LIX SPbSU Championship

May 12, 2024

Problem A: Element-Wise Comparison

Idea:	Dmitry Belichenko
Development:	Dmitry Belichenko
	Nikita Gaevoy
Editorial:	Ivan Bochkov

Element-Wise Comparison

- Given a permutation p, we need to find a number of pairs of subarrays of a given length such that the left one is element-wise smaller than the right one.

Element-Wise Comparison

- Given a permutation p, we need to find a number of pairs of subarrays of a given length such that the left one is element-wise smaller than the right one.
- Consider Boolean matrix $C_{\ell,s}$ such that $C_{\ell,s} = 1$ iff $p_\ell < p_{\ell+s}$.

Element-Wise Comparison

- Given a permutation p, we need to find a number of pairs of subarrays of a given length such that the left one is element-wise smaller than the right one.
- Consider Boolean matrix $C_{\ell,s}$ such that $C_{\ell,s} = 1$ iff $p_\ell < p_{\ell+s}$.
- How do we do that? Bitsets! (or pragmas, they also help)

Element-Wise Comparison

- Given a permutation p, we need to find a number of pairs of subarrays of a given length such that the left one is element-wise smaller than the right one.
- Consider Boolean matrix $C_{\ell,s}$ such that $C_{\ell,s} = 1$ iff $p_\ell < p_{\ell+s}$.
- How do we do that? Bitsets! (or pragmas, they also help)
- We can construct this matrix explicitly in time $O(n^2/w)$.
- We can choose pivot elements with step by m rows and then compute prefix- and suffix-OR between pivot elements to compute the answer. This part also takes $O(n^2/w)$ time.

Problem B: Schoolgirls

Idea: Mikhail Ivanov
Development: Mikhail Ivanov
Editorial: Mikhail Ivanov

Schoolgirls

- We are given the vertices of a regular polygon on the plane

Schoolgirls

- We are given the vertices of a regular polygon on the plane
- We can extend a triangle to form a parallelogram, adding one new point to our set

Schoolgirls

- We are given the vertices of a regular polygon on the plane
- We can extend a triangle to form a parallelogram, adding one new point to our set
- After several such operations, check the regularity of polygons with vertices from our set

Schoolgirls

- Each vertex can be associated with a vector $v \in \mathbb{R}^2$

Schoolgirls

- Each vertex can be associated with a vector $v \in \mathbb{R}^2$
- The operation on vertices A, B, C yields $D = A + C - B$

Schoolgirls

- Each vertex can be associated with a vector $v \in \mathbb{R}^2$
- The operation on vertices A, B, C yields $D = A + C - B$
- How to check that $A_1, \ldots, A_n$ is regular?

Schoolgirls

- Each vertex can be associated with a vector $v \in \mathbb{R}^2$
- The operation on vertices A, B, C yields $D = A + C - B$
- How to check that $A_1, \ldots, A_n$ is regular?
- Firstly, let $M = \frac{A_1 + \ldots + A_n}{n}$, $A'_i = A_i - M$

Schoolgirls

- Each vertex can be associated with a vector $v \in \mathbb{R}^2$
- The operation on vertices A, B, C yields $D = A + C - B$
- How to check that $A_1, \ldots, A_n$ is regular?
- Firstly, let $M = \frac{A_1 + \ldots + A_n}{n}$, $A'_i = A_i - M$
- Check that $A'_1, \ldots, A'_n$ is regular with zero center

Schoolgirls

- Each vertex can be associated with a vector $v \in \mathbb{R}^2$
- The operation on vertices A, B, C yields $D = A + C - B$
- How to check that $A_1, \ldots, A_n$ is regular?
- Firstly, let $M = \frac{A_1 + \ldots + A_n}{n}$, $A'_i = A_i - M$
- Check that $A'_1, \ldots, A'_n$ is regular with zero center
  - Rotate each vector by $\frac{2\pi}{n}$ and check that the rotated vector is in the set

Schoolgirls

- Each vertex can be associated with a vector $v \in \mathbb{R}^2$
- The operation on vertices A, B, C yields $D = A + C - B$
- How to check that $A_1, \ldots, A_n$ is regular?
- Firstly, let $M = \frac{A_1 + \ldots + A_n}{n}$, $A'_i = A_i - M$
- Check that $A'_1, \ldots, A'_n$ is regular with zero center
  - Rotate each vector by $\frac{2\pi}{n}$ and check that the rotated vector is in the set
- To avoid fractions, perform the same check with nA'_i instead of A'_i

Schoolgirls

- $\mathbb{R}^2$ is a two-dimensional vector space over $\mathbb{R}$

Schoolgirls

- $\mathbb{R}^2$ is a two-dimensional vector space over $\mathbb{R}$
- Store two real coordinates of each point, add and check regularity accordingly

Schoolgirls

- $\mathbb{R}^2$ is a two-dimensional vector space over $\mathbb{R}$
- Store two real coordinates of each point, add and check regularity accordingly
- However, modern computers are uncapable of storing an element of $\mathbb{R}$

Schoolgirls

- $\mathbb{R}^2$ is a two-dimensional vector space over $\mathbb{R}$
- Store two real coordinates of each point, add and check regularity accordingly
- However, modern computers are uncapable of storing an element of $\mathbb{R}$
- Precision errors

Schoolgirls

- $\mathbb{R}^2$ is a two-dimensional vector space over $\mathbb{R}$
- Store two real coordinates of each point, add and check regularity accordingly
- However, modern computers are uncapable of storing an element of $\mathbb{R}$
- Precision errors
- For any $n \neq 4$, we can construct a sequence which exponentially tends to a point but never reaches it

Schoolgirls

- $\mathbb{R}^2$ is a two-dimensional vector space over $\mathbb{R}$
- Store two real coordinates of each point, add and check regularity accordingly
- However, modern computers are uncapable of storing an element of $\mathbb{R}$
- Precision errors
- For any $n \neq 4$, we can construct a sequence which exponentially tends to a point but never reaches it
- Therefore, no finite precision is enough

Schoolgirls

- Instead, let us work in a vector space over $\mathbb{Q}$

Schoolgirls

- Instead, let us work in a vector space over $\mathbb{Q}$
- Regular n-gon generates a vector space of dimension $\varphi(n)$

