中国计算机学会
China Computer Federation

IOI Problems: Past, Present, and Future

Richard Peng (彭泱)

NOI 2021 冬令营

What this talk is about

- Origins of the modern IOI
- Problem setting process
- Source of problems
- Future predictions

- What
- How, who, why
- Where
- When

What this talk is NOT about

- What problems should be
- What contests should be / how to change contests
- How to prepare for this type of contests

What this talk is about

http://ioi.te.lv/locations/ioi02/contest/report.pdf

IOI 2002
Yong-In, Korea

INTRODUCTION

First we will explain the procedure used by the ISC (International Scientific Committee) and the HSC (Host Scientific Committee) to prepare and select the competition tasks. Then we will give an overview of the tasks, including solutions for and the basic theory behind them.

After the "Call for Tasks" announcement our HSC received 20 problems from Korea and abroad. Of these, we excluded 4 problems which were not mature enough to be used in IOI-style competition tasks. The HSC then prepared all solutions and test data sets (20 test cases for each task) and discussed these sets with the ISC members in July. The meetings held concerning IOI2002 task preparation were follows:

IOI 02: Frog

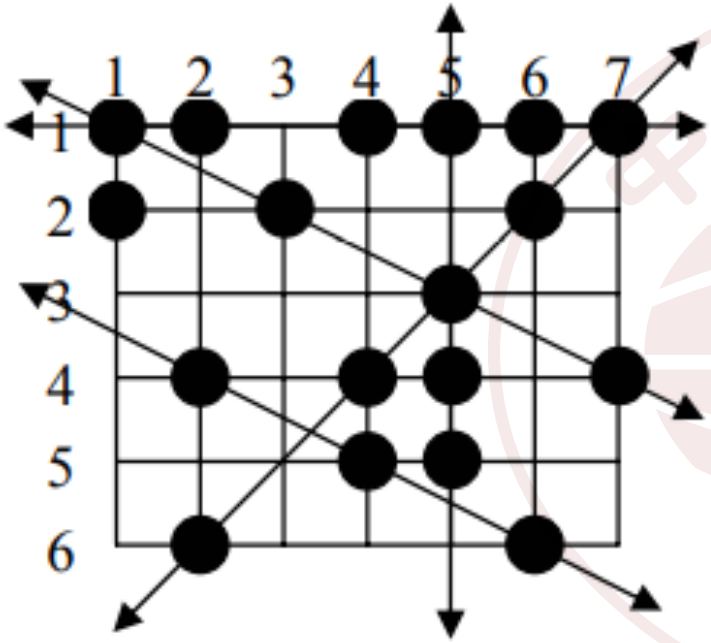
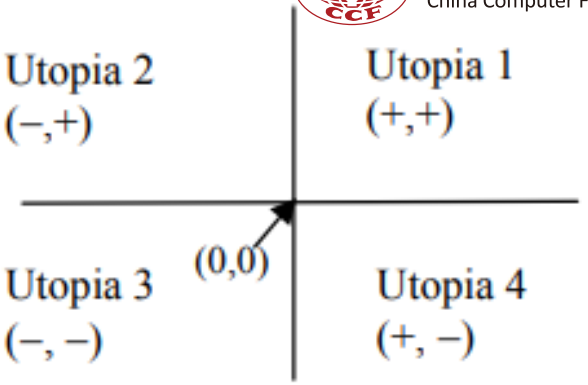


Figure-6: The maximum number of plants flattened by a frog is 4.

- n <= 5000

IOI 02: Utopia



You will receive as input a sequence of $2N$ code numbers and are to write them as a sequence of N code pairs, placing a plus or a minus sign before each number. If you are currently at the point (x,y) and use the code pair $(+u,-v)$, you will be teleported to the point $(x+u, y-v)$. You have the $2N$ numbers, and you can use them in any order you like, each with a plus or a minus sign.

Suppose you have code numbers 7, 5, 6, 1, 3, 2, 4, 8 and are to guide the teleporter according to the sequence of region numbers 4, 1, 2 ,1. The sequence of code pairs (+7,–1), (–5,+2), (–4,+3), (+8,+6) achieves this as it teleports you from (0,0) to the locations (7,–1), (2,1), (–2,4) and (6,10) in that order. These points are located in Utopia 4, Utopia 1, Utopia 2, and Utopia 1, respectively.

TASK

You are given $2N$ distinct code numbers and a sequence of N region numbers indicating where the teleporter is to land. Construct a sequence of code pairs from the given numbers that guide the teleporter to go through the given region sequence.

- $n <= 10^6$

IOI 02: XOR (output only)

As an example, consider the image in Figure-3. Applying XOR(2,4,2,6) to an all white image gives the image in Figure-1. Applying XOR(3,6,4,7) to the image of Figure-1 gives the image in Figure-2, and applying XOR(1,3,3,5) to the image in Figure-2 finally gives the image in Figure-3.

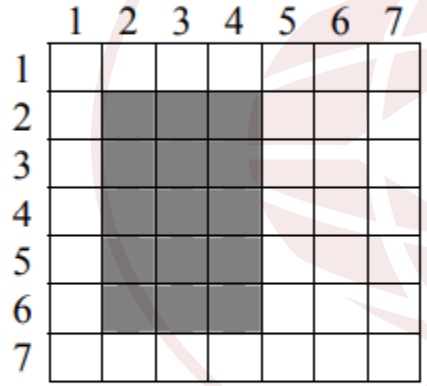


Figure-1

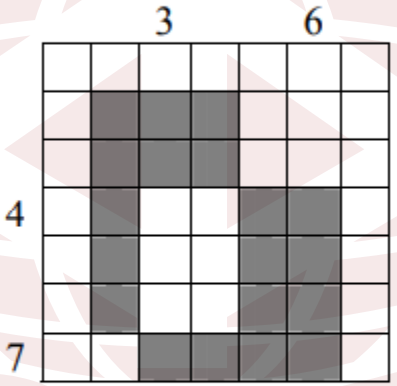


Figure-2

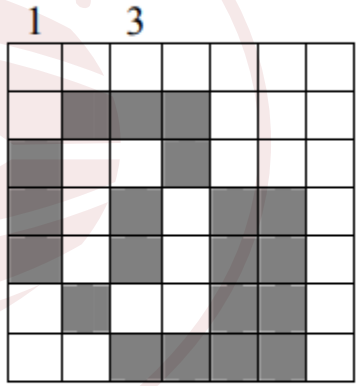


Figure-3

Given a set of black-and-white pictures, your task is to generate each picture from an initially white screen using as few XOR calls as you can. You are given the input files describing the images, and you are to submit files including the required XOR call parameters, not a program to create these files.

