Lower Bounds

T.S. Jayram (IBM Almaden)

MADALGO Summer School

Lecture II

Outline

- L_∞
- Probabilistic Streams
- Read/Write Streams

L_∞ distances

For $x, y \in U^n$,

$$\|x-y\|_\infty = \max_i |x_i - y_i|$$

Sketching protocol for L_∞:

$$\|x-y\|_\infty \le \|x-y\|_2 \le \sqrt{n}\,\|x-y\|_\infty$$

- Divide input into blocks of size $n^{4\varepsilon}$
- Use AMS sketch for each block

→ $n^{1-4\varepsilon}$ space

→ n^{ε} factor approximation

L_∞ Promise Problem ($GapL_\infty$)

YES : $\|x-y\|_4 \geq m, \quad m \geq 2$

NO: $\|x-y\|_4 \leq 1$

Theorem. [Saks, Sun] [Bar-Yossef, Jayram, Kumar, Sivakumar]

The C.C. of $GapL_\infty$ is fi $\frac{q}{p^{5}}$,

Direct-Sum methodology

I. Define the small problem

DIST(u,v):

YES: $|u - v| \geq m$

NO: $|u - v| \leq 1$

Note:

$\mathrm{GapL}_\infty(x,y) = \vee_i \mathrm{DIST}(x_i,y_i)$

Direct-Sum methodology

II. Define an appropriate conditionally independent input distribution

Distribution κ for DIST:

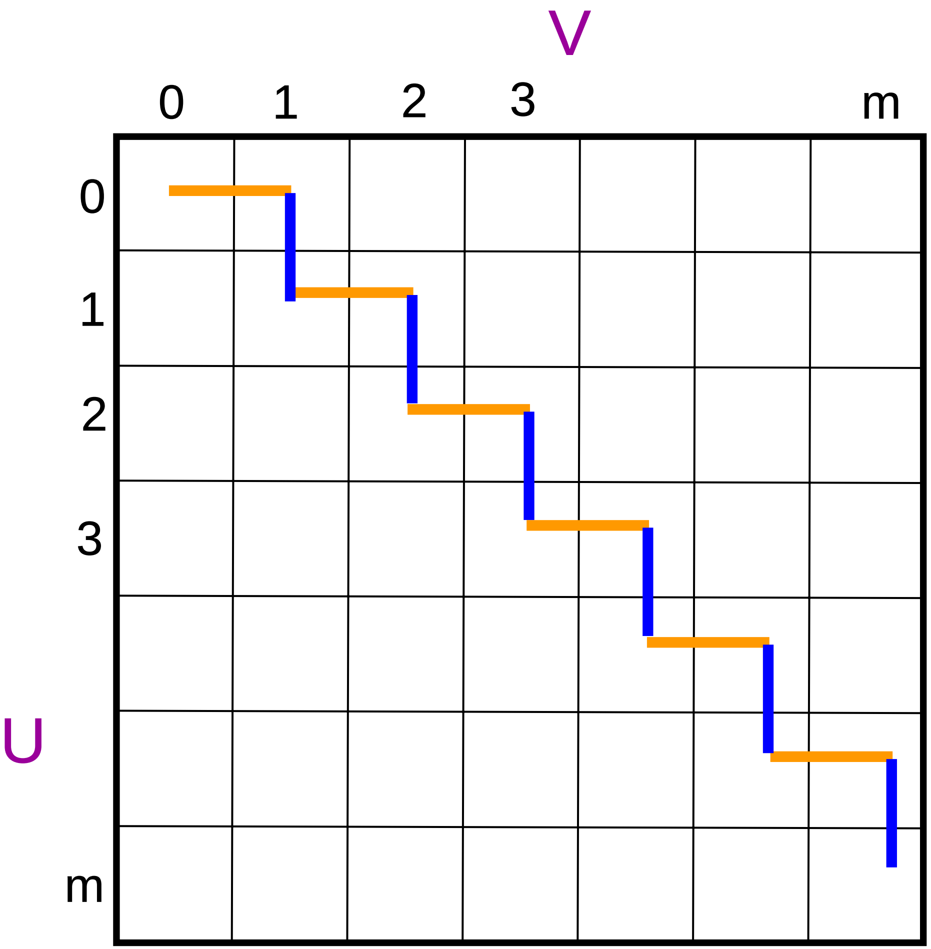
- $(U,V,D,S) \sim \kappa$
- $U \perp V \sim D,S$
- (U,V) is a NO instance
  - $|U - V| \leq 1$

