KTH Challenge 2021
Solutions

2021-05-08

Made by Björn Martinsson

B - Bus Lines

Author: Nils Gustafsson

Given nodes labled 1 to n. Create a connected graph with m edges such that the sums of edge endpoints are distinct.

B - Bus Lines
Author: Nils Gustafsson

Given nodes labled 1 to n. Create a connected graph with m edges such that the sums of edge endpoints are distinct.

Here is one possible construction:

B - Bus Lines
Author: Nils Gustafsson

Given nodes labled 1 to n. Create a connected graph with m edges such that the sums of edge endpoints are distinct.

Here is one possible construction:

- Connect node 1 with all other nodes (connecting the graph).

B - Bus Lines
Author: Nils Gustafsson

Given nodes labled 1 to n. Create a connected graph with m edges such that the sums of edge endpoints are distinct.

Here is one possible construction:

- Connect node 1 with all other nodes (connecting the graph).
- Connect node n with all other nodes.

B - Bus Lines
Author: Nils Gustafsson

Given nodes labled 1 to n. Create a connected graph with m edges such that the sums of edge endpoints are distinct.

Here is one possible construction:

- Connect node 1 with all other nodes (connecting the graph).
- Connect node n with all other nodes.

Note This uses all possible sums. So it is not possible to add more edges.

D - DEX Save

Author: Per Austrin

Given the model of a dexterity saving throw in B&B, determine the probability of success. (not related to D&D)

D - DEX Save

Author: Per Austrin

Given the model of a dexterity saving throw in B&B, determine the probability of success. (not related to D&D)

Key insight Few dice, few sides.

D - DEX Save

Author: Per Austrin

Given the model of a dexterity saving throw in B&B, determine the probability of success. (not related to D&D)

Key insight Few dice, few sides.

Only at most $20^2 \cdot 10^5 = 4 \cdot 10^7$ possible outcomes.

D - DEX Save

Author: Per Austrin

Given the model of a dexterity saving throw in B&B, determine the probability of success. (not related to D&D)

Key insight Few dice, few sides.

Only at most $20^2 \cdot 10^5 = 4 \cdot 10^7$ possible outcomes.

Solution Loop over all possible outcomes.

D - DEX Save

Author: Per Austrin

Given the model of a dexterity saving throw in B&B, determine the probability of success. (not related to D&D)

Key insight Few dice, few sides.

Only at most $20^2 \cdot 10^5 = 4 \cdot 10^7$ possible outcomes.

Solution Loop over all possible outcomes.

Implementation Number of dice can vary, so recursion simplifies

A - Alto Singing
Author: Johan Sannemo

Given a song (list of tones) and a vocal range. Minimize the number accidentals by transposing (shifting). Also count how many such transpositions that exist.

$$D4 \quad D4 \quad E4 \quad D4 \quad G4 \quad F\sharp 4 \quad \nearrow_5 \quad G4 \quad G4 \quad A4 \quad G4 \quad C5 \quad B4$$

A - Alto Singing
Author: Johan Sannemo

Given a song (list of tones) and a vocal range. Minimize the number accidentals by transposing (shifting). Also count how many such transpositions that exist.

$$D4 \quad D4 \quad E4 \quad D4 \quad G4 \quad F\sharp 4 \quad \nearrow_5 \quad G4 \quad G4 \quad A4 \quad G4 \quad C5 \quad B4$$

Key insight There are fundamentally 12 different kinds of transpositions.

A - Alto Singing
Author: Johan Sannemo

Given a song (list of tones) and a vocal range. Minimize the number accidentals by transposing (shifting). Also count how many such transpositions that exist.

$$D4 \quad D4 \quad E4 \quad D4 \quad G4 \quad F\sharp4 \qquad \nearrow_5 \qquad G4 \quad G4 \quad A4 \quad G4 \quad C5 \quad B4$$

Key insight There are fundamentally 12 different kinds of transpositions.

Solution Check which of the 12 transpositions minimizes #accidentals in $O(n)$.