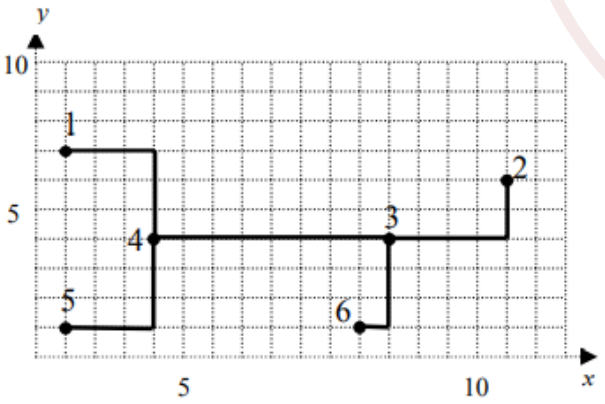
IOI 02 day2: hull DP + L_1 min diameter spanning tree + interactive

This problem can be solved using dynamic programming. Let C_i be the minimum total cost of all partitionings of jobs $J_i, J_{i+1}, \ldots, J_N$ into batches. Let $C_i(k)$ be the minimum total cost when the first batch is selected as $\{J_i, J_{i+1}, \ldots, J_{k-1}\}$. That is, $C_i(k) = C_k + (S + T_i + T_{i+1} + \ldots + T_{k-1}) * (F_i + F_{i+1} + \ldots + F_N)$.

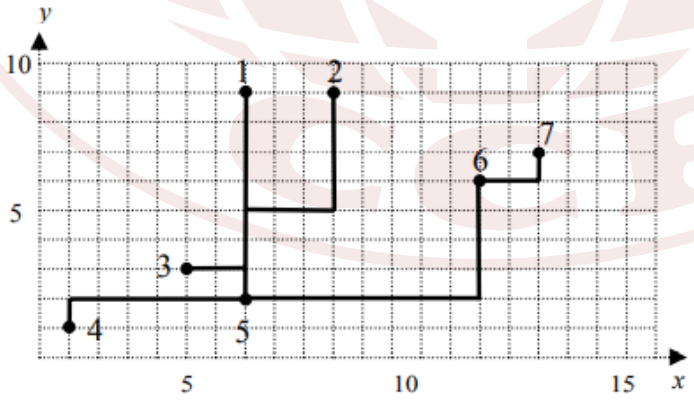
Then we have that

$C_i = \min \{ C_i(k) \mid k = i+1, \ldots, N+1 \}$ for $1 \le i \le N$,

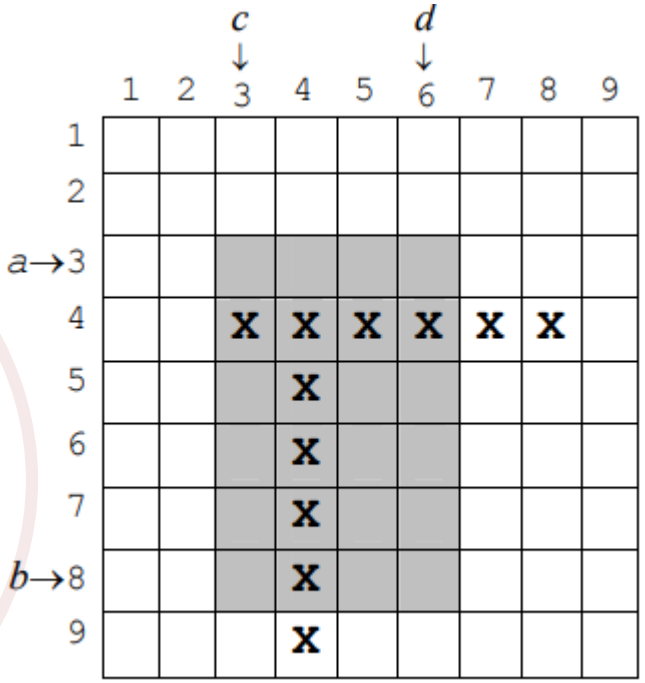
and $C_{N+1} = 0$.

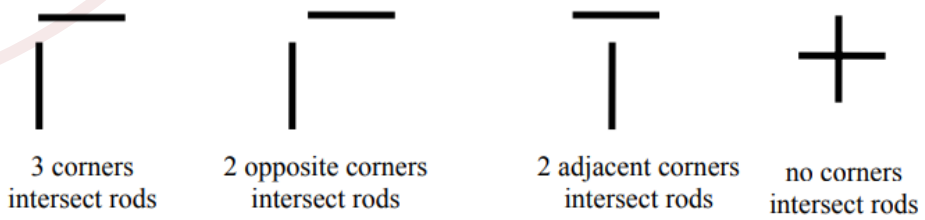


Bus network for Example 1



Bus network for Example 2





3 corners intersect rods

2 opposite corners intersect rods

2 adjacent corners intersect rods

no corners intersect rods

IOI 02 Problems

- Frog
- Utopia
- XOR
- Batch
- Bus
- Rods
- DP + data structures
- Construction (sequence)
- Output only, grid
- DP + convex hull
- Min-diameter spanning tree
- Interactive

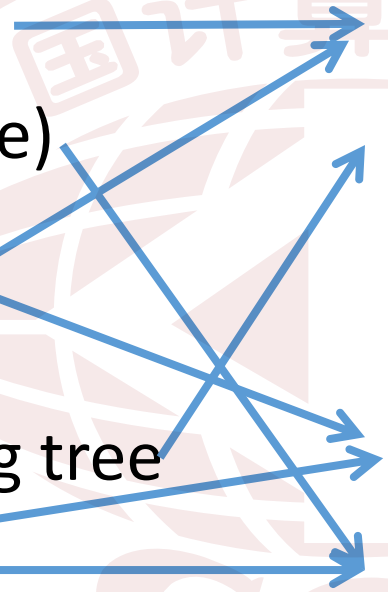
IOI 20 Problems

- Plants
- Supertrees
- Tickets
- Biscuits
- Mushrooms
- Stations

- Adhoc + data structures
- Graph, construction
- Greedy, grid-like
- Counting DP
- Interactive
- Construction (tree)

IOI 02 → 20

- DP + data structures
- Construction (sequence)
- Output only, grid
- DP + convex hull
- Min-diameter spanning tree
- Interactive



- Adhoc + data structures
- Graph, construction
- Greedy, grid-like
- Counting DP
- Interactive
- Construction (tree)