Direct-Sum methodology

$D \in_R \{ -, | \}$
$S \in_R \{0,\ldots,m-1\}$

If $D = -$, then
$U = S$, $V \in_R \{S, S+1\}$

If $D = |$, then
$U \in_R \{S, S+1\}$, $V = S+1$



Direct-Sum methodology

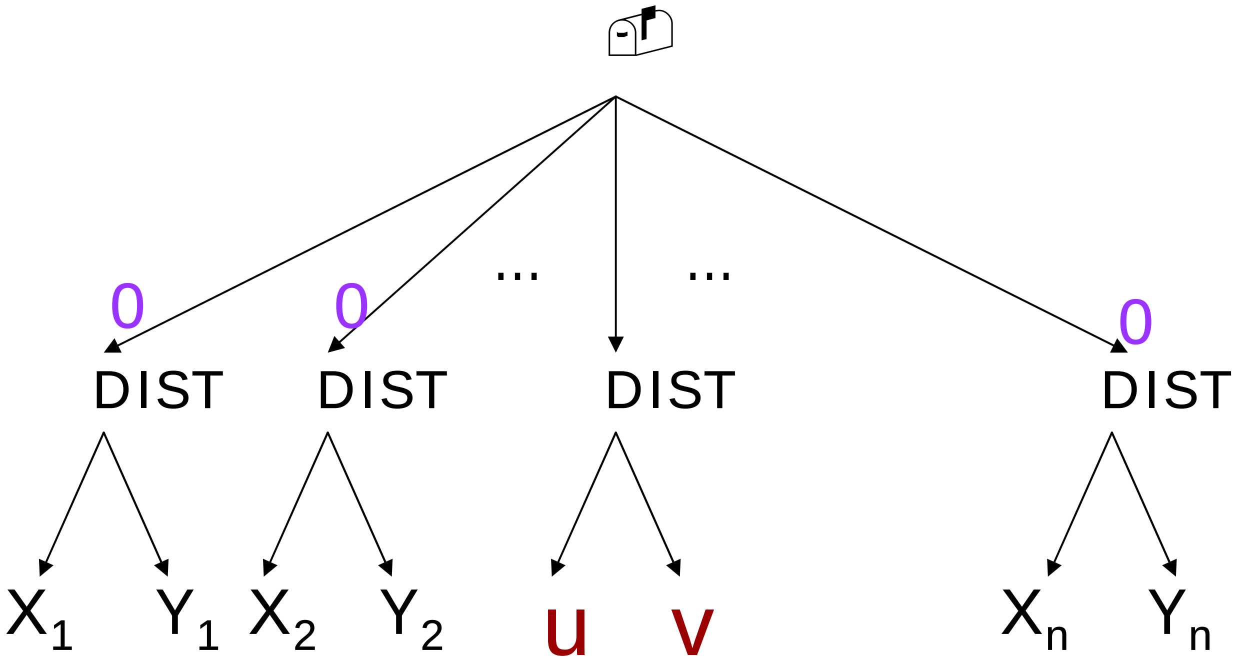
- Distribution μ for instances of GapL_∞ is defined to be n independent copies of κ

- This always produces NO instances of GapL_∞

Direct-Sum methodology

III. Apply Direct-Sum Theorem

$$\mathrm{IC}_\mu(\mathrm{GapL}_\infty) \geq \mathrm{n} \cdot \mathrm{IC}_\kappa(\mathrm{DIST})$$



