Communication Complexity and Applications

Swagato Sanyal

IIT Kharagpur

Indo-Slovenia Pre-Conference School on Algorithms and Combinatorics

February 12-13 2024

Communication Complexity

- Two-party communication model [Yao,89].
- More restrictive (one-way) and more general (multi-party) models.
- General “lower-bound technique” in algorithms and complexity.
- Applications:
  - Streaming algorithms
  - Data structures
  - Boolean formula size and depth
  - VLSI chip design
  - …

One-way communication model

$g(m, y)$

$$f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$$

$m(x)$

$x \in \mathcal{X}$

$y \in \mathcal{Y}$

$$\Pi = (m, g) \text{ computes } f : \forall x \in \mathcal{X}, y \in \mathcal{Y}, g(m(x), y) = f(x, y)$$

$$\mathsf{CC}^{\rightarrow}(\Pi) := \max_{x \in \mathcal{X}} |m(x)|$$

$$\mathsf{CC}^{\rightarrow}(f) := \min_{\Pi \text{ computing } f} \mathsf{CC}^{\rightarrow}(\Pi)$$

[Image credit: Internet]

One-way randomized (public coin)

$g(m, y, \mathcal{R})$

$$f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$$

Public randomness $\mathcal{R}$

$m(x, \mathcal{R})$

$x \in \mathcal{X}$

$y \in \mathcal{Y}$

$$\Pi = (m, g) \text{ computes } f : \forall x \in \mathcal{X}, y \in \mathcal{Y}, \Pr_{\mathcal{R}}[g(m(x, \mathcal{R}), y, \mathcal{R}) = f(x, y)] \geq \frac{2}{3}$$

$$\mathsf{RCC}^{\to}(\Pi) := \max_{x \in \mathcal{X}, \mathcal{R}} |m(x, \mathcal{R})|$$

$$\mathsf{RCC}^{\to}(f) := \min_{\Pi \text{ computing } f} \mathsf{RCC}^{\to}(\Pi)$$

[Image credit: Internet]

Examples

Equality

- $\mathsf{RCC}^{\rightarrow}(f) \leq \mathsf{CC}^{\rightarrow}(f) \leq \lceil \log_2 |\mathcal{X}| \rceil$
- $\mathsf{EQ} : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}$:

$$\mathsf{EQ}(x,y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases}$$

- $\mathsf{CC}^{\rightarrow}(\mathsf{EQ}) = n$ (pigeon-hole principle)
- $\mathsf{RCC}^{\rightarrow}(\mathsf{EQ}) = O(1)$

Disjointness

- $\mathsf{DISJ} : 2^{[n]} \times 2^{[n]} \to \{0,1\}$:

$$\mathsf{DISJ}(S,T) = \begin{cases} 1 & \text{if } S \cap T = \emptyset, \\ 0 & \text{otherwise} \end{cases}$$

- $\mathsf{CC}^{\rightarrow}(\mathsf{DISJ}) = n$ (pigeon-hole principle)

- $\mathsf{RCC}^{\rightarrow}(\mathsf{DISJ}) = \Omega(n)$

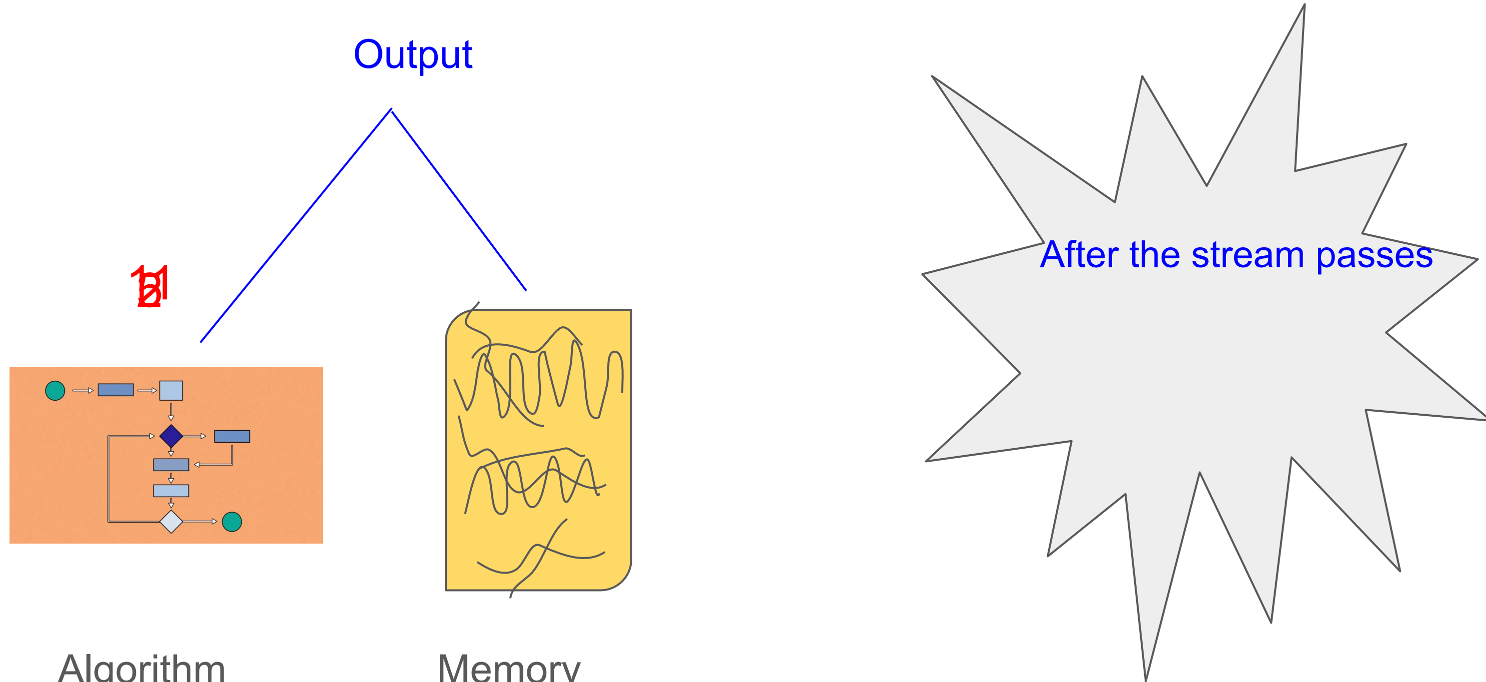
The streaming model: estimating frequency moments