A - Alto Singing
Author: Johan Sannemo

Given a song (list of tones) and a vocal range. Minimize the number accidentals by transposing (shifting). Also count how many such transpositions that exist.

$$D4 \quad D4 \quad E4 \quad D4 \quad G4 \quad F\sharp 4 \quad \nearrow_5 \quad G4 \quad G4 \quad A4 \quad G4 \quad C5 \quad B4$$

Key insight There are fundamentally 12 different kinds of transpositions.

Solution Check which of the 12 transpositions minimizes #accidentals in $O(n)$.

Implementation Start by transposing the song down to the lowest allowed tone.

A - Alto Singing
Author: Johan Sannemo

Given a song (list of tones) and a vocal range. Minimize the number accidentals by transposing (shifting). Also count how many such transpositions that exist.

$$D4 \quad D4 \quad E4 \quad D4 \quad G4 \quad F\sharp 4 \quad \nearrow_5 \quad G4 \quad G4 \quad A4 \quad G4 \quad C5 \quad B4$$

Key insight There are fundamentally 12 different kinds of transpositions.

Solution Check which of the 12 transpositions minimizes #accidentals in $O(n)$.

Implementation Start by transposing the song down to the lowest allowed tone.

Use the highest tone in the song to count number of allowed octave transpositions.

C - Costly Contest
Author: Björn Martinsson

Given n participants and m available problems. You are given the task of designing a contest of duration t, split into k age-divisions, in order to minimize the total number of prize winners (participants solving the most problems in each division).

Participant i has a *slowness-factor* s_i and problem j has *difficulty rating* d_j. The time it takes participant i to solve problem j is $s_i \cdot d_j$.

C - Costly Contest
Author: Björn Martinsson

Given n participants and m available problems. You are given the task of designing a contest of duration t, split into k age-divisions, in order to minimize the total number of prize winners (participants solving the most problems in each division).

Participant i has a *slowness-factor* s_i and problem j has *difficulty rating* d_j. The time it takes participant i to solve problem j is $s_i \cdot d_j$.

Case $k = 1$ (Single division):

C - Costly Contest
Author: Björn Martinsson

Given n participants and m available problems. You are given the task of designing a contest of duration t, split into k age-divisions, in order to minimize the total number of prize winners (participants solving the most problems in each division).

Participant i has a *slowness-factor* s_i and problem j has *difficulty rating* d_j. The time it takes participant i to solve problem j is $s_i \cdot d_j$.

Case $k = 1$ (Single division):

Insight 1 Fastest participant will win a prize

C - Costly Contest

Author: Björn Martinsson

Given n participants and m available problems. You are given the task of designing a contest of duration t, split into k age-divisions, in order to minimize the total number of prize winners (participants solving the most problems in each division).

Participant i has a *slowness-factor* s_i and problem j has *difficulty rating* d_j. The time it takes participant i to solve problem j is $s_i \cdot d_j$.

Case $k = 1$ (Single division):

Insight 1 Fastest participant will win a prize

Insight 2 Minimize number of winners by picking a problemset minimizing the dead time for the fastest participant.

C - Costly Contest
Author: Björn Martinsson

Case $k = 1$ (single division):

Insight 1 Fastest participant will win a prize

Insight 2 Minimize number of winners by picking a problemset minimizing the dead time for the fastest participant.

Solution

C - Costly Contest
Author: Björn Martinsson

Case $k = 1$ (single division):

Insight 1 Fastest participant will win a prize

Insight 2 Minimize number of winners by picking a problemset minimizing the dead time for the fastest participant.

Solution

Find fastest participant. $O(n)$

C - Costly Contest

Author: Björn Martinsson

Case $k = 1$ (single division):

Insight 1 Fastest participant will win a prize

Insight 2 Minimize number of winners by picking a problemset minimizing the dead time for the fastest participant.

Solution

Find fastest participant. $O(n)$

Solve the subset sum minimizing dead time for fastest participant. $O(m\,t)$

C - Costly Contest

Author: Björn Martinsson

Case $k = 1$ (single division):

Insight 1 Fastest participant will win a prize

Insight 2 Minimize number of winners by picking a problemset minimizing the dead time for the fastest participant.