μ produces NO instances for GapL_∞

Conditional independence

Direct-Sum methodology

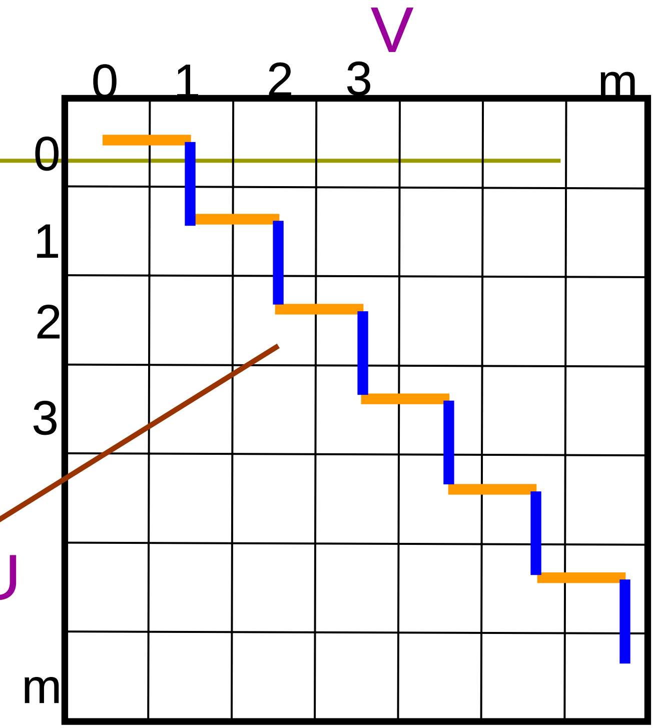
IV. Show that the information complexity of DIST is $\Omega(1/m^2)$

Let P be a protocol for DIST

Define $(U,V,D,S) \sim \kappa$

$U \perp V \mid D,S$

Information Cost of P



$I(U,V : P \mid D, S)$

$= (1/2m) \cdot$
$\quad \sum_s I(U,V : P \mid D = -, S = s)$
$\quad + I(U,V : P \mid D = |, S = s)$

$= (1/2m) \cdot$
$\quad \sum_s I(V : P(s,V) \mid D = -, S = s)$
$\quad + I(U : P(U,s+1) \mid D = |, S = s)$

Binary Valued

Hellinger Distance

- Z is a binary r.v.
- $Q(Z,R)$ is another r.v
- $Z \perp R$

- $Q(0,R) \sim Q_0$
- $Q(1,R) \sim Q_1$
- Then, $I(Z : Q(Z,R)) \geq h^2(Q_0,Q_1)$

Information Cost of P

$I(U,V : P(U,V) \mid D,S)$

$= (1/2m) \cdot \sum_s I(V : P(s,V) \mid D = -, S = s)$
$\quad + I(U : P(U,s+1) \mid D = |, S = s)$

$\geq (1/2m) \cdot \sum_s h^2(P_{s,s}, P_{s,s+1}) + h^2(P_{s,s+1}, P_{s+1,s+1})$

$\geq (1/2m^2) \cdot h^2(P_{00}, P_{mm})$

[Cauchy-Schwartz and Metric]

After some elementary calculations…

DKK. 5,492,050 question:

Why should P_{00} and P_{mm} be far apart?

☹ (0,0) and (m,m) are both NO instances

Z-Lemma

P is deterministic:

$P(x,y) = t = P(u,v)$

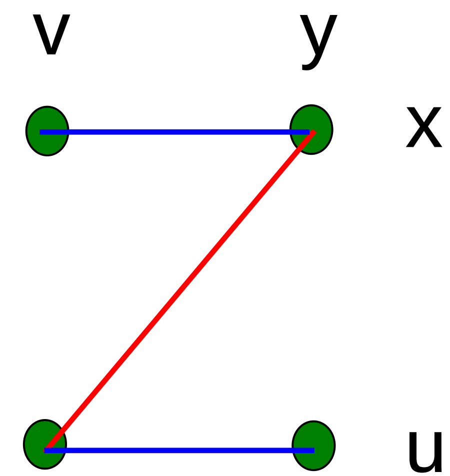
➔ $P(x,v) = t = P(y,v)$

P is randomized:

$h^2(P_{xy}, P_{uv})$ ❍

$\frac{1}{4}\,(h^2(P_{xy}, P_{xv}) + h^2(P_{uy}, P_{uv}))$

(Hint: Use AM-GM)



Finishing it up

By **Z**-Lemma,

$h^2(P(x,y), P(u,v))$ ❍

¼ $(h^2(P(0,0), P(0,m))$

$+ h^2(P(m,0), P(m,m))$

The latter two quantities are both $\Omega(1)$

$I(U,V : P(U,V) \mid D,S)$

$\geq (1/2m^2) \cdot h^2(P_{00}, P_{mm})$

$= \Omega(1/m^2)$

Other applications

[Jayram,Kumar,Sivakumar]