Differences

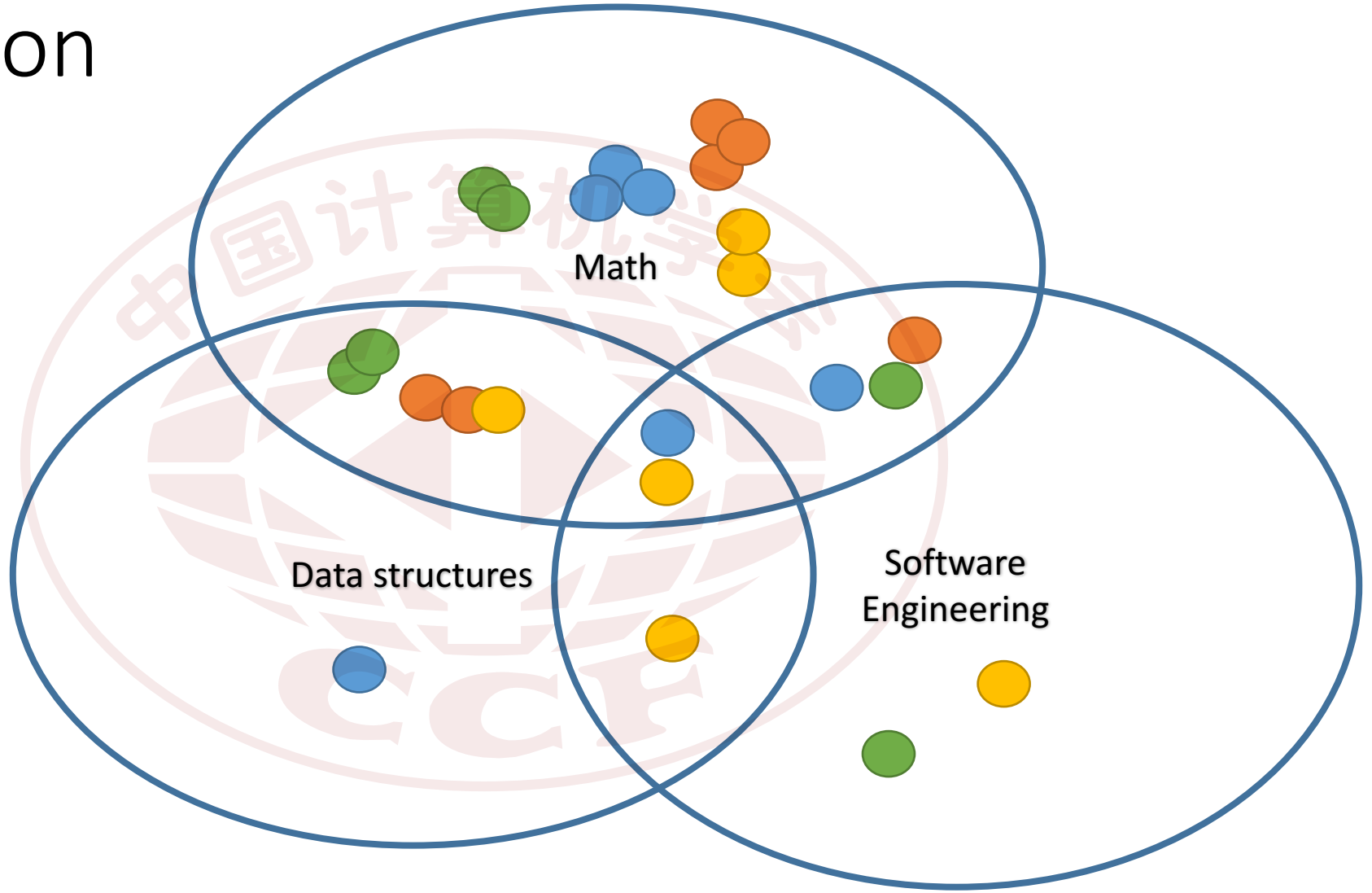
- 02: most tasks goals are `find optimum solution’
- 20: most tasks’ goals are `check if possible, guess some unknown, exhibit one such object’

Similarities

- Proofs based thinking
- Data structures
- Construction / combinatorics
- Graphs, sequences

Classification

- 2020
- 2019
- 2018
- 2017



中国计算机学会
CCF

Why

1997

Score	
Abs.	Rel.
462	77.00%
461	76.83%
450	75.00%
449	74.83%
446	74.33%
444	74.00%
431	71.83%
423	70.50%
422	70.33%
418	69.67%

2000

Score	
Abs.	Rel.
700	100.00%
690	98.57%
690	98.57%
690	98.57%
680	97.14%
680	97.14%
670	95.71%
670	95.71%
660	94.29%
610	87.14%

2002

Score	
Abs.	Rel.
510	85.00%
458	76.33%
448	74.67%
415	69.17%
398	66.33%
390	65.00%
385	64.17%
378	63.00%
372	62.00%
361	60.17%

2018

Score	
Abs.	Rel.
499	83.17%
469	78.17%
466	77.67%
463	77.17%
430	71.67%
428	71.33%
428	71.33%
404	67.33%
397	66.17%
390	65.00%

2019

Score	
Abs.	Rel.
547.09	91.18%
511.22	85.20%
494.33	82.39%
491.46	81.91%
482.39	80.40%
477.14	79.52%
466.87	77.81%
464.11	77.35%
459.26	76.54%
452.65	75.44%

2020

Score	
Abs.	Rel.
600.00	100.00%
593.00	98.83%
592.62	98.77%
592.62	98.77%
592.62	98.77%
592.62	98.77%
592.24	98.71%
567.62	94.60%
567.62	94.60%
536.62	89.44%
533.58	88.93%
532.94	88.82%
529.43	88.24%
521.43	86.91%

¯_(ツ)_/¯
¯_(ツ)_/¯
¯_(ツ)_/¯
¯_(ツ)_/¯

Why

Aside from GSB, 2 ‘unspoken’ thresholds:

- Top 3
- Top 10

Modern IOI organization spend an unreasonable amount of time ensuring separation among those

- ‘standard’ problems usually get between 25 – 40 solves
- Rank 15~40: practice practice practice practice practice
- Below rank 15: distinguish using something not very well known

How

After the "Call for Tasks" announcement our HSC received 20 problems from Korea and abroad. Of these, we excluded 4 problems which were not mature enough to be used in IOI-style competition tasks. The HSC then prepared all solutions and test data sets (20 test cases for each task) and discussed these sets with the ISC members in July. The meetings held concerning IOI2002 task preparation were follows:

Call for Tasks announcement (Oct. 2001)
1st HSC meeting at KAIST (Jan. 4)
- discussed computer specification
- discussed Competition Rules

2nd HSC meeting in Pusan (Feb. 4-6)
- discussed submitted tasks
- selected 17 candidate tasks

International Committee Meeting in Seoul (Feb. 17)
- reported the current state of the tasks to IC

3rd HSC meeting at KAIST (Mar. 29-30)
- reviewed the first draft of candidate tasks
- programmed solutions

4th HSC meeting at KAIST (Apr. 26-28)
- reviewed solution programs and test data
- tested solution programs for timing

5th HSC meeting at KAIST (May 4)

ISC meeting at KAIST (May 11-18)
- HSC and ISC reviewed and discussed the first draft of tasks
- Prof. Kunsoo Park explained our evaluation system and discussed with ISC
- ISC selected 7 competition tasks out of 16 tasks
- ISC improved the presentation style of 7 selected tasks.

6th HSC meeting at KAIST (Jul. 3-5)

7th HSC meeting at KAIST (Jul. 22-23)
- demonstrated task evaluation system
- loaded and tested solution programs and task library on the evaluation server

Evaluation system was tested on-line and off-line (Jul. 25)
Internet Practice Session Opened (Aug. 6)

8th HSC meeting at KAIST (Aug. 7)
- finalized tasks and solutions

Who: https://ioinformatics.org/page/history-of-the-isc/47

- Kyung-Yong Chwa, Korea (2000 - 2003, Chair IOI'02 HSC)

professor of computer science, KAIST
Verified email at kaist.ac.kr
algorithm

	All
Citations	3026
h-index	24
i10-index	53

- Ian Munro, Canada (2000 - 2003, elected)

University Professor, University of Waterloo
Verified email at uwaterloo.ca - Homepage
Algorithms Data Structures Analysis of Algorithms

	All
Citations	11206
h-index	51
i10-index	136

- Kunsoo Park, Korea (2001 - 2002, IOI'02 HSC)

Professor of Computer Science, Seoul National University
Verified email at theory.snu.ac.kr
Computer Theory String Algorithm Bioinformatics
Cryptography Financial Engineering