Solution

Find fastest participant. $O(n)$

Solve the subset sum minimizing dead time for fastest participant. $O(m\,t)$

Count number of participants tying fastest participant. $O(n)$

C - Costly Contest
Author: Björn Martinsson

Case $k > 1$ (multiple divisions):

C - Costly Contest
Author: Björn Martinsson

Case $k > 1$ (multiple divisions):

Key insight We can reuse solution for $k = 1$.

C - Costly Contest
Author: Björn Martinsson

Case $k > 1$ (multiple divisions):

Key insight We can reuse solution for $k = 1$.

For example, suppose that the winner for $k = 1$ would have been

$$L \quad W \quad W \quad L \quad W \quad L \quad L$$

C - Costly Contest
Author: Björn Martinsson

Case $k > 1$ (multiple divisions):

Key insight We can reuse solution for $k = 1$.

For example, suppose that the winner for $k = 1$ would have been

$$L \quad W \quad W \quad L \quad W \quad L \quad L$$

We can then find an optimal partition for $k = 4$ by doing

$$[\, L \quad W \,][\, W \quad L \,][\, W \quad L \,][\, L \,]$$

C - Costly Contest
Author: Björn Martinsson

Case $k > 1$ (multiple divisions):

Key insight We can reuse solution for $k = 1$.

C - Costly Contest
Author: Björn Martinsson

Case $k > 1$ (multiple divisions):

Key insight We can reuse solution for $k = 1$.

Solution Output $\max(k, \text{solution}(k = 1))$

C - Costly Contest

Author: Björn Martinsson

Case $k > 1$ (multiple divisions):

Key insight We can reuse solution for $k = 1$.

Solution Output $\max(k, \text{solution}(k = 1))$

Time complexity The solution takes $O(n + m\,t)$

E - Even Electricity

Author: Nils Gustafsson

Given a series of data over n days. On the i-th day

E - Even Electricity

Author: Nils Gustafsson

Given a series of data over n days. On the i-th day

s_i units of solar powered electricity was generated.

E - Even Electricity
Author: Nils Gustafsson

Given a series of data over n days. On the i-th day

s_i units of solar powered electricity was generated.

w_i units of water flowed into the reservoir (can on demand be converted to energy).

E - Even Electricity
Author: Nils Gustafsson

Given a series of data over n days. On the i-th day

s_i units of solar powered electricity was generated.

w_i units of water flowed into the reservoir (can on demand be converted to energy).

Let e_i be the amount of energy produced on day i.

E - Even Electricity

Author: Nils Gustafsson

Given a series of data over n days. On the i-th day

s_i units of solar powered electricity was generated.

w_i units of water flowed into the reservoir (can on demand be converted to energy).

Let e_i be the amount of energy produced on day i.

Task Assuming you need to use up all the water.

$$\text{Minimize:} \quad \max_i (e_i) - \min_i (e_i).$$

E - Even Electricity
Author: Nils Gustafsson

Insight 1 It is possbible to make function $\text{check}(L, R)$ that in $O(n)$ time checks if it is possible to keep $L \leqslant e_i \leqslant R$ for all i.

E - Even Electricity
Author: Nils Gustafsson

Insight 1 It is possbibile to make function $\text{check}(L, R)$ that in $O(n)$ time checks if it is possible to keep $L \leqslant e_i \leqslant R$ for all i.

This is done by keeping track of how much reservoir water you can have at the end of each day (it will be an interval).

E - Even Electricity
Author: Nils Gustafsson

Insight 1 It is possbible to make function $\text{check}(L, R)$ that in $O(n)$ time checks if it is possible to keep $L \leqslant e_i \leqslant R$ for all i.

This is done by keeping track of how much reservoir water you can have at the end of each day (it will be an interval).

Insight 2 The function $\text{check}(L, R)$ can actually be split into testing L and R seperately. So essentially $\text{check}(L, R) = \text{check}(L, \infty) \wedge \text{check}(-\infty, R)$.

E - Even Electricity
Author: Nils Gustafsson

Insight 1 It is possbible to make function $\text{check}(L, R)$ that in $O(n)$ time checks if it is possible to keep $L \leqslant e_i \leqslant R$ for all i.