Rand. C.C. LB for “AND-OR tree” function

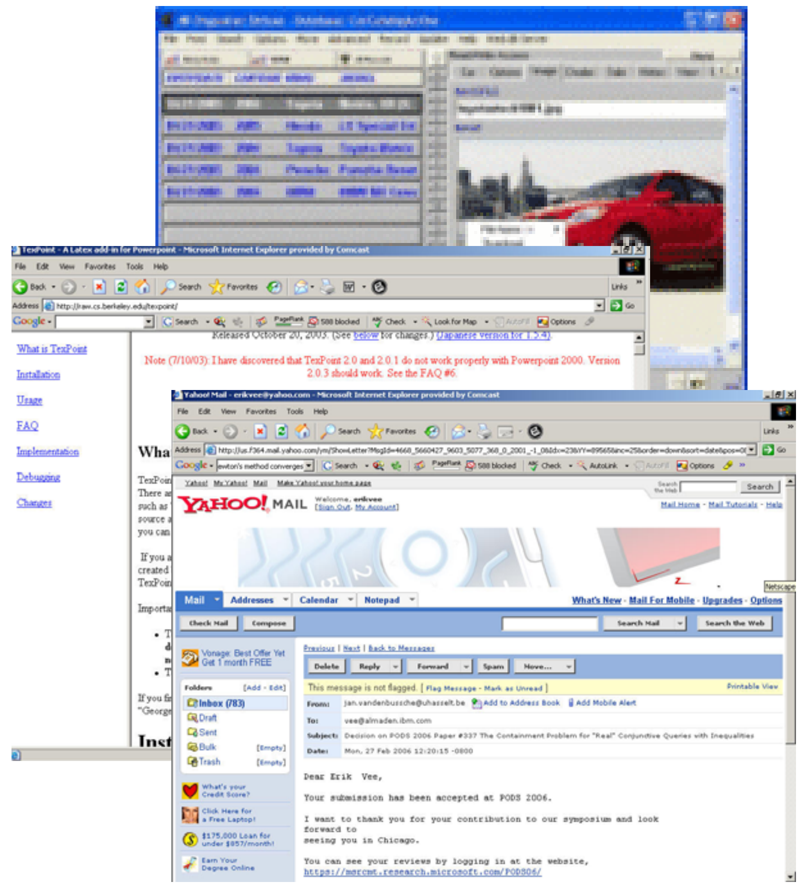
[Jain, Radhakrishnan, Sen]

Extended direct-sum paradigm to quantum C.C and obtained l.b.’s for set-disjointness

Estimating Statistical Aggregates on Probabilistic Data Streams

Motivation: Probabilistic Data

- Data that is incomplete, imprecise, error-prone
- Probabilistic data is everywhere
  - Automated data extraction:
    - Emails, Web page, Blogs
  - Recommendation systems
    - Fuzzy, incomplete ratings
  - Inherently noisy
    - Sensor data
  - Data cleansing



Probabilistic Databases

- Avatar Semantic Search (IBM)
- HeisenData (Intel/Berkeley)
- MystiQ (U Washington)
- Orion (Purdue)
- Trio (Stanford)

- Others…

How do we calculate aggregates?

- Given database of probabilistic data, can we compute simple aggregates efficiently?

- SUM, COUNT, MIN, MAX
- MEDIAN
- AVG
- DISTINCT (F_0)
- REPEAT-RATE (F_2)

Probabilistic stream

Input: $(a_1, p_1), (a_2, p_2), \ldots (a_n, p_n)$

- Means w.p. p_i, the i^{th} item in stream has value a_i
  - Otherwise, not in stream
- Generally, want expected value of aggregate

Notation

Input: $(a_1,p_1), (a_2,p_2), \ldots, (a_n,p_n)$

Define X,Y

- with probability p_i
  - $Y_i = 1$ and $X_i = a_i$
- with probability $1-p_i$
  - $Y_i = 0$ and $X_i = 0$
- X_i and Y_i are correlated

SUM and COUNT

$\text{COUNT} = E[\sum_i X_i] = \sum_i p_i$

$\text{SUM} = E[\sum_i X_i] = \sum_i a_i \, p_i$

Linearity of Expectation

- Easy to compute in one pass

AVERAGE

$AVG = E[\sum_i X_i / \sum_i Y_i]$

Linearity of expectation fails!

