UNIVERSITÄT DES SAARLANDES

max planck institut informatik

Sublinear Algorithms

Lecture 11: Applications III – String Algorithms

Karl Bringmann

July 9, 2020

European Research Council

Established by the European Commission

Recap: Convolution

Convolution: Let $u, v \in \mathbb{R}^n$

Their convolution is the vector $u * v \in \mathbb{R}^{2n}$ with $(u * v)_i = \sum_{j=0}^{i} u_j v_{i-j}$

It can be computed in time $\tilde{O}(out) = \tilde{O}(n)$

$\tilde{O}$ hides factor polylog(n)

Variant: IP-Convolution

Convolution: Let $u, v \in \mathbb{R}^n$

Their convolution is the vector $u * v \in \mathbb{R}^{2n}$ with $(u * v)_i = \sum_{j=0}^{i} u_j v_{i-j}$

It can be computed in time $\tilde{O}(out) = \tilde{O}(n)$

IP-Convolution: Let $u \in \mathbb{R}^n, v \in \mathbb{R}^m$ with $n \geq m$

Their IP-convolution is the vector $u *_{IP} v \in \mathbb{R}^{n-m+1}$ with $(u *_{IP} v)_i = \boxed{\sum_{j=1}^{m} u_{i+j} v_j}$

Compute inner product of v *and* ***each window*** $\langle u[i+1 \dots i+m], v \rangle$

Variant: IP-Convolution

Convolution: Let $u, v \in \mathbb{R}^n$

Their convolution is the vector $u * v \in \mathbb{R}^{2n}$ with $(u * v)_i = \sum_{j=0}^{i} u_j v_{i-j}$

It can be computed in time $\tilde{O}(out) = \tilde{O}(n)$

IP-Convolution: Let $u \in \mathbb{R}^n, v \in \mathbb{R}^m$ with $n \geq m$

Their IP-convolution is the vector $u *_{IP} v \in \mathbb{R}^{n-m+1}$ with $(u *_{IP} v)_i = \sum_{j=1}^{m} u_{i+j} v_j$

It can be computed in time $\tilde{O}(out) = \tilde{O}(n)$

$x = (0, v_1, \dots, v_m, 0, \dots, 0) \in \mathbb{R}^{n+1}$

$y = (u_n, \dots, u_1, 0, \dots, 0) \in \mathbb{R}^{n+1}$

Consider $m \leq i \leq n$

$x_j = v_j$

$y_j = u_{n-j}$

Then $(x * y)_i = \sum_{j=0}^{i} x_j y_{i-j}$

$= \sum_{j=1}^{m} v_j u_{n-(i-j)} = \sum_{j=1}^{m} u_{(n-i)+j} v_j$

So $(u *_{IP} v)_i = (x * y)_{n-i}$

Pattern Matching

Given two strings: text T of length n and pattern P of length $m \le n$

Does P appear as a substring of T?

optimal classic algorithm:

Knuth, Morris, Pratt '77: time $O(n)$

Let's consider generalizations!

T:	**a**	**b**	**c**	**b**	**b**	**c**	**a**	**a**	
P:	b	b	c	a					✗
		b	b	c	a				✗
			b	b	c	a			✗
i=3				b	b	c	a		✓
					b	b	c	a	✗

window $T_i = T[i+1..i+m]$

Is $P = T_i$ for some i?

Text-to-Pattern Hamming Distance

Given two strings: text T of length n and pattern P of length $m \le n$

Compute Hamming distance between P and each window T_i

$\Sigma = \{a, b, c\}$

Algorithm: [Abrahamson'87] Time $\tilde{O}(|\Sigma| n)$

For each symbol σ in alphabet Σ:

Construct vector u with $u_j = [T[j] = \sigma] = \begin{cases} 1, & \text{if } T[j] = \sigma \\ 0, & \text{ow.} \end{cases}$

Construct vector v with $v_j = [P[j] = \sigma]$

Compute vector w^σ = **IP-convolution** of u and v $\tilde{O}(n)$

// Then $w_i^\sigma = \sum_j u_{i+j} v_j = \underbrace{\sum_{j=1}^{m} [T[i+j] = \sigma] \cdot [P[j] = \sigma]}_{\#\text{matches of } \sigma \text{ for shift } i}$

Return $d_i = m - \sum_{\sigma \in \Sigma} w_i^\sigma$ for all i

a	b	c	b	b	c	a	a	
b	b	c	a					2
	b	b	c	a				3
		b	b	c	a			3
			b	b	c	a		0
				b	b	c	a	2

