CMPSCI 711: More Advanced Algorithms

Lower Bounds 1: : Lower Bounds and Communication Complexity

Andrew McGregor

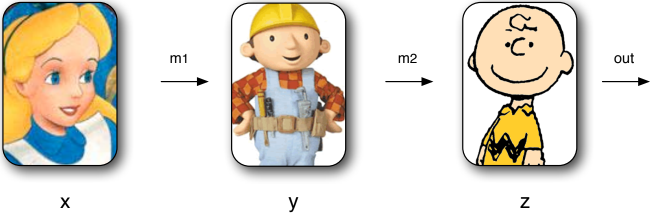
Last Compiled: March 27, 2018

Basic Communication Complexity

- Three friends Alice, Bob, and Charlie each have some information x, y, z and Charlie wants to compute some function $P(x, y, z)$.

Basic Communication Complexity

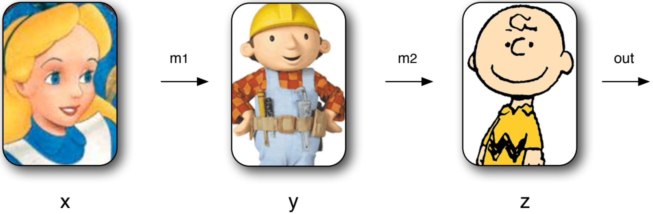
- Three friends Alice, Bob, and Charlie each have some information x, y, z and Charlie wants to compute some function $P(x, y, z)$.



- To help Charlie, Alice sends a message m_1 to Bob, and then Bob sends a message m_2 to Charlie.

Basic Communication Complexity

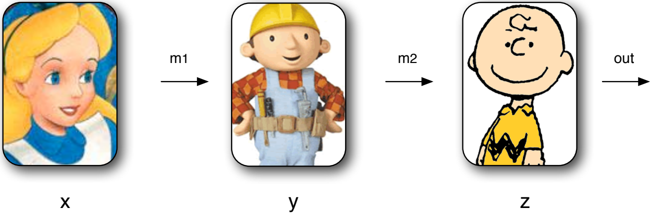
- Three friends Alice, Bob, and Charlie each have some information x, y, z and Charlie wants to compute some function $P(x, y, z)$.



- To help Charlie, Alice sends a message m_1 to Bob, and then Bob sends a message m_2 to Charlie.
- *Question:* How large must be $|m_1| + |m_2|$ be if Charlie is to evaluate $P(x, y, z)$ correctly in the worst case over possible x, y, z?

Basic Communication Complexity

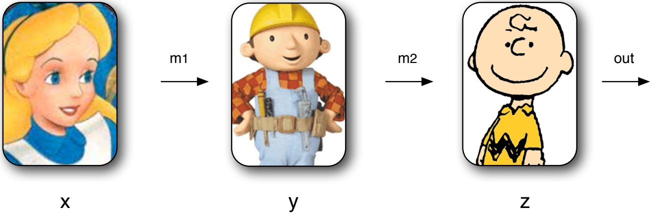
- Three friends Alice, Bob, and Charlie each have some information x, y, z and Charlie wants to compute some function $P(x, y, z)$.



- To help Charlie, Alice sends a message m_1 to Bob, and then Bob sends a message m_2 to Charlie.
- *Question:* How large must be $|m_1| + |m_2|$ be if Charlie is to evaluate $P(x, y, z)$ correctly in the worst case over possible x, y, z?
  - *Deterministic:* $m_1(x)$, $m_2(m_1, y)$, $\mathrm{out}(m_2, z) = P(x, y, z)$

Basic Communication Complexity

- Three friends Alice, Bob, and Charlie each have some information x, y, z and Charlie wants to compute some function $P(x, y, z)$.