- It can be shown that $\Omega(n)$ space is needed to compute AVG exactly

Approximating AVG

- Easy approximation

  $$\text{AVG} \approx \text{SUM}/\text{COUNT}$$

  - Works well when COUNT is large (use Chernoff's bound)
  - What about small COUNT?

Generating Functions

- Suppose we want to compute

  H ˝ S4 / 4. ı\ı ¨

- Define the generating function

  j+{, @ H ˝" S4 / 4. ı\ı j{ S 4. ı\ı ¨

- Required answer g(1)

Calculus

Take the derivative of g(x)

j 3+{, @ H 7+4 . [\1, f 4 S / 4 . 1\1 f{ S 1\18

@ H k S { 1\1 1 1 5 6

@ H 7 \ {\18

@ \ H 1 k { \1 1

@ \1 +s1{ . 4 ¥ s1, 1

Implication

- $g'(x) = \prod_i (p_i x + 1 - p_i)$
  - for any x, $g'(x)$ can be computed in one pass
- Further, we know

H ˝ 4 S / 4.]\] ... @ j+4, @ U 4 3 j 3 +{,g{

- Need to calculate an integral over a data stream!

The integral for AVG

```
        ]  4   C \                    [   d1s1
DYJ @             +4 ¥ s1},D C            ------- D g}
        3        1                 1    4 ¥ s1}

            ┘  4
DYJ @           j+},k+},g}
               3
```

- Taylor approximation for h(z) is fine
- Taylor approximation for g(z) is horrible
- BUT approximation for log g(z) can work!

Read/Write Streams

What happens if…

- There are multiple streams ?
- The algorithms can modify the streams ?

- Modern memory organization
  - Main memory: fast and scarce
  - Disk memory : slow and abu
    - Random access is very costly
    - Sequential access is cheap
      - E.g., Cache prefetching
- Disk Memory is Read/Write