	All
Citations	8226
h-index	37
i10-index	128

The Net Result

- A lot of thought go into IOI problems: >100 person/hours per problem
- Discussions inevitably focuses on the very top contestants
- But the problem topics have to be accessible
- With a few exceptions, data + task batching is strong
- Hard end of topics becomes very hard (next)
- Advanced topics tend to only appear on hard problems

IOI 2008 PBS (author: Richard)

- Based off of https://acm.timus.ru/problem.aspx?space=1&num=1235 (NEERC `02): find max empty axis-aligned square with n obstacles.
- Black book: rectilinear rectangle union in O(nlogn)
- + binary search: $O(n\log^2 n)$
- + POI `01 goldmine: each rectangle has cost, find point on plane w. min total cost (summed over covering rectangle)
- Change problem: empty → total weight at most < LIM

IOI 2008 PBS …

- Tester thought this was too easy
- Host didn’t want a 600
- Someone on the HSC spent 2 weeks coding another solution
- …and came up with an O(nlogn) for the LIM = 0 case

IOI 2008 PBS …

- Tester thought this was too easy
- Host didn't want a 600
- Someone on the HSC spent 2 weeks coding another solution
- …and came up with an O(nlogn) for the LIM = 0 case

CONSTRAINTS AND GRADING

Your program will be graded on three disjoint sets of tests. For all of them, the following constraints apply:

$1 \le M, N \le 1{,}000{,}000$	The dimensions of the grid.
$1 \le C_i \le 7{,}000$	The cost of removing the i^{th} obstacle.
$1 \le X_{i1} \le X_{i2} \le M$	X coordinates of the leftmost and the rightmost cells of the i^{th} obstacle.
$1 \le Y_{i1} \le Y_{i2} \le N$	Y coordinates of the bottommost and the topmost cells of the i^{th} obstacle.

In the first set of tests worth 35 points:

$B = 0$	The budget you have. (You cannot remove any obstacles.)
$1 \le P \le 1{,}000$	The number of obstacles in the grid.

In the second set of tests worth 35 points:

$0 < B \le 2{,}000{,}000{,}000$	The budget you have.
$1 \le P \le 30{,}000$	The number of obstacles in the grid.

In the third set of tests worth 30 points:

$B = 0$	The budget you have. (You cannot remove any obstacles.)
$1 \le P \le 400{,}000$	The number of obstacles in the grid.

Similar Problems

- Already combine many problems
- Someone on HSC, ISC, or the GA, add 1 more step…

- IOI09 Archery
- IOI12 Odometer
- IOI13 Game
- IOI16 Aliens
- IOI18 Meetings
- IOI20 Mushrooms

中国计算机学会
CCF

Where to get problems that are

- Easy to understand / explain
- Hard, for even the best contestants
- Considered accessible

https://ucalendar.uwaterloo.ca/2021/COURSE/course-CO.html

CO 330 LEC 0.50 Course ID: 003891

Combinatorial Enumeration

The algebra of formal power series. The combinatorics of the ordinary and exponential generating series. Lagrange's implicit function theorem, applications to the enumeration of permutations, functions, trees and graphs. Integer partitions, geometric methods, enumerating linear transformations. Introduction to the pattern algebra, applications to the enumeration of strings. Lattice paths, Wiener-Hopf factorization. Enumeration under symmetries. [Offered: F]

Prereq: MATH 239 or 249

CO 331 LEC 0.50 Course ID: 003892

Coding Theory

A first course in error-correcting codes. Linear block codes, Hamming-Golay codes, and multiple error-correcting BCH codes are studied. Various encoding and decoding schemes are considered. [Offered: W]

Prereq: MATH 225 or 235 or 245

CO 342 LEC 0.50 Course ID: 003893

Introduction to Graph Theory

An introduction to some of the key topics in graph theory: connectivity, planarity, and matchings. Connectivity: Menger's theorem, 3-connected graphs. Planarity: Kuratowski's theorem, uniqueness of planar embeddings. Matchings: Review of Konig's theorem, Tutte's theorem. [Offered: F,S]

Prereq: MATH 239 or 249

CO 351 LEC 0.50 Course ID: 003896

Network Flow Theory

Review of linear programming. Shortest path problems. The max-flow min-cut theorem and applications. Minimum cost flow problems. Network simplex and primal-dual algorithms. Applications to problems of transportation, distribution, job assignments, and critical-path planning. [Offered: F,S]

Prereq: One of CO 250 or 255 or 352

CO 353 LAB,LEC 0.50 Course ID: 011442

Computational Discrete Optimization

Formulations of combinatorial optimization problems, greedy algorithms, dynamic programming, branch-and-bound, cutting plane algorithms, decomposition techniques in integer programming, approximation algorithms.

Prereq: One of CO 250 or 255 or 352

CO 446 LEC 0.50 Course ID: 013337

Matroid Theory

This is an introductory course on matroid theory, with particular emphasis on graphic matroids and on topics that are applicable to graph theory. The topics include matroid intersection and partition, graphic matroids, matroid connectivity, regular matroids, and representable matroids. Applications include matching, disjoint paths, disjoint spanning trees, the 8-flow theorem for graphs, planarity testing, and recognizing totally unimodular matrices. [Offered: S]

Prereq: CO 342; Cumulative overall average of at least 80%

CO 450 LEC 0.50 Course ID: 003910

Combinatorial Optimization

Characterizations of optimal solutions and efficient algorithms for optimization problems over discrete structures. Topics include network flows, optimal matchings, T-joins and postman tours, matroid optimization. [Offered: F]

Prereq: CO 255 or 351; Cumulative overall average of at least 80%

https://cs.uwaterloo.ca/current-graduate-students/courses/course-calendar/700-level-courses

CS 761 Randomized Algorithms (0.50) LEC, Course ID: 011296

Introduction to the design and analysis of algorithms that make use of randomization. Topics include review of basic probabiloity and introduction to randomized algorithms; game theoretic techniques; uses of Markov and Chebyshev inequalities; tail inequalities; Markov chains and random walks; algebraic techniques; data structures and graph algorithms.

CS 762 Graph-Theoretic Algorithms (0.50) LEC, Course ID: Unknown

To give the student further exposure to the design, analysis, and application of algorithms for problems defined on graphs.

CS 763 Computational Geometry (0.50) LEC, Course ID: 011297

Introduction to the design, analysis and application of algorithms for geometric problems. Topics include convex hull algorithms in two and three dimensions; Voronoi diagrams, Delaunay triangulations, and their applications; linear programming in low dimensions; line segments, planar subdivision, and polygons; range searching.

CS 764 Computational Complexity (0.50) LEC, Course ID: 011298

Further exposure to the classification of problems based on their computational requirements and to mathematical tools designed to explore the structural consequences of such classifications. Topics include relativization, alternation, provably intractable problems, feasible parallel computation; fixed-parameter tractability and the W-hierarchy; Kolmogorov complexity, including algorithmic and algorithmic prefix complexity and their applications.

Antirequisite: CS 664

CS 765 Algorithmic Number Theory (0.50) LEC, Course ID: 011299

(Cross-listed with CO 785)

Fundamental problems of elementary and algebraic number theory from an algorithmic and computational complexity point of view with emphasis placed on analysis of algorithms. Topics include basic arithmetic algorithms; computation over finite fields; primality testing; algorithms for integer factorization; algorithms in number fields.