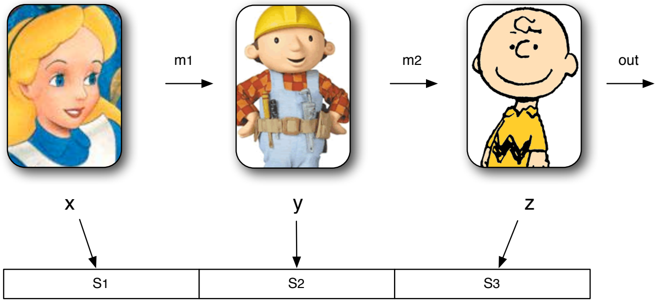


- To help Charlie, Alice sends a message m_1 to Bob, and then Bob sends a message m_2 to Charlie.
- *Question:* How large must be $|m_1| + |m_2|$ be if Charlie is to evaluate $P(x, y, z)$ correctly in the worst case over possible x, y, z?
  - *Deterministic:* $m_1(x)$, $m_2(m_1, y)$, $\text{out}(m_2, z) = P(x, y, z)$
  - *Random:* $m_1(x, r)$, $m_2(m_1, y, r)$, $\text{out}(m_2, z, r)$ where r is public random bits. Require $\mathbb{P}[\text{out}(m_2, z) = P(x, y, z)] \geq 9/10$.

Stream Algorithms Yield Communication Protocols

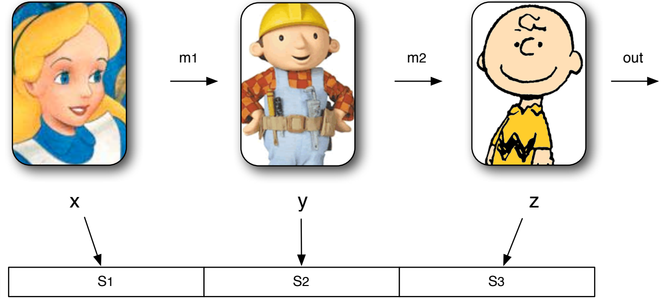
Stream Algorithms Yield Communication Protocols

- Let Q be some stream problem. Suppose there's a reduction $x \to S_1$, $y \to S_2$, $z \to S_3$ such that knowing $Q(S_1 \circ S_2 \circ S_3)$ solves $P(x, y, z)$.



Stream Algorithms Yield Communication Protocols

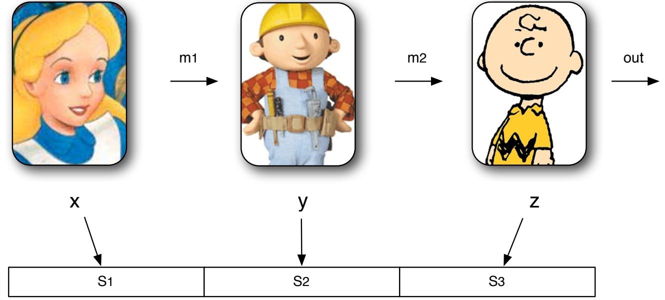
- Let Q be some stream problem. Suppose there's a reduction $x \to S_1$, $y \to S_2$, $z \to S_3$ such that knowing $Q(S_1 \circ S_2 \circ S_3)$ solves $P(x, y, z)$.



- An s-bit stream algorithm $\mathcal{A}$ for Q yields $2s$-bit protocol for P:

Stream Algorithms Yield Communication Protocols

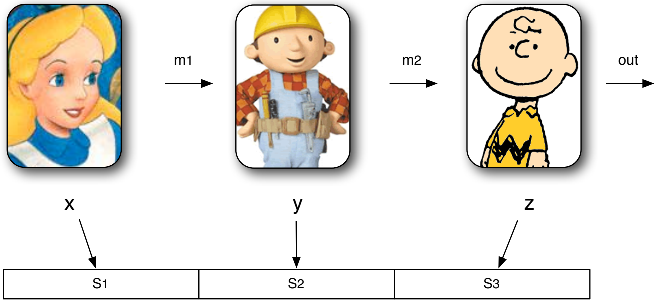
- Let Q be some stream problem. Suppose there's a reduction $x \to S_1$, $y \to S_2$, $z \to S_3$ such that knowing $Q(S_1 \circ S_2 \circ S_3)$ solves $P(x, y, z)$.



