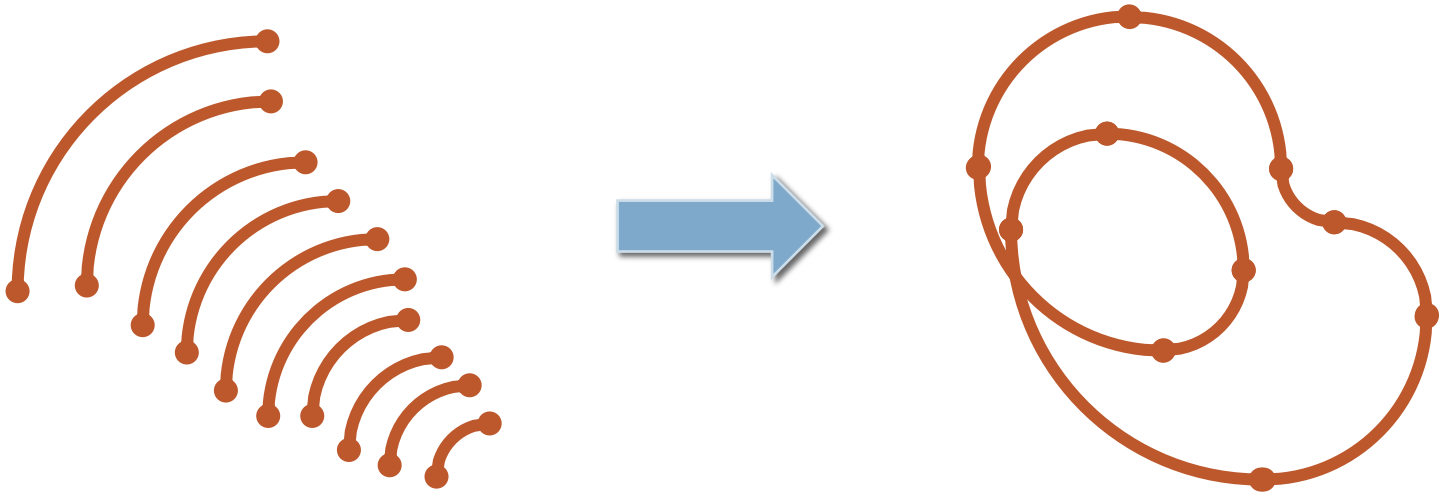
F: Make a Loop

PROPOSER: YOICHI IWATA
AUTHOR: YOICHI IWATA

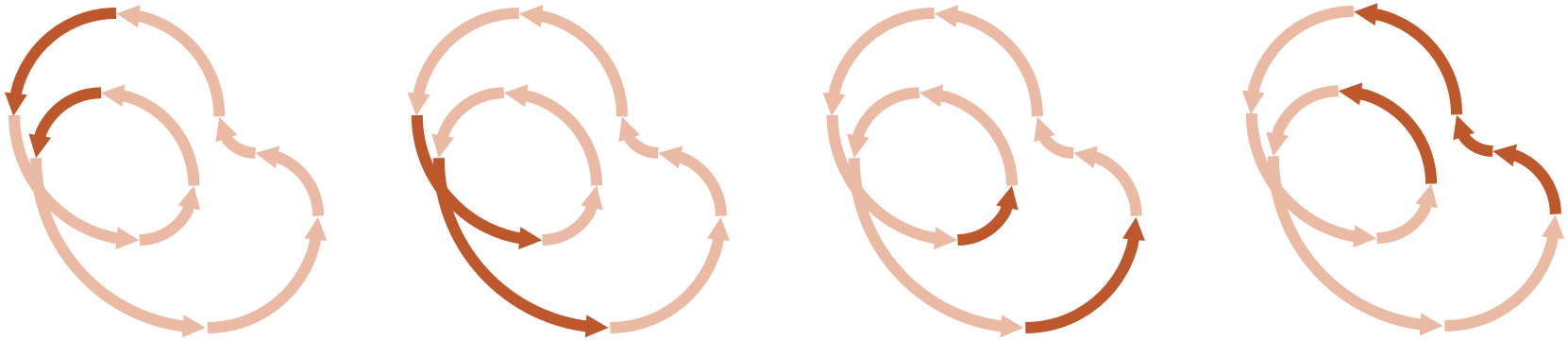
Problem

Given a set of arcs with a right central angle, is it possible to construct a single loop using all the arcs?



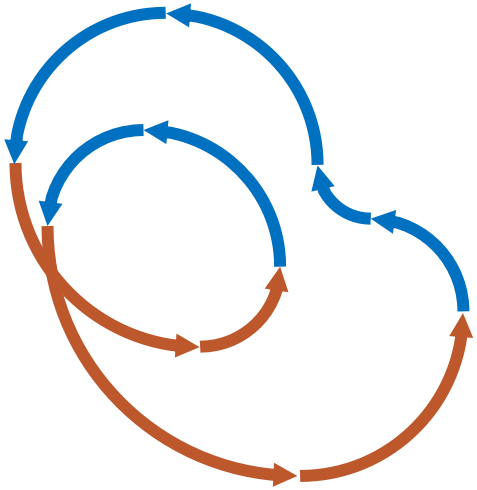
Necessary condition

Classify arcs into 4 groups.

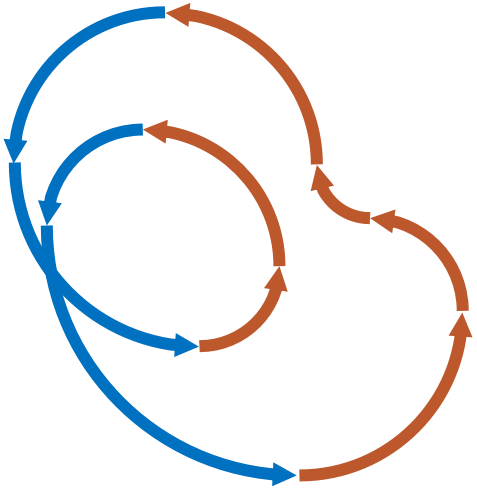


$S_{-,-}$ $S_{+,-}$ $S_{+,+}$ $S_{-,+}$

Necessary condition (1)

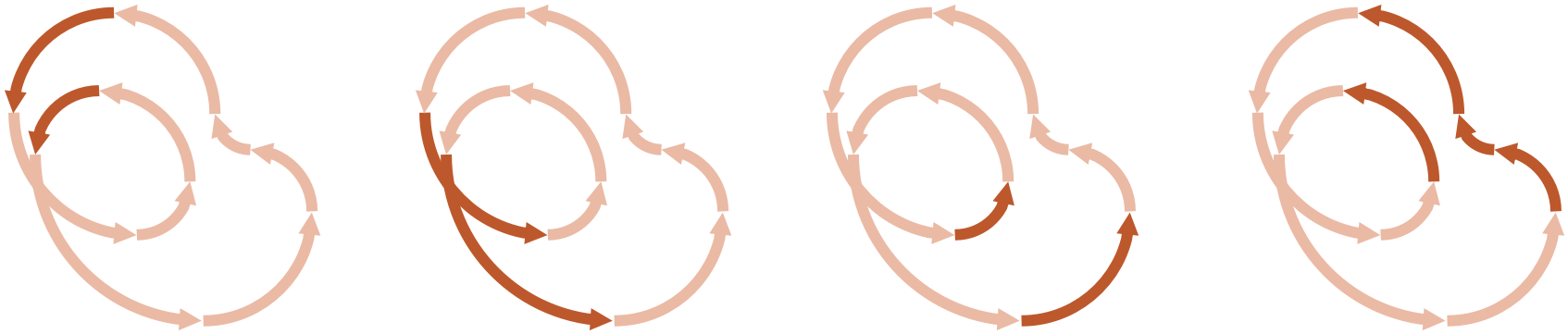


$$\sum_{i \in S_{-,-} \cup S_{-,+}} r_i = \sum_{i \in S_{+,-} \cup S_{+,+}} r_i$$



$$\sum_{i \in S_{-,-} \cup S_{+,-}} r_i = \sum_{i \in S_{-,+} \cup S_{+,+}} r_i$$

Necessary condition (2)



$$|S_{-,-}| \equiv |S_{-,+}| \equiv |S_{+,-}| \equiv |S_{+,+}| \pmod 2$$

$$S_{-,-}, S_{-,+}, S_{+,-}, S_{+,+} \neq \emptyset$$

These are sufficient

$$\{0, \dots, n-1\} = S_{-,-} \sqcup S_{-,+} \sqcup S_{+,-} \sqcup S_{+,+}$$

$$\sum_{i \in S_{-,-} \cup S_{-,+}} r_i = \sum_{i \in S_{+,-} \cup S_{+,+}} r_i$$

$$\sum_{i \in S_{-,-} \cup S_{+,-}} r_i = \sum_{i \in S_{-,+} \cup S_{+,+}} r_i$$

$$|S_{-,-}| \equiv |S_{-,+}| \equiv |S_{+,-}| \equiv |S_{+,+}| \pmod 2$$

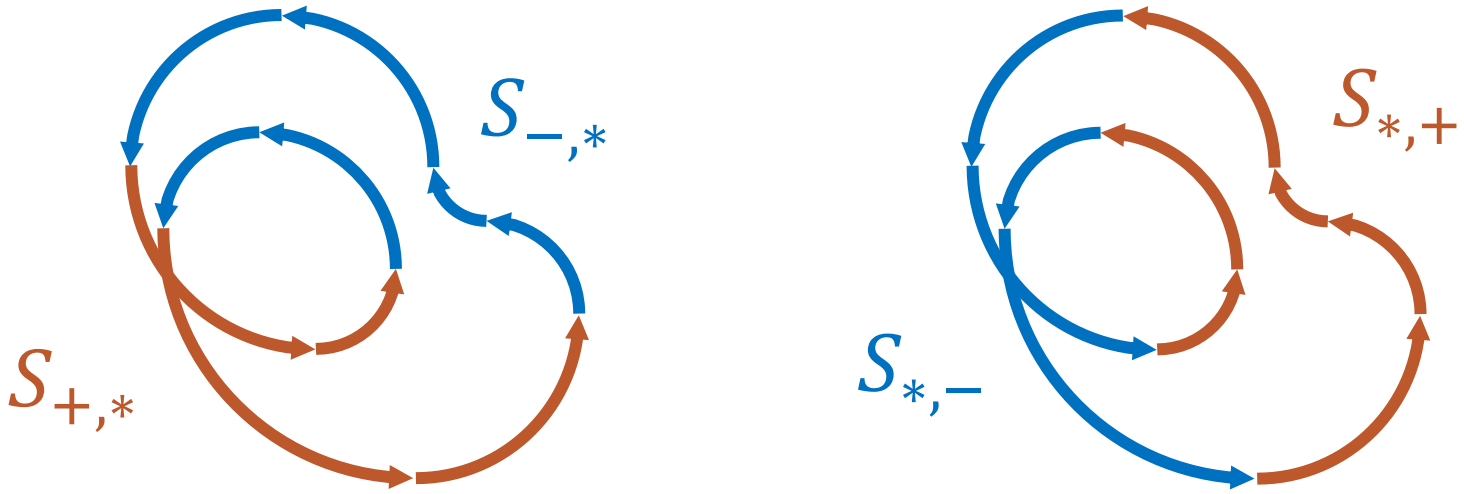
$$S_{-,-}, S_{-,+}, S_{+,-}, S_{+,+} \neq \emptyset$$

This can be solved in $O(n^3 r^2)$ time using subset sum DP, but too slow ☹

Equivalent conditions

Define $S_{-,*} := S_{-,-} \cup S_{-,+}$, …

Then $S_{-,-} = S_{-,*} \cap S_{*,-}$, …



Equivalent conditions

$$\{0, \dots, n-1\} = S_{-,*} \sqcup S_{+,*} = S_{*,-} \sqcup S_{*,+}$$

$$\sum_{i \in S_{-,*}} r_i = \sum_{i \in S_{+,*}} r_i$$

$$\sum_{i \in S_{*,-}} r_i = \sum_{i \in S_{*,+}} r_i$$

$$|S_{-,*}| \equiv |S_{+,*}| \equiv |S_{*,-}| \equiv |S_{*,+}| \equiv 0 \pmod 2$$

$$S_{-,*}, S_{+,*}, S_{*,-}, S_{*,+} \neq \emptyset, S_{-,*} \neq S_{*,-}$$

$\Leftrightarrow$ there are at least two even bisections

Algorithm

Compute the number of even bisections in $O(n^2 r)$ time using subset sum DP.