T_0

$d_H(X, Y) = \#\{j \mid X[j] \neq Y[j]\}$

$T_i = T[i+1..i+m]$

$d_i = d_H(P, T_i)$

New Tool: Subset Shifts

Given $A, B \subseteq \mathbb{Z}$, determine all x such that $A + x \subseteq B$

Subset Shifts:

Given $A, B \subseteq \{0, \dots, d\}$

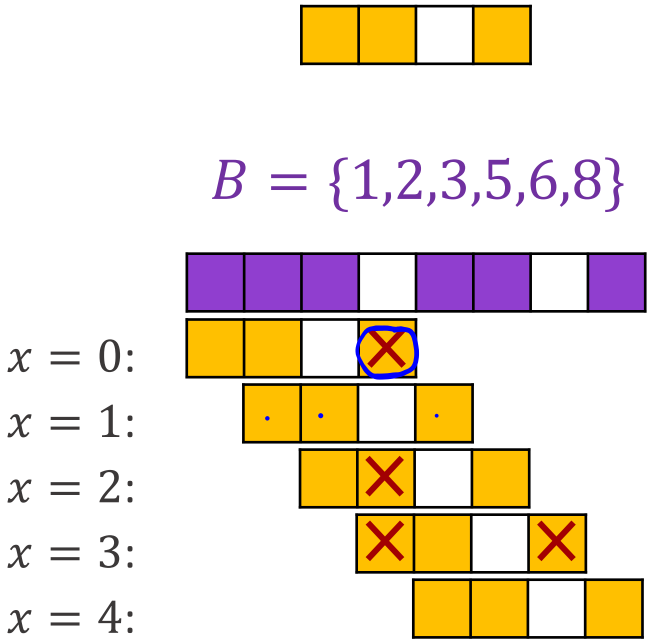
Compute all x with $A + x \subseteq B$:

$$\underline{A \Uparrow B} := \{x \mid A + x \subseteq B\}$$

This is in time $\tilde{O}(|A| + |B|)$ w.h.p.

$A = \{1,2,4\}$

$B = \{1,2,3,5,6,8\}$



$A \Uparrow B = \boxed{\{1,4\}}$

Wildcard Matching

Given two strings: text T of length n and pattern P of length $m \leq n$

P contains wildcards „$*$“ that match any symbol. Does P match any substring of T?

Algorithm: [Cole,Hariharan‘02] Time $\tilde{O}(n)$

For each symbol σ in alphabet Σ:

Construct set $A^\sigma = \{j \mid P[j] = \sigma\}$

Construct set $B^\sigma = \{j \mid T[j] = \sigma\}$

Compute $A^\sigma \Uparrow B^\sigma$ *// = all shifts s.t. all σ‘s in P are matched*

Return $\left(\bigcap_{\sigma\in\Sigma} \boxed{A^\sigma \Uparrow B^\sigma}\right) \cap \{0,\dots,n-m\}$

$\sum_\sigma \tilde{O}(|A^\sigma| + |B^\sigma|) = \tilde{O}(n+m)$

T:	a	b	c	b	b	c	a	a	
P:	b	$*$	$*$	a					X
		b	$*$	$*$	a				X
			b	$*$	$*$	a			X
				b	$*$	$*$	a		✓
					b	$*$	$*$	a	✓

$T \in \Sigma^n, P \in (\Sigma \cup \{*\})^m$

P matches T_i if for all $1 \leq j \leq m$:

$P[j] = *$ or $P[j] = T[i+j]$

Subset Matching

Given two sequences: text T of length n and pattern P of length $m \leq n$

Each position is a set of alphabet symbols: $T[j], P[j] \subseteq \Sigma$.

P matches window T_i if for all $1 \leq j \leq m$ we have $P[j] \subseteq T[i+j]$

Does P match any window T_i?