- An s-bit stream algorithm $\mathcal{A}$ for Q yields $2s$-bit protocol for P: Alice runs $\mathcal{A}$ of S_1;

Stream Algorithms Yield Communication Protocols

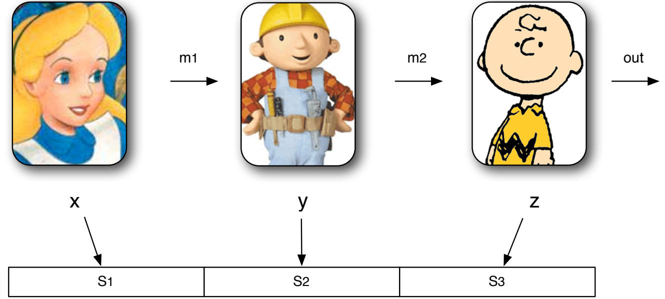
- Let Q be some stream problem. Suppose there's a reduction $x \to S_1$, $y \to S_2$, $z \to S_3$ such that knowing $Q(S_1 \circ S_2 \circ S_3)$ solves $P(x, y, z)$.



- An s-bit stream algorithm $\mathcal{A}$ for Q yields $2s$-bit protocol for P: Alice runs $\mathcal{A}$ of S_1; sends memory state to Bob;

Stream Algorithms Yield Communication Protocols

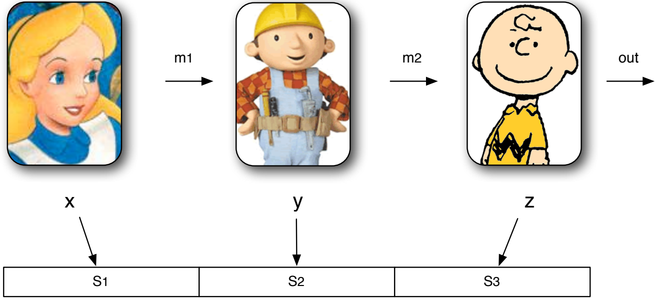
- Let Q be some stream problem. Suppose there's a reduction $x \to S_1$, $y \to S_2$, $z \to S_3$ such that knowing $Q(S_1 \circ S_2 \circ S_3)$ solves $P(x, y, z)$.



- An s-bit stream algorithm $\mathcal{A}$ for Q yields $2s$-bit protocol for P: Alice runs $\mathcal{A}$ of S_1; sends memory state to Bob; Bob instantiates $\mathcal{A}$ with state and runs it on S_2;

Stream Algorithms Yield Communication Protocols

- Let Q be some stream problem. Suppose there's a reduction $x \to S_1$, $y \to S_2$, $z \to S_3$ such that knowing $Q(S_1 \circ S_2 \circ S_3)$ solves $P(x, y, z)$.



- An s-bit stream algorithm $\mathcal{A}$ for Q yields $2s$-bit protocol for P: Alice runs $\mathcal{A}$ of S_1; sends memory state to Bob; Bob instantiates $\mathcal{A}$ with state and runs it on S_2; sends state to Charlie who finishes running $\mathcal{A}$ on S_3 and infers $P(x, y, z)$ from $Q(S_1 \circ S_2 \circ S_3)$.

Communication Lower Bounds imply Stream Lower Bounds

- Had there been t players, the s-bit stream algorithm for Q would have lead to a $(t-1)s$ bit protocol P.

Communication Lower Bounds imply Stream Lower Bounds

- Had there been t players, the s-bit stream algorithm for Q would have lead to a $(t-1)s$ bit protocol P.
- Hence, a lower bound of L for P implies $s = \Omega(L/t)$.

Outline

Indexing

- Consider a binary string $x \in \{0,1\}^n$ and $j \in [n]$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \qquad \text{and} \qquad j = 3$$

and define $\textsc{Index}(x, j) = x_j$

Indexing

- Consider a binary string $x \in \{0,1\}^n$ and $j \in [n]$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \qquad \text{and} \qquad j = 3$$

  and define $\textsc{Index}(x, j) = x_j$
- Suppose Alice knows x and Bob knows j.

Indexing

- Consider a binary string $x \in \{0,1\}^n$ and $j \in [n]$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \qquad \text{and} \qquad j = 3$$

  and define $\textsc{Index}(x, j) = x_j$
- Suppose Alice knows x and Bob knows j.
- How many bits need to be sent by Alice for Bob to determine $\textsc{Index}(x, j)$ with probability $9/10$?