- Universe $\mathcal{U} = \{1, \ldots, n\}$.
- Stream $s : a_1, \ldots, a_m$. Each $a_i \in \mathcal{U}$.
- $\forall i \in [n], f_i := |\{j \in [m] \mid a_j = i\}|$.
- Algorithm $\mathcal{A}$ with bounded memory.
- $\mathcal{A}$ has "one-pass access" to the stream.

- Task is to estimate $F_k := \sum_{i=1}^{n} f_i^k$:

  Output a number in $[0.9F_k, 1.1F_k]$.

The streaming model



Algorithm updates memory

[Image credit: Internet]

The streaming model: estimating frequency moments

- $k = 1 : F_k = \text{m}$. Easy: maintain a counter. $O(\log m)$ space.

  -Deterministic and exact.

- Any k: Maintain frequency vector $(f_1, \ldots, f_n)$. $O(n \log m)$ space.

  -Deterministic and exact.

- [Alon, Matias, Szegedy, 1999] There is a $O(\log n + \log m)$

  space algorithm for $k = 0$ and 2.

  -Randomized and approximate. Gödel Prize 2005!

Hardness of estimating F_∞

- $F_\infty := \max_{i=1}^{n} f_i.$

Theorem [Alon, Matias, Szegedy 1999]. Every randomized streaming algorithm that, for every data stream of length m, outputs a number in the range [0.9 F_∞, 1.1 F_∞] with probability at least ⅔, uses space Ω(min{m, n}).

Proof idea: If there is such a streaming algorithm with space o(min{m, n}), then there is a randomized one-way protocol for disjointness of complexity o(n).

Hardness of estimating F_∞ : continued

Proof.

Let $\mathcal{A}$ be a streaming algorithm that outputs an estimate in $[0.9F_\infty, 1.1F_\infty]$ with probability ⅔, and runs in space s.

We will use $\mathcal{A}$ to construct a protocol for disjointness.

Hardness of estimating F_∞: continued

Protocol:

1. Alice runs $\mathcal{A}$ on the sequence of elements of S.
2. Alice sends the contents of her memory to Bob.
3. Bob continues the run of $\mathcal{A}$ with the communicated memory image on the sequence of elements of T. .
4. If the output of $\mathcal{A}$ is at most 1.5, output “disjoint”. Else, output “intersecting”.

Communication Complexity: s

Hardness of estimating F_∞ : continued

Correctness:

Case 1: The sets are intersecting. Let the sets intersect at $i \in [n]$.

Then, there will be two occurrences of i in the stream. This implies that $F_\infty = 2$ (note that no element has frequency more than 2).

Thus with probability at least ⅔ $\mathcal{A}$ outputs a number that is at least 1.8. Thus, with probability ⅔, the protocol gives correct output.

Hardness of estimating F_∞: continued

Correctness (continued):

Case 2: The sets are disjoint.

Then, each element occurs at most once in the stream. Hence $F_\infty \leq 1$

Thus with probability at least ⅔ $\mathcal{A}$ outputs a number that is at most 1.1. Thus, with probability ⅔, the protocol gives correct output.

Conclusion: Since the randomized communication complexity of disjointness is $\Omega(n)$,we conclude that $s = \Omega(n)$.