Claim

- Modern OI is more of an applied math contest than a programming contest
- But it still needs to be ‘computer science’

Source of Problems

Academic

- POI: stringology, automatas
- CCC/CCO: SODA + ML

Outreach / pop sci

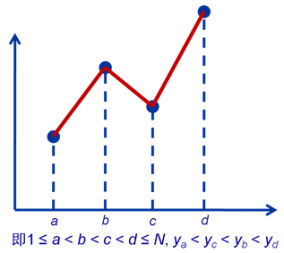
- Combinatorics
- Puzzles
- Physics

Source of Problems

Academic

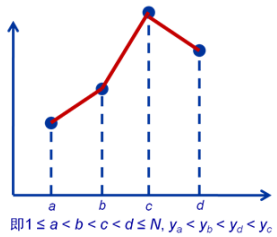
- POI: stringology, automatas
- CCC/CCO: SODA + ML

小布的团队打算研究在这幅壁画中包含着多少个图腾，其中闪电图腾的定义如下：

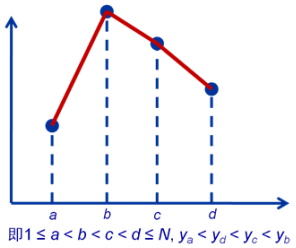


即$1 \le a < b < c < d \le N, y_a < y_c < y_b < y_d$

崇拜森林的部落曾经有过一次分裂，因而图腾有两种形式，分别如下（左边为 A 类，右边 B 类，图腾的形式只与 4 个纵坐标的相对顺序有关）：

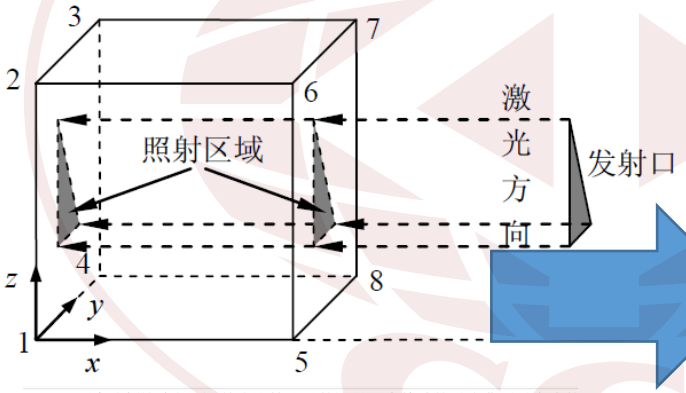


即$1 \le a < b < c < d \le N, y_a < y_b < y_d < y_c$

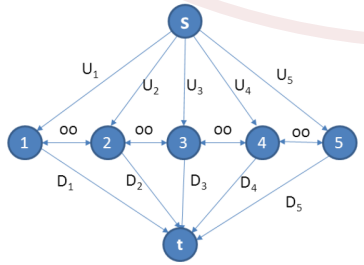


即$1 \le a < b < c < d \le N, y_a < y_d < y_c < y_b$

小布的团队希望知道，这 N 个点中两个部落图腾数目的差值。因此在本题中，

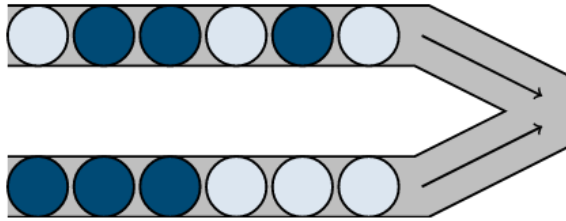


下面来分析这个问题：首先比较明显的这是一个特殊的最小费用最大流的模型。模型如下：$N+2$ 个点，$4N-2$ 条边。



Outreach / pop sci

- Combinatorics
- Puzzles
- Physics



- 卡池里总共有 N 种卡，第 i 种卡有一个权值 W_i，小刘同学不知道 W_i 具体的值是什么。但是他通过和网友交流，他了解到 W_i 服从一个分布。
- 具体地，对每个 i，小刘了解到三个参数 $p_{i,1}, p_{i,2}, p_{i,3}$，$W_i$ 将会以 $p_{i,j}$ 的概率取值为 j，保证 $p_{i,1} + p_{i,2} + p_{i,3} = 1$。

小刘开始玩游戏了，他每次会氪一元钱来抽一张卡，其中抽到卡 i 的概率为：

$$\frac{W_i}{\sum_j W_j}$$

小刘会不停地抽卡，**直到**他手里集齐了全部 N 种卡。

抽卡结束之后，服务器记录下来了小刘**第一次**得到每张卡的时间 T_i。游戏公司在这里设置了一个彩蛋：公司准备了 $N-1$ 个二元组 (u_i, v_i)，如果对任意的 i，成立 $T_{u_i} < T_{v_i}$，那么游戏公司就会认为小刘是极其幸运的，从而送给他一个橱柜的手办作为幸运大奖。

游戏公司为了降低获奖概率，它准备的这些 (u_i, v_i) 满足这样一个性质：对于任意的 $\varnothing \neq S \subsetneq \{1, 2, \ldots, N\}$，总能找到 (u_i, v_i) 满足：$u_i \in S, v_i \notin S$ **或者** $u_i \notin S, v_i \in S$。

请你求出小刘同学能够得到幸运大奖的概率，可以保证结果是一个有理数，请输出它对 998244353 取模的结

2 Kinds of IOIs

Odd years

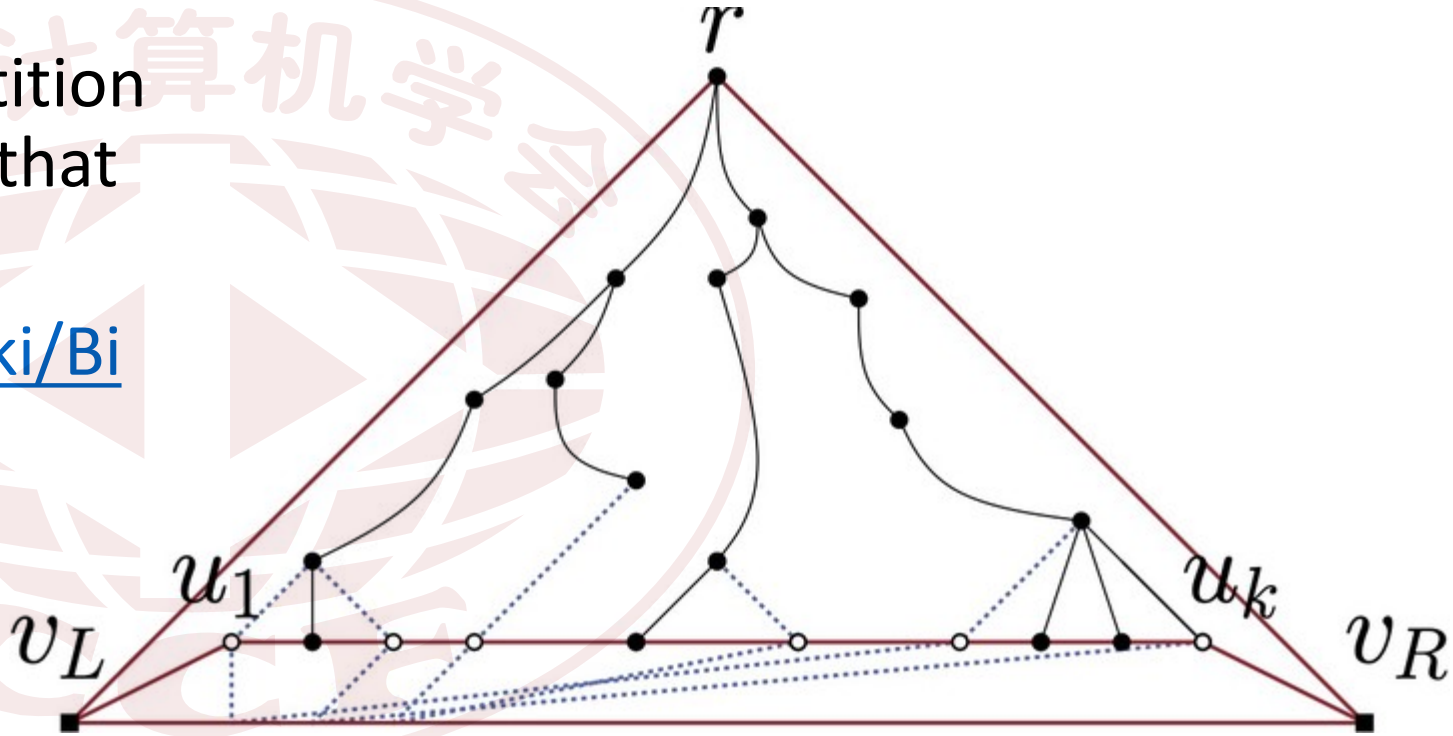
- ~400 lines of code per day
- Hard usually hard in standard ways