Indexing

- Consider a binary string $x \in \{0,1\}^n$ and $j \in [n]$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \qquad \text{and} \qquad j = 3$$

  and define $\textsc{Index}(x, j) = x_j$
- Suppose Alice knows x and Bob knows j.
- How many bits need to be sent by Alice for Bob to determine $\textsc{Index}(x, j)$ with probability $9/10$? $\Omega(n)$

Application: Median Finding

- *Thm:* Any algorithm that returns the exact median of length $2n - 1$ stream requires $\Omega(n)$ memory.

Application: Median Finding

- *Thm:* Any algorithm that returns the exact median of length $2n - 1$ stream requires $\Omega(n)$ memory.
- *Reduction from indexing on input $x \in \{0, 1\}^n, j \in [n]$*: Alice generates: $S_1 = \{2i + x_i : i \in [n]\}$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \to \{2, 5, 6, 9, 11, 12\}$$

Application: Median Finding

- *Thm:* Any algorithm that returns the exact median of length $2n-1$ stream requires $\Omega(n)$ memory.
- *Reduction from indexing on input $x \in \{0,1\}^n, j \in [n]$*: Alice generates: $S_1 = \{2i + x_i : i \in [n]\}$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \to \{2, 5, 6, 9, 11, 12\}$$

Bob generates: $S_2 = \{n - j$ copies of 0 and $j - 1$ copies of $2n + 2\}$, e.g.,

$$j = 3 \longrightarrow \{0, 0, 0, 14, 14\}$$

Application: Median Finding

- *Thm:* Any algorithm that returns the exact median of length $2n-1$ stream requires $\Omega(n)$ memory.
- *Reduction from indexing on input $x \in \{0,1\}^n, j \in [n]$*: Alice generates: $S_1 = \{2i + x_i : i \in [n]\}$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \to \{2, 5, 6, 9, 11, 12\}$$

  Bob generates: $S_2 = \{n - j$ copies of 0 and $j - 1$ copies of $2n + 2\}$, e.g.,

$$j = 3 \longrightarrow \{0, 0, 0, 14, 14\}$$

- Then $\text{median}(S_1 \cup S_2) = 2j + x_j$ and parity determines $\textsc{Index}(x, j)$

Application: Median Finding

- *Thm:* Any algorithm that returns the exact median of length $2n - 1$ stream requires $\Omega(n)$ memory.
- *Reduction from indexing on input $x \in \{0,1\}^n, j \in [n]$*: Alice generates: $S_1 = \{2i + x_i : i \in [n]\}$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \to \{2, 5, 6, 9, 11, 12\}$$

  Bob generates: $S_2 = \{n - j \text{ copies of } 0 \text{ and } j - 1 \text{ copies of } 2n + 2\}$, e.g.,

$$j = 3 \longrightarrow \{0, 0, 0, 14, 14\}$$

- Then $\text{median}(S_1 \cup S_2) = 2j + x_j$ and parity determines $\textsc{Index}(x, j)$
- An s-space algorithm gives an s-bit protocol so

$$s = \Omega(n)$$

  by the one-way communication complexity of indexing.

Multi-Party Set-Disjointness

- Consider a $t \times n$ matrix where column has weight $0, 1$, or t, e.g.,

$$M = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Multi-Party Set-Disjointness

- Consider a $t \times n$ matrix where column has weight $0, 1$, or t, e.g.,

$$M = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

- Define $\textsc{disj}_t(M) = \bigvee_j \textsc{and}_t(M_{1,j}, \ldots, M_{t,j})$, i.e., $\textsc{disj}_t(M) = 1$ iff there is an all 1's column.

Multi-Party Set-Disjointness

- Consider a $t \times n$ matrix where column has weight $0, 1$, or t, e.g.,

$$M = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

- Define $\textsc{disj}_t(M) = \bigvee_j \textsc{and}_t(M_{1,j}, \ldots, M_{t,j})$, i.e., $\textsc{disj}_t(M) = 1$ iff there is an all 1's column.
- Consider t players where P_i knows i-th row of M.