Two-way communication model

$$f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$$

$m_1(x)$

$m_2(m_1, y)$

$\vdots$

$x \in \mathcal{X}$

$y \in \mathcal{Y}$

output $m_r(y, m_1, \ldots, m_{r-1})$

$$\Pi = (m_1, \ldots, m_r) \text{ computes } f : \forall x \in \mathcal{X}, y \in \mathcal{Y}, \Pi(x, y) = f(x, y)$$

$$\mathsf{CC}(\Pi) := \max_{x \in \mathcal{X}, y \in \mathcal{Y}} |m_1| + |m_2| + \ldots + |m_r|$$

$$\mathsf{CC}(f) := \min_{\Pi \text{ computing } f} \mathsf{CC}(\Pi)$$

[Image credit: Internet]

Two-way randomized (public coin)

$$f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$$

$\mathcal{R}$: 

$m_1(x, \mathcal{R})$ $\longrightarrow$

$\longleftarrow$ $m_2(m_1, y, \mathcal{R})$

$\vdots$

$\longleftarrow$

$x \in \mathcal{X}$

output $m_r(y, m_1, \ldots, m_{r-1}, \mathcal{R})$

$y \in \mathcal{Y}$

$$\Pi = (m_1, \ldots, m_r) \text{ computes } f : \forall x \in \mathcal{X}, y \in \mathcal{Y}, \Pr_{\mathcal{R}}[\Pi(x, y, \mathcal{R}) = f(x, y)] \geq \frac{2}{3}$$

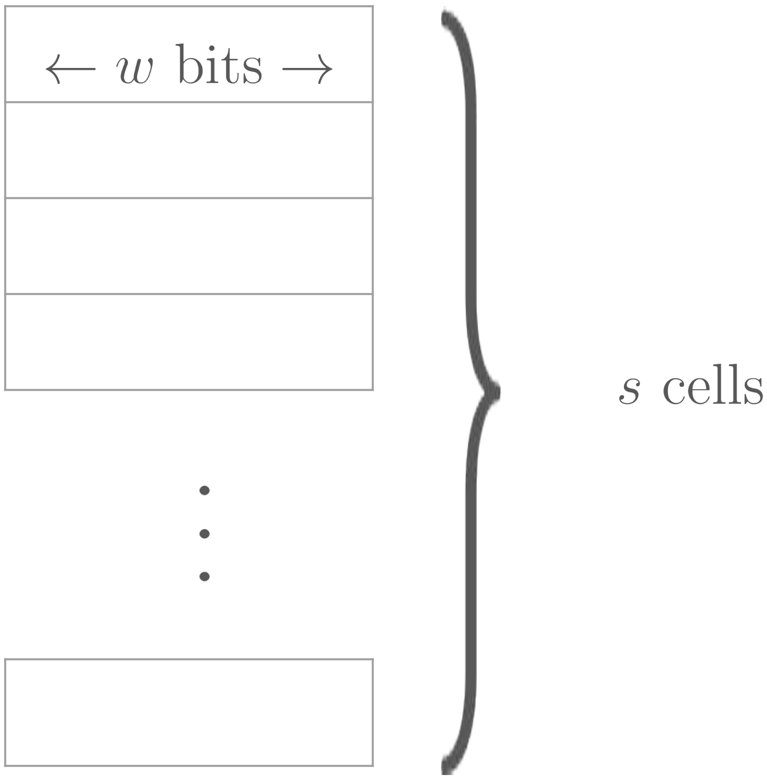
$$\mathsf{RCC}(\Pi) := \max_{x \in \mathcal{X}, y \in \mathcal{Y}, \mathcal{R}} |m_1| + |m_2| + \ldots + |m_r|$$

$$\mathsf{RCC}(f) := \min_{\Pi \text{ computing } f} \mathsf{RCC}(\Pi)$$

[Image credit: Internet]

Data structures: cell-probe model

- Memory: s cells organized in w-bit words.



Parameters to optimize (minimize) :

- Space s
- Word size w
- Query time t

- *Static:* Store data such that *queries* can be supported. No updation.

Set-intersection

- $\mathcal{U} = \{1, \ldots, n\}$.

- Preprocessing: Store an arbitrary $Y \subseteq \mathcal{U}$ in memory, using space s (worst case over Y).

- Objective: Support queries of the form "Is $X \cap Y$ empty?" in as low a time t (worst case over X and Y) as possible.

Scheme-1

- Store Y as a string of n bits broken up into words of size w.

- $s, t = \lceil n/w \rceil$.

Scheme-2

- For every set X store whether Y intersects X.

- $s = 2^n/w, t = 1.$

A lower bound

Theorem: Any data structure that solves the set intersection problem must have

$$t(\lceil \log s \rceil + w) \geq n + 1.$$

Proof idea:

Fact: $\mathsf{CC}(\mathsf{DISJ}) = n + 1.$

Proof of the lower bound

Proof: We will show that Alice and Bob may use a data structure to deterministically solve Disjointness with at most $t(\lceil \log s \rceil + w)$communication.