Even years

- ~200 lines code per day
- Hard usually combinatorics

IOI 19 Split

- Given connected graph, partition into 3 sets of sizes a, b, c so that at least 2 are connected
- https://en.wikipedia.org/wiki/Bipolar_orientation



IOI 19 Sky Walking

Shortest Paths in an Arrangement with k Line Orientations

David Eppstein [1] David W. Hart [2]

Abstract

Suppose one has a line arrangement in which many lines are parallel, so that the number of different line orientations is k, and one wants to find a shortest path from one point on a line in the arrangement to another such point. Using known techniques one can find the shortest path in time and space $O(n^2)$. We present an algorithm that can find the shortest path in time and space $O(n + k^2)$.

derived from the nonconvex cells of the arrangement. However, these methods do not seem to generalize to arrangements with more than two orientations.

In this paper, we consider arrangements with few distinct line orientations.[3] We show that only a constant number of lines of each orientation can participate in the shortest path, and that these lines can be identified quickly. Hence, if there are k orientations, the shortest path can be found in time and space $O(n+k^2)$. If the n lines have n different orientations, this time

中国计算机学会
CCF

Sources of Hard Problems (< 10 solvers)

- 2020 plants (4): data structure
- 2020 mushrooms (4-): information theory, ???
- 2019 split attractions (0): graph drawing, WD
- 2019 sky walking (0): computational geometry, WADS
- 2018 highway (1): graph theory (interactive)
- 2018 seats (0.5): planar geometry, math/combinatorics
- 2018 meetings (0): data structure
- 2017 toytrain (1): game theory / complexity theory,
- 2017 simurgh (1.5): interactive + graph theory
- 2016 shortcut (3): data structure
- 2016 railroad (2): graph theory
- 2016 aliens (1): optimization/geometry/math, WADS, CEOI`11/CTSC`12

Main Sources of Hard Problems

- Data structures
- Graph theory
- Geometry
- Information theory / communication complexity

Data structures

- Usually self contained, can produce ‘original’ hard problems
- Very very rare for IOI data structures to NOT FOLLOW from some CTSC report
- Issue: usualy 4~5 years delay

Counting / Signal Processing

- Generating functions
- FFT
- Structured matrices

- Not sure how well known this is generally
- This might start to become popular
- But the math ceiling here is unlimited

Counting / Signal Processing

INTRODUCTION TO FAST MULTIPOLE METHODS

LONG CHEN

ABSTRACT. This is a simple introduction to fast multipole methods for the N-body summation problems. Low rank approximation plus hierarchical decomposition leads to fast $\mathcal{O}(N)$ or $\mathcal{O}(N \log N)$ algorithms for the summation problem or equivalently the computation of a matrix-vector product.

We consider a summation problem in the form:

$$u_i = \sum_{j=1}^{N} \phi_{ij} q_j, \quad i = 1, \cdots, N. \tag{1}$$

In the matrix form, if we denote

$$\boldsymbol{u} = (u_i) \in \mathbb{R}^N, \ \boldsymbol{q} = (q_j) \in \mathbb{R}^N, \ \boldsymbol{\Phi} = (\phi_{ij}) \in \mathbb{R}^{N \times N},$$

the sums in (1) are equivalent to the multiplication of matrix $\boldsymbol{\Phi}$ with vector $\boldsymbol{q}$

$$\boldsymbol{u} = \boldsymbol{\Phi} \boldsymbol{q}. \tag{2}$$

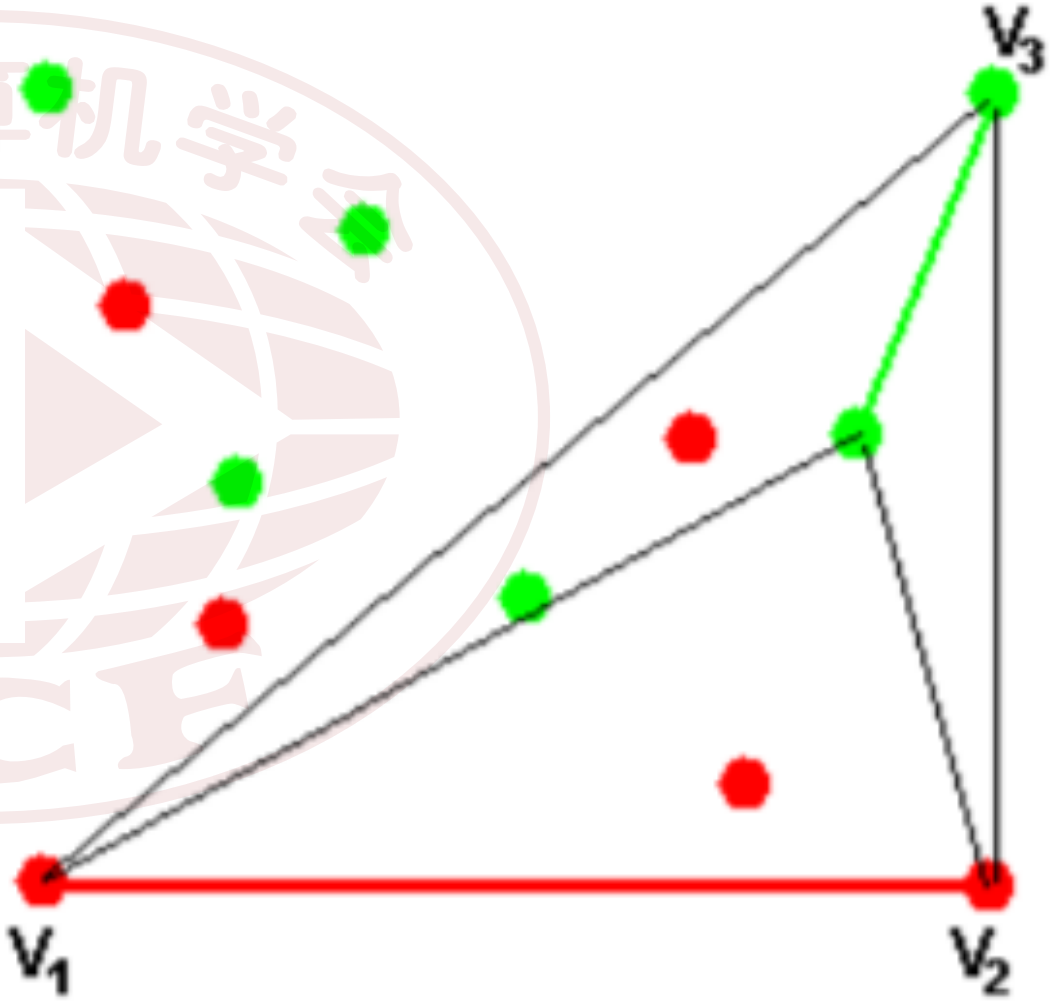
A direct evaluation obviously requires $\mathcal{O}(N^2)$ operations. It is unavoidable for general dense matrices. But if the matrix has certain structure, we may calculate the sums in linearithmic, i.e., $\mathcal{O}(N \log N)$ operation or even linear, i.e., $\mathcal{O}(N)$ operations.