Multi-Party Set-Disjointness

- Consider a $t \times n$ matrix where column has weight $0, 1$, or t, e.g.,

$$M = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

- Define $\textsc{disj}_t(M) = \bigvee_j \textsc{and}_t(M_{1,j}, \ldots, M_{t,j})$, i.e., $\textsc{disj}_t(M) = 1$ iff there is an all 1's column.
- Consider t players where P_i knows i-th row of M.
- How many bits need to be communicated between the players to determine $\textsc{disj}_t(M)$?

Multi-Party Set-Disjointness

- Consider a $t \times n$ matrix where column has weight $0, 1$, or t, e.g.,

$$M = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

- Define $\textsc{disj}_t(M) = \bigvee_j \textsc{and}_t(M_{1,j}, \ldots, M_{t,j})$, i.e., $\textsc{disj}_t(M) = 1$ iff there is an all 1's column.
- Consider t players where P_i knows i-th row of M.
- How many bits need to be communicated between the players to determine $\textsc{disj}_t(M)$? $\Omega(n/t)$

Application: Frequency Moments

- *Thm:* A 2-approximation algorithm for F_k needs $\Omega(n^{1-2/k})$ space.
- *Reduction from multi-party set disjointness on input $M \in \{0,1\}^{t \times n}$:*

Application: Frequency Moments

- *Thm:* A 2-approximation algorithm for F_k needs $\Omega(n^{1-2/k})$ space.
- *Reduction from multi-party set disjointness on input $M \in \{0,1\}^{t \times n}$*:
  P_i generates set $S_i = \{j : M_{ij} = 1\}$, e.g.,

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \longrightarrow \{4, 1, 4, 5, 2, 4, 4\}$$

Application: Frequency Moments

- *Thm:* A 2-approximation algorithm for F_k needs $\Omega(n^{1-2/k})$ space.
- *Reduction from multi-party set disjointness on input $M \in \{0,1\}^{t \times n}$*:
  P_i generates set $S_i = \{j : M_{ij} = 1\}$, e.g.,

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \longrightarrow \{4, 1, 4, 5, 2, 4, 4\}$$

- If all columns have weight 0 or 1: $F_k(S) \leq n$

Application: Frequency Moments

- *Thm:* A 2-approximation algorithm for F_k needs $\Omega(n^{1-2/k})$ space.
- *Reduction from multi-party set disjointness on input $M \in \{0,1\}^{t\times n}$*:
  P_i generates set $S_i = \{j : M_{ij} = 1\}$, e.g.,

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \longrightarrow \{4,1,4,5,2,4,4\}$$

- If all columns have weight 0 or 1: $F_k(S) \leq n$
- If there's column of weight t: $F_k(S) \geq t^k$

Application: Frequency Moments

- *Thm:* A 2-approximation algorithm for F_k needs $\Omega(n^{1-2/k})$ space.
- *Reduction from multi-party set disjointness on input $M \in \{0,1\}^{t \times n}$*:
  P_i generates set $S_i = \{j : M_{ij} = 1\}$, e.g.,

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \longrightarrow \{4, 1, 4, 5, 2, 4, 4\}$$

- If all columns have weight 0 or 1: $F_k(S) \leq n$
- If there's column of weight t: $F_k(S) \geq t^k$
- If $t > 2^{1/k} n^{1/k}$ then a 2 approximation of $F_k(S)$ distinguishes cases.

Application: Frequency Moments

- *Thm:* A 2-approximation algorithm for F_k needs $\Omega(n^{1-2/k})$ space.
- *Reduction from multi-party set disjointness on input $M \in \{0,1\}^{t\times n}$*:
  P_i generates set $S_i = \{j : M_{ij} = 1\}$, e.g.,

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \longrightarrow \{4,1,4,5,2,4,4\}$$

- If all columns have weight 0 or 1: $F_k(S) \leq n$
- If there's column of weight t: $F_k(S) \geq t^k$
- If $t > 2^{1/k} n^{1/k}$ then a 2 approximation of $F_k(S)$ distinguishes cases.
- An s-space 2-approximation gives a $s(t-1)$ bit protocol so

$$s = \Omega(n/t^2) = \Omega(n^{1-2/k})$$

  by the communication complexity of set-disjointness.