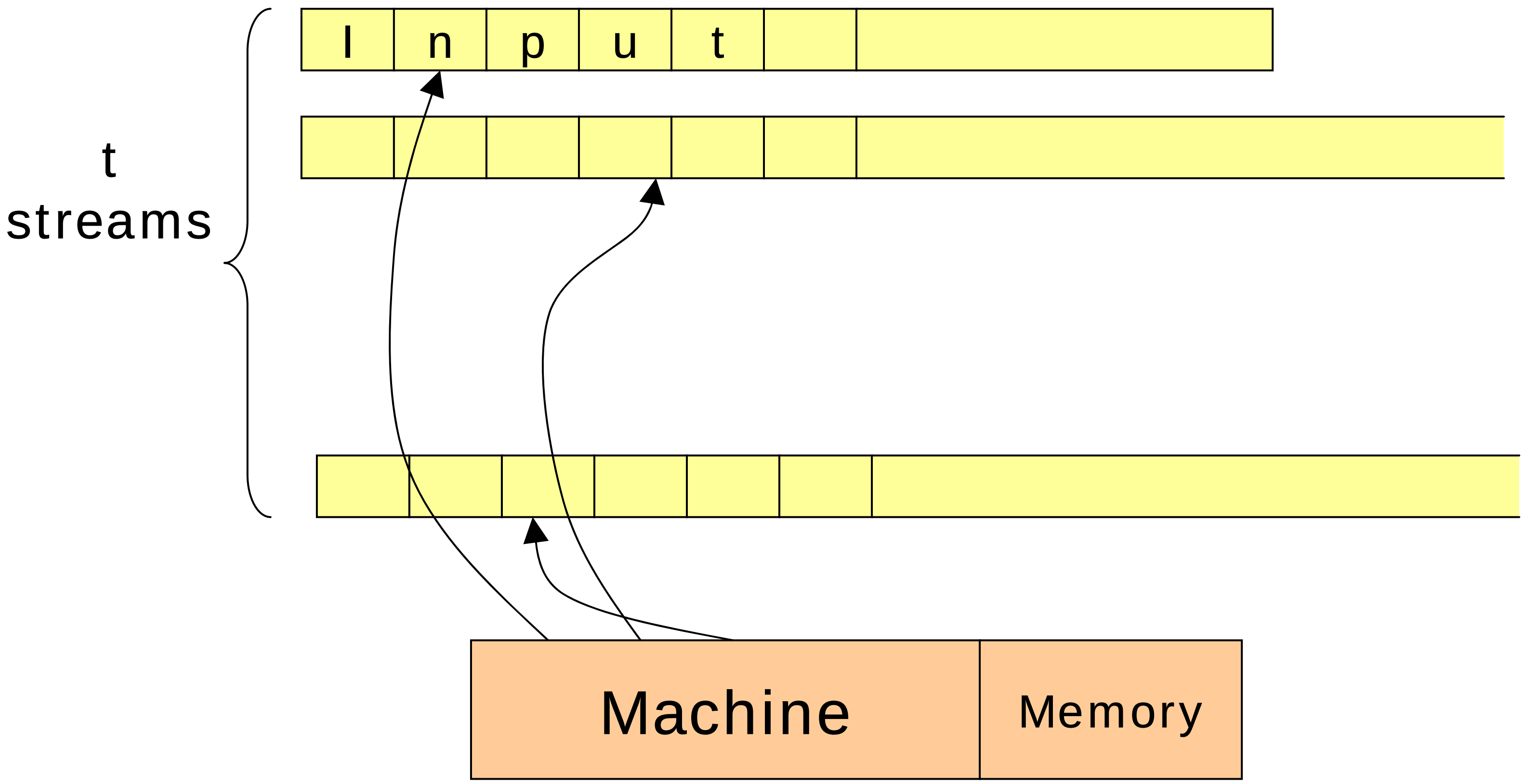
Not true for data streams

Beyond Data Streams

- Efficient access to external memory is possible in restricted ways
  - I/O rates for sequential read/write access to disks are as good as random access to main memory

- New models of I/O-efficient computing
  - W-streams [Demetrescu, Finocchi, Ribichini]
  - Read/write streams [Grohe,Schweikardt; Grohe,Hernich, Schweikardt]
  - Stream-Sort [Aggarwal,Datar,Rajagopalan,Ruhl]
  - Map-reduce [Dean,Ghemawat]

Read/Write Streams



- Also called Reversal Turing Machines

Critical Resources

- #tapes t
- space s
- No constraint on the length of streams
- But #reversals is at most r

➔ (r,s,t) read/write stream algorithm

Read/write Streams are Powerful

- Sorting can be done with $O(\log N)$ passes (variant
  - space is $O(1)$
  - 3 Read/Write streams
  - After sorting, computing frequency moments takes 1 pass and $O(\log N)$ space

Data stream algorithms require $N^{\Omega(1)}$ passes to approximate F_k for $k>2$

But…

- $\Theta(\log N)$ passes is a lot!

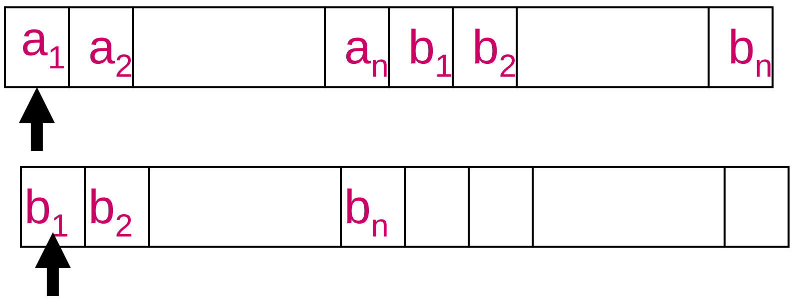
- What if the Read/Write stream algorithm has $o(\log N)$ passes?

Read/write Stream Lower Bounds

- $o(\log N)$ passes implies $N^{\Omega(1)}$ space for deterministic algorithms:
  - Sorting [Grohe-Schweikardt]
- $o(\log N)$ passes implies $N^{\Omega(1)}$ space for one-sided error algorithms:
  - Set Equality [Grohe-Hernich-Schweikardt]
- $o(\log N/\log\log N)$ passes implies near $\Omega(N)$ space for two-sided error algorithms:
  - Set Disjointness [Beame-Jayram-Rudra]

Set Disjointness

- Given sets A and B as characteristic vectors
  - Is $A \cap B = \emptyset$?
  - $A, B \in \{0,1\}^N$
- Communication complexity = $\Omega(N)$
  - yields $\Omega(N)$ space LBs for data streams
- Easy for a Read/Write Stream algorithm



Key Idea

- Keep the first vector $(a_1,\ldots,a_n)$ and permute the second vector $(b_1,..,b_n)$
- Any Read/Write Stream algorithm with few passes cannot "compare" many pairs (a_i,b_i)
  - Can "mix and match" values in this pair
- Develop a new combinatorial structural property to formalize this intuition
  - [Grohe,Schweikardt; Grohe,Hernich, Schweikardt]