- Given x[1]< x[2] < … < x[n]
- For each i, compute $\Sigma_{j<i}$ (x[i] – x[j])$^{-1}$ to 1e^{-6} relative accuracy

Graph Theory / Geometry

- Usually exist in some math contest combinatorics
- USAMO & Tournament of towns are very good starting points
- Too many math contest problems
- Too many conference results to remember even (e.g. lex BFS / cographs)

IOI `06 Points



Information theory / complexity theory

- Classic topic
- Many courses / notes
- Easy to convert to problems: easier to set than data structures
- Short solutions

- Main issue: rare topic in computer science
- Corollary: when people who know this topic set problems, they are very hard

USACO Camp…

MoooOoOMOoooooOMOooooOOMOoOooo
MOoOOooMOoOOOOMOOoOOOMOooOoOMOOooOoMOoooOOMOooooOMOOooOOMOooOoO
MOooooOMOoOOooMOOooooMOoOooooMOooooOMOooooOoMOooOoOMOOoOoo
MOoooOOMOoOoooMOooooOMOOooOoMOooooOMOooooOOMOOoOooMOooOoOMOOooOo
MOoOooOMOOooOO MOooOoOMOoOOOoMOoooOOMOoOOOOMOooOooMOooOoOMOooOoo
MOOoOoOMOOooOOMOoOooOMOoOOOoMOooOOO MOoOooOMOOoOooMOOooOO
MOooooOMOOooOOMOooooOOMOoOooOMOoOooO
MOoooOoMOoOooOMOoOOOoMOooooOMOOooOoMOOOooO
MOOooOoMOooOoOMOOooooMOOooOoMOooOoOMOOooOOMOooOoOMOoOOOoMOOoOooMOooooOMOOoOooMOoOooOMOoO
OOOMOoOOOo MOOoOOOMOoOooOMOOoOooMOoOooo ’MooOOOO’ MOooooOMOoOOOoMOooOoo ’MOoOOOO’
MOoooOOMOoOoooMOooooOMOOooOoMOooooOMOooOOMOOoOooMOooOoOMOOooOoMOOooOO
MOOooOoMOooOoOMOOooooMOOooOoMOooOoOMOOooOOMOooOoOMOoOOOoMOOoOooMOoOooOMOoOOOoMOooOOO
MOoOOOOMOoOOOoMOooOoO MOooooOMOoOOOoMOooOoo MOOOoOoMOooOoOMOOooOoMOoOOOO.
MoOoOooMOoOoooMOooOoO MOoOOooMOooOoOMOooooOMOooOooMOoOooOMOoOOOoMOooOOO
MOooOooMOoOooOMOooOOOMOoOooOMOOoOoo MOoOooOMOOooOO
MOOoOooMOoOoooMOooOoOMOoOOOo
MOOooOoMOooOoOMOOooooMOoOOooMOooooOMOooOOMOooOoOMOooOoo
MOOoOOOMOoOooOMOOoOooMOoOooo ’MooOOoO’ MOOoOooMOoOOOO
MOOooOoMOooOoOMOoOOoOMOoOOOOMOOoOOoMOooOoO
MOooooOMOoooOoMOoOooOMOooOOOMOOoOoOMOoOooOMOOoOooMOOOooO.
MoOooOoMOooOoOMOooooOMOooOoo MOOoOooMOoOoooMOooOoO
MOoOooOMOoOOOoMOOooooMOOoOoOMOOoOoo MOooooOMOoOOOoMOooOoo
MOOoOoOMOoOOOoMOooOooMOoOOOO MOOoOooMOoOoooMOooOoO
MOooOoOMOoOOOoMOoooOOMOoOOOOMOooOooMOoOooOMOoOOOoMOooOOO.

IOI 2010

Jack and Jill play a game called *Hotter, Colder*. Jill has a number between 1 and N, and Jack makes repeated attempts to guess it.

Each of Jack's guesses is a number between 1 and N. In response to each guess, Jill answers *hotter*, *colder* or *same*. For Jack's first guess, Jill answers *same*. For the remaining guesses Jill answers:

- *hotter* if this guess is closer to Jill's number than his previous guess
- *colder* if this guess is farther from Jill's number than his previous guess
- *same* if this guess is neither closer to nor further from Jill's number than his previous guess.

You are to implement a procedure **HC(N)** that plays Jack's role. This implementation may repeatedly call **Guess(G)**, with **G** a number between 1 and N. **Guess(G)** will return 1 to indicate *hotter*, -1 to indicate *colder* or 0 to indicate *same*. **HC(N)** must return Jill's number.

Subtask 4 [up to 25 points]

Let W be the largest integer, such that $2^W \leq 3N$. For this subtask your solution will score:

- 0 points, if **HC(N)** calls **Guess(G)** 2W times or more,
- 25α points, where α is the largest real number, such that $0<\alpha<1$ and **HC(N)** calls **Guess(G)** at most $2W-\alpha W$ times,
- 25 points, if **HC(N)** calls **Guess(G)** at most W times.