Hamming Distance

- Consider 2 binary vectors $x, y \in \{0,1\}^n$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

$$y = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Hamming Distance

- Consider 2 binary vectors $x, y \in \{0,1\}^n$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

$$y = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$

- Define the Hamming distance $\Delta(x, y) = |\{i : x_i \neq y_i\}|$.

Hamming Distance

- Consider 2 binary vectors $x, y \in \{0,1\}^n$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

$$y = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$

- Define the Hamming distance $\Delta(x, y) = |\{i : x_i \neq y_i\}|$.
- Suppose Alice knows x and Bob knows y.

Hamming Distance

- Consider 2 binary vectors $x, y \in \{0,1\}^n$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

$$y = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$

- Define the Hamming distance $\Delta(x,y) = |\{i : x_i \neq y_i\}|$.
- Suppose Alice knows x and Bob knows y.
- How many bits need to be communicated to estimate $\Delta(x,y)$ up to an additive $\sqrt{n}$ error?

Hamming Distance

- Consider 2 binary vectors $x, y \in \{0,1\}^n$, e.g.,

$$x = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

$$y = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$

- Define the Hamming distance $\Delta(x, y) = |\{i : x_i \neq y_i\}|$.
- Suppose Alice knows x and Bob knows y.
- How many bits need to be communicated to estimate $\Delta(x, y)$ up to an additive $\sqrt{n}$ error? $\Omega(n)$ bits.

Application: Distinct Elements

- *Thm:* A $(1+\epsilon)$-approximation algorithm for F_0 needs $\Omega(\epsilon^{-2})$ space.
- *Reduction from Hamming Distance on input $x, y \in \{0,1\}^n$*: Alice and Bob generate sets $S_1 = \{j : x_j = 1\}$ and $S_2 = \{j : y_j = 1\}$, e.g.,

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \longrightarrow \{2,4,5,1,2,5,6\}$$

Application: Distinct Elements

- *Thm:* A $(1+\epsilon)$-approximation algorithm for F_0 needs $\Omega(\epsilon^{-2})$ space.
- *Reduction from Hamming Distance on input $x, y \in \{0,1\}^n$:* Alice and Bob generate sets $S_1 = \{j : x_j = 1\}$ and $S_2 = \{j : y_j = 1\}$, e.g.,

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \longrightarrow \{2,4,5,1,2,5,6\}$$

- Note that $2F_0(S) = |x| + |y| + \Delta(x, y)$.

Application: Distinct Elements

- *Thm:* A $(1+\epsilon)$-approximation algorithm for F_0 needs $\Omega(\epsilon^{-2})$ space.
- *Reduction from Hamming Distance on input $x, y \in \{0,1\}^n$*: Alice and Bob generate sets $S_1 = \{j : x_j = 1\}$ and $S_2 = \{j : y_j = 1\}$, e.g.,

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \longrightarrow \{2,4,5,1,2,5,6\}$$

- Note that $2F_0(S) = |x| + |y| + \Delta(x,y)$.
- We may assume $|x|$ and $|y|$ are known Bob. Hence, a $(1+\epsilon)$ approximation of F_0 yields an additive approximation to $\Delta(x,y)$ of

$$\epsilon(|x| + |y| + \Delta(x,y))/2 \leq n\epsilon$$

Application: Distinct Elements

- *Thm:* A $(1+\epsilon)$-approximation algorithm for F_0 needs $\Omega(\epsilon^{-2})$ space.
- *Reduction from Hamming Distance on input $x, y \in \{0,1\}^n$*: Alice and Bob generate sets $S_1 = \{j : x_j = 1\}$ and $S_2 = \{j : y_j = 1\}$, e.g.,

$$\left(\begin{array}{cccccc} 0 & 1 & 0 & 1 & 1 & 0 \end{array} \right), \left(\begin{array}{cccccc} 1 & 1 & 0 & 0 & 1 & 1 \end{array} \right) \longrightarrow \{2,4,5,1,2,5,6\}$$