If the number ≥ 2, answer Yes; otherwise, answer No.

K: New Year Festival

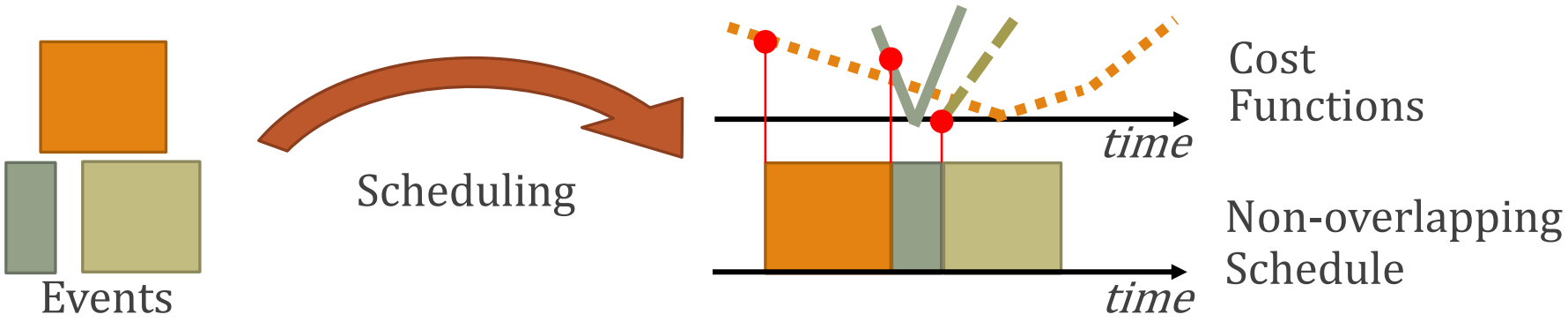
PROPOSER: SHINYA SHIROSHITA
AUTHOR: SHINYA SHIROSHITA

Problem Overview

You need to schedule n events.

Each event has a polygonal line cost function whose input is the start time.

You need to calculate the minimum total costs such that no two events have overlap.

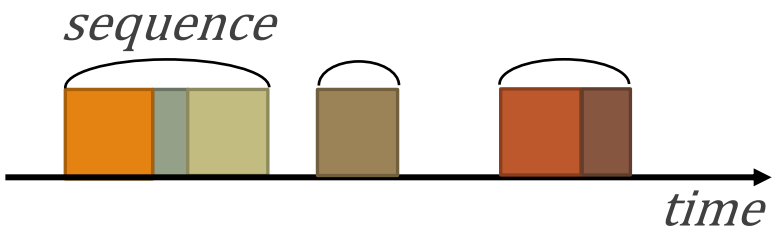


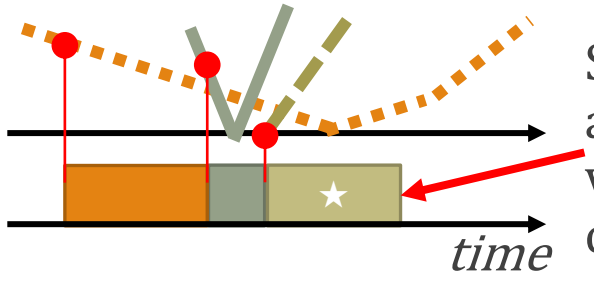
Consideration

Each solution consists of a series of consecutive events (*sequences*).
We can assume that **each sequence has an event whose start time is at a vertex of its cost function**.

Proof idea:
We can slide the sequence without increasing the total cost. This slide ends with either of

- Some event reaches a vertex of its cost function, or
- Collide with another sequence.
  → We can merge both sequences and continue sliding.



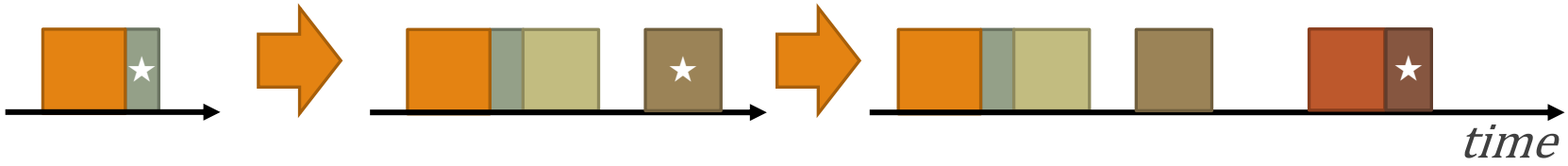


Some event starts at one of its vertices of the cost function

Solution

Dynamic Programming (DP) memorizing vertices of cost functions whose events' start times are at the vertices (*vertices with events*).

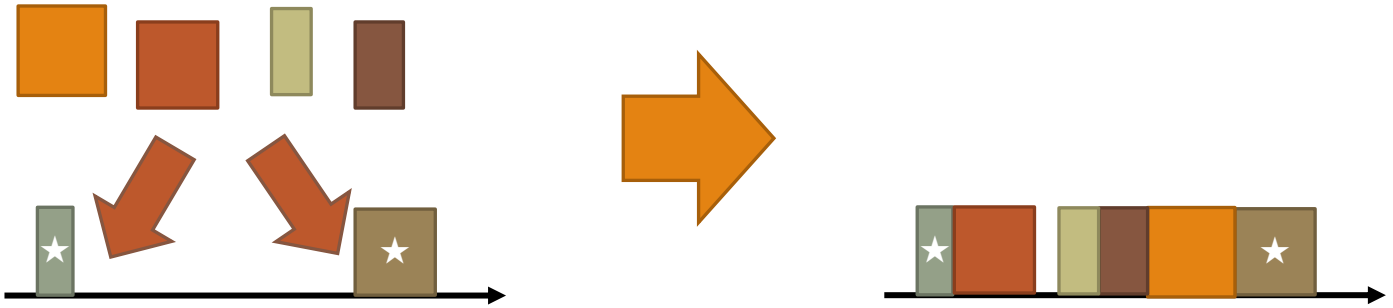
- DP State: [previous vertex w/ event][used event set].
- Events between vertices w/ events are appended to either the left or the right event. → Next slide



DP transitions
★: previous vertex with event

Solution

We can precompute the minimum cost for appending interval events to left/right in $O(m^2 3^n)$ where m is the total number of the vertices of the cost functions. $(3^n = (\text{left}, \text{right}, \text{outside})^n)$



The main DP transition part can also be calculated in $O(m^2 3^n)$. $(3^n = (\text{used}, \text{use now}, \text{not used})^n)$

C: Secure the Top Secret

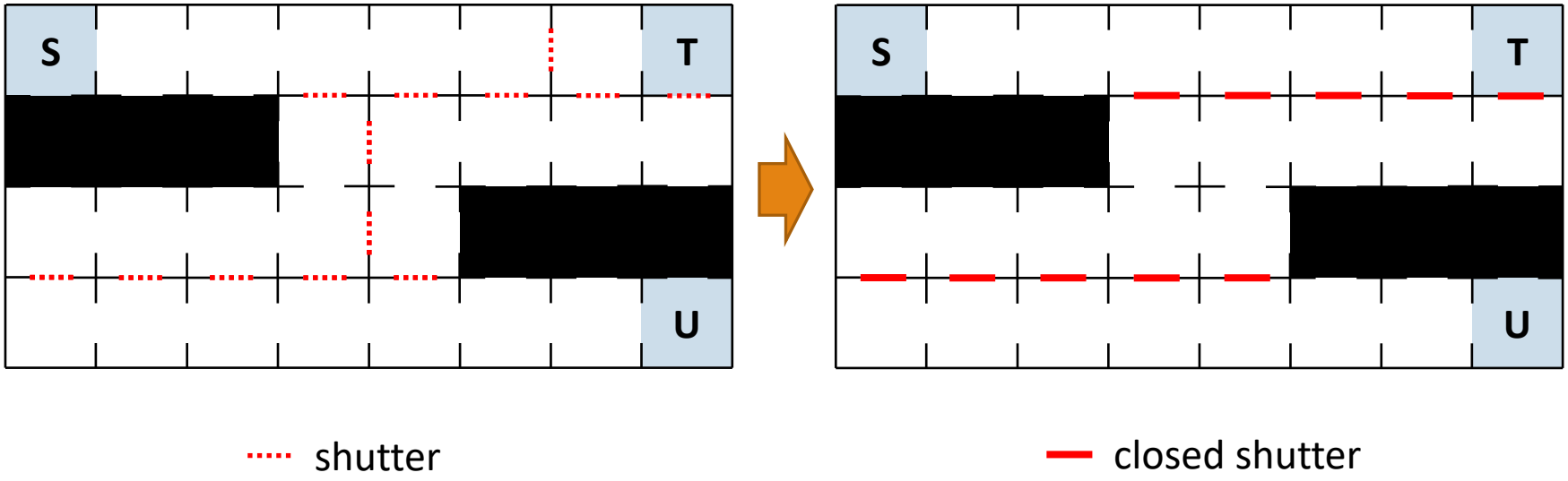
PROPOSER: MASATOSHI KITAGAWA
AUTHOR: MASATOSHI KITAGAWA

Problem

A grid graph with some special edges (shutters) is given.

Find the minimum number of shutters to close to satisfy

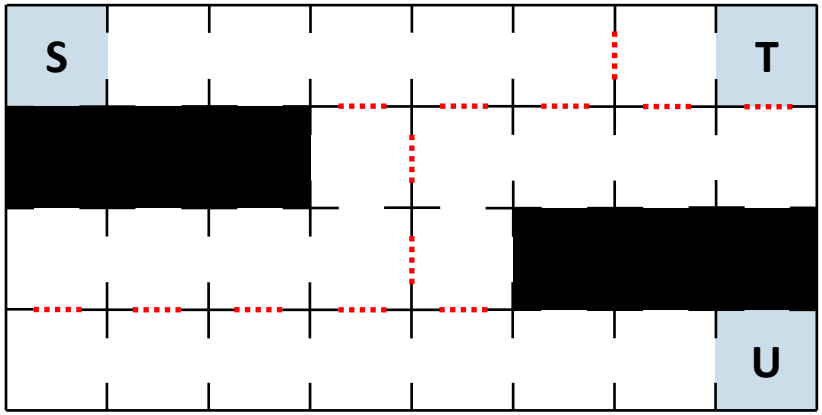
1. There exists a path from S to T with no closed shutters on it.
2. Any path from U to T contains at least **two** closed shutters.