Schoolgirls

- Instead, let us work in a vector space over $\mathbb{Q}$
- Regular n-gon generates a vector space of dimension $\varphi(n)$
  - To understand this vector space better, please refer to MemSQL Start[c]UP 3.0, Round 1, problem G: Circle of Numbers

Schoolgirls

- Instead, let us work in a vector space over $\mathbb{Q}$
- Regular n-gon generates a vector space of dimension $\varphi(n)$
  - To understand this vector space better, please refer to MemSQL Start[c]UP 3.0, Round 1, problem G: Circle of Numbers
- Now we store $\varphi(n)$ integers

Schoolgirls

- Instead, let us work in a vector space over $\mathbb{Q}$
- Regular n-gon generates a vector space of dimension $\varphi(n)$
  - To understand this vector space better, please refer to MemSQL Start[c]UP 3.0, Round 1, problem G: Circle of Numbers
- Now we store $\varphi(n)$ integers
- Choose the basis from the vertices of the initial polygon

Schoolgirls

- Instead, let us work in a vector space over $\mathbb{Q}$
- Regular n-gon generates a vector space of dimension $\varphi(n)$
  - To understand this vector space better, please refer to MemSQL Start[c]UP 3.0, Round 1, problem G: Circle of Numbers
- Now we store $\varphi(n)$ integers
- Choose the basis from the vertices of the initial polygon
- To rotate a vector by $\frac{2\pi}{n}$ radians, replace each basis vector with the representation of the next vertex in the polygon

Schoolgirls

- To avoid operations with very long integers, choose a random prime, and calculate the sum modulo this prime

Schoolgirls

- To avoid operations with very long integers, choose a random prime, and calculate the sum modulo this prime
- Asymptotics:

Schoolgirls

- To avoid operations with very long integers, choose a random prime, and calculate the sum modulo this prime
- Asymptotics:
  - $\mathcal{O}(\varphi(n))$ per parallelogram

Schoolgirls

- To avoid operations with very long integers, choose a random prime, and calculate the sum modulo this prime
- Asymptotics:
  - $\mathcal{O}(\varphi(n))$ per parallelogram
  - $\mathcal{O}(\varphi(n)^2)$ per one $\frac{2\pi}{n}$ rotation

Schoolgirls

- To avoid operations with very long integers, choose a random prime, and calculate the sum modulo this prime
- Asymptotics:
  - $\mathcal{O}(\varphi(n))$ per parallelogram
  - $\mathcal{O}(\varphi(n)^2)$ per one $\frac{2\pi}{n}$ rotation
  - $\mathcal{O}(n\varphi(n)^2)$ per one polygon check

Schoolgirls

- To avoid operations with very long integers, choose a random prime, and calculate the sum modulo this prime
- Asymptotics:
  - $\mathcal{O}(\varphi(n))$ per parallelogram
  - $\mathcal{O}(\varphi(n)^2)$ per one $\frac{2\pi}{n}$ rotation
  - $\mathcal{O}(n\varphi(n)^2)$ per one polygon check
- Still quite slow

Schoolgirls

- Let us embed this structure into $\mathbb{F}_p$ for some prime p

Schoolgirls

- Let us embed this structure into $\mathbb{F}_p$ for some prime p
- For that, n should divide $p - 1$

Schoolgirls

- Let us embed this structure into $\mathbb{F}_p$ for some prime p
- For that, n should divide $p - 1$
- Iterate over large numbers of form $kn + 1$ and check primality, find its generating root and take its k^{th} power g

Schoolgirls

- Let us embed this structure into $\mathbb{F}_p$ for some prime p
- For that, n should divide $p - 1$
- Iterate over large numbers of form $kn + 1$ and check primality, find its generating root and take its k^{th} power g
- Rotation around zero is $x \mapsto gx$, polygon is $g, g^2, \ldots, g^{n-1}, g^n = 1$

Schoolgirls

- What about m-gons, $m \neq n$?

Schoolgirls

- What about m-gons, $m \neq n$?
- If m does not divide $\operatorname{lcm}(n, 2)$, it is definitely not regular

Schoolgirls

- What about m-gons, $m \neq n$?
- If m does not divide $\operatorname{lcm}(n, 2)$, it is definitely not regular
- If n is odd, then a regular $2n$-gon is constructible, so let us start with $2n$ from the beginning

Schoolgirls

- What about m-gons, $m \neq n$?
- If m does not divide $\operatorname{lcm}(n, 2)$, it is definitely not regular
- If n is odd, then a regular $2n$-gon is constructible, so let us start with $2n$ from the beginning
- If m divides n, rotation $\frac{2\pi}{m}$ is $x \mapsto g^{n/m}x$

Problem C: Cherry Picking

Idea: Anton Maidel
Development: Anton Maidel
Editorial: Mikhail Ivanov

Cherry Picking

- You played n games of chess
- You are given the n chess ratings of your opponents
- For each game, you also know whether it was a win or a loss
- Find the maximum x such that, among the games against players with rating $\geq x$, there were k wins in a row

Cherry Picking

- Note that binary search is impossible: no monotonicity

Cherry Picking

- Note that binary search is impossible: no monotonicity
- There are two solutions: with a segment tree-like data structure and with DSU

Cherry Picking

- Data structures:

Cherry Picking

- Data structures:
  - Unpicked game $= 0$
  - Picked victory $= 1$
  - Picked defeat $= -\infty$

Cherry Picking

- Data structures:
  - Unpicked game $= 0$
  - Picked victory $= 1$
  - Picked defeat $= -\infty$
- Standard divide-and-conquer method for maximizing the sum on a subarray

Cherry Picking

- DSU:

Cherry Picking

- DSU:
- Gradually increase x

Cherry Picking

- DSU:
- Gradually increase x
- At first, we have many segments between defeats

Cherry Picking

- DSU:
- Gradually increase x
- At first, we have many segments between defeats
- What happens?
  - As a defeat disappears, two segments merge

Cherry Picking

- DSU:
- Gradually increase x
- At first, we have many segments between defeats
- What happens?
  - As a defeat disappears, two segments merge
  - As a victory disappears, a unit of value is lost

Cherry Picking

- DSU:
- Gradually increase x
- At first, we have many segments between defeats
- What happens?
  - As a defeat disappears, two segments merge
  - As a victory disappears, a unit of value is lost
- Instead of DSU, one can use `std::set` of pairs of integers
  - (moment of defeat, the chain of victories that it cut off)