This is done by keeping track of how much reservoir water you can have at the end of each day (it will be an interval).

Insight 2 The function $\text{check}(L, R)$ can actually be split into testing L and R seperately. So essentially $\text{check}(L, R) = \text{check}(L, \infty) \wedge \text{check}(-\infty, R)$.

Solution Use binary search on $\text{check}(L, \infty)$ and $\text{check}(-\infty, R)$ to find largest feasible L and smallest feasible R.

E - Even Electricity
Author: Nils Gustafsson

Insight 1 It is possbible to make function $\mathrm{check}(L,R)$ that in $O(n)$ time checks if it is possible to keep $L \leqslant e_i \leqslant R$ for all i.

This is done by keeping track of how much reservoir water you can have at the end of each day (it will be an interval).

Insight 2 The function $\mathrm{check}(L,R)$ can actually be split into testing L and R seperately. So essentially $\mathrm{check}(L,R) = \mathrm{check}(L,\infty) \wedge \mathrm{check}(-\infty,R)$.

Solution Use binary search on $\mathrm{check}(L,\infty)$ and $\mathrm{check}(-\infty,R)$ to find largest feasible L and smallest feasible R.

Time complexity $O(n \log m)$, where $m = 10^9$.

F - Forgotten Homework

Author: Björn Martinsson

Given a $n \times n$ matrix A, two indices i and j, and the sequence

$$A^1(i,j), A^2(i,j), \ldots, A^{2n-1}(i,j)$$

Output $A^{2n}(i,j)$ modulo $10^9 + 7$.

F - Forgotten Homework

Author: Björn Martinsson

Given a $n \times n$ matrix A, two indices i and j, and the sequence

$$A^1(i,j), A^2(i,j), \ldots, A^{2n-1}(i,j)$$

Output $A^{2n}(i,j)$ modulo 10^9+7.

Let $a_k \overset{\text{def}}{=\!=} A^k(i,j)$.

F - Forgotten Homework
Author: Björn Martinsson

Given a $n \times n$ matrix A, two indices i and j, and the sequence

$$A^1(i,j), A^2(i,j), \ldots, A^{2n-1}(i,j)$$

Output $A^{2n}(i,j)$ modulo $10^9 + 7$.

Let $a_k \overset{\text{def}}{=\!=} A^k(i,j)$.

Key insight The sequence $\{a_k\}_{k \geqslant 1}$ has a *linear recurrence* of length n.

F - Forgotten Homework
Author: Björn Martinsson

Given a $n \times n$ matrix A, two indices i and j, and the sequence

$$A^1(i,j), A^2(i,j), \ldots, A^{2n-1}(i,j)$$

Output $A^{2n}(i,j)$ modulo $10^9 + 7$.

Let $a_k \overset{\text{def}}{=\!=} A^k(i,j)$.

Key insight The sequence $\{a_k\}_{k \geqslant 1}$ has a *linear recurrence* of length n.

Example of a linear recurrence of length 2: The Fibonacci sequence

Key insight The sequence $\{a_k\}_{k\geqslant 1}$ has a linear recurrence of length n.

Key insight The sequence $\{a_k\}_{k \geqslant 1}$ has a linear recurrence of length n.

Why?

Key insight The sequence $\{a_k\}_{k\geqslant 1}$ has a linear recurrence of length n.

Why?

Because of Cayley-Hamilton Theorem.

Key insight The sequence $\{a_k\}_{k\geqslant 1}$ has a linear recurrence of length n.

Why?

Because of Cayley-Hamilton Theorem.

The theorem essentially states that there exists integers $c_0, \ldots, c_{n-1}$ such that

$$A^n \;=\; c_{n-1}\,A^{n-1} + c_{n-2}\,A^{n-2} + \ldots + c_0\,I_n.$$

Key insight The sequence $\{a_k\}_{k \geqslant 1}$ has a linear recurrence of length n.

Why?

Because of Cayley-Hamilton Theorem.