Protocol: Bob stores T as per the data structure pre-processing. Alice "queries S". Alice then invokes the query service routine of the data structure and solves the set disjointness by accessing t memory cells of the data structure in t rounds as follows: In each round, Alice requests for the content of a memory cell to Bob, by sending him the address of the cell. This requires bits of communication. Alice responds to each of those queries by sending the content (bits). Thus, the communication complexity is $t(\lceil \log s \rceil + w)$.

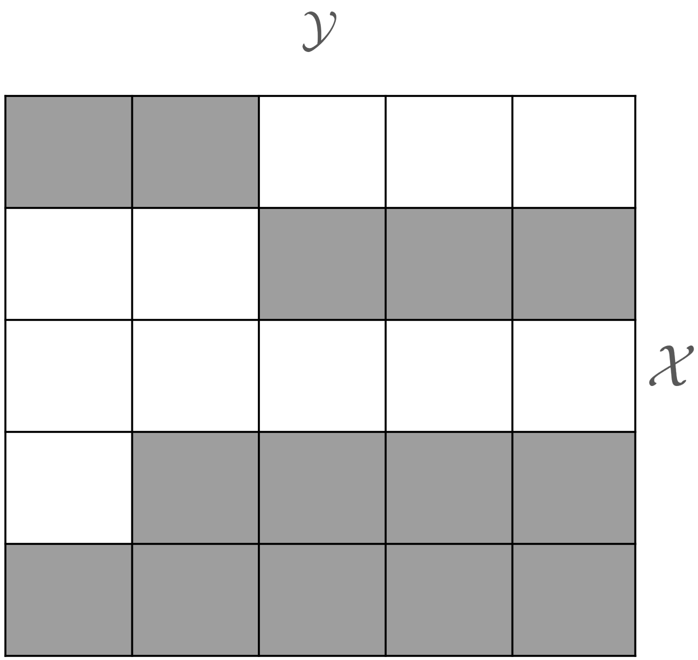
The proof follows from the fact $\mathsf{CC}(\mathsf{DISJ}) = n + 1.$

Structure of communication protocols

- Two way. Deterministic.

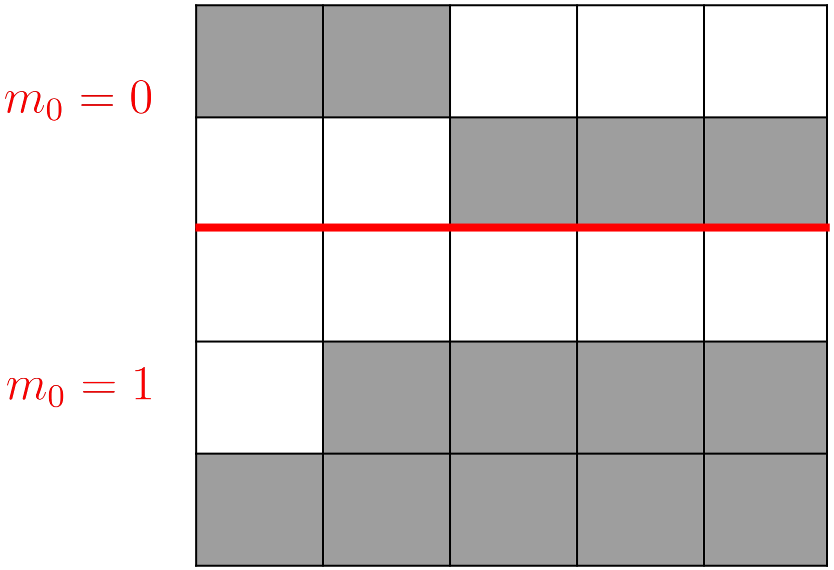
- $\mathcal{Z} = \{0,1\}$, i.e., $f : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$.