Problem D: Dwarfs' Bedtime

Idea: Ivan Kazmenko
Development: Ivan Kazmenko
Editorial: Ivan Kazmenko

Dwarfs' Bedtime

- Snow White and n dwarfs live in the house
- Each dwarf is asleep for consecutive 12 hours each day (periodic), and awake also for consecutive 12 hours each day
- We have one day, from 00:00 to 23:59, to ask questions
- For each dwarf, we can interactively ask whether he is asleep or awake at most 50 times
- For each dwarf, find the exact minute when he goes to sleep
- Twist: we cannot return back in time to ask a question

Dwarfs' Bedtime

- Dwarfs are independent, let us solve the problem for one dwarf
- The constraints allow us, at each minute from 00:00 to 23:59, to check whether we have a question for each dwarf

Dwarfs' Bedtime

For every dwarf:

- First, ask at 00:00
- If the dwarf is awake, he will turn asleep and then turn awake
- If the dwarf is asleep, he will turn awake and then turn asleep
- The solutions are symmetric: just compare with state at 00:00, and add 12 hours at the end if needed

Dwarfs' Bedtime

For every dwarf:

- What if we could go back in time?
- Binary search: $12 \cdot 60 = 720$ minutes means 10 more questions

Dwarfs' Bedtime

For every dwarf:

- What are the key moments we can look for?
- 1. The dwarf changes state from 00:01 until 12:00
- 2. The dwarf changes state from 12:01 until 24:00
- But we can't ask at 24:00, so take care with the last minute
- (How: if we didn't find the answer, then the answer is 00:00)
- Idea: find the first moment approximately, then find the second moment precisely

Dwarfs' Bedtime

For every dwarf:

- Square-root approach: separate $729 > 720$ minutes into 27 sections of 27 minutes
- Until $12{:}00$, ask at the start of each section
- Until $24{:}00$, ask at each minute of the appropriate section
- $1 + 27 + 27 = 55 > 50$, a bit not enough

Dwarfs' Bedtime

For every dwarf:

- Refined square-root approach: separate $741 > 720$ minutes into 38 sections of $38, 37, 36, \ldots, 3, 2, 1$ minutes
- Until $12:00$, ask at the start of each section
- Until $24:00$, ask at each minute of the appropriate section
- If we got inside section k, it contains $39 - k$ minutes to ask
- The total number of questions will be $1 + 39 = 40 < 50$, which is quite enough

Problem E: Fake Coin and Lying Scales

Idea: Ivan Bochkov
Development: Ivan Bochkov
Editorial: Ivan Bochkov

Fake Coin and Lying Scales

- We have n coins and two-pan scales which may lie up to k times. One coin is fake, heavier than others.

Fake Coin and Lying Scales

- We have n coins and two-pan scales which may lie up to k times. One coin is fake, heavier than others.
- We need to find the fake coin.

Fake Coin and Lying Scales

- We have n coins and two-pan scales which may lie up to k times. One coin is fake, heavier than others.
- We need to find the fake coin.
- We may make up to $3k$ wrong guesses.

Fake Coin and Lying Scales

- We have n coins and two-pan scales which may lie up to k times. One coin is fake, heavier than others.
- We need to find the fake coin.
- We may make up to $3k$ wrong guesses.
- Our goal is to find the maximum possible n such that it is doable, with some accuracy (10 on the logarithmic scale).

Fake Coin and Lying Scales

- Suppose we made some weighings. What information do we have for any coin?

Fake Coin and Lying Scales

- Suppose we made some weighings. What information do we have for any coin?
- Only one number up to k: the number of times the scales lied to us if this coin is fake.

Fake Coin and Lying Scales

- Suppose we made some weighings. What information do we have for any coin?
- Only one number up to k: the number of times the scales lied to us if this coin is fake.
- Define a potential $p(\ell, v)$: the potential of coin if we may make up to ℓ weighings and lie up to v times, if this coin is fake.

Fake Coin and Lying Scales

- Suppose we made some weighings. What information do we have for any coin?
- Only one number up to k: the number of times the scales lied to us if this coin is fake.
- Define a potential $p(\ell, v)$: the potential of coin if we may make up to ℓ weighings and lie up to v times, if this coin is fake.
- p are taken in such a way that $p(0,0) = 1$ and $p(\ell, v) = 2p(\ell - 1, v - 1) + p(\ell - 1, v)$.

Fake Coin and Lying Scales

- Suppose we made some weighings. What information do we have for any coin?
- Only one number up to k: the number of times the scales lied to us if this coin is fake.
- Define a potential $p(\ell, v)$: the potential of coin if we may make up to ℓ weighings and lie up to v times, if this coin is fake.
- p are taken in such a way that $p(0, 0) = 1$ and $p(\ell, v) = 2p(\ell - 1, v - 1) + p(\ell - 1, v)$.
- The potential of a state is defined as the sum of potentials of all its coins.

Fake Coin and Lying Scales

- With this potential, we may note that the sum of potentials of 3 possible results of weighings is equal to the potential of the initial state.

Fake Coin and Lying Scales

- With this potential, we may note that the sum of potentials of 3 possible results of weighings is equal to the potential of the initial state.
- So $n \leq \frac{(3k+1)3^n}{p(n,k)}$.

Fake Coin and Lying Scales

- With this potential, we may note that the sum of potentials of 3 possible results of weighings is equal to the potential of the initial state.
- So $n \leq \frac{(3k+1)3^n}{p(n,k)}$.
- Also, we may note that we can split the potential almost equally on each step, so this approximation is good enough.

Fake Coin and Lying Scales

- With this potential, we may note that the sum of potentials of 3 possible results of weighings is equal to the potential of the initial state.
- So $n \leq \frac{(3k+1)3^n}{p(n,k)}$.
- Also, we may note that we can split the potential almost equally on each step, so this approximation is good enough.
- $p(n, k) = \sum_{j \leq k} C_n^j 2^j$.

Fake Coin and Lying Scales

- With this potential, we may note that the sum of potentials of 3 possible results of weighings is equal to the potential of the initial state.
- So $n \leq \frac{(3k+1)3^n}{p(n,k)}$.
- Also, we may note that we can split the potential almost equally on each step, so this approximation is good enough.
- $p(n,k) = \sum_{j \leq k} C_n^j 2^j$.
- All we need is to approximate this sum. This may be done in many ways, probably the easiest one is the following:

Fake Coin and Lying Scales

- Consider the maximal summand m instead of the whole sum.