The theorem essentially states that there exists integers $c_0, \ldots, c_{n-1}$ such that

$$A^n \; = \; c_{n-1} \, A^{n-1} + c_{n-2} \, A^{n-2} + \ldots + c_0 \, I_n.$$

Since we can multiply LHS and RHS by A, we have that for $k \geqslant n$

$$A^k \; = \; c_{n-1} \, A^{k-1} + c_{n-2} \, A^{k-2} + \ldots + c_0 \, A^{k-n}.$$

Key insight The sequence $\{a_k\}_{k \geqslant 1}$ has a linear recurrence of length n.

Why?

Because of Cayley-Hamilton Theorem.

The theorem essentially states that there exists integers $c_0, \ldots, c_{n-1}$ such that

$$A^n \;=\; c_{n-1}\, A^{n-1} + c_{n-2}\, A^{n-2} + \ldots + c_0\, I_n.$$

Since we can multiply LHS and RHS by A, we have that for $k \geqslant n$

$$A^k \;=\; c_{n-1}\, A^{k-1} + c_{n-2}\, A^{k-2} + \ldots + c_0\, A^{k-n}.$$

This is a matrix equality, so in particular it will also hold that

$$A^k(i,j) \;=\; c_{n-1}\, A^{k-1}(i,j) + c_{n-2}\, A^{k-2}(i,j) + \ldots + c_0\, A^{k-n}(i,j).$$

F - Forgotten Homework
Author: Björn Martinsson

Key insight The sequence $\{a_k\}_{k \geqslant 1}$ has a linear recurrence of length n.

F - Forgotten Homework
Author: Björn Martinsson

Key insight The sequence $\{a_k\}_{k\geqslant 1}$ has a linear recurrence of length n.

So how do we figure out the linear recurrence of a_k?

F - Forgotten Homework
Author: Björn Martinsson

Key insight The sequence $\{a_k\}_{k\geqslant 1}$ has a linear recurrence of length n.

So how do we figure out the linear recurrence of a_k?

We use Berlekamp-Massey's algorithm!

F - Forgotten Homework
Author: Björn Martinsson

Key insight The sequence $\{a_k\}_{k \geqslant 1}$ has a linear recurrence of length n.

So how do we figure out the linear recurrence of a_k?

We use Berlekamp-Massey's algorithm!

The algorithm recovers the shortest linear recurrence from a sequence. As input it needs two times the length of the shortest linear recurrence.

F - Forgotten Homework
Author: Björn Martinsson

Key insight The sequence $\{a_k\}_{k \geqslant 1}$ has a linear recurrence of length n.

So how do we figure out the linear recurrence of a_k?

We use Berlekamp-Massey's algorithm!

The algorithm recovers the shortest linear recurrence from a sequence. As input it needs two times the length of the shortest linear recurrence.

The linear recurrence can be n long, so Berlekamp-Massey algorithms needs the first $2n$ values. But we only have $2n-1$ values.

F - Forgotten Homework
Author: Björn Martinsson

Problem We need $2n$ the first values of a_k, but we only know of $2n-1$ values. We need one more value.

F - Forgotten Homework
Author: Björn Martinsson

Problem We need $2n$ the first values of a_k, but we only know of $2n-1$ values. We need one more value.

Solution Look back at Cayley-Hamilton Theorem

$$A^n(i,j) \;=\; c_{n-1}A^{n-1}(i,j) + c_{n-2}A^{n-2}(i,j) + \ldots + c_0 I_n(i,j).$$

F - Forgotten Homework
Author: Björn Martinsson

Problem We need $2n$ the first values of a_k, but we only know of $2n-1$ values. We need one more value.