Structure of communication protocols



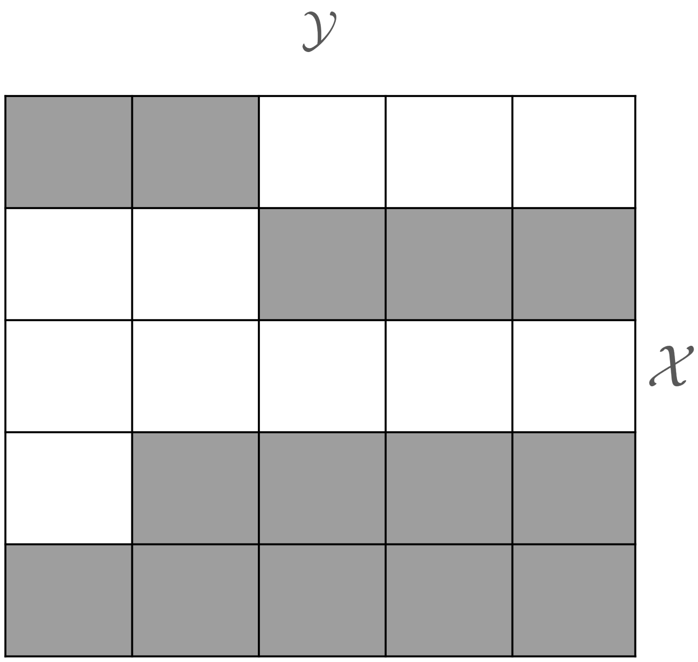
$$M_f(x, y) = f(x, y).$$

Communication matrix M_f



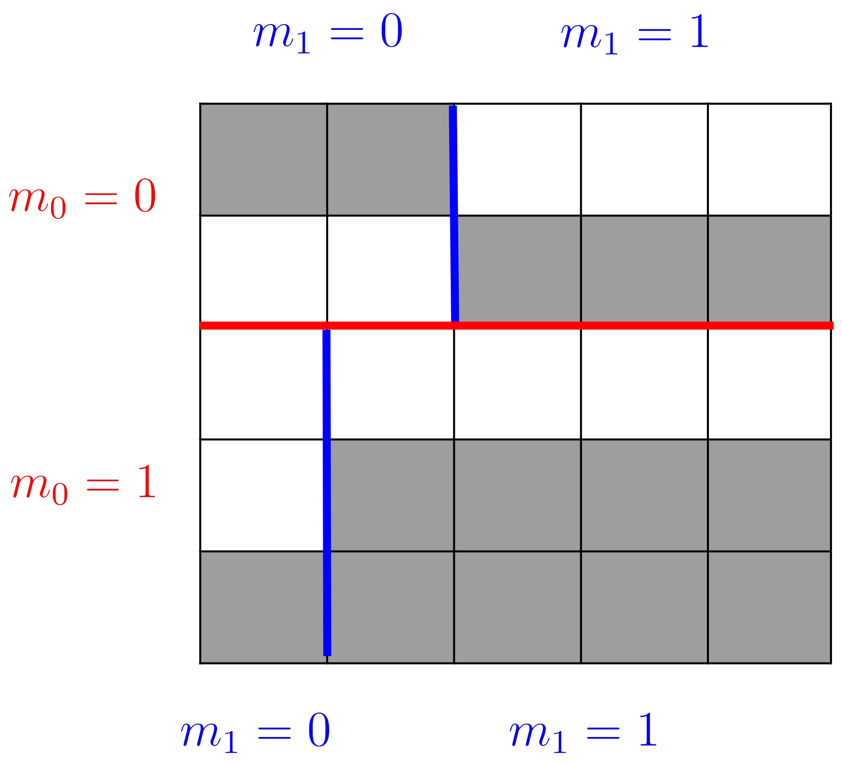
Round 1

Structure of communication protocols



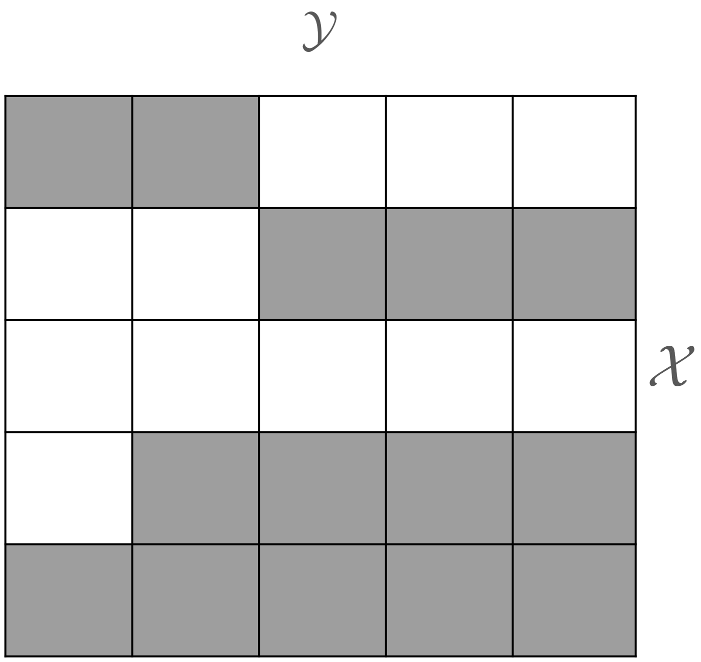
$M_f(x, y) = f(x, y).$

Communication matrix M_f



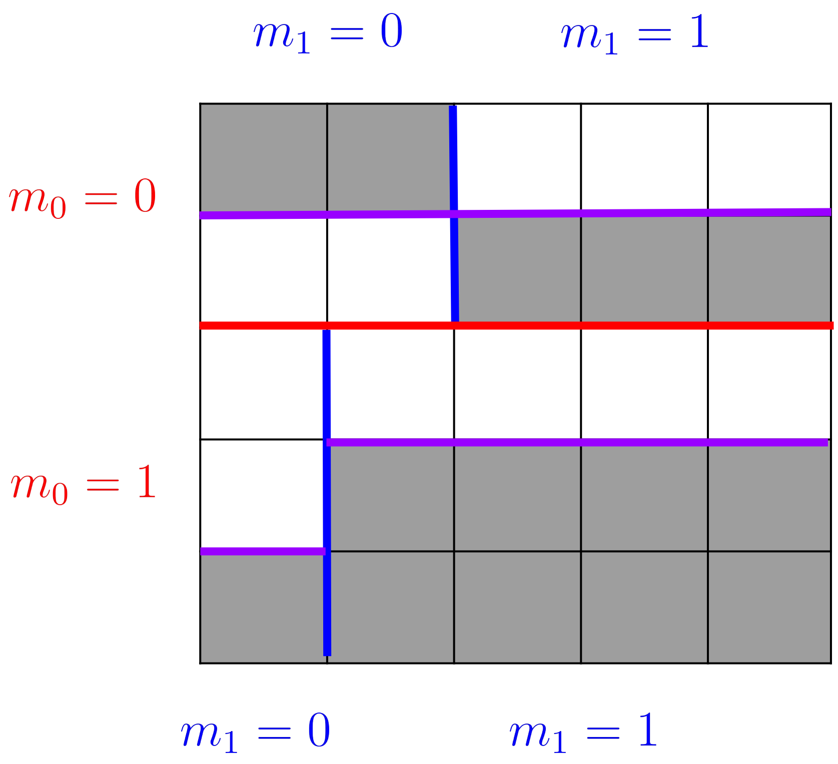
Round 2

Structure of communication protocols