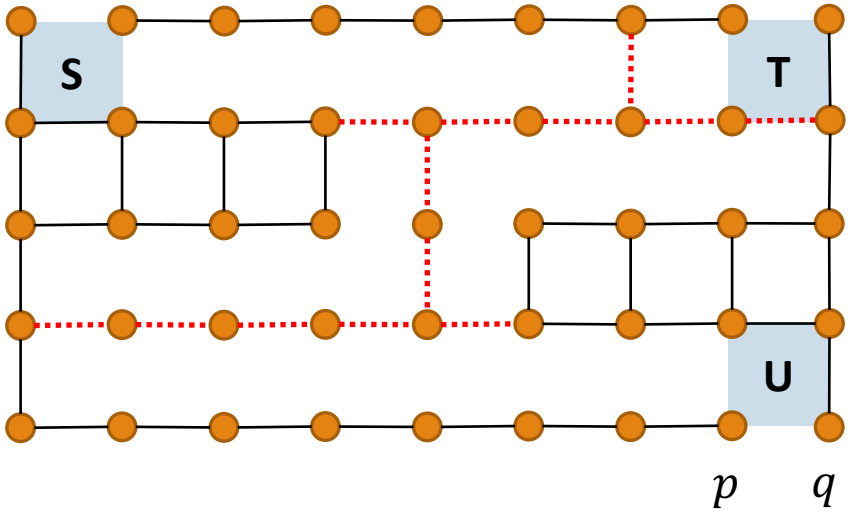


Solution

Find the minimum cost flow in the ‘dual’ graph.

Graph





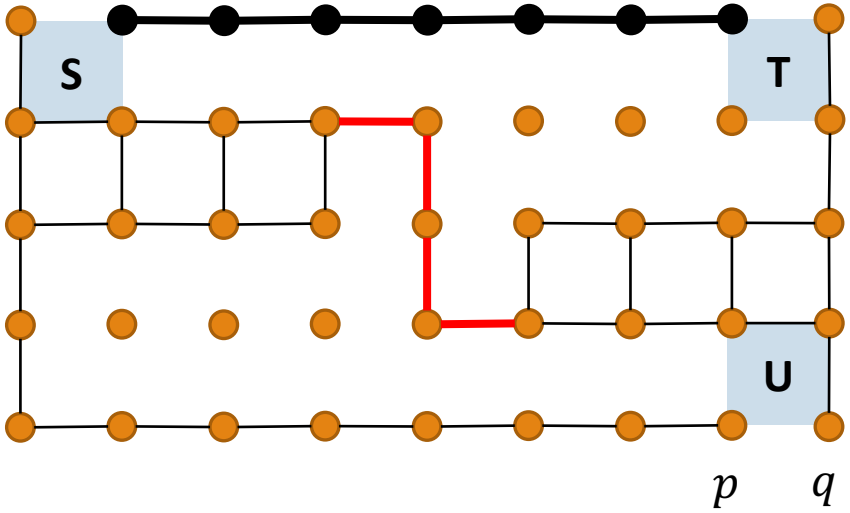
- vertex of a cell → vertex
- wall → black edge (cost 0, capacity ∞)
- shutter → dotted edge (cost 1, capacity 1)

Remove an outer wall (edge (p, q)) of U.
Remove an outer wall of S (and T).

Translation

Minimum U-T cut in the original graph
in which S and T belongs to the same connected component

Shortest p-q path with no black vertices



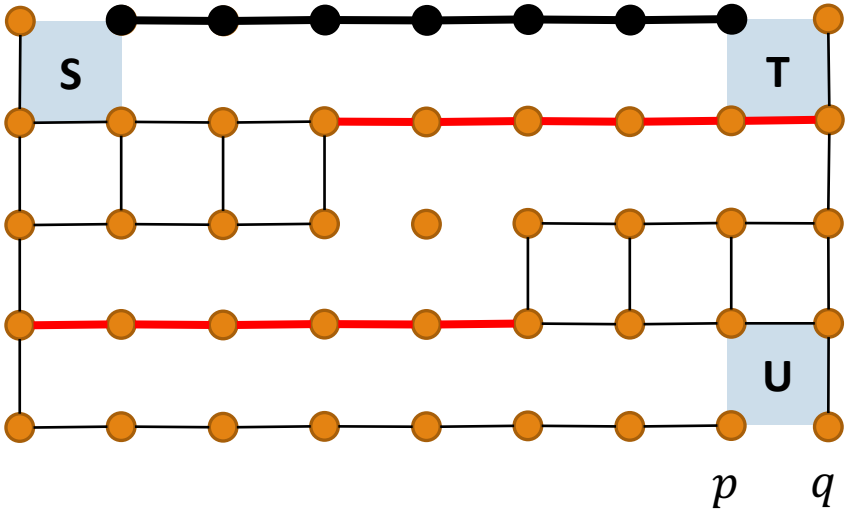
The bold edges = the walls used in the left-hand rule from S to T

Translation

The original problem

= Find the minimum cost of shutter-disjoint two p-q paths with no black vertices.

= Find the minimum cost of flows through no black vertices with amount of flow 2.



A: Hasty Santa Claus

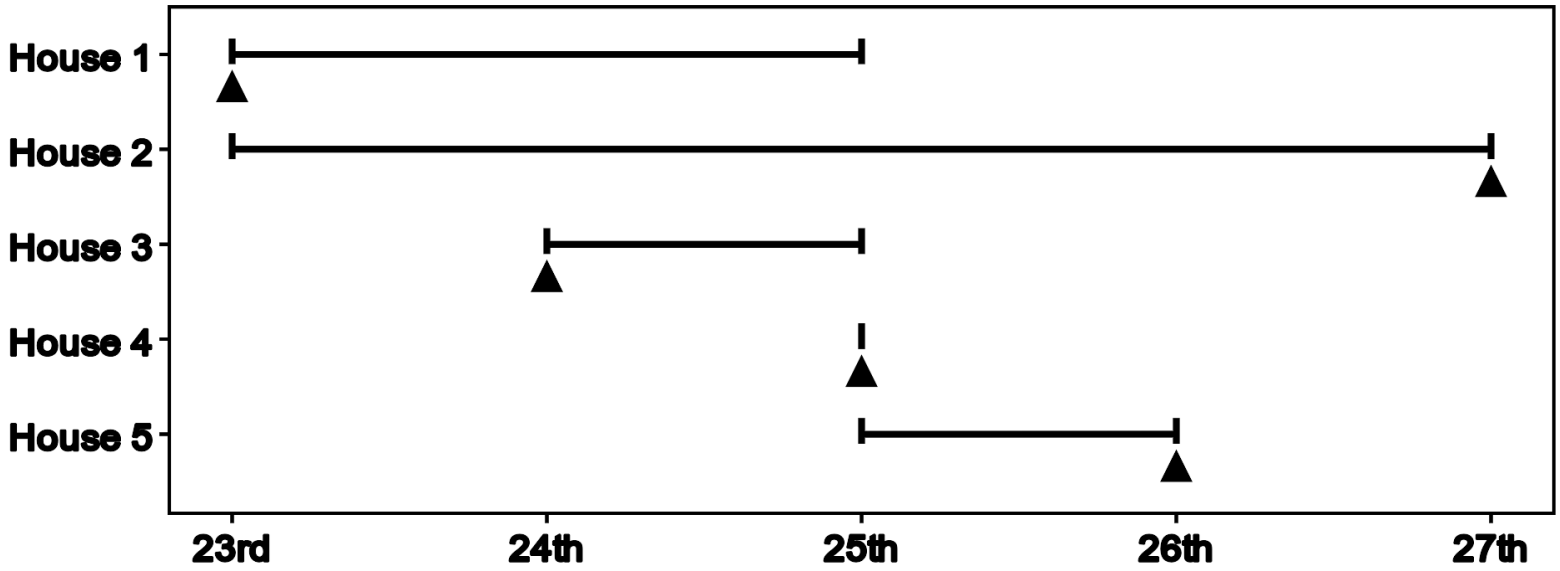
PROPOSER: KAZUHIRO INABA
AUTHOR: TOMOHIRO OKA

Problem

- **Given n intervals and an integer k**
  - $[a_i, b_i]$
- **Find a date assignment for each intervals**
  - $a_i \leqq date_i \leqq b_i$
- **The frequency of a date should be no more than k**
  - $\#\{i \mid date_i = d\} \leqq k$

Sample Input 1

- `n=5, k=1`



Solution

- **Greedy assignment**
  - **Select a house that has minimum b_i**

- Loop n times
  - i ⬅ the house not assigned yet has minimum b_i
  - d ⬅ earliest date that is in $[a_i, b_i]$ and $count_d < k$
  - $date_i$ ⬅ d
  - $count_d$ += 1

E: Incredibly Cute Penguin Chicks

PROPOSER: SOH KUMABE

AUTHOR: SOH KUMABE

Story

Count the way to cut given string into ICPC-ish substrings.

ICPCCPPPCIICCP

Design

The string consists of I,C,P is ICPC-ish if
- two of them appear same number of times, and
- the other one appears more than them.

ICPCCPPPCIICCP

Input and Output

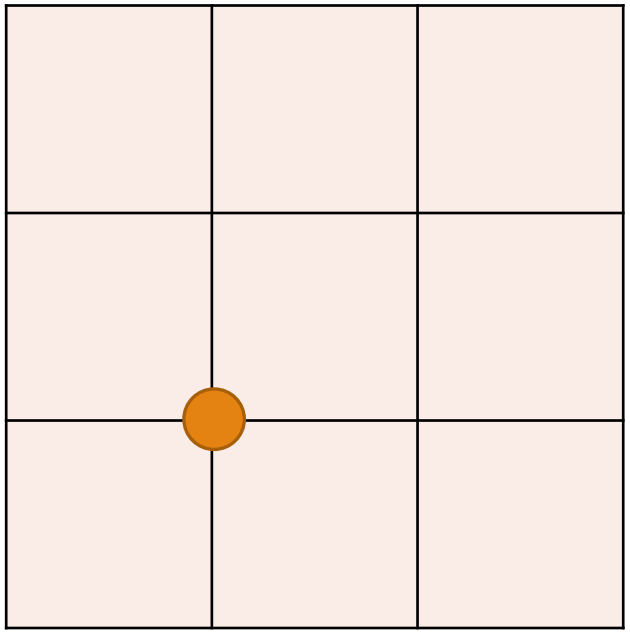
1. Input: string S.
2. Output: #ways to cut S into ICPC-ish substrings, modulo 998244353.