Fake Coin and Lying Scales

- Consider the maximal summand m instead of the whole sum.
- Then $\frac{1}{4k^{\frac{1}{2}}}p(n,k) \leq m \leq p(n,k)$.

Fake Coin and Lying Scales

- Consider the maximal summand m instead of the whole sum.
- Then $\frac{1}{4k^{\frac{1}{2}}}p(n,k) \leq m \leq p(n,k)$.
- So we may take $\frac{m}{2k^{\frac{1}{4}}}$ as an approximation.

Fake Coin and Lying Scales

- Consider the maximal summand m instead of the whole sum.
- Then $\frac{1}{4k^{\frac{1}{2}}}p(n,k) \leq m \leq p(n,k)$.
- So we may take $\frac{m}{2k^{\frac{1}{4}}}$ as an approximation.
- Accuracy is $O(k^{\frac{1}{4}})$, which is good enough.

Fake Coin and Lying Scales

- Consider the maximal summand m instead of the whole sum.
- Then $\frac{1}{4k^{\frac{1}{2}}}p(n, k) \leq m \leq p(n, k)$.
- So we may take $\frac{m}{2k^{\frac{1}{4}}}$ as an approximation.
- Accuracy is $O(k^{\frac{1}{4}})$, which is good enough.
- It is possible to approximate with constant accuracy though.

Problem F: Whole World

Idea:	Mikhail Ivanov
	Ivan Bochkov
Development:	Ivan Bochkov
Editorial:	Ivan Bochkov

Whole World

- A polynomial is *whole* if it takes integer values at all integer points.

Whole World

- A polynomial is *whole* if it takes integer values at all integer points.
- We have some points (x_i, y_i) with $x_i \leq 30$.

Whole World

- A polynomial is *whole* if it takes integer values at all integer points.
- We have some points (x_i, y_i) with $x_i \leq 30$.
- What is the smallest degree of a *whole* polynomial taking these values in these points?

Whole World

- Whole polynomials are linear combinations of binomial coefficients.

Whole World

- Whole polynomials are linear combinations of binomial coefficients.
- So at least one such whole polynomial exists.

Whole World

- Whole polynomials are linear combinations of binomial coefficients.
- So at least one such whole polynomial exists.
- Let us first forget the condition that the polynomial should be whole.

Whole World

- Whole polynomials are linear combinations of binomial coefficients.
- So at least one such whole polynomial exists.
- Let us first forget the condition that the polynomial should be whole.
- Then we may just interpolate the given points and obtain some polynomial.

Whole World

- If it is whole, we win. And this condition is enough to check in points $1, 2, \ldots, d$, where d is its degree.

Whole World

- If it is whole, we win. And this condition is enough to check in points $1, 2, \ldots, d$, where d is its degree.
- If no, we may note that all denominators have divisors only from small powers of prime numbers up to 29.

Whole World

- If it is whole, we win. And this condition is enough to check in points $1, 2, \ldots, d$, where d is its degree.
- If no, we may note that all denominators have divisors only from small powers of prime numbers up to 29.
- Then it is enough to solve problem modulo these powers of primes, and take the maximum.

Whole World

- If it is whole, we win. And this condition is enough to check in points $1, 2, \ldots, d$, where d is its degree.
- If no, we may note that all denominators have divisors only from small powers of prime numbers up to 29.
- Then it is enough to solve problem modulo these powers of primes, and take the maximum.
- We can do a binary search by degree. How to check that a whole polynomial of given degree d exists?

Whole World

- If we take vectors $v_i = (C^i_{x_1}, \ldots, C^i_{x_n})$, we need to check that some given number is the linear combination of v_i.

Whole World

- If we take vectors $v_i = (C^i_{x_1}, \ldots, C^i_{x_n})$, we need to check that some given number is the linear combination of v_i.
- It is enough to check it modulo small powers of small primes.

Whole World

- If we take vectors $v_i = (C^i_{x_1}, \ldots, C^i_{x_n})$, we need to check that some given number is the linear combination of v_i.
- It is enough to check it modulo small powers of small primes.
- So, we need to solve some linear system modulo powers of primes, which may be done by a diagonalization process close to Gauss elimination.

Whole World

- If we take vectors $v_i = (C^i_{x_1}, \dots, C^i_{x_n})$, we need to check that some given number is the linear combination of v_i.
- It is enough to check it modulo small powers of small primes.
- So, we need to solve some linear system modulo powers of primes, which may be done by a diagonalization process close to Gauss elimination.
- By the way, you may prove that first part with interpolation and checking polynomial isn't necessary here. It is enough just to solve the system modulo prime powers.

Whole World

- If we take vectors $v_i = (C^i_{x_1}, \ldots, C^i_{x_n})$, we need to check that some given number is the linear combination of v_i.
- It is enough to check it modulo small powers of small primes.
- So, we need to solve some linear system modulo powers of primes, which may be done by a diagonalization process close to Gauss elimination.
- By the way, you may prove that first part with interpolation and checking polynomial isn't necessary here. It is enough just to solve the system modulo prime powers.
- Bonus. Solve it with $x_i \leq 10^9$.

Problem G: Unusual Case

Idea: Sergey Kopeliovich
Development: Sergey Kopeliovich
Editorial: Mikhail Ivanov

Unusual Case

- You are given a random undirected graph with n vertices and m edges
- Find k non-intersecting Hamiltonian paths in the given graph
- $n = 10\,000$, $m = 200\,000$, $k = 8$

Unusual Case

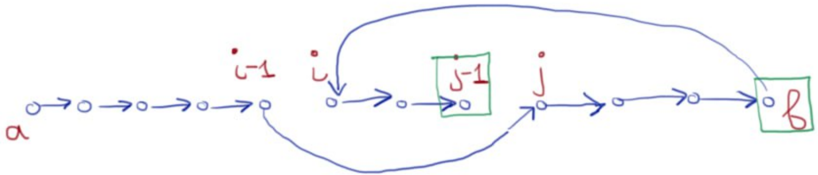
- How to find one path?