Solution Look back at Cayley-Hamilton Theorem

$$A^n(i,j) \;=\; c_{n-1}A^{n-1}(i,j) + c_{n-2}A^{n-2}(i,j) + \ldots + c_0 I_n(i,j).$$

This means that it is natural to make

$$a_0 \;=\; \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

F - Forgotten Homework
Author: Björn Martinsson

Conclusion

1. Let $a_0 \stackrel{\text{def}}{=\!=} \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$

F - Forgotten Homework
Author: Björn Martinsson

Conclusion

1. Let $a_0 \stackrel{\text{def}}{=\!=} \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$

2. Run Berlekamp-Massey on $a_0, \ldots, a_{2n-1}$ (takes $O(n^2)$ time)

F - Forgotten Homework

Author: Björn Martinsson

Conclusion

1. Let $a_0 \overset{\text{def}}{=\!=} \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$
2. Run Berlekamp-Massey on $a_0, \ldots, a_{2n-1}$ (takes $O(n^2)$ time)
3. Calculate a_{2n} using the linear recurrence. (takes $O(n)$ time)

G - Guessing Circle

Author: Johan Sannemo

There are n numbers laid out on a circle. The same number can occur multiple times. Alf and Beta are playing a game where Alf thinks of a number x on the circle and Beta tries to guess it.

G - Guessing Circle
Author: Johan Sannemo

There are n numbers laid out on a circle. The same number can occur multiple times. Alf and Beta are playing a game where Alf thinks of a number x on the circle and Beta tries to guess it.

Beta picks a y and Alf answers which direction (clock-wise or counter-clockwise) is closest to x. If there are multiple possible answers then Alf can pick whichever answer he wants.

G - Guessing Circle
Author: Johan Sannemo

There are n numbers laid out on a circle. The same number can occur multiple times. Alf and Beta are playing a game where Alf thinks of a number x on the circle and Beta tries to guess it.

Beta picks a y and Alf answers which direction (clock-wise or counter-clockwise) is closest to x. If there are multiple possible answers then Alf can pick whichever answer he wants.

Task Output all x that it is possible for Beta to guess given that Beta can ask as many questions as he wants.

G - Guessing Circle

Author: Johan Sannemo

Cubic solution In $O(n^2)$ time create a matrix M where

$$M(x, y) = \begin{cases} 1 & \text{if Alf needs to answer CW on query } (x, y) \\ 0 & \text{otherwise.} \end{cases}$$

G - Guessing Circle
Author: Johan Sannemo

Cubic solution In $O(n^2)$ time create a matrix M where

$$M(x, y) = \begin{cases} 1 & \text{if Alf needs to answer CW on query } (x, y) \\ 0 & \text{otherwise.} \end{cases}$$

Let x_1 and x_2 be two possible values that Alf could be thinking of. Beta can distinguish between the two iff there exists a y such that

$$M(x_1, y) = M(y, x_2) \quad \text{or} \quad M(y, x_1) = M(x_2, y).$$

G - Guessing Circle
Author: Johan Sannemo

Cubic solution In $O(n^2)$ time create a matrix M where

$$M(x, y) = \begin{cases} 1 & \text{if Alf needs to answer CW on query } (x, y) \\ 0 & \text{otherwise.} \end{cases}$$

Let x_1 and x_2 be two possible values that Alf could be thinking of. Beta can distinguish between the two iff there exists a y such that

$$M(x_1, y) = M(y, x_2) \quad \text{or} \quad M(y, x_1) = M(x_2, y).$$

This allows for a $O(n^3)$ solution. Or $n^3/64$ using bitset.

G - Guessing Circle
Author: Johan Sannemo

Cubic solution In $O(n^2)$ time create a matrix M where

$$M(x, y) = \begin{cases} 1 & \text{if Alf needs to answer CW on query } (x, y) \\ 0 & \text{otherwise.} \end{cases}$$

Let x_1 and x_2 be two possible values that Alf could be thinking of. Beta can distinguish between the two iff there exists a y such that

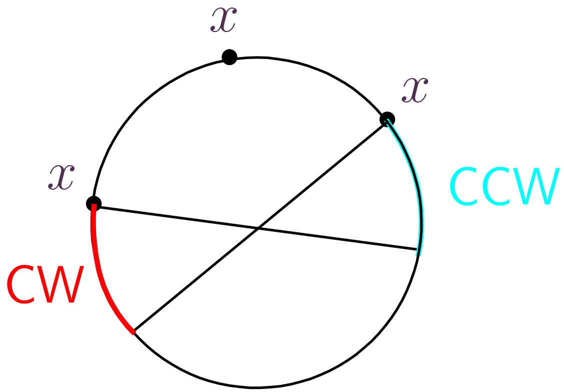
$$M(x_1, y) = M(y, x_2) \quad \text{or} \quad M(y, x_1) = M(x_2, y).$$

This allows for a $O(n^3)$ solution. Or $n^3/64$ using bitset.