Solution

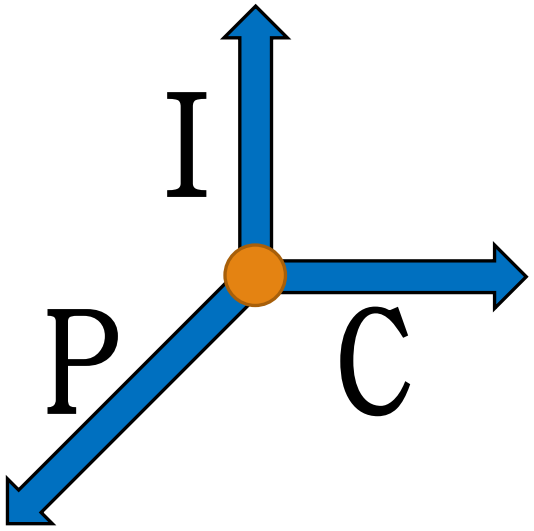
DP[t]: #ways to cut first t letters into ICPC-ish substrings

$O(|S|^2)$, TLE

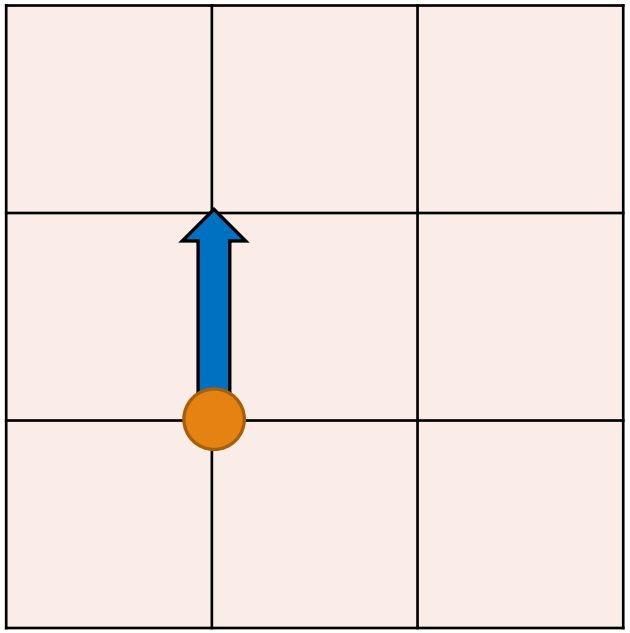
Solution

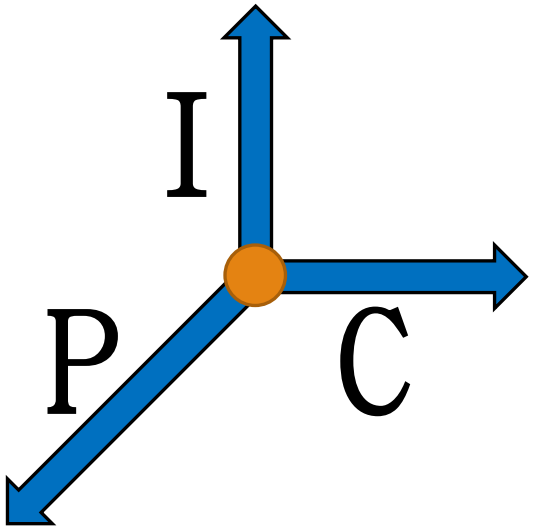


ICPC



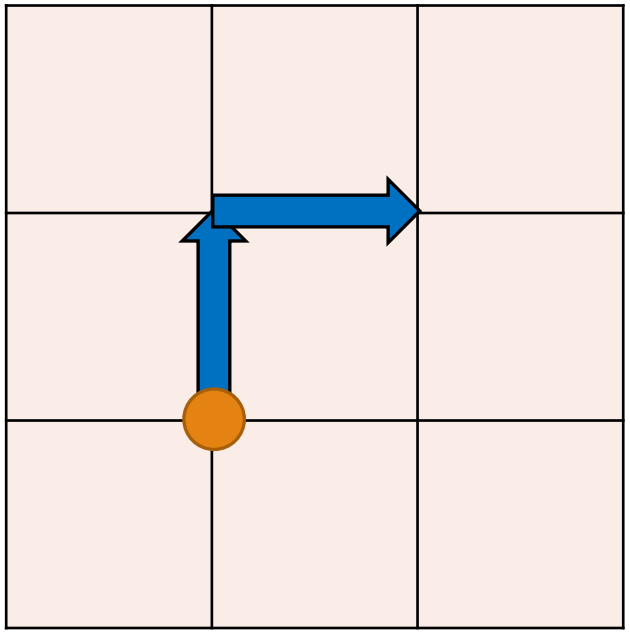
Solution

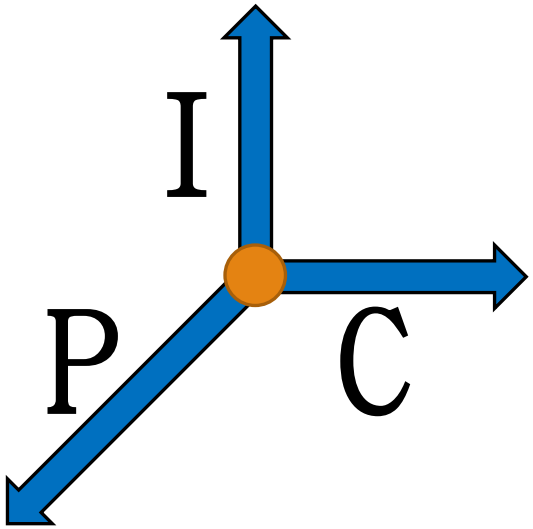




ICPC

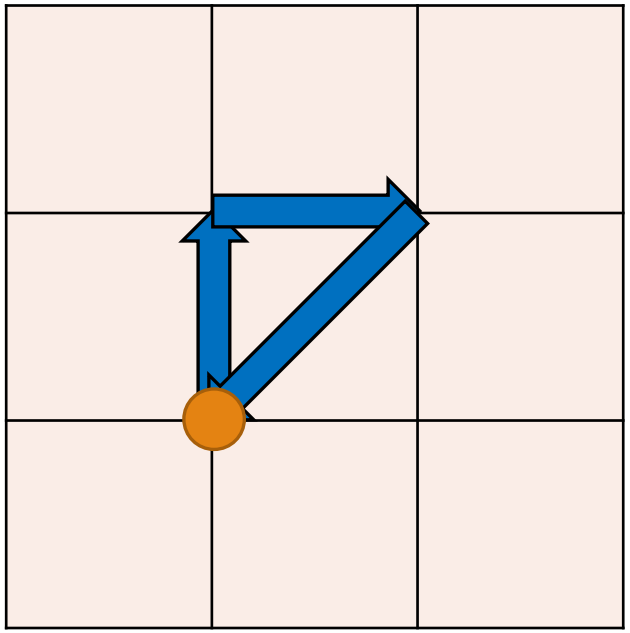
Solution

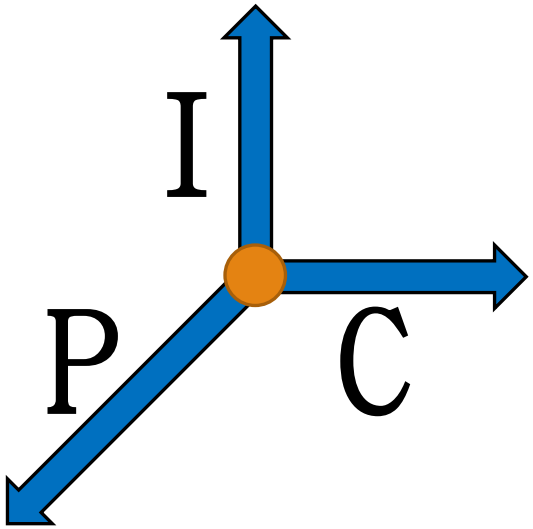




ICPC

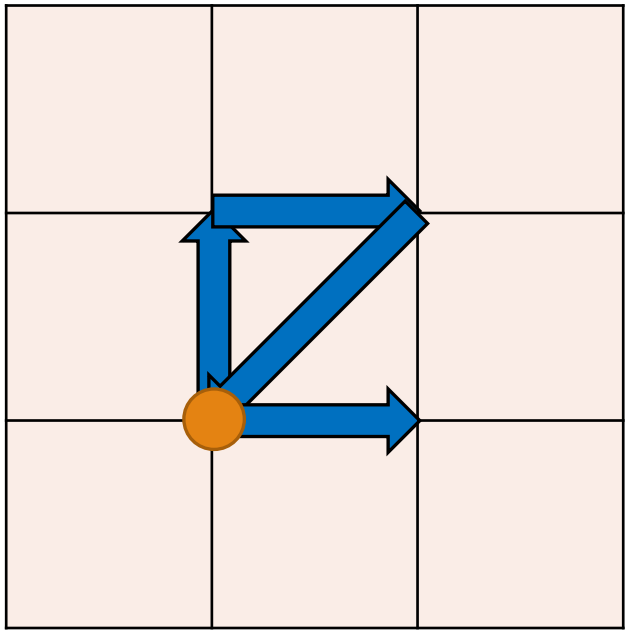
Solution

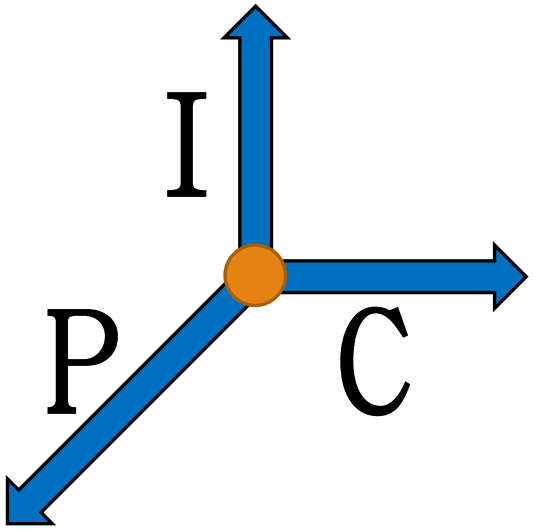




ICPC

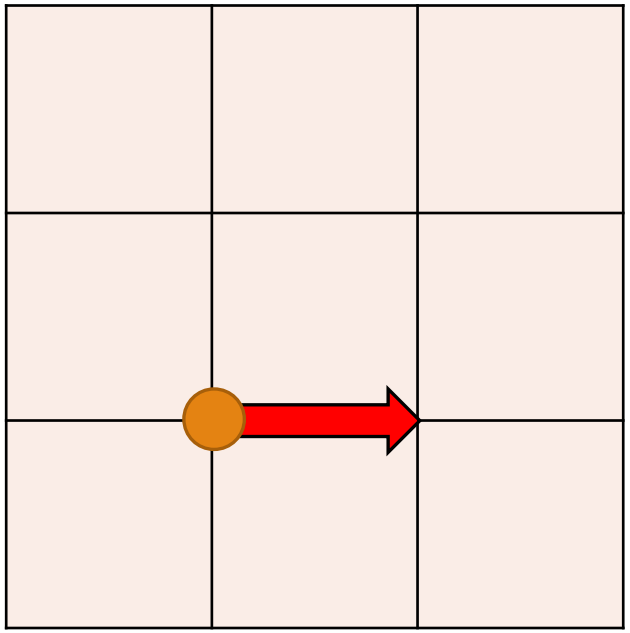
Solution

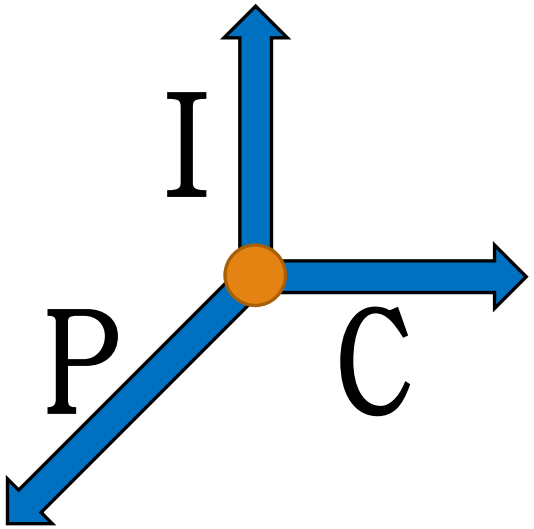




ICPC

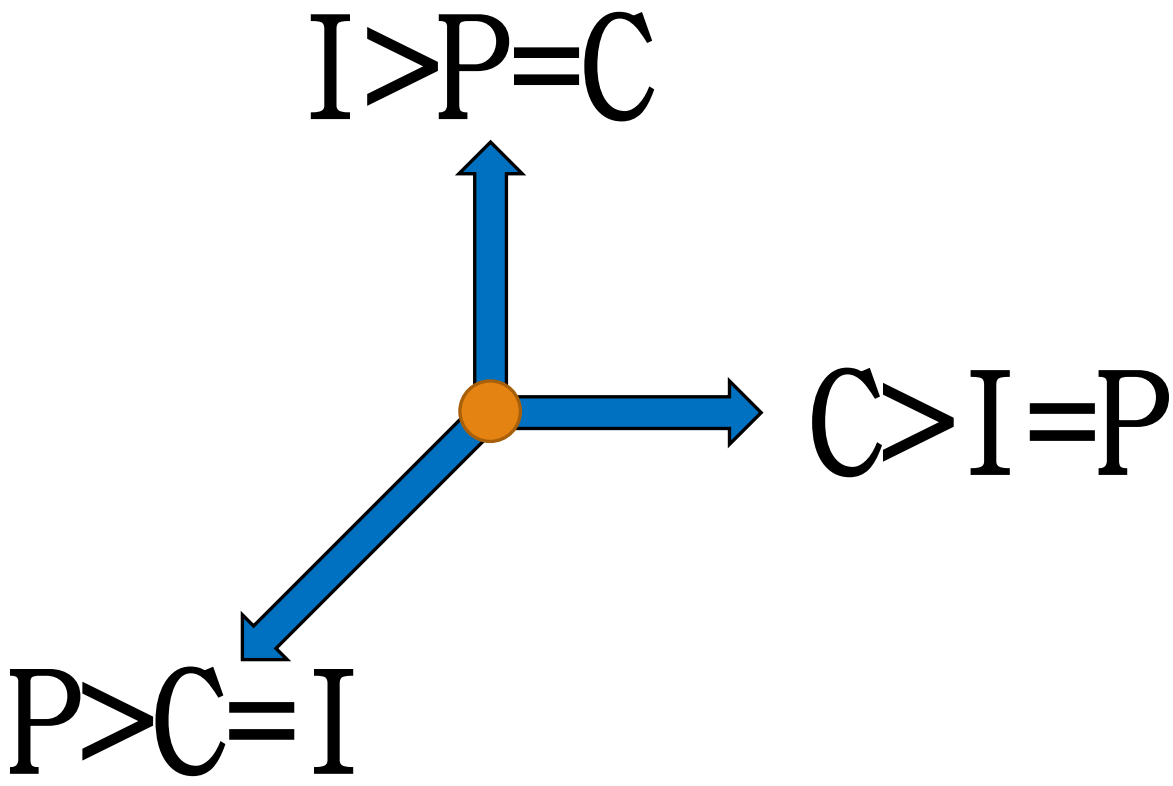