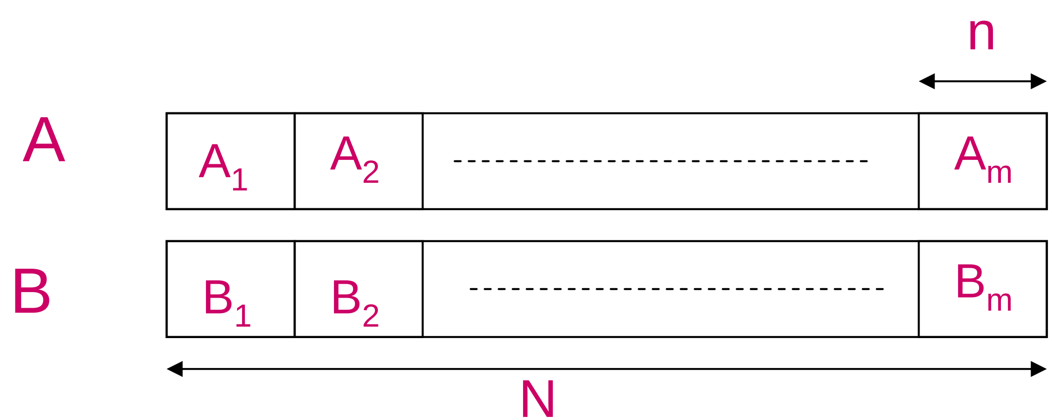
Hard Instances

- $\mathrm{Disj}(A,B) = 0$ iff $A \cap B = \emptyset$
  - $A,B \in \{0,1\}^N$
- $\mathrm{Disj}(A,B) = \bigvee_i \mathrm{Disj}(A_i,B_i)$
  - $N = nm$
- Reorder pairs of blocks to be compared by permutation ϕ on $\{1,\ldots,m\}$

ϕ has low "sortedness"

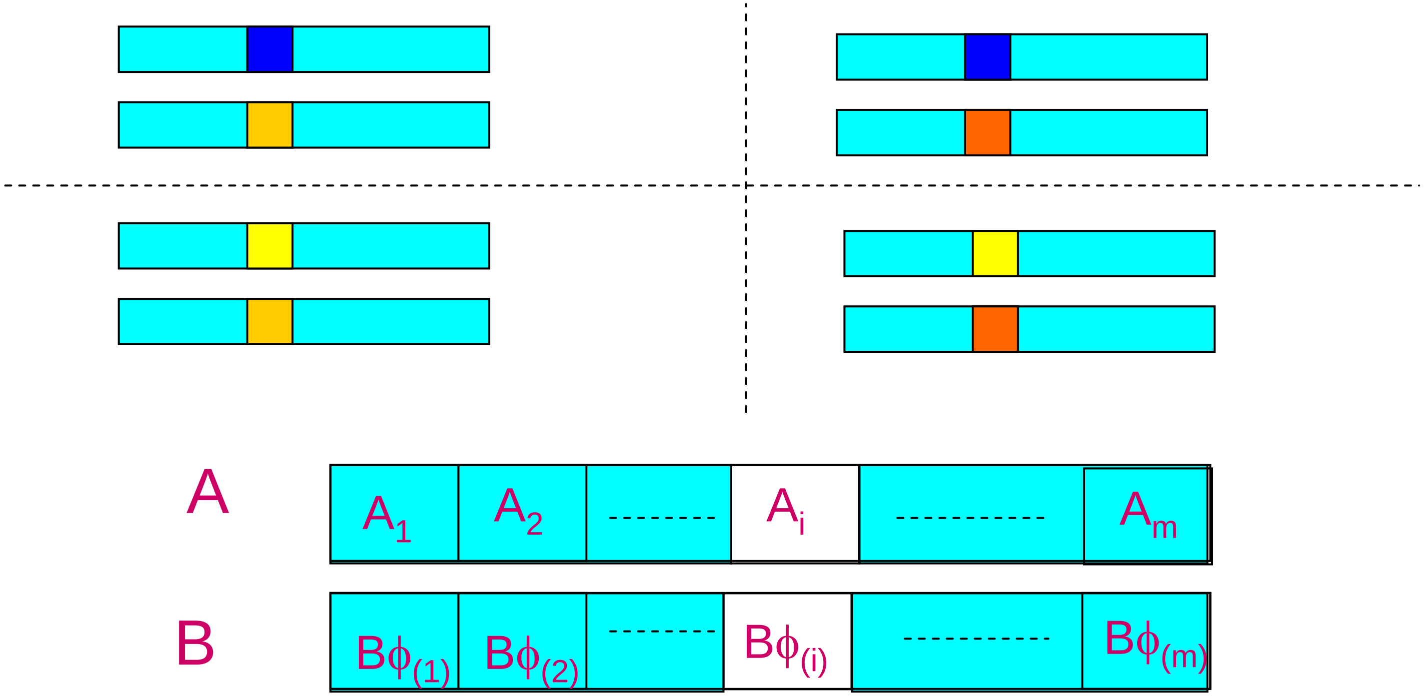
Sortedness: longest monotone subsequence