Algorithm: [Cole,Hariharan'02] Time $\tilde{O}(s)$

For each symbol σ in alphabet Σ:

Construct set $A^\sigma = \{j \mid \sigma \in P[j]\}$

Construct set $B^\sigma = \{j \mid \sigma \in T[j]\}$

Compute $A^\sigma \Uparrow B^\sigma$ *// = all shifts s.t. all σ's in P are matched*

Return $\left(\bigcap_{\sigma\in\Sigma} A^\sigma \Uparrow B^\sigma\right) \cap \{0, \dots, n-m\}$

$\sum_\sigma |A^\sigma| = \sum_j |P[j]|$

$\tilde{O}(|A^\sigma| + |B^\sigma|)$

a,b	**b**	**c**	**b**	**a,b**	**a,b**	**∅**	
b	∅	∅	a,b				✗
	b	∅	∅	a,b			✓
		b	∅	∅	a,b		✗
			b	∅	∅	a,b	✗

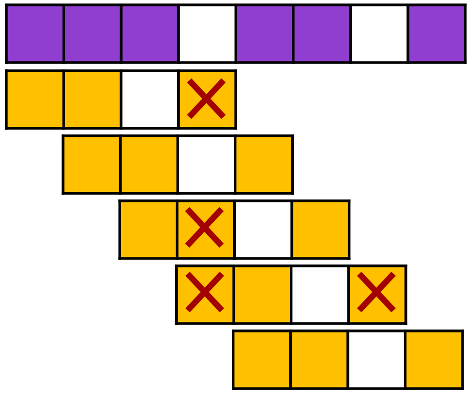
Input size $s = \boxed{\sum_j |T[j]| + \sum_j |P[j]|}$

Subset Shifts

Given A, B, determine all x such that $A + x \subseteq B$

$A = \{1,2,4\}$

$B = \{1,2,3,5,6,8\}$



$A \Uparrow B = \{1,4\}$

Subset Shifts:

Given $A, B \subseteq \{0, \dots, d\}$

Compute all x with $A + x \subseteq B$:

$$A \Uparrow B := \{x \mid A + x \subseteq B\}$$

This is in time $\tilde{O}(|A| + |B|)$ w.h.p.

Write

$A = \{a_1, \dots, a_{|A|}\}$,

$B = \{b_1, \dots, b_{|B|}\}$

We must have $|A| \leq |B|$ otherwise $A \Uparrow B = \emptyset$, since $|A + x| = |A| > |B|$

$A \Uparrow B \subseteq B - a_1 = \{b_1 - a_1, b_2 - a_1, \dots, b_{|B|} - a_1\}$ since a_1 must be aligned to one of $b_1, \dots, b_{|B|}$

$\left(A \Uparrow B = \bigcap_{a \in A} (B - a)\right)$

In particular, $|A \Uparrow B| \leq |B|$

Subset Shifts

Given A, B, determine all x such that $A + x \subseteq B$

[Cole,Hariharan'02]

Subset Shifts:

Given $A, B \subseteq \{0, \dots, d\}$

Compute all x with $A + x \subseteq B$:

$A \Uparrow B := \{x \mid A + x \subseteq B\}$

This is in time $\tilde{O}(|A| + |B|)$ w.h.p.

Direct Use of IP-Convolution: $\tilde{O}(d)$

1) Construct u with $u_i = [i \in A]$
2) Construct v with $v_i = [i \in B]$
3) $w := v *_{IP} u$
4) Return $\{x \mid w_x = |A|\}$

Correctness: $w_x = (v *_{IP} u)_x = \sum_{i=0}^{d} v_{x+i} u_i$

$= \sum_{i=0}^{d} [x + i \in B] \cdot [i \in A] \le |A|$

$w_x = |A| \Leftrightarrow \forall a \in A: a + x \in B$

Convolution counts for each shift the number of matches

$A \Uparrow B = \{x \mid A + x \subseteq B\} \subseteq B - a_1, |A| \le |B|$

Subset Shifts

Given A, B, determine all x such that $A + x \subseteq B$

[Cole,Hariharan'02]

Subset Shifts:

Given $A, B \subseteq \{0, \dots, d\}$

Compute all x with $A + x \subseteq B$:

$$A \Uparrow B := \{x \mid A + x \subseteq B\}$$

This is in time $\underline{\tilde{O}(|A| + |B|)}$ w.h.p.

Direct Use of IP-Convolution: $\tilde{O}(d)$

1) Construct u with $u_i = [i \in A]$
2) Construct v with $v_i = [i \in B]$
3) $w := \underline{v *_{IP} u}$ $\tilde{O}(d) \to \tilde{O}(out)$
4) Return $\{x \mid w_x = |A|\}$

Outputsensitive IP-Convolution takes time

$\tilde{O}(out) = \tilde{O}(\underline{|A + B|}) \gg |B|$ *in general!*

$$A \Uparrow B = \{x \mid A + x \subseteq B\} \subseteq B - a_1,\ |A| \le |B|$$

Subset Shifts

Subset Shifts:

Given $A, B \subseteq \{0, \dots, d\}$

Compute all x with $A + x \subseteq B$:

$$A \Uparrow B := \{x \mid A + x \subseteq B\}$$

This is in time $\tilde{O}(|A| + |B|)$ w.h.p.