Solution





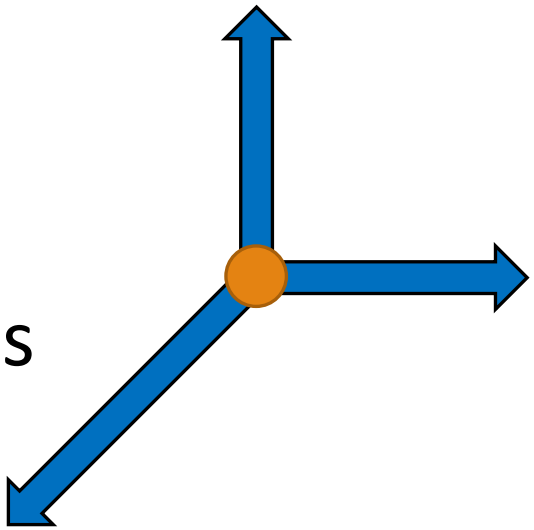
ICPC

Solution



Solution

DP[●]=sum of DP values on these directions



Solution

DP[●]=sum of DP values on these directions

Use Fenwick Tree to compute the sum

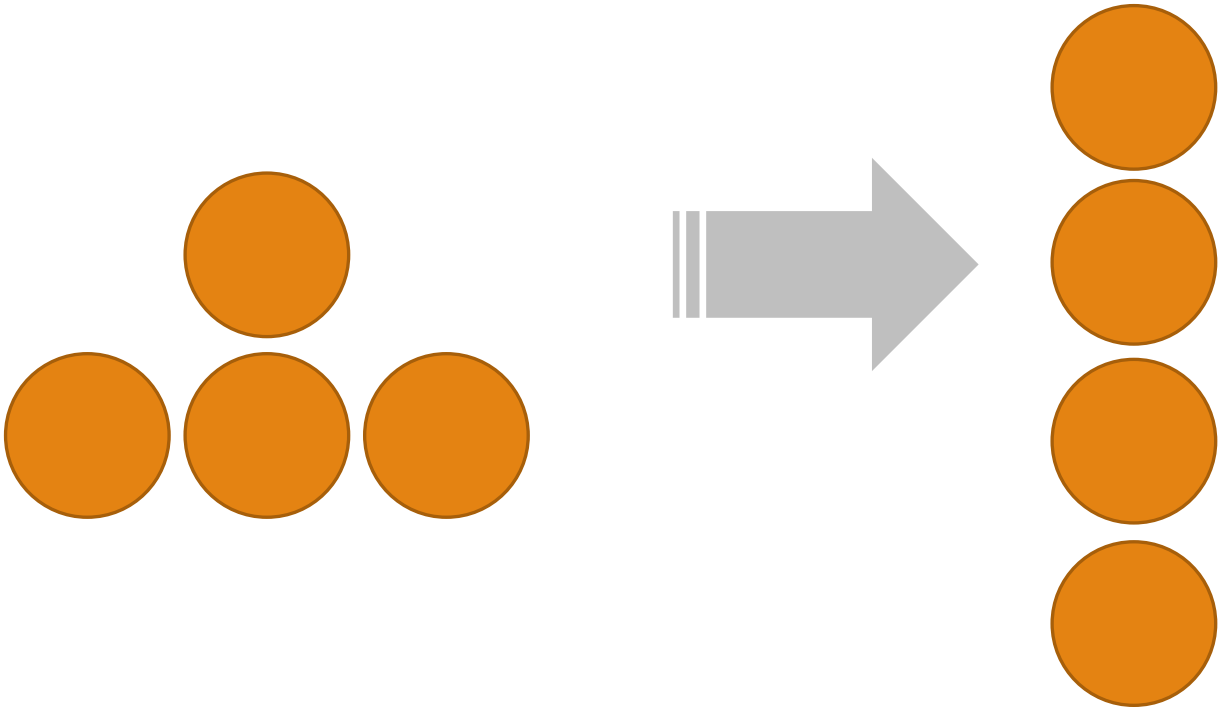
$O(|S| \log |S|)$, AC

D: Move One Coin

PROPOSER: KAZUHIRO INABA
AUTHOR: KAZUHIRO INABA

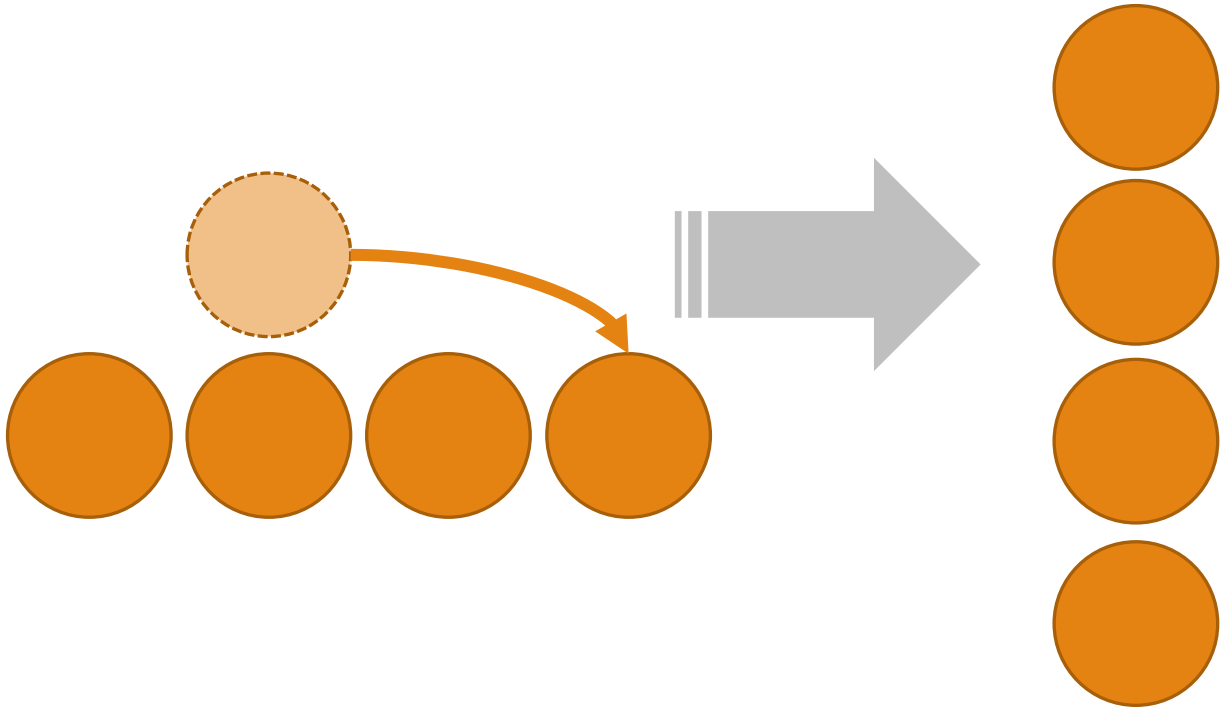
Problem

Match the left pattern to the right pattern (up to rotation), by **moving exactly one coin**.

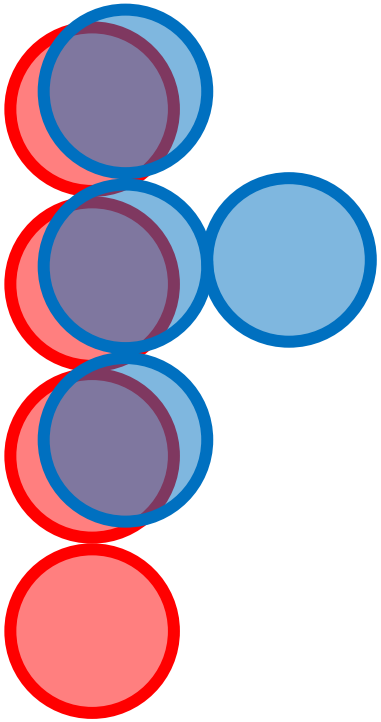


Problem

Match the left pattern to the right pattern (up to rotation), by **moving exactly one coin**.