Unusual Case

- How to find one path?
- Greedy random walk

Unusual Case

- How to find one path?
- Greedy random walk
- If nowhere to go, rebuild as in the picture:



Unusual Case

- After finding k paths, start finding path $k + 1$ the same way

Unusual Case

- After finding k paths, start finding path $k+1$ the same way
- If we did not succeed to find 8 paths, start over

Unusual Case

- In 2021, it was proven that one path can be found in $\mathcal{O}(n)$

Unusual Case

- In 2021, it was proven that one path can be found in $\mathcal{O}(n)$
- After removing several random Hamiltonian paths, the graph is still pretty random

Problem H: Page on vdome.com

Idea: Mikhail Ivanov
Development: Anastasia Grigorieva
Editorial: Anastasia Grigorieva

Page on vdome.com

- Write down all the numbers from 1 to N, each one backwards.
- Remove all leading zeros.
- Find the Minimum EXcluded number (MEX).

Page on vdome.com

- For almost all N, the answer is 10.

Page on vdome.com

- For almost all N, the answer is 10.
- Because there are no page addresses where 0 is placed between "`id`" and the first significant digit.

Page on vdome.com

- For almost all N, the answer is 10.
- Because there are no page addresses where 0 is placed between "`id`" and the first significant digit.
- Thus, 10 cannot exist in the set of resulting numbers. And 10 will be the MEX for all $N \geqslant 10$.

Page on vdome.com

- For almost all N, the answer is 10.
- Because there are no page addresses where 0 is placed between "`id`" and the first significant digit.
- Thus, 10 cannot exist in the set of resulting numbers. And 10 will be the MEX for all $N \geqslant 10$.
- The answer for $N < 10$ is $N + 1$.

Problem I: Spin & Rotate!

Idea:	Mikhail Ivanov
Development:	Mikhail Ivanov
Editorial:	Mikhail Ivanov

Spin & Rotate!

- Consider a tangle of two ropes AB and CD

Spin & Rotate!

- Consider a tangle of two ropes AB and CD
- Two operations:

Spin & Rotate!

- Consider a tangle of two ropes AB and CD
- Two operations:
  - $\mathtt{S}$ — *spin*: spin the square $ABCD$ 90° ccw
  - $\mathtt{R}$ — *rotate*: swap ends A and D, rotating around each other ccw

Spin & Rotate!

- Consider a tangle of two ropes AB and CD
- Two operations:
  - S — *spin*: spin the square $ABCD$ 90° ccw
  - R — *rotate*: swap ends A and D, rotating around each other ccw
- You are given some initial sequence of operations

Spin & Rotate!

- Consider a tangle of two ropes AB and CD
- Two operations:
  - `S` — *spin*: spin the square $ABCD$ 90° ccw
  - `R` — *rotate*: swap ends A and D, rotating around each other ccw
- You are given some initial sequence of operations
- Perform more operations to disentangle the ropes

Spin & Rotate!

- The problem is based on a known plot

Spin & Rotate!

- The problem is based on a known plot
- Conway's Rational Tangles

Spin & Rotate!

- The problem is based on a known plot
- Conway's Rational Tangles
- Feel free to search it and watch some videos with people playing with two ropes!

Spin & Rotate!

- Redefine operation S

Spin & Rotate!

- Redefine operation $\mathbb{S}$
- Instead of rotating everything 90° ccw, let us imagine Ka-BAN going to next side cw

Spin & Rotate!

- Redefine operation $\mathtt{S}$
- Instead of rotating everything 90° ccw, let us imagine Ka-BAN going to next side cw
- Now $\mathtt{R}$ rotates not A and D, but along the side Ka-BAN is currently close to

Spin & Rotate!

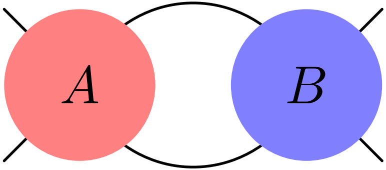
- Define operation $\dotplus$ — *horizontal sum*

Spin & Rotate!

- Define operation $\div$ — *horizontal sum*
- $A \div B$ is a tangle obtained by attaching B to the right of A

Spin & Rotate!

- Define operation $\div$ — *horizontal sum*
- $A \div B$ is a tangle obtained by attaching B to the right of A



Spin & Rotate!

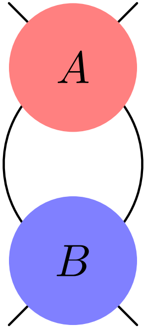
- Define operation $\cdot|\cdot$ — *vertical sum*

Spin & Rotate!

- Define operation $\cdot|\cdot$ — *vertical sum*
- $A \cdot|\cdot B$ is a tangle obtained by attaching B to the bottom of A

Spin & Rotate!

- Define operation $\cdot\!|\!\cdot$ — *vertical sum*
- $A \cdot\!|\!\cdot B$ is a tangle obtained by attaching B to the bottom of A



Spin & Rotate!

- Two basic tangles:

Spin & Rotate!

- Two basic tangles:
  - Horizontal unit H

Spin & Rotate!

- Two basic tangles:
  - Horizontal unit H

Spin & Rotate!

- Two basic tangles:
  - Horizontal unit H

  - Vertical unit V

Spin & Rotate!

- Two basic tangles:
  - Horizontal unit H

  - Vertical unit V

Spin & Rotate!

- Therefore, there are four possible $\mathtt{R}$ applied to a rational tangle T:

Spin & Rotate!

- Therefore, there are four possible $\mathtt{R}$ applied to a rational tangle T:
  - $T \mapsto T \cdot\!|\!\cdot V$
  - $T \mapsto T \div H$
  - $T \mapsto V \cdot\!|\!\cdot T$
  - $T \mapsto H \div T$

Spin & Rotate!

- Let us call a tangle *rational* if it is reachable from the initial tangle 0 via a sequence of `R` and `S`

Spin & Rotate!

- Let us call a tangle *rational* if it is reachable from the initial tangle 0 via a sequence of `R` and `S`
- Two rational tangles are *equivalent* if they are reachable from each other by smooth deformation above the square

Spin & Rotate!

Theorem

Any rational tangle is equivalent to a horizontally/vertically flipped one.

Spin & Rotate!

Theorem

Any rational tangle is equivalent to a horizontally/vertically flipped one.

Proof.

By induction. □

Spin & Rotate!

Theorem

Any rational tangle is equivalent to a horizontally/vertically flipped one.

Proof.

By induction. □

Corollary

$\div$ *and* $\cdot|\cdot$ *are commutative:* $A \div B = B \div A$, $A \cdot|\cdot B = B \cdot|\cdot A$.

Spin & Rotate!

- $T \div H = H \div T$

Spin & Rotate!