Outputsensitive: $\tilde{O}(|B|)$

1) Pick set P of $\log d$ random primes in the range $\Theta(|B| \log^2 d)$

2) For each $p \in P$:

3) $\Delta_p :=$ the set of all $x \in \{0, \dots, p-1\}$ with $(A + x) \bmod p \subseteq (B \bmod p)$

4) For each $b \in B$: $x := b - a_1$

5) If $x \bmod p \in \Delta_p$ for all $p \in P$: print x

$A \Uparrow B = \{x \mid A + x \subseteq B\} \subseteq B - a_1$, $|A| \leq |B|$

Subset Shifts

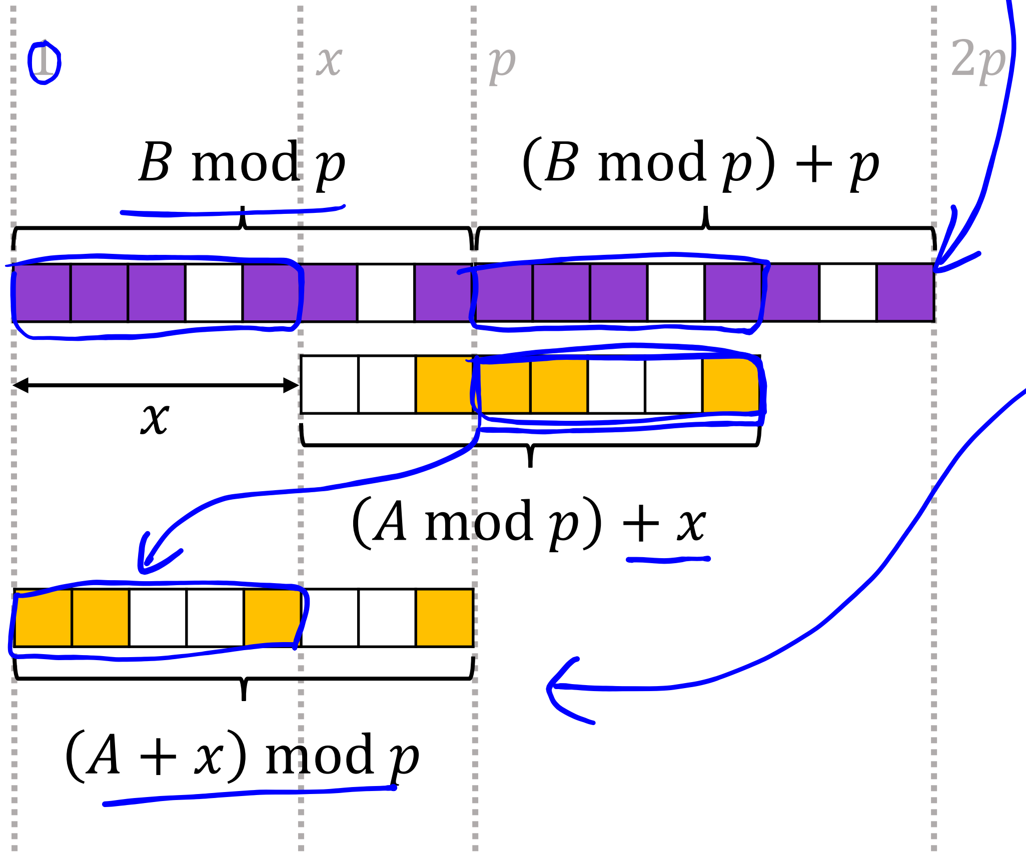
$|A| \le |B| \le p$

Lem: Δ_p is equal to

$$\{0,\dots,p-1\} \cap \left((A \bmod p) \Uparrow ((B \bmod p) + \{0,p\})\right)$$

and thus Δ_p can be computed in time $\tilde{O}(p)$

Proof:



Outputsensitive: $\tilde{O}(|B|)$

1) Pick set P of $\log d$ random primes in the range $\Theta(|B| \log^2 d)$
2) For each $p \in P$:
3) $\Delta_p :=$ the set of all $x \in \{0,\dots,p-1\}$ with $(A+x) \bmod p \subseteq (B \bmod p)$
4) For each $b \in B$: $x := b - a_1$
5) If $x \bmod p \in \Delta_p$ for all $p \in P$: print x