Idea



If the src & dst patterns are already on the same location, we just need to scan and spot the two differences.

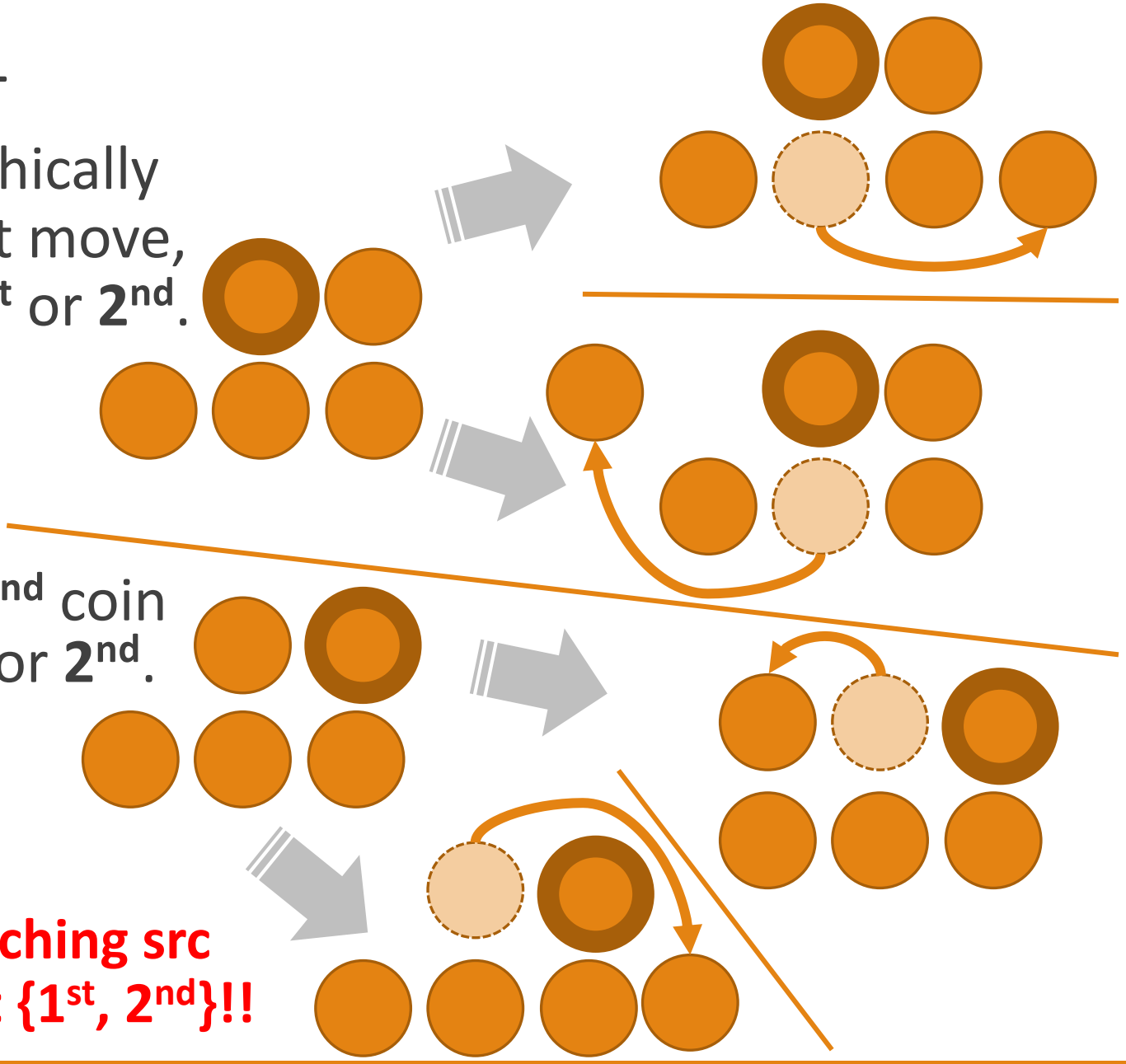
How to find the right…

- rotation? → try all 4 cases!
- parallel displacement?

Solution 1

If the lexicographically **1st coin** does not move, the coin stays **1st** or **2nd**.

If it moves, **2nd** coin becomes **1st** or **2nd**.

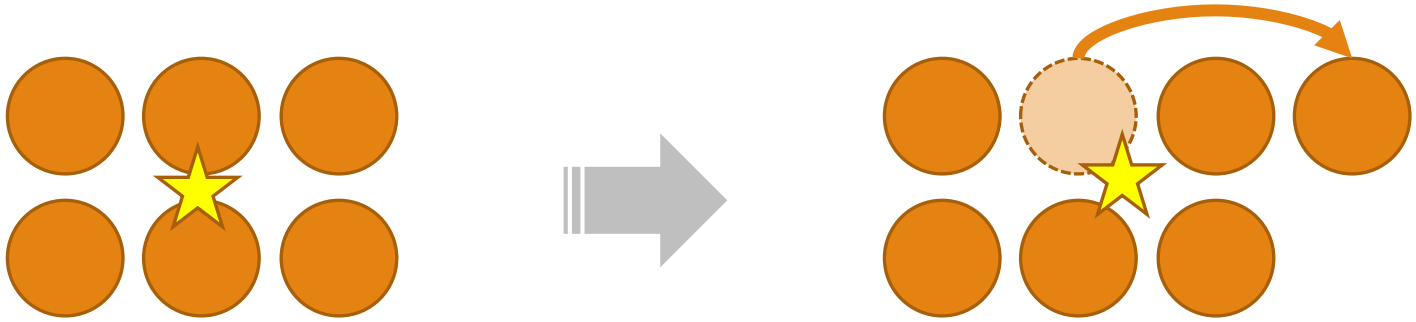


Try 4 offsets matching src {1st, 2nd} with dst {1st, 2nd}!!

Solution 2

- If coins aren't many (N<500), brute force search.
  - $O(N^3)$ by testing all (src, dst) pairs.

- If coins are many (N>=500), **one-coin move does not change the average of xy-coords too much.**
(Because max possible move is within ±1000.)
Try the offsets matching average points within ±2!



More Solutions…

- Two patterns are *very similar*, because after all they differ by only one coin.
- Exploit such similarity in some way, then you'll reach to a solution.
  - Many other approaches are possible.

B: Interactive Number Guessing

PROPOSER : MITSURU KUSUMOTO
AUTHOR : MITUSRU KUSUMOTO

Problem

- **The first interactive problem in this regional!**
- Judge has a secret number x.
- You should guess it by using queries, where you specify a number a and you receive digitsum($a + x$).
- Query limit ≤ 75
- $0 \leq a, x < 10^{18}$

Solution

Obtain d_0 = digitsum(x) by query a=0.

Now assume that you query a=500.

- If x is like x=..4.. or x=..3.. (“.” stands for arbitrary number in decimal notation), then digitsum(x+a)=d_0+5 is returned.
- If x is like x=..5.. or x=..6.., then digitsum(x+a) < d_0+5 is returned due to carry.

Using this observation, you can identify each digit by binary search.

Total query required is $1 + 18 \cdot \lceil \log_2 10 \rceil = 73$.

H: Cake Decoration

PROPOSER: AKIFUMI IMANISHI
AUTHOR: AKIFUMI IMANISHI

Problem

Find the number of combinations of four integers tuple (a,b,c,d):

*** a,b,c,d is different**
*** L <= a+b < R**
*** abcd <= X**
*** (a+1)bcd > X**
*** a(b+1)cd > X**
*** ab(c+1)d > X**
*** abc(d+1) > X**

Solution

Sort (a,b,c,d) by increasing order

* **a < b < c < d**
* **abcd <= X < abc(d+1)**

Find (*4) of sum of numbers of:

* **L <= a+b < R**
* **L <= a+c < R**
* **L <= a+d < R**
* **L <= b+c < R**
* **L <= b+d < R**
* **L <= c+d < R**

Solution

*** abcd <= X < abc(d+1)**

<=> d = floor(X / abc)

Algorithm:

```
For a in 1..X^(1/4)
  For b in 1..X^(1/3)
    Binary search:
    Count the number of c
```

Time complexity

Algorithm:

```
For a in 1..X^(1/4)
  For b in 1..X^(1/3)
    Binary search:
    Count the number of c
```

Loop: $\sum_{a=1}^{X^{1/4}} \left(\frac{X}{a}\right)^{1/3} \approx \int_{1}^{X^{1/4}} \left(\frac{X}{a}\right)^{1/3} da = O(X^{1/2})$

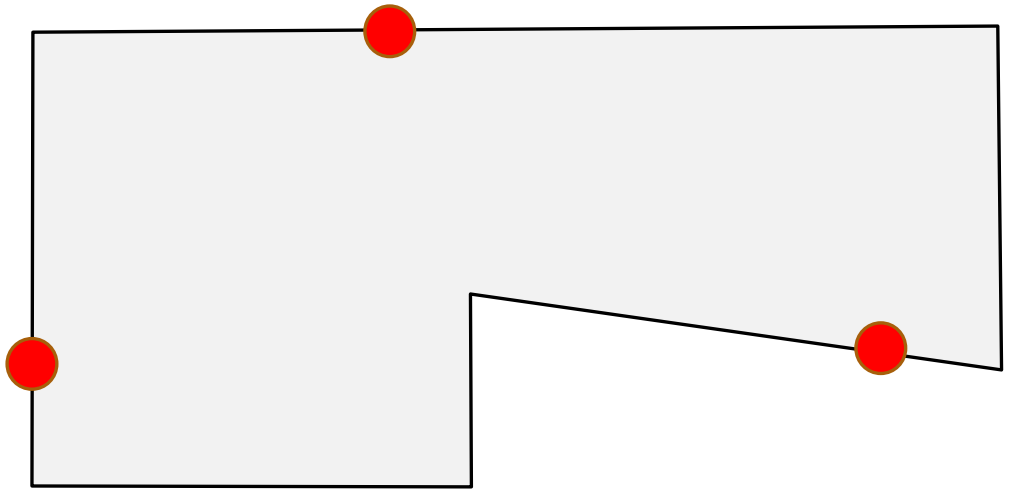
Time complexity: $O(\sqrt{X}\log X)$

J: Traveling Salesman in an Island

PROPOSER: SHUICHI HIRAHARA
AUTHOR: SHUICHI HIRAHARA

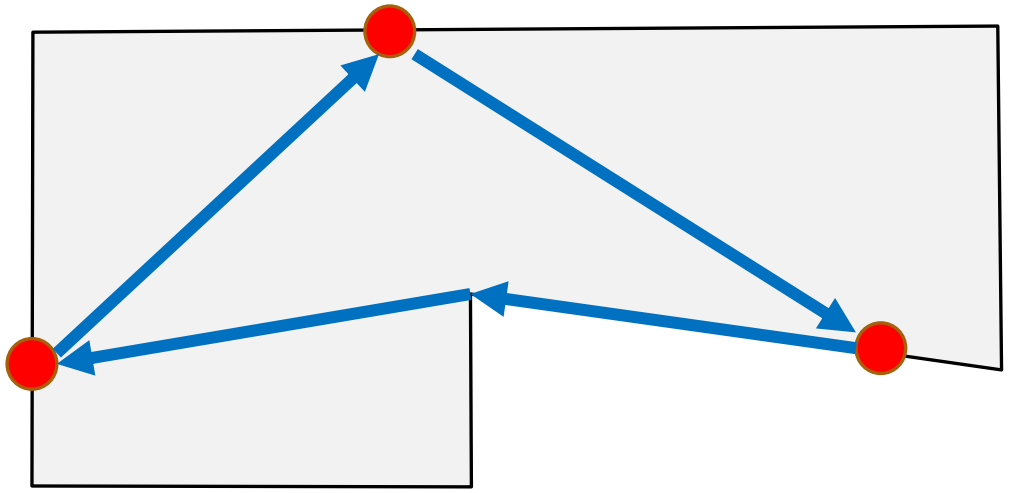
Problem

Given a simple polygon and points on its boundary, solve the Traveling Salesperson Problem (TSP).



