ACM Training

Greedy, Enumeration and Guess

Outlines

- Enumeration
  - Simplest method
  - How to reduce the instances you have to enumerate?
- Greedy
  - Every time pick the current “best” choice
  - How to define best?
  - How to prove that optimal solution can be obtained in this way?

Example 1

- Coin Flipping
  - There are N (N<10000) rows of coins
  - Each row consists of 9 coins
  - They formulate a matrix of size N*9
  - Some coins are heads up and some are tails up
  - We can flip a whole row or a whole column every time
  - Find a flipping method that can make the number of “heads up” coins maximum

Example 2

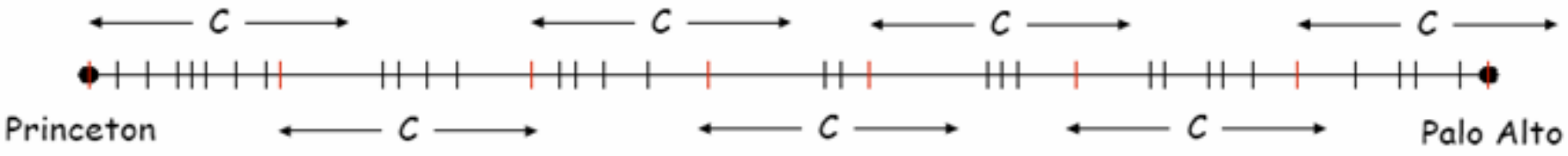
- Discrete Function
  - A discrete function is defined on {1,2,3,…,N}
  - Function values are between 2^{-32} and 2^{32}
  - Find two points on the function curve such that
    - Any function point between these two points are below the line connecting these two points
    - The slope of the line connecting these two points are as large as possible
  - $O(n^3)$ → $O(n^2)$
  - $O(n^2)$ → $O(n)$

Example 3 – Selecting Breakpoints

Selecting breakpoints.

- Road trip from Princeton to Palo Alto along fixed route.
- Refueling stations at certain points along the way.
- Fuel capacity = C.
- Goal: makes as few refueling stops as possible.

Greedy algorithm. Go as far as you can before refueling.