$A \Uparrow B = \{x \mid A + x \subseteq B\} \subseteq B - a_1,\ |A| \le |B|$

Subset Shifts

Correctness: „$A + x \subseteq B \Longrightarrow$ print x“

If $A + x \subseteq B$ then $(A + x) \bmod p \subseteq (B \bmod p)$

So $(x \bmod p) \in \Delta_p$ for all $p \in P$

So we print x

Outputsensitive: $\tilde{O}(|B|)$

1) Pick set P of $\log d$ random primes in the range $\Theta(|B| \log^2 d)$

2) For each $p \in P$:

3) Δ_p := the set of all $x \in \{0, \dots, p-1\}$ with $(A + x) \bmod p \subseteq (B \bmod p)$

4) For each $b \in B$: $x := b - a_1$

5) If $x \bmod p \in \Delta_p$ for all $p \in P$: print x

$A \Uparrow B = \{x \mid A + x \subseteq B\} \subseteq B - a_1, |A| \leq |B|$

Subset Shifts

Correctness: „$A + x \not\subseteq B \implies$ w.h.p. don't print x"

If $A + x \not\subseteq B$ then $\underline{a + x \notin B}$ for some $a \in A$

$\mathbb{P}[\underline{x \bmod p \in \Delta_p}] \le \mathbb{P}[(a+x) \bmod p \in (B \bmod p)]$ $=$

$\le |B| \cdot \mathbb{P}[(a+x) \bmod p = b \bmod p]$

$\le |B| \cdot \mathbb{P}[(\underbrace{a + x - b}_{\neq 0}) \bmod p = 0]$ $\leftarrow p \mid (a+x-b)$

$\le |B| \cdot \dfrac{\log d}{\#\text{primes in the range } \Theta(|B| \log^2 d)} \le \boxed{\dfrac{1}{2}}$

$\uparrow \Omega(|B| \log d)$

Thus, $\mathbb{P}[\underline{x \bmod p \in \Delta_p} \text{ for all } p \in P] \le \boxed{\frac{1}{d}}$

Outputsensitive: $\tilde{O}(|B|)$

1) Pick set P of $O(\log d)$ random primes in the range $\Theta(|B| \log^2 d)$

2) For each $p \in P$:

3) $\Delta_p :=$ the set of all $x \in \{0, \dots, p-1\}$ with $\underline{(A + x) \bmod p} \subseteq \underline{(B \bmod p)}$

$(a+x) \bmod p \in$

4) For each $b \in B$: $x := b - a_1$

5) If $x \bmod p \in \Delta_p$ for all $p \in P$: $\underline{\text{print } x}$

$A \Uparrow B = \{x \mid A + x \subseteq B\} \subseteq B - a_1$, $|A| \le |B|$

Subset Shifts

Similarity to previous lectures:

$$\mathbb{1}[B \bmod p]_i = \overset{\Sigma}{\bigvee_{j:j \bmod p = i}} \mathbb{1}[B]_{i\,j}$$

$\mathbb{1}[B \bmod p]$ is a Boolean version of $\mathrm{fold}(\mathbb{1}[B], p)$

Outputsensitive: $\tilde{O}(|B|)$

1) Pick set P of $\log d$ random primes in the range $\Theta(|B| \log^2 d)$

2) For each $p \in P$:

    3) $\Delta_p :=$ the set of all $x \in \{0, \dots, p-1\}$ with $(A + x) \bmod p \subseteq \underline{(B \bmod p)}$

4) For each $b \in B$: $x := b - a_1$

    5) If $x \bmod p \in \Delta_p$ for all $p \in P$: print x

$A \Uparrow B = \{x \mid A + x \subseteq B\} \subseteq B - a_1, |A| \leq |B|$

Subset Shifts

Subset Shifts:

Given $A, B \subseteq \{0, \dots, d\}$

Compute all x with $A + x \subseteq B$:

$$A \Uparrow B := \boxed{\{x \mid A + x \subseteq B\}}$$

This is in time $\underline{\tilde{O}(|A| + |B|)}$ w.h.p.