- $T \div H = H \div T$
- $T \cdot\!|\!\cdot V = V \cdot\!|\!\cdot T$

Spin & Rotate!

- $T \div H = H \div T$
- $T \cdot|\cdot V = V \cdot|\cdot T$
- Therefore, there are only two possible $\mathtt{R}$:

Spin & Rotate!

- $T \div H = H \div T$
- $T \cdot\!|\!\cdot V = V \cdot\!|\!\cdot T$
- Therefore, there are only two possible $\mathtt{R}$:
  - $T \mapsto T \div H$
  - $T \mapsto T \cdot\!|\!\cdot V$

Spin & Rotate!

- $T \div H = H \div T$
- $T \cdot\!|\!\cdot V = V \cdot\!|\!\cdot T$
- Therefore, there are only two possible $\mathtt{R}$:
  - $T \mapsto T \div H$
  - $T \mapsto T \cdot\!|\!\cdot V$
- Also, $\mathtt{S}$ undoes an $\mathtt{S}$

Spin & Rotate!

- $T \div H = H \div T$
- $T \cdot\!|\!\cdot V = V \cdot\!|\!\cdot T$
- Therefore, there are only two possible $\mathtt{R}$:
  - $T \mapsto T \div H$
  - $T \mapsto T \cdot\!|\!\cdot V$
- Also, $\mathtt{S}$ undoes an $\mathtt{S}$
- $\mathtt{S}^{-1} \sim \mathtt{S}$

Spin & Rotate!

- How to undo an `R`?

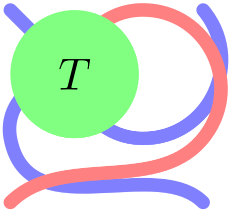
Spin & Rotate!

- How to undo an R?
- For instance, how to transform $T \div H \mapsto T$?

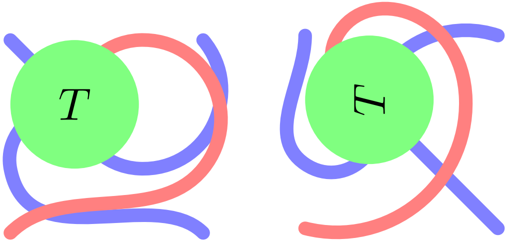
Spin & Rotate!

- How to undo an `R`?
- For instance, how to transform $T \div H \mapsto T$?
- Let us try to add a vertical unit: $(T \div H) \cdot\!|\!\cdot V$

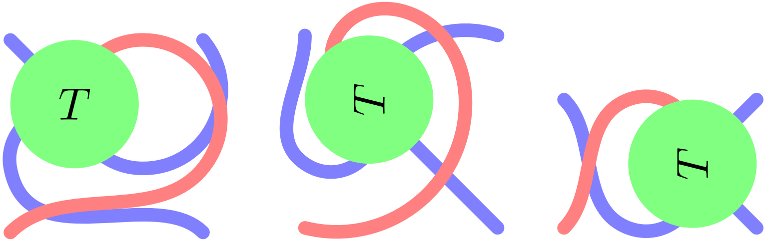
Spin & Rotate!



Spin & Rotate!



Spin & Rotate!



Spin & Rotate!

- So $((T \div H) \mathbin{\cdot\!|\!\cdot} V) \div H$ is actually just a rotated T

Spin & Rotate!

- So $((T \div H) \cdot|\cdot V) \div H$ is actually just a rotated T
- After three S the robot also changes its orientation

Spin & Rotate!

- So $((T \div H) \cdot\!|\!\cdot V) \div H$ is actually just a rotated T
- After three $\mathtt{S}$ the robot also changes its orientation
- Therefore, $\mathtt{RSRSRS} \sim \mathrm{id}$

Spin & Rotate!

- So $((T \div H) \cdot|\cdot V) \div H$ is actually just a rotated T
- After three S the robot also changes its orientation
- Therefore, $\mathrm{RSRSRS} \sim \mathrm{id}$
- $\mathrm{R}^{-1} \sim \mathrm{SRSRS}$

Spin & Rotate!

- Therefore, we can *somehow* undo any sequence

Spin & Rotate!

- Therefore, we can *somehow* undo any sequence
- Reverse the sequence, replace each `R` with `SRSRS`

Spin & Rotate!

- Therefore, we can *somehow* undo any sequence
- Reverse the sequence, replace each `R` with `SRSRS`
- But will it be the shortest one?

Spin & Rotate!

- Therefore, we can *somehow* undo any sequence
- Reverse the sequence, replace each `R` with `SRSRS`
- But will it be the shortest one?
- Probably no:

Spin & Rotate!

- Therefore, we can *somehow* undo any sequence
- Reverse the sequence, replace each `R` with `SRSRS`
- But will it be the shortest one?
- Probably no:
  - a substring `SS` can be removed

Spin & Rotate!

- Therefore, we can *somehow* undo any sequence
- Reverse the sequence, replace each `R` with `SRSRS`
- But will it be the shortest one?
- Probably no:
  - a substring `SS` can be removed
  - a substring `RSRSRS` can be removed

Spin & Rotate!

- Therefore, we can *somehow* undo any sequence
- Reverse the sequence, replace each `R` with `SRSRS`
- But will it be the shortest one?
- Probably no:
  - a substring `SS` can be removed
  - a substring `RSRSRS` can be removed
  - a suffix `RS` can be replaced with `S` (if we end with zero tangle)

Spin & Rotate!

- Therefore, we can *somehow* undo any sequence
- Reverse the sequence, replace each `R` with `SRSRS`
- But will it be the shortest one?
- Probably no:
  - a substring `SS` can be removed
  - a substring `RSRSRS` can be removed
  - a suffix `RS` can be replaced with `S` (if we end with zero tangle)
- Actually, these are enough!

Spin & Rotate!

- Let us assign a rational number (or ∞) to each rational tangle

Spin & Rotate!

- Let us assign a rational number (or ∞) to each rational tangle
- Initial tangle is 0

Spin & Rotate!

- Let us assign a rational number (or ∞) to each rational tangle
- Initial tangle is 0
- $x \overset{S}{\mapsto} -\frac{1}{x}$

Spin & Rotate!

- Let us assign a rational number (or ∞) to each rational tangle
- Initial tangle is 0
- $x \overset{S}{\mapsto} -\frac{1}{x}$
- $x \overset{R}{\mapsto} x + 1$

Spin & Rotate!