But both are too slow. $n = 15000$.

G - Guessing Circle
Author: Johan Sannemo

Quadratic solution: For each x there are two fundamental intervals of positions for which Alf answers CW or CCW.



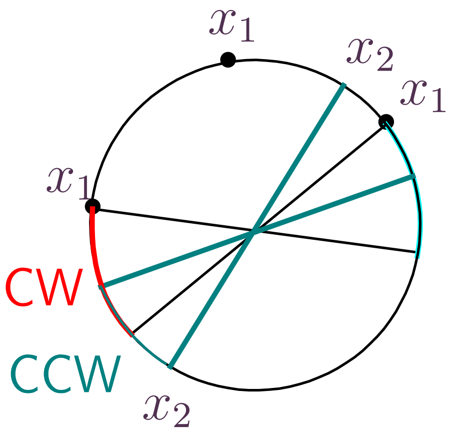
Note: An interval works for y iff all occurences of y lies inside the interval.

Suppose we want to see if x_1 and x_2 can be distiguished asking queries.

This means that we need to find a y such that Alf is forced to answer differently for $\mathrm{query}(x_1, y)$ and $\mathrm{query}(x_2, y)$.

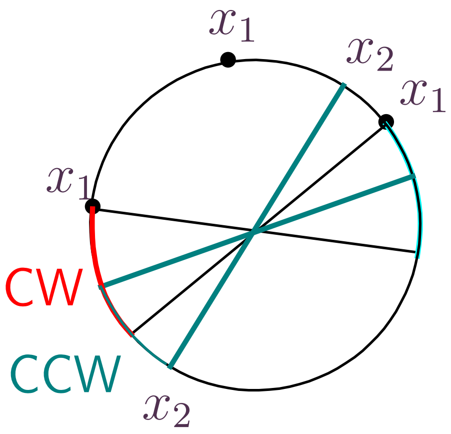
Suppose we want to see if x_1 and x_2 can be distiguished asking queries.

This means that we need to find a y such that Alf is forced to answer differently for $\mathrm{query}(x_1, y)$ and $\mathrm{query}(x_2, y)$.



Suppose we want to see if x_1 and x_2 can be distiguished asking queries.

This means that we need to find a y such that Alf is forced to answer differently for $\mathrm{query}(x_1, y)$ and $\mathrm{query}(x_2, y)$.



A y forces CW for x_1 and CCW for x_2 iff all occurrences of y lies inside the intersection of CW and CCW.

So we need to quickly be able to:

So we need to quickly be able to:

Given an interval. Does there exists a y occurring only inside the interval?

So we need to quickly be able to:

Given an interval. Does there exists a y occurring only inside the interval?

This is a monotone property. So we can precalculate all answers with DP in $O(n^2)$ time with $O(n)$ memory.

So we need to quickly be able to:

Given an interval. Does there exists a y occurring only inside the interval?

This is a monotone property. So we can precalculate all answers with DP in $O(n^2)$ time with $O(n)$ memory.

Time complexity $O(n^2)$

So we need to quickly be able to:

Given an interval. Does there exists a y occurring only inside the interval?

This is a monotone property. So we can precalculate all answers with DP in $O(n^2)$ time with $O(n)$ memory.

Time complexity $O(n^2)$

Memory complexity $O(n)$

So we need to quickly be able to:

Given an interval. Does there exists a y occurring only inside the interval?

This is a monotone property. So we can precalculate all answers with DP in $O(n^2)$ time with $O(n)$ memory.

Time complexity $O(n^2)$

Memory complexity $O(n)$

Extra challenge: Try solving the problem in linear time. It is possible.

Thanks for participating in KTH Challenge 2021!

Orginizers

- Per Austrin (KTH)
- Nils Gustafsson (Depict.ai)
- Björn Martinsson (KTH)
- Johan Sannemo (Kognity)