$M_f(x, y) = f(x, y).$

Communication matrix M_f



Round 3

Partition number

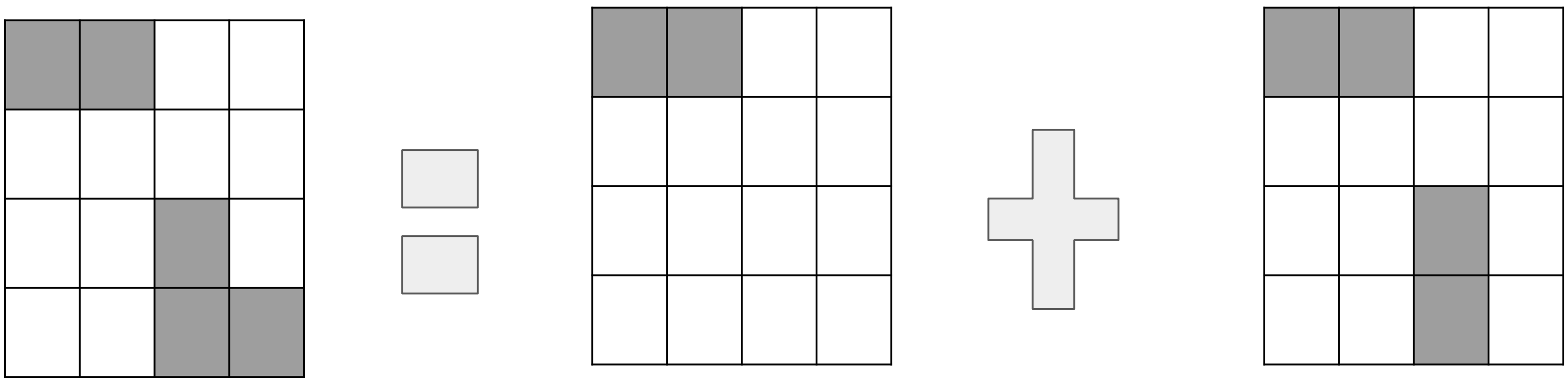
- Partition number $P(f)$ of a function f is the minimum number k such that M_f can be partitioned into at most k monochromatic rectangles.

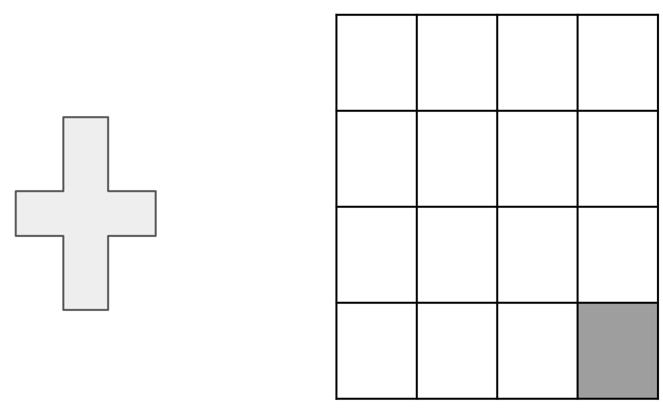
- Clearly $\mathsf{CC}(f) \geq \lceil \log_2 P(f) \rceil$.

- $\mathsf{CC}(f) = O(\log P(f))^2$ (Aho, Ullman, Yannakakis 1983).