$\mathrm{Disj}^{\phi}(A,B) = \bigvee_i \mathrm{Disj}(A_i,B_{\phi(i)})$



Skeletons

- Describes the information flow in terms of the locations of elements that are compared



Formally,

Given a skeleton

- There exists many indices $i \in \{1,\ldots,m\}$
- For every assignment to $(A_j, B_{\phi(j)})$, $j \neq i$
- Inputs of the skeleton projected to $(i, \phi(i))$ is a rectangle
- The rectangles do not form a partition of the inputs of the skeleton

Fundamental Theorem of R/W Streams

Theorem.

The skeletons partition the input domain such that

(1) #skeletons is “small”

(2) output depends only on the skeleton

(3) Each skeleton satisfies a weak rectangle-like property

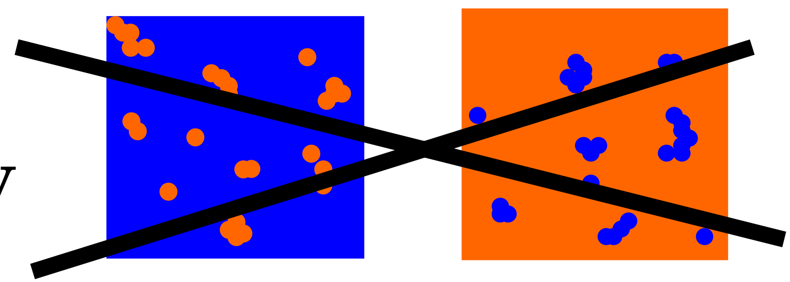
Direct-Sum

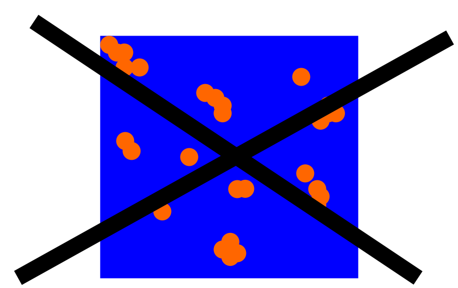
- Given $f: \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$
- Permutation ϕ on $\{1,\ldots,m\}$ with low sortedness
  - $XOR_{f,\phi}(A_1,\ldots,A_m,B_1,\ldots,B_m) = \bigoplus f(A_i,B_{\phi(i)})$
  - $OR_{f,\phi}(A_1,\ldots,A_m,B_1,\ldots,B_m) = \bigvee f(A_i,B_{\phi(i)})$

Hardness measures for functions

- $f:\{0,1\}^n \times \{0,1\}^n \to \{0,1\}$
- Hardness measure for 2-sided lower bounds
  - Defined on rectangles
  - f has low discrepancy
  - f has low corruption
    - Set disjointness

0s and 1s





Results

- f has low discrepancy or corruption
  - $o(\log mn)$ passes implies large space for $XOR_{f,\,\phi}$
- If f has low corruption
  - $o(\log(mn)/\log\log(mn))$ passes implies large space for $OR_{f,\,\phi}$

Remarks

- Currently, our direct-sum framework works for primitive functions that have high discrepancy or corruption
  - Open problem: derive an information complexity based approach
  - Application: frequency moments
- We consider two kinds of composition operators: $\oplus$ and $\vee$
- Yields lower bounds for Intersection Size Mod 2 (Inner Product)