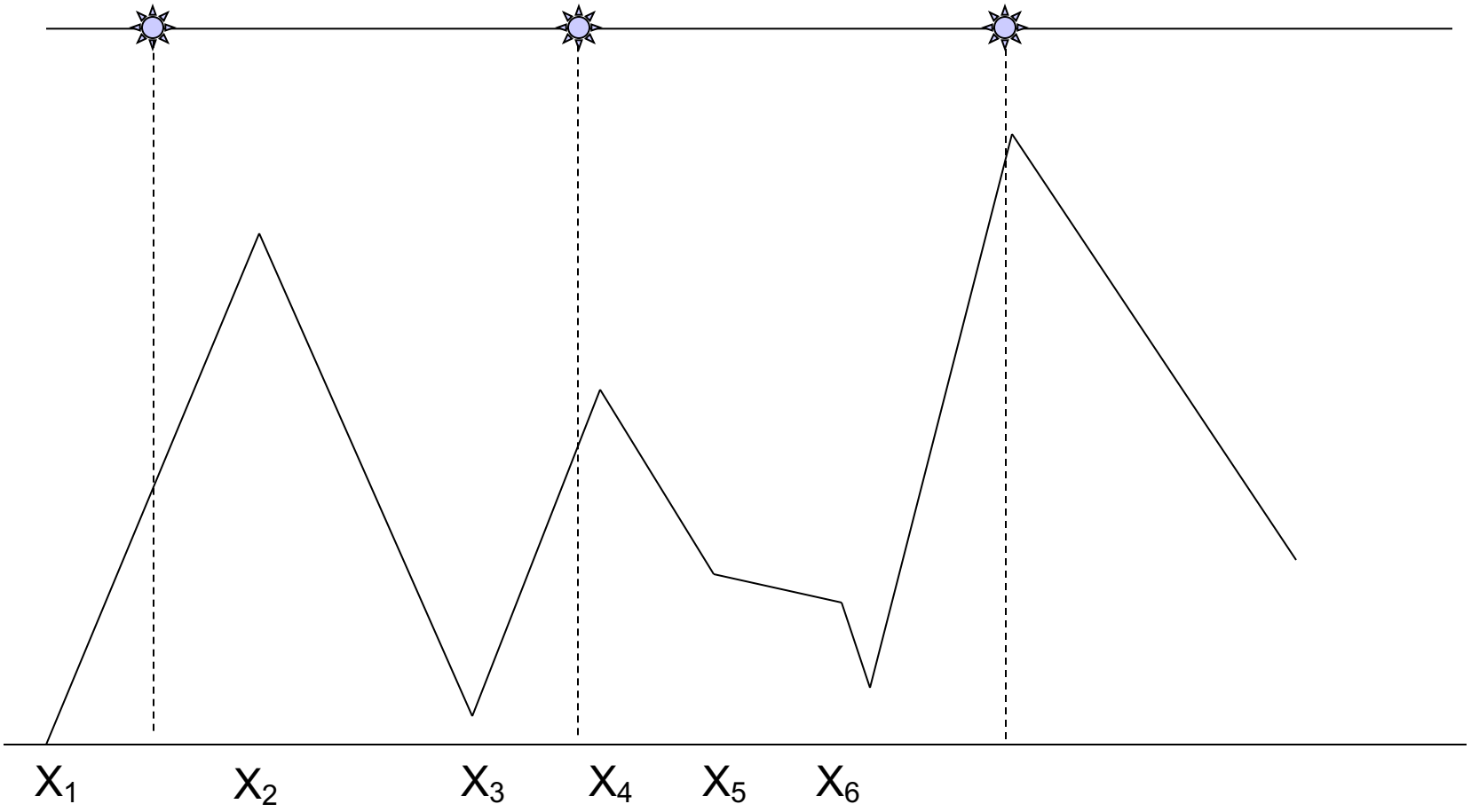


1	2	3	4	5	6	7

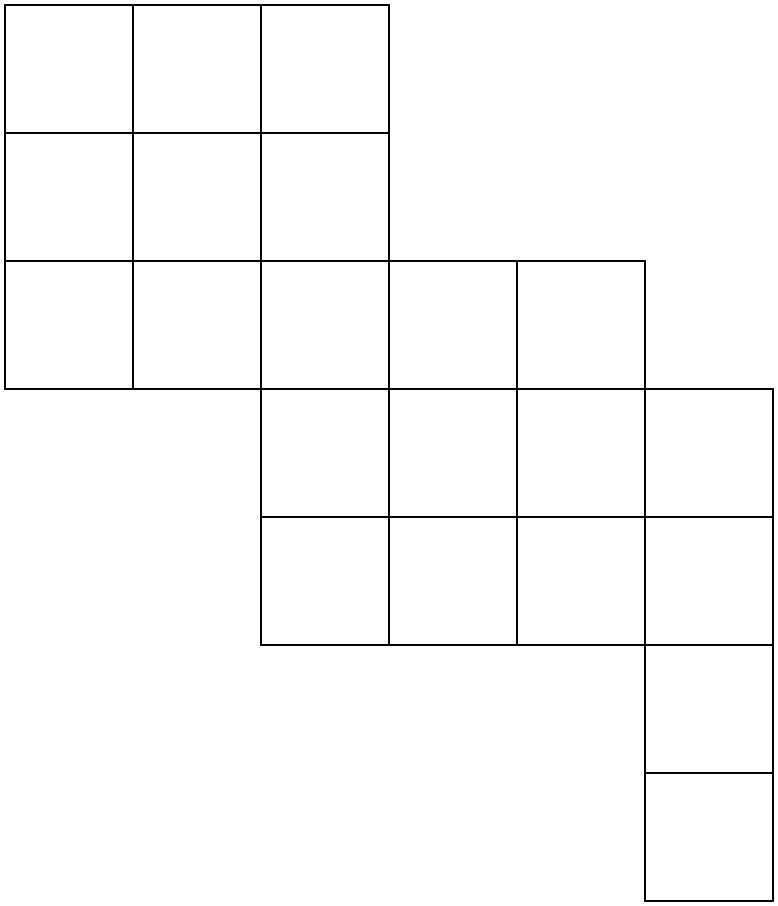
Example 4 – Gone Fishing

- There are *n* fishing lakes on a line
- Labeled 1, 2, 3, …, n from left to right
- H hours' fishing time
- Start from lake 1
- $5*T_i$ minutes to reach lake i+1 from lake i
- At the same lake i
  - *The first 5 minutes can produce F_i fishes*
  - *Every 5 minutes more will see a decrease of D_i in the number of fishes produced*
- *How to get the maximum number of fishes?*

Example 5 – Enlightened Landscape



Example 6 – The best way to put rooks (open for your try)



Game of Matches

- There is a pile of *n* matches
  - Player 1 pick some matches (*k* matches) from the pile but he cannot take them all
  - Player 2 pick *t* matches (at least 1, at most 2*k*)
  - Player 1 pick *s* matches (at least 1, at most 2*t*)
  - …
- The one who have no match to pick from the pile will lose
- Input: n
- Output: whether the first player can win