- Tight (Göös, Pitassi, Watson 2017).

Lower bounding Partition number: Rank





$M_f = \text{Sum of } P(f) \text{ rank one matrices.}$

Rank (continued)

- Thus, $\mathsf{rank}(M_f) \leq P(f)$.

- Example: Equality (again).

- M_{EQ} is the identity matrix of dimension $2^n \times 2^n$.

- Thus, $\mathsf{CC}(\mathsf{EQ}) \geq \lceil \log_2 P(\mathsf{EQ}) \rceil \geq \lceil \log_2 \mathsf{rank}(M_{\mathsf{EQ}}) \rceil = n$.

- Exercise: Show that the rank of M_{DISJ} is 2^n.

The Log-rank conjecture

- $\mathsf{CC}(f) \geq \lceil \log_2 \mathsf{rank}(M_f) \rceil$.

- How much larger can $\mathsf{CC}(f)$ be than $\log_2 \mathsf{rank}(f)$?

- Rank over real numbers.

Log-rank conjecture (Lovász and Saks 1988): $\exists k > 0$ such that $\forall f$, $\mathsf{CC}(f) = O(\log \mathsf{rank}(M_f)^k)$.

The Log-rank conjecture

• Easy: $\mathsf{CC}(f) = O(\mathsf{rank}(M_f))$.

• Best known bound: $\mathsf{CC}(f) = O(\sqrt{\mathsf{rank}(\mathsf{M_f})} \log \mathsf{rank}(\mathsf{M_f}))$. (Lovett 2014)

• Best lower bound: $\exists f$ such that $\mathsf{CC}(\mathsf{f}) = \Omega(\log P(f))^2 = \Omega(\log \mathsf{rank}(M_f))^2$. (Göös, Pitassi, Watson 2017)

• *Log-approximate-rank conjecture:* refuted by Chattopadhyay, Mande and Sherif (2020).

Fooling sets

- Consider the Equality problem.

- Consider the set of all its 1-inputs $\mathsf{EQ}^{-1}(1) = \{(x, x) \mid x \in \{0,1\}^n\}$.

- Any rectangle that contains both (x, x) and (y, y) also contains (x, y) and (y, x).

  Cannot be monochromatic.

- $\mathsf{EQ}^{-1}(1)$ is a "fooling set".

Fooling Sets continued

- $P(\mathsf{EQ}) \geq\ 2^n$ (to cover $\mathsf{EQ}^{-1}(1)$)$+\,1$ (to cover $\mathsf{EQ}^{-1}(0)$).

- $\mathsf{CC}(\mathsf{EQ}) \geq \lceil \log_2 P(\mathsf{EQ}) \rceil \geq n + 1$.

- Tight bound for equality.

- Exercise: Find a 2^n-sized fooling set for disjointness.

- May give exponentially worse bounds sometimes.

References

1. “Communication Complexity (for Algorithm Designers)” by Tim Roughgarden. https://arxiv.org/abs/1509.06257

2. “Communication Complexity and Applications” by Anup Rao and Amir Yehudayoff. Cambridge University Press.

3. “Communication Complexity” by Eyal Kushilevitz and Noam Nisan. Cambridge University Press.

Thank you.

Index function

- $\mathsf{INDEX} : \{0,1\}^n \times [n] \to \{0,1\}$:

  $\mathsf{INDEX}(x, i) = x_i$

- $\mathsf{CC}^{\rightarrow}(\mathsf{INDEX}) = n$ (pigeon-hole principle)