There will be at most 1 000 000 calls to **HC(N)** with N between 1 and 500 000 000.

https://ioi2010.org/competitiontask/day1/hottercolder/hottercolder.c

```
int endgame(int end, int n){
   // returns Jill's number, assuming
   // end = value on edge (1 or N); n = number of remaining candidate values
   int z, g;
   for (z=0; t[z]<n; z++);
   if (n == 2) {
      g = Guess(fix(end, 1));
      if (g > 0) return fix(end, 1);
      return fix(end, 2);
   }
   if (n == 3) {
      g = Guess(fix(end, 1));
      if (g > 0) return fix(end, 1);
      if (g == 0) return fix(end, 2);
      if (g < 0) return fix(end, 3);
   }
   if (n == 4 || n == 5) {
      g = Guess(fix(end, 3));
      if (g < 0) return fix(end, n);
      if (g == 0) return fix(end, 4);
      g = Guess(fix(end, 1));
      if (g > 0) return fix(end, 1);
      if (g == 0) return fix(end, 2);
      if (g < 0) return fix(end, 3);
   }
   if (n == 6) {
      g = Guess(fix(end, 1));
      if (g > 0) {
         g = Guess(fix(end, 3));
         if (g > 0) return fix(end, 3);
         if (g == 0) return fix(end, 2);
         if (g < 0) return fix(end, 1);
      }
      g = Guess(fix(end, 9));
      if (g > 0) return fix(end, 6);
      if (g == 0) return fix(end, 5);
   }
   if (n == 7) {
      g = Guess(fix(end, 1));
      if (g == 0) return fix(end, 4);
      if (g > 0) {
         int g = Guess(fix(end, 3));
         if (g > 0) return fix(end, 3);
         if (g == 0) return fix(end, 2);
         return fix(end, 1);
      }
      g = Guess(fix(end, 11));
      if (g > 0) return fix(end, 7);
      if (g == 0) return fix(end, 6);
      if (g < 0) return fix(end, 5);
   }
   g = Guess(fix(end, t[z-2]-2));
   if (g == 0) return fix(end, (t[z-2]-2+n)/2);
   if (g < 0) return midgame(fix(end, t[z-2]-2), fix(end, (t[z-2]-2+n)/2+1), fix(end, n));
   // g > 0
   g = Guess(fix(end, t[z-2]));
   if (g < 0) return endgame(end, t[z-2]);
   if (g == 0) return fix(end, t[z-2]-1);
   return midgame(fix(end, t[z-2]), fix(end, t[z-2]), fix(end, (t[z-2]-2+n-1)/2));
   return 0;
}
```

Information theory / complexity theory

- Solution for n = 35 hardest
- Search over all possible decision trees!
- Very very very details heavy: # of operations can distinguish a lot
- Is this really computer science?
- Who sets this kind of problems?!?!

USACO Camp

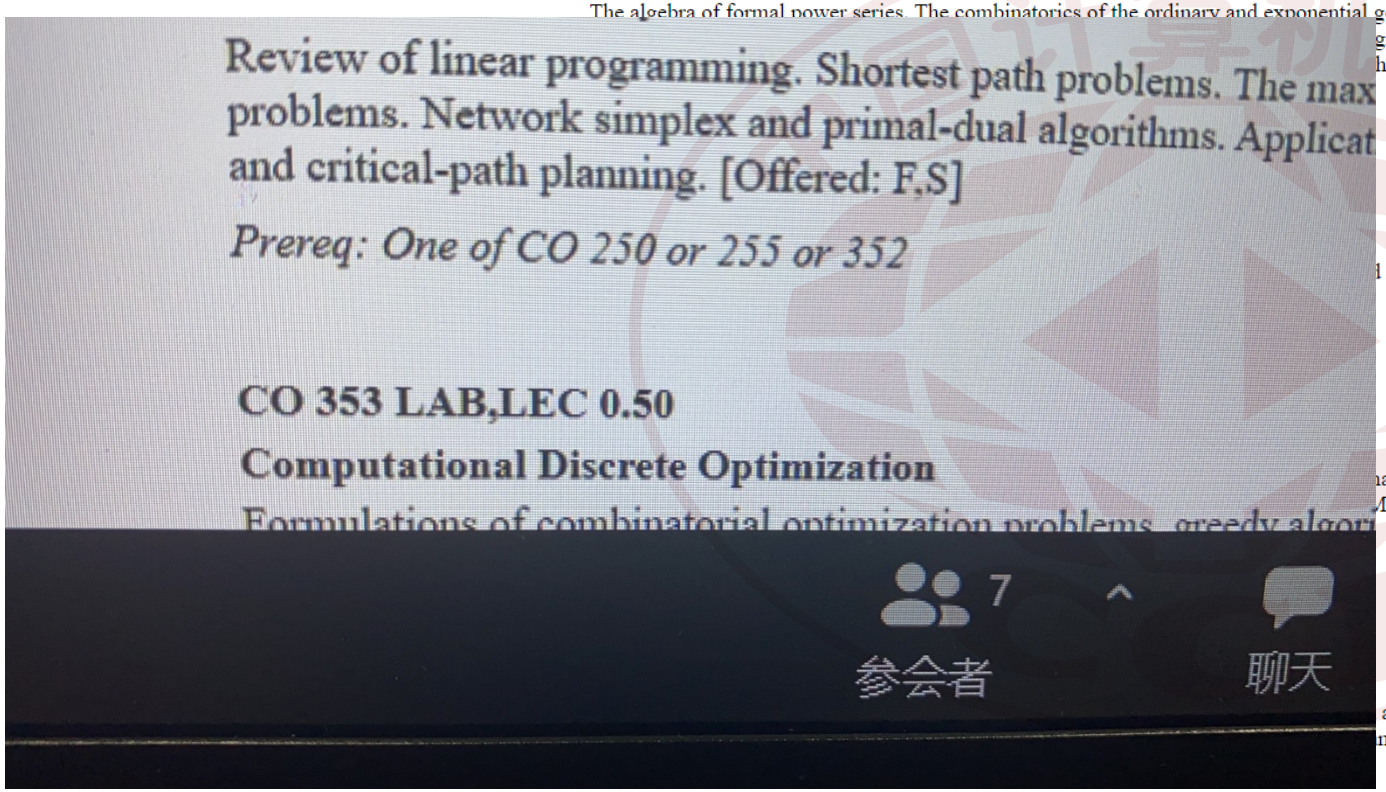
- 2 people have subsets of $[1, 2^L]$
- Compute median of the combined list using O(L) bits of communication

Codeforces

So you don't have a choice but to solve the riddle. Sphinx has an array $a_1, a_2, \ldots, a_n$ of **nonnegative** integers **strictly smaller** than 2^b and asked you to find the maximum value among its elements. Of course, she will not show you the array, but she will give you n and b. As it is impossible to answer this riddle blindly, you can ask her some questions. For given i, y, she'll answer you whether a_i is **bigger** than y. As sphinxes are not very patient, you can ask at most $3 \cdot (n + b)$ such questions.

Extra Slides After This

EXTRA: 调位置, BOTTOM 3 LINES HIDDEN BY ZOOM



CO 330 LEC 0.50 Course ID: 003891

Combinatorial Enumeration

The algebra of formal power series. The combinatorics of the ordinary and exponential generating series. Lagrange's implicit
graphs. Integer partitions, geometric methods,
he enumeration of strings. Lattice paths,

Course ID: 003892

l multiple error-correcting BCH codes are

Course ID: 003893

atchings. Connectivity: Menger's theorem, 3-
Matchings: Review of Konig's theorem,

Course ID: 003896

and applications. Minimum cost flow
nsportation, distribution, job assignments,

Prereq: One of CO 250 or 255 or 352

CO 353 LAB,LEC 0.50 Course ID: 011442

Computational Discrete Optimization

Formulations of combinatorial optimization problems, greedy algorithms, dynamic programming, branch-and-bound, cutting plane algorithms, decomposition techniques in integer programming, approximation algorithms.

Prereq: One of CO 250 or 255 or 352