Outputsensitive: $\underline{\tilde{O}(|B|)}$

1) Pick set P of $\log d$ random primes in the range $\underline{\Theta(|B| \log^2 d)}$

2) For each $p \in P$:

   3) $\Delta_p :=$ the set of all $x \in \{0, \dots, p-1\}$ with $(A + x) \bmod p \subseteq (B \bmod p)$

4) For each $b \in B$: $x := b - a_1$

   5) If $x \bmod p \in \Delta_p$ for all $p \in P$: print x

$A \Uparrow B = \{x \mid A + x \subseteq B\} \subseteq B - a_1,\ |A| \leq |B|$

More Material

Lecture based on:

[Cole, Hariharan „Verifying Candidate Matches in Sparse and Wildcard Matching“ 2002]

[Abrahamson „Generalized string matching“ 1987]

See also:

[Chan, Golan, Kociumaka, Kopelovitz, Porat „Approximating text-to-pattern Hamming distances“ 2020]

Course Overview

NP: $O(2^n)$ $O(2^{\sqrt{n}})$ $O(n^n)$

P: $O(n^{100})$ $O(n^2)$ $O(n \log n)$ $O(n)$

Space $\boldsymbol{o(n)}$?

#Measurements $\boldsymbol{o(n)}$?

Time $\boldsymbol{o(n)}$?

Course Overview

NP: $O(2^n)$ $O(2^{\sqrt{n}})$ $O(n^n)$

P: $O(n^{100})$ $O(n^2)$ $O(n \log n)$ $O(n)$

Space $o(n)$?

#Measurements $o(n)$?

Time $o(n)$?

Streaming Algorithms:

Data stream $x_1, x_2, \dots, x_n$

Make one pass over the stream

Working memory $o(n) / O(\log n)$

≈ low-space data structures

©Stefan Funke / Wikipedia

Typical problems:

- Compute number of distinct x_i's
- Compute all numbers that appear $\geq \varepsilon n$ times
- Maintain a vector and its frequency moments

Course Overview

NP: $O(2^n)$ $O(2^{\sqrt{n}})$ $O(n^n)$

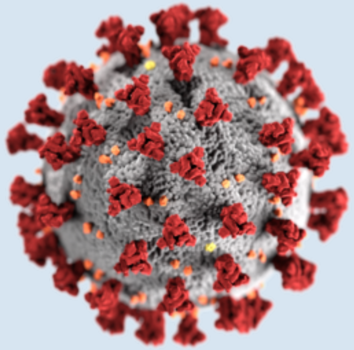
P: $O(n^{100})$ $O(n^2)$ $O(n \log n)$ $O(n)$

Space $o(n)$?

#Measurements $o(n)$?

Time $o(n)$?

Randomized Trials:



Estimate the infected population
by testing random individuals

Combinatorial Group Testing:

Mix samples of a group of individuals → test tells us whether at least one individual is positive

Find *all* positive individuals using $o(n)$ group tests

Medical Imaging:

Reconstruct a sparse vector
from few Fourier measurements

Course Overview

NP: $O(2^n)$ $O(2^{\sqrt{n}})$ $O(n^n)$

P: $O(n^{100})$ $O(n^2)$ $O(n \log n)$ $O(n)$

Space $\boldsymbol{o(n)}$?

#Measurements $\boldsymbol{o(n)}$?

Time $\boldsymbol{o(n)}$?

Property Testing:

Really sublinear time $o(n)$!

"What can we find out about $x_1, x_2, \dots, x_n$

using $o(n)$ random accesses?"

Typical problems:

Is $x_1, x_2, \dots, x_n$ sorted or *far* from sorted?

Is a graph connected or *far* from connected?

Course Overview

NP: $O(2^n)$ $O(2^{\sqrt{n}})$ $O(n^n)$

P: $O(n^{100})$ $O(n^2)$ $O(n \log n)$ $O(n)$

Space $\boldsymbol{o(n)}$?

#Measurements $\boldsymbol{o(n)}$?

Time $\boldsymbol{o(n)}$?

Course Outline:

3x Streaming (Space)

3x Vector Reconstruction (Measurements)

2x Property Testing (Time)

3x Applications

Exam

Oral exam on July 20 via Zoom

If you do not know your exam slot yet, contact me!

You must be registered in LSF

All course material is relevant

Questions of two types:

- explain algorithm X from the lecture
- simple problems in the spirit of the exercise sheets

Have some ID ready (e.g. student ID card)

You are allowed a blank sheet of paper + pens, and no other materials

Be prepared to show us the room that you are in

The exam is not recorded

Thanks!