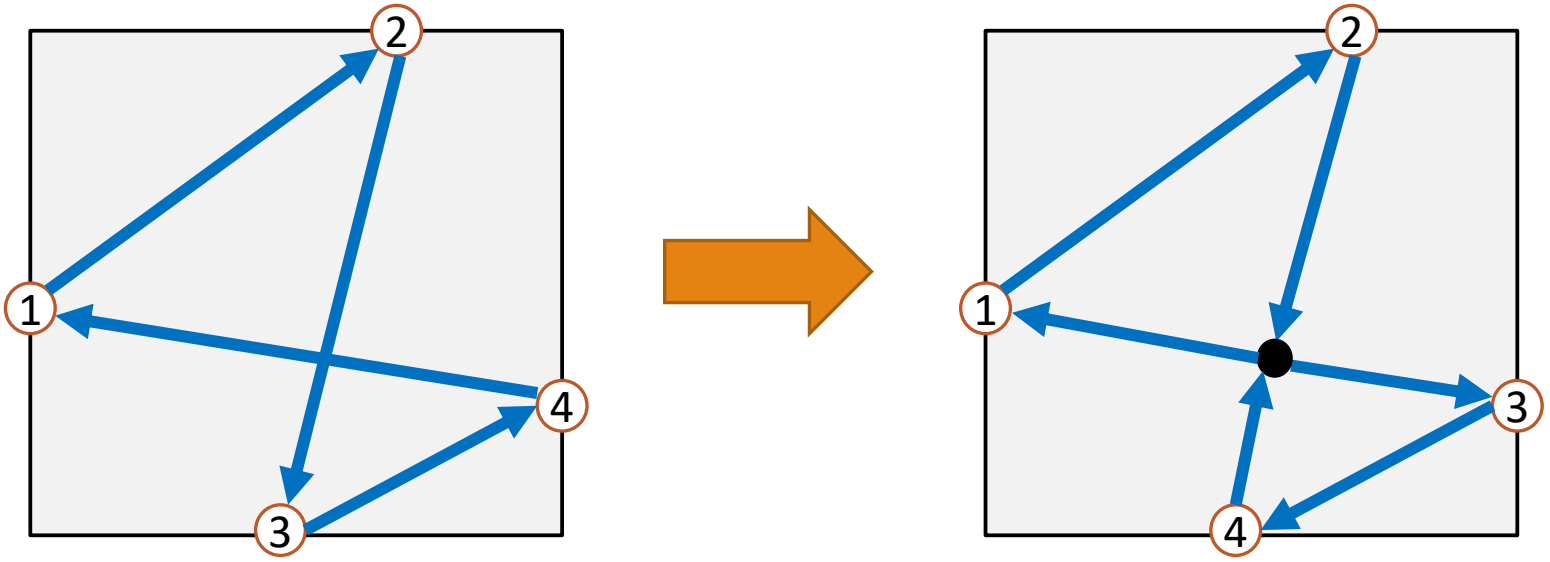
Problem

Given a simple polygon and points on its boundary, solve the Traveling Salesperson Problem (TSP).



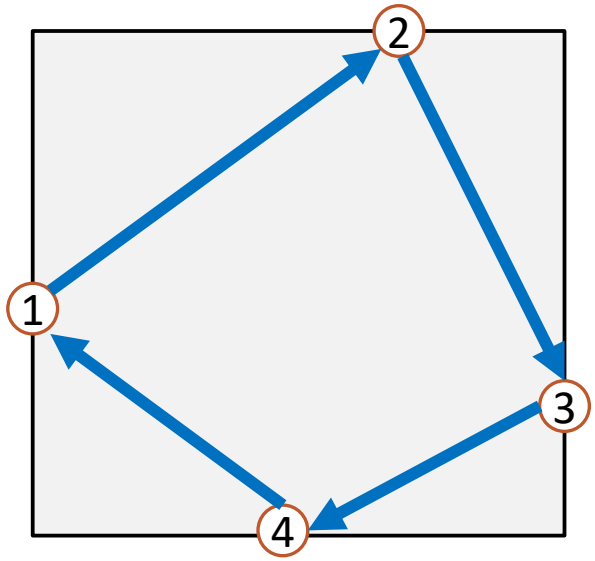
Solution

- TSP is NP-hard, but this special case is easy.
- Without loss of generality, the shortest tour visits the points in clockwise order.



Solution

1. Sort all the points in clockwise order.
2. Compute the shortest distance (inside the polygon) between i-th and $(i+1)$-th vertices.
3. Output the sum of the distances.



I: Quiz Contest

PROPOSER: RYOTARO SATO
AUTHOR: RYOTARO SATO

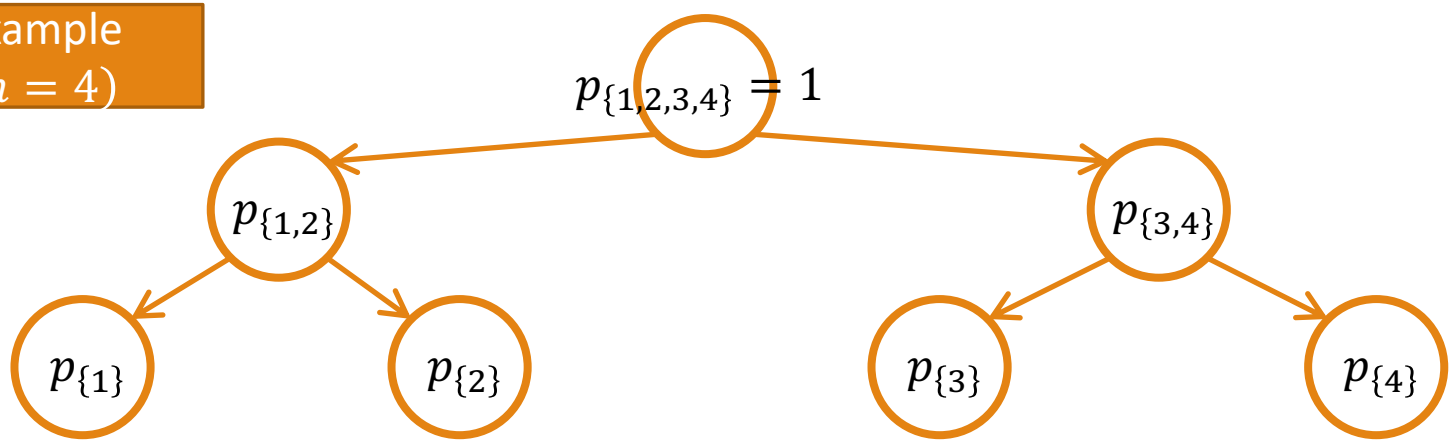
Problem Summary

- Quiz Contest by n participants is ongoing
- The first participant to answer the goal number of questions is winner
- Participant i can answer a_i of m questions not proposed yet
- Participant i has to answer additional b_i questions to win
- Count the number of question orders such that participant i will be the winner **for each** $i = 1, \ldots, n$, modulo $119 \times 2^{23} + 1$
- Constraints:
  - $1 \leq n \leq m \leq 2 \times 10^5$
  - Each question is answered by exactly one participant $(a_1 + \cdots + a_n = m)$.
  - Every participant has a chance to win $(b_i \leq a_i)$.

Solution Overview

- Build tree structure of participants
- We want to “distribute" winning probability of subtree p_S from root $(\{1, 2, \dots, n\})$ and finally get $p_{\{i\}}$ of each participant i

Example $(n = 4)$



- Instead of direct calculation of p_S, we consider **conditional probability**

$$p(\text{Winner is in } S \mid \text{Winner is decided by } i \text{ th question})$$

Solution Structure

- Key idea of fast counting: **Two step divide-and-conquer strategy**
  1. **Bottom-up DP to solve the auxiliary problem**: "When will the winner be decided?"
  2. **Top-down DP to solve the main problem**: "Who will be the winner and when?," by fully utilizing previous results.
- Both steps are significantly speed up by Fast Fourier Transform (FFT) and convolution!
  - Note: You can use 3 as the primitive root of multiplicative group of $\mathbb{F}_{119\times 2^{23}+1}$ to find primitive 2^d-th roots $(d \leq 23)$ for FFT.
- Overall complexity: $O(m \log n \log m)$

Notation

Introduce symbols:

- $U := (\text{set of all participants}) = \{1, \ldots, n\}$
- $a(S) := \sum_{i \in S} a_i$
  - $\rightarrow \quad a(U) = m$ holds.
- $f(S, i) := (\#\text{ of perms. of } a(S) \text{ questions s.t. the winner is decided just after } i\text{-th question})$

$$(i = 1, \ldots, a(S))$$

Step 1: Bottom-up DP to solve "When someone wins?"

- Start from $\{i\}$s for $i = 1, \ldots, n$. Merge them to make U.
- $f(S+T,\ \cdot)$ can be calculated from only $f(S,\ \cdot)$ and $f(T,\ \cdot)$:

$$f(S+T,i) = \binom{a(S+T)}{a(S)}^{-1} \sum_{j+k=i} \left(f(S,j) \left(\sum_{k'=k+1}^{a(T)} f(T,k') \right) \binom{j-1+k}{k} + f(T,k) \left(\sum_{j'=j+1}^{a(S)} f(S,j') \right) \binom{j+k-1}{j} \right) \binom{a(S+T)-i}{a(T)-k}$$

→ Convolution, $O(a(S+T)\log a(S+T))$

Example (Sample Input 2)

i	1	2	3	4	5	6
$f(\{1,2,3,4\},i)$	480	240	0	0	0	0

i	1	2
$f(\{1,2\},i)$	2	0

i	1	2	3	4
$f(\{3,4\},i)$	12	12	0	0

i	1
$f(\{1\},i)$	1

i	1
$f(\{2\},i)$	1

i	1	2
$f(\{3\},i)$	2	0

i	1	2
$f(\{4\},i)$	0	2

Participant 1 $(a_1, b_1) = (1, 1)$

Participant 2 $(a_2, b_2) = (1, 1)$

Participant 3 $(a_3, b_3) = (2, 1)$

Participant 4 $(a_4, b_4) = (2, 2)$

Step 2: Top-down DP to solve "When and Who wins?"