- Let us assign a rational number (or ∞) to each rational tangle
- Initial tangle is 0
- $x \overset{\mathrm{S}}{\mapsto} -\frac{1}{x}$
- $x \overset{\mathrm{R}}{\mapsto} x+1$
- $\frac{1}{0} = \infty, \quad \frac{1}{\infty} = 0, \quad \infty + 1 = \infty$

Spin & Rotate!

Theorem

Two rational tangles are equivalent if and only if their rational numbers are equal.

Theorem

If a sequence of $\mathcal{R}$ and $\mathcal{S}$ obtaining x from 0 does not contain a substring $\mathcal{SS}$, $\mathcal{RSRSRS}$, or a prefix $\mathcal{SR}$, it cannot be shortened.
If a sequence of $\mathcal{R}$ and $\mathcal{S}$ obtaining 0 from x does not contain a substring $\mathcal{SS}$, $\mathcal{SRSRSR}$, or a suffix $\mathcal{RS}$, it cannot be shortened.

Problem J: First Billion

Idea:	Sergey Kopeliovich
Development:	Sergey Kopeliovich
Editorial:	Mikhail Ivanov

First Billion

- We generated two sets of positive integers, each of size n and with sum 10^9

First Billion

- We generated two sets of positive integers, each of size n and with sum 10^9
- They are merged and shuffled into a set of size $N = 2n$

First Billion

- We generated two sets of positive integers, each of size n and with sum 10^9
- They are merged and shuffled into a set of size $N = 2n$
- Restore a subset of any size with sum 10^9

First Billion

- If there are $N \leq 18$ elements, we can solve in $\mathcal{O}(2^N)$

First Billion

- If there are $N \leq 18$ elements, we can solve in $\mathcal{O}(2^N)$
- If there are $N \leq 36$ elements, we can use meet-in-the-middle approach to solve in $\mathcal{O}(N \cdot 2^N)$

First Billion

- If there are $N \leq 18$ elements, we can solve in $\mathcal{O}(2^N)$
- If there are $N \leq 36$ elements, we can use meet-in-the-middle approach to solve in $\mathcal{O}(N \cdot 2^N)$
- What if $N > 36$?

First Billion

- Greedily distribute the numbers among $B = 36$ buckets

First Billion

- Greedily distribute the numbers among $B = 36$ buckets
- Solve in $\mathcal{O}^*(2^{B/2})$ time

First Billion

- Greedily distribute the numbers among $B = 36$ buckets
- Solve in $\mathcal{O}^*(2^{B/2})$ time
- Since 2^B is much larger than 10^9, the solution exists

Problem K: Tasks And Bugs

Idea: Nikolay Dubchuk
Development: Nikolay Dubchuk
Editorial: Nikolay Dubchuk

Tasks And Bugs

- There is a list of bugs, and for each bug, there is a list of tasks

Tasks And Bugs

- There is a list of bugs, and for each bug, there is a list of tasks
- Create a list of tasks with a list of bugs for each task

Tasks And Bugs

- Idea: create a map for tasks, add bugs

Tasks And Bugs

- Idea: create a map for tasks, add bugs
- Carefully output the result, sorting in numerical order, not lexicographical

Problem L: Candies

Idea:	Ivan Bochkov
Development:	Ivan Bochkov
Editorial:	Ivan Bochkov

Candies

- We have three integers x_1, x_2, x_3, initially zeroes.

Candies

- We have three integers x_1, x_2, x_3, initially zeroes.
- In one step, we increase one of them by 1, but x_1 should be the maximal one during the process.

Candies

- We have three integers x_1, x_2, x_3, initially zeroes.
- In one step, we increase one of them by 1, but x_1 should be the maximal one during the process.
- Calculate the number of way to obtain $x_1 = a$, $x_2 = b$, $x_3 = c$.

Candies

- We may generate answers for $a, b, c < 500$ using a dynamic programming solution in $\mathcal{O}(abc)$ time.

Candies

- We may generate answers for $a, b, c < 500$ using a dynamic programming solution in $\mathcal{O}(abc)$ time.
- Turns out that answers for $a = b$ (and $a = c$ by symmetry) may be described by a simple formula.

Candies

- We may generate answers for $a, b, c < 500$ using a dynamic programming solution in $\mathcal{O}(abc)$ time.
- Turns out that answers for $a = b$ (and $a = c$ by symmetry) may be described by a simple formula.
- Namely, if the answer to the problem is $f(a, b, c)$, then $f(a, a, 0) = \frac{(2n)!}{n!(n+1)!}$: the Catalan number.

Candies

- We may generate answers for $a, b, c < 500$ using a dynamic programming solution in $\mathcal{O}(abc)$ time.
- Turns out that answers for $a = b$ (and $a = c$ by symmetry) may be described by a simple formula.
- Namely, if the answer to the problem is $f(a, b, c)$, then $f(a, a, 0) = \frac{(2n)!}{n!(n+1)!}$: the Catalan number.
- Moreover, $f(a, a, k) = \frac{(2a+k)!}{a!(a+1)!} k! \cdot \prod_{m=a-k+1}^{a} \frac{2m}{2m+1}$.

Candies

- How to prove this? Well, it is some equality with hyperheometric coefficients, and it can be proved using the polynomial recurrence technique.

Candies

- How to prove this? Well, it is some equality with hyperheometric coefficients, and it can be proved using the polynomial recurrence technique.
- You may read about it, for example, in the book “A=B” by Doron Zeilberger.

Candies

- How to prove this? Well, it is some equality with hyperheometric coefficients, and it can be proved using the polynomial recurrence technique.
- You may read about it, for example, in the book "A=B" by Doron Zeilberger.
- I don't know the combinatorial meaning of this formula. If anyone has the idea, please share!

Candies

- What to to with general (a, b, c)?

Candies

- What to to with general (a, b, c)?
- Consider all ways to obtain (a, b, c) if we drop the condition $x_1 \geq x_2, x_3$.

Candies

- What to to with general (a, b, c)?
- Consider all ways to obtain (a, b, c) if we drop the condition $x_1 \geq x_2, x_3$.
- This will count some extra ways as well. What do they look like?

Candies

- What to to with general (a, b, c)?
- Consider all ways to obtain (a, b, c) if we drop the condition $x_1 \geq x_2, x_3$.
- This will count some extra ways as well. What do they look like?
- We reach the point (x, x, y) or (x, y, x) for some x, y, make a step to $(x, x+1, y)$ or $x, y, x+1$, and then somehow reach (a, b, c).