- $\mathsf{RCC}^{\rightarrow}(\mathsf{INDEX}) = \Omega(n)$

Gap-Hamming

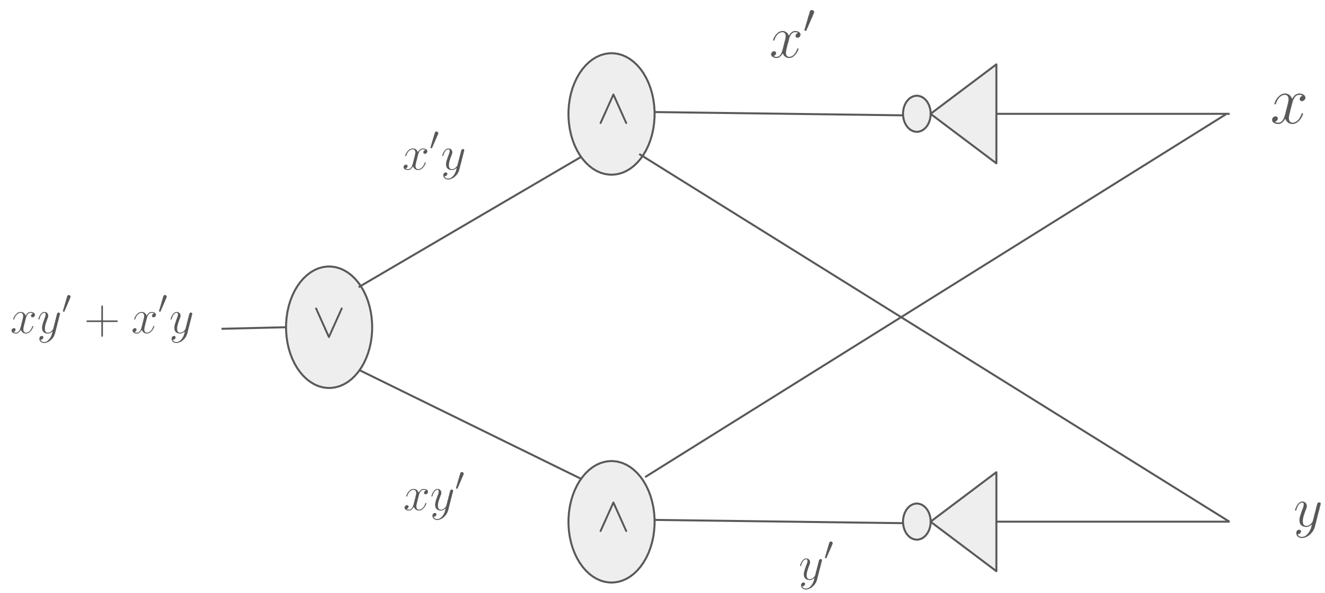
- $\mathsf{GapHam} : 2^{[n]} \times 2^{[n]} \to \{0,1\}$:

$$\mathsf{GapHam}(S,T) = \begin{cases} 1 & \text{if } |S \triangle T| \geq \frac{n}{2} + \sqrt{n}, \\ 0 & \text{if } |S \triangle T| \leq \frac{n}{2} - \sqrt{n}, \\ * & \text{otherwise} \end{cases}$$

- $\mathsf{CC}^{\to}(\mathsf{GapHam}) = n$ (pigeon-hole principle)

- $\mathsf{RCC}^{\to}(\mathsf{GapHam}) = \Omega(n)$

Circuit complexity



- 2-input AND and OR Gates.
- Depth=max. length of an i/p-o/p path = 2.
- Size=no. of gates=3.
- $T(n)$ step algorithm $\Rightarrow \widetilde{O}(T(n))$ sized circuit.

Size-depth trade-off

Open question: Can every function that is computable using circuits of size polynomial in n be computed by circuits of depth O(log n)?

Monotone functions and circuits

- **Monotone functions:** Changing an input variable from 0 to 1 does not change the function value from 1 to 0.
- **Example**: AND, OR.
- **“Algorithmic examples”**: Matching, Connectivity.
- **Monotone circuits:** No NOT gate.
- Any monotone function can be computed by a monotone circuit.
- However, use of NOT gates may lead to cheaper circuits. Example: The **perfect matching** function has polynomial sized non-monotone circuit (**perfect matching** has a polytime algorithm) but no polynomial sized monotone circuit (Razborov 1985).

Monotone Karchmer-Wigderson game

- $f : \{0,1\}^n \to \{0,1\}$ is a monotone function.

- Alice is given $x \in f^{-1}(0)$, Bob is given $y \in f^{-1}(1)$.

- Task: find an index i such that $x_i = 0$ and $y_i = 1$.

Theorem (**Karchmer-Wigderson 1990**): Communication complexity of the KW-game=circuit depth complexity of f.

The **match** function

$$\mathsf{match}(G) = \begin{cases} 1 & \text{if } G \text{ has a matching of size at least } n/3 + 1, \\ 0 & \text{otherwise.} \end{cases}$$

- Monotone, has a polysized circuit.
- Is there a low-depth circuit?

Theorem (Raz-Wigderson 1992): Any monotone circuit computing **match on input graphs with n vertices** has depth Ω(n).

Proof idea

Recall:

Theorem (**Karchmer-Wigderson 1990**): Communication complexity of the KW-game=circuit depth complexity of .

Idea: Show a lower bound on the KW game for **match.**

$$\text{Fact: } \mathsf{RCC}(\mathsf{DISJ}) = \Omega(n).$$

Randomized reduction from DISJ to KW game .

Proof idea (contd.)

- Showing that any protocol $\prod$ for the KW game given by **match** has communication complexity Ω(n).
- Randomized reduction from DISJ. Given inputs S, T, the parties produce inputs G, G' to KW game by public randomness and no communication. G and G' are graphs with Θ(n) vertices.
- Let e be the edge that $\prod$ returns.
- Bob examines e and answers whether S and T intersect with no additional communication.
- For every S, T, Bob's answer is correct with probability at least ⅔.
- Proof follows from the Fact.

Scheme-3

- Parameter p.

- For every subset V of size at most p store whether or not Y intersects V.

- Every set X can be expressed as a union of at most $\lceil n/p \rceil$ disjoint sets.

- $s = \dfrac{\sum_{i=1}^{p} \binom{n}{i}}{w}, t = \lceil n/p \rceil.$