- Consider uniform distribution over $a(U)!$ permutations and introduce

$$g(S,i) := P\left(\text{The winner is in } S \;\middle|\; \begin{array}{l}\text{If the questions answered by } U-S \text{ are erased,}\\ \text{the winner is decided just after } i\text{-th question}\end{array}\right)$$

→ $g(S,\ \cdot)$ can be calculated from **ONLY** $f(T,\ \cdot)$ **AND** $g(S+T,\ \cdot)$:

$$g(S,i) = \binom{a(S+T)}{a(S)}^{-1} \sum_{j=i}^{i+a(T)} \binom{j-1}{i-1}\binom{a(S+T)-j}{a(S)-i}\left(\sum_{k=j-i+1}^{a(T)} f(T,k)\right) g(S+T,j)$$ → **Convolution again!**

- Finally, output $a(U)!\, g(\{i\},\ b_i)$ for each i .

i	1	2
$f(\{1,2\},i)$	2	0
$g(\{1,2\},i)$	$\frac{7}{15}$	$*$

i	1	2	3	4	5	6
$f(\{1,2,3,4\},i)$	480	240	0	0	0	0
$g(\{1,2,3,4\},i)$	1	1	$*$	$*$	$*$	$*$

i	1	2	3	4
$f(\{3,4\},i)$	12	12	0	0
$g(\{3,4\},i)$	$\frac{2}{3}$	$\frac{2}{5}$	$*$	$*$

i	1
$f(\{1\},i)$	1
$g(\{1\},i)$	$\frac{7}{30}$

i	1
$f(\{2\},i)$	1
$g(\{2\},i)$	$\frac{7}{30}$

i	1	2
$f(\{3\},i)$	2	0
$g(\{3\},i)$	$\frac{7}{15}$	$*$

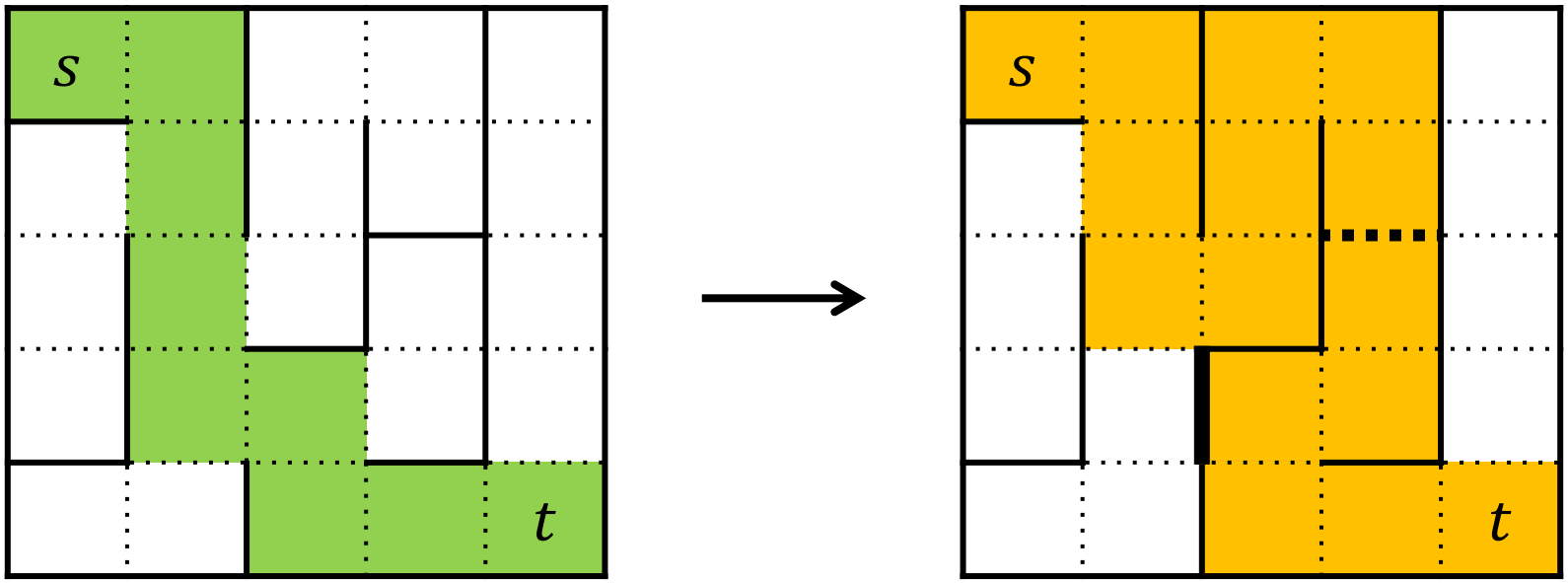
i	1	2
$f(\{4\},i)$	0	2
$g(\{4\},i)$	$*$	$\frac{1}{15}$

G: Remodeling the Dungeon

PROPOSER: TOMOHARU UGAWA
AUTHOR: YUTARO YAMAGUCHI

Story

Enhance the security of the castle by remodeling the dungeon.

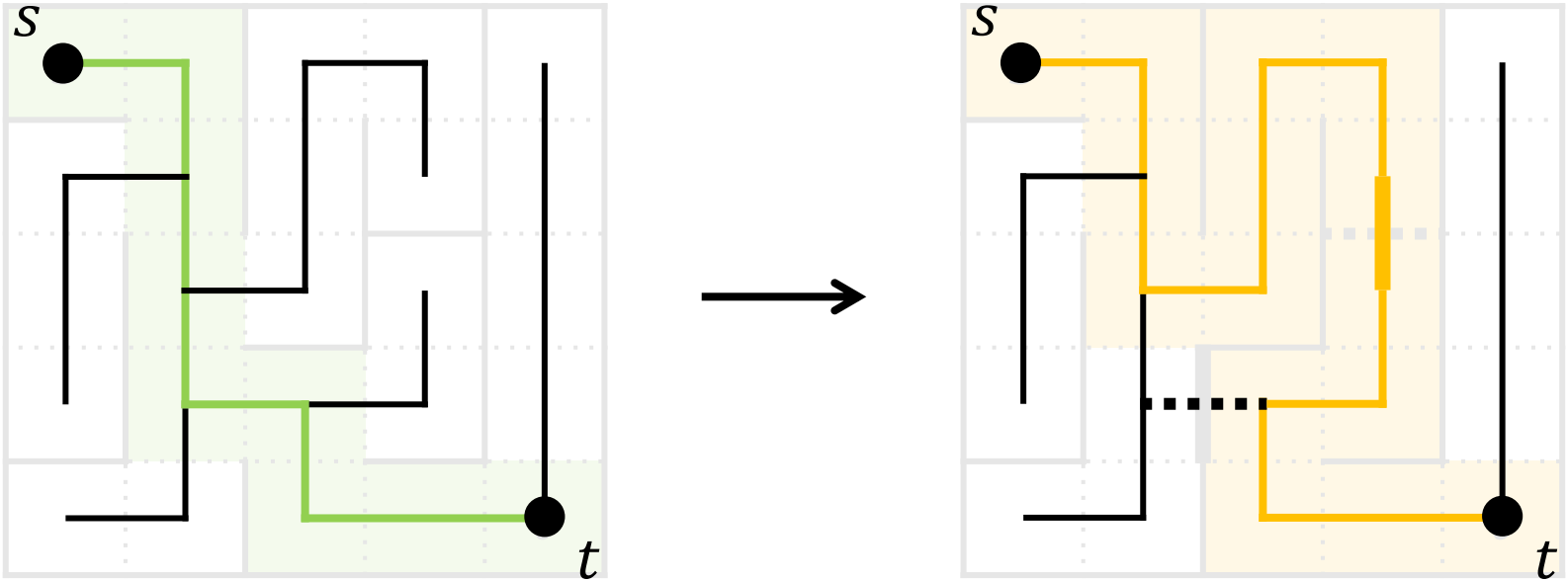


9

15

Story

Enhance the security of the castle by remodeling the dungeon.



9

15

$$n = h \times w \leq 2.5 \times 10^5$$
$$\ell < 2n \leq 5 \times 10^5$$

Problem

Given a tree of $n = h \times w$ vertices.

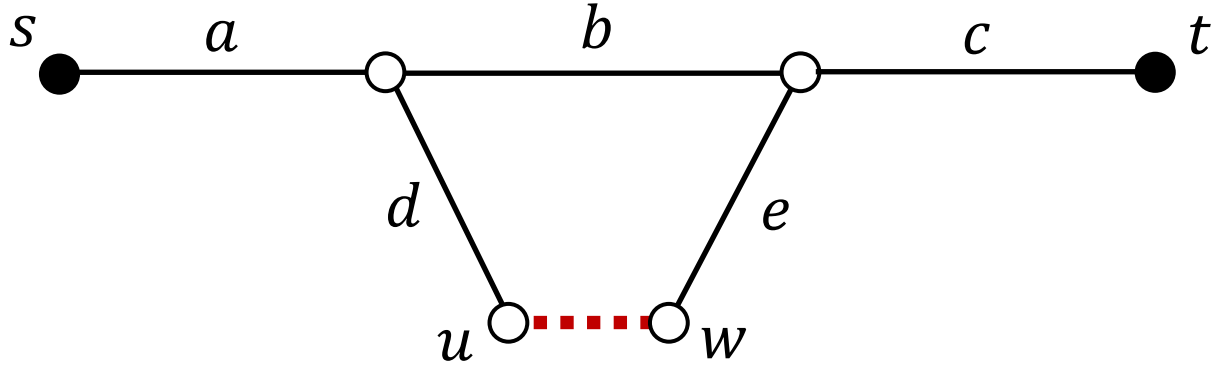
Given $\ell = (h-1) \times w + h \times (w-1)$ possible new edges.

Maximize the distance between s and t by removing one edge and adding one new edge instead so that the result is also a tree.

$$n = h \times w \leq 2.5 \times 10^5$$
$$\ell < 2n \leq 5 \times 10^5$$

Solution

Maximize the distance between s and t by removing one edge and adding one new edge instead so that the result is also a tree.



The new route through $\{u, w\}$ consists of $a + d + 1 + e + c$ edges.

$$\begin{aligned} &\mathrm{dist}(s,u) + \mathrm{dist}(t,w) + 1 \\ &= \mathrm{dist}(s,w) + \mathrm{dist}(t,u) - 2b + 1 \\ &< \mathrm{dist}(s,w) + \mathrm{dist}(t,u) + 1 \end{aligned}$$

$n = h \times w \leq 2.5 \times 10^5$

$\ell < 2n \leq 5 \times 10^5$

Solution

Maximize the distance between s and t by removing one edge and adding one new edge instead so that the result is also a tree.

1. Compute $\mathrm{dist}(s, v)$ and $\mathrm{dist}(t, v)$ for all vertices v. $\Theta(n)$ time
2. For each possible new edge $\{u, w\}$,
   if $\mathrm{dist}(s, u) + \mathrm{dist}(t, w) \neq \mathrm{dist}(s, w) + \mathrm{dist}(t, u)$,
   then the minimum of them + 2 is a candidate of the answer. $\Theta(\ell)$ time

The new route through $\{u, w\}$ consists of $a + d + 1 + e + c$ edges.

$\mathrm{dist}(s, u) + \mathrm{dist}(t, w) + 1$

$= \mathrm{dist}(s, w) + \mathrm{dist}(t, u) - 2b + 1$

$< \mathrm{dist}(s, w) + \mathrm{dist}(t, u) + 1$