Candies

- What to to with general (a, b, c)?
- Consider all ways to obtain (a, b, c) if we drop the condition $x_1 \geq x_2, x_3$.
- This will count some extra ways as well. What do they look like?
- We reach the point (x, x, y) or (x, y, x) for some x, y, make a step to $(x, x+1, y)$ or $x, y, x+1$, and then somehow reach (a, b, c).
- If we fix x, y, this number may be calculated using $f(x, x, y)$.

Candies

- What to to with general (a, b, c)?
- Consider all ways to obtain (a, b, c) if we drop the condition $x_1 \geq x_2, x_3$.
- This will count some extra ways as well. What do they look like?
- We reach the point (x, x, y) or (x, y, x) for some x, y, make a step to $(x, x+1, y)$ or $x, y, x+1$, and then somehow reach (a, b, c).
- If we fix x, y, this number may be calculated using $f(x, x, y)$.
- We may check all pairs x, y. Asymptotic is $\mathcal{O}(a^2)$.

Candies

Any combinatorial meaning?

Problem M: Toilets

Idea: Leonid Dyachkov, Nikita Gaevoy
Development: Nikita Gaevoy
Editorial: Ivan Bochkov

Toilets

- Consider a circular office with toilets.
- Employees move around the office in one of two possible directions, looking for an empty toilet.
- Employees ignore occupied toilets, and when they find a vacant one, they occupy it for an amount of time, individual for each employee.
- We need to determine, for each employee, which toilet they will occupy and when.
- Ties when two employees contest for a toilet are broken with the time of walking or, equivalently, by employees' indices.

Toilets

- We want to simulate the process.
- We need to handle three possible situations:
  1. An employee finds a free toilet.
  2. A toilet becomes available.
  3. A new employee starts the journey.

Toilets

- We want to simulate the process.
- We need to handle three possible situations:
  1. An employee finds a free toilet.
  2. A toilet becomes available.
  3. A new employee starts the journey.
- All our events are essentially additions and removals of toilets and employees, so we win if we can maintain the most recent future event under these queries.

Optimizing the number of events

- The first idea is to maintain all such events in a heap.
- However, there are $\Theta(n^2)$ of them, so we can't do that directly.

Optimizing the number of events

- The first idea is to maintain all such events in a heap.
- However, there are $\Theta(n^2)$ of them, so we can't do that directly.
- We are interested only in the closest toilet to each employee and in two (one per direction) closest employees for each toilet.
- We can find those using `std::set` in $\mathcal{O}(\log(n + m))$ time.

Optimizing the number of events

- The first idea is to maintain all such events in a heap.
- However, there are $\Theta(n^2)$ of them, so we can't do that directly.
- We are interested only in the closest toilet to each employee and in two (one per direction) closest employees for each toilet.
- We can find those using `std::set` in $\mathcal{O}(\log(n+m))$ time.
- The remaining observation is that we can update only the nearest toilets and employees after each change, making only a constant number of additional events per query.
- Time complexity is $\mathcal{O}(n \log(n+m))$.

Problem N: (Un)labeled Graphs

Idea: Mikhail Ivanov
Development: Mikhail Ivanov
Editorial: Mikhail Ivanov

(Un)labeled Graphs

- You are given a labeled graph G

(Un)labeled Graphs

- You are given a labeled graph G
- Encode it with an unlabeled graph H

(Un)labeled Graphs

- You are given a labeled graph G
- Encode it with an unlabeled graph H
- Preceding decoding, the vertices of H shall be shuffled

(Un)labeled Graphs

- Idea: copy the initial graph G, write each vertex' number in binary

(Un)labeled Graphs

- Idea: copy the initial graph G, write each vertex' number in binary
- Create $\ell = \lceil \log_2 n \rceil$ auxiliary vertices $B_0, \ldots, B_{\ell-1}$ which encode these numbers

(Un)labeled Graphs

- Idea: copy the initial graph G, write each vertex' number in binary
- Create $\ell = \lceil \log_2 n \rceil$ auxiliary vertices $B_0, \ldots, B_{\ell-1}$ which encode these numbers
- How to distinguish between main and auxiliary vertices?

(Un)labeled Graphs

- Add two more vertices T_0, T_1, and connect them with all main vertices

(Un)labeled Graphs

- Add two more vertices T_0, T_1, and connect them with all main vertices
- Now T_0 and T_1 are the only vertices with coinciding neighborhood

(Un)labeled Graphs

- Add two more vertices T_0, T_1, and connect them with all main vertices
- Now T_0 and T_1 are the only vertices with coinciding neighborhood
- We can find the main vertices, we only need to enumerate them

(Un)labeled Graphs

- How to find the order on the auxiliary vertices?

(Un)labeled Graphs

- How to find the order on the auxiliary vertices?
- Add new vertex B_ℓ, add a path $B_0 B_1 \ldots B_\ell$

(Un)labeled Graphs

- How to find the order on the auxiliary vertices?
- Add new vertex B_ℓ, add a path $B_0 B_1 \ldots B_\ell$
- Now B_ℓ is the only auxiliary leaf

(Un)labeled Graphs

- How to find the order on the auxiliary vertices?
- Add new vertex B_ℓ, add a path $B_0 B_1 \ldots B_\ell$
- Now B_ℓ is the only auxiliary leaf
- $n + \lceil \log_2 n \rceil + 3$ vertices in total

Problem O: Mysterious Sequence

Idea: Nikolay Dubchuk
Development: Nikolay Dubchuk
Editorial: Nikolay Dubchuk

Mysterious Sequence

- There is a formula:

$$X_{i+2} = A \cdot X_{i+1} + B \cdot X_i$$

Mysterious Sequence

- There is a formula:

$$X_{i+2} = A \cdot X_{i+1} + B \cdot X_i$$

- The task is to reconstruct all the elements of the sequence knowing only the first and last numbers: X_1 and X_N

Mysterious Sequence

- Use binary search, find X_2, achieving the required precision with X_N

Mysterious Sequence

- Use binary search, find X_2, achieving the required precision with X_N
- Or a mathematical solution: after calculating a power of the matrix $\begin{pmatrix} A & B \\ 1 & 0 \end{pmatrix}$, we calculate X_2 using X_1 and X_N