheduling (activity selection problem)

References

- [Algorithms by Jeff Erickson [Chap 4]]
- [Algorithm Design by Kleinberg and Tardos [Chap 4]]

Quiz/Practice session starts today (for Section 13 and 14) Good luck!
and on Friday (for Section 12 and 15)

TAs are leading the quiz/practice sessions and have announced the logistics on Canvas. They will provide more details during the session. Stay tuned!