- Note that $2F_0(S) = |x| + |y| + \Delta(x,y)$.
- We may assume $|x|$ and $|y|$ are known Bob. Hence, a $(1+\epsilon)$ approximation of F_0 yields an additive approximation to $\Delta(x,y)$ of

$$\epsilon(|x| + |y| + \Delta(x,y))/2 \leq n\epsilon$$

- This is less than $\sqrt{n}$ if $\epsilon < 1/\sqrt{n}$

Application: Distinct Elements

- *Thm:* A $(1+\epsilon)$-approximation algorithm for F_0 needs $\Omega(\epsilon^{-2})$ space.
- *Reduction from Hamming Distance on input $x, y \in \{0,1\}^n$*: Alice and Bob generate sets $S_1 = \{j : x_j = 1\}$ and $S_2 = \{j : y_j = 1\}$, e.g.,

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \longrightarrow \{2,4,5,1,2,5,6\}$$

- Note that $2F_0(S) = |x| + |y| + \Delta(x,y)$.
- We may assume $|x|$ and $|y|$ are known Bob. Hence, a $(1+\epsilon)$ approximation of F_0 yields an additive approximation to $\Delta(x,y)$ of

$$\epsilon(|x| + |y| + \Delta(x,y))/2 \leq n\epsilon$$

- This is less than $\sqrt{n}$ if $\epsilon < 1/\sqrt{n}$
- An s-space $(1+\epsilon)$-approximation gives a s bit protocol so

$$s = \Omega(n) = \Omega(1/\epsilon^2)$$

by communication complexity of approximating Hamming distance.

Outline

Hamming Distance Lower Bound

Some communication results can be proved via a reduction from other communication results.

Theorem

Alice and Bob have $x \in \{0,1\}^n$ *and* $y \in \{0,1\}^n$ *respectively. If Bob wants to determine* $\Delta(x,y)$ *up to* $\pm\sqrt{n}$ *with probability* $9/10$ *then Alice must send* $\Omega(n)$ *bits.*

Hamming Distance Lower Bound

- Reduction from INDEX problem: Alice knows $z \in \{0,1\}^t$ and Bob knows $j \in [t]$. Let's assume $|z| = t/2$ and this is odd.

Hamming Distance Lower Bound

- Reduction from INDEX problem: Alice knows $z \in \{0,1\}^t$ and Bob knows $j \in [t]$. Let's assume $|z| = t/2$ and this is odd.
- Alice and Bob pick $r \in_R \{-1,1\}^t$ using public random bits.

Hamming Distance Lower Bound

- Reduction from INDEX problem: Alice knows $z \in \{0,1\}^t$ and Bob knows $j \in [t]$. Let's assume $|z| = t/2$ and this is odd.
- Alice and Bob pick $r \in_R \{-1,1\}^t$ using public random bits.
- Alice computes $\text{sign}(r.z)$ and Bob computes $\text{sign}(r_j)$

Hamming Distance Lower Bound

- Reduction from INDEX problem: Alice knows $z \in \{0,1\}^t$ and Bob knows $j \in [t]$. Let's assume $|z| = t/2$ and this is odd.
- Alice and Bob pick $r \in_R \{-1,1\}^t$ using public random bits.
- Alice computes $\text{sign}(r.z)$ and Bob computes $\text{sign}(r_j)$
- *Lemma:* For some constant $c > 0$,

$$\mathbb{P}\left[\text{sign}(r.z) = \text{sign}(r_j)\right] = \begin{cases} 1/2 & \text{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \text{if } z_j = 1 \end{cases}$$

Hamming Distance Lower Bound

- Reduction from INDEX problem: Alice knows $z \in \{0,1\}^t$ and Bob knows $j \in [t]$. Let's assume $|z| = t/2$ and this is odd.
- Alice and Bob pick $r \in_R \{-1,1\}^t$ using public random bits.
- Alice computes $\text{sign}(r.z)$ and Bob computes $\text{sign}(r_j)$
- *Lemma:* For some constant $c > 0$,

$$\mathbb{P}\left[\text{sign}(r.z) = \text{sign}(r_j)\right] = \begin{cases} 1/2 & \text{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \text{if } z_j = 1 \end{cases}$$

- Repeat $n = 25t/c^2$ times to construct

$$x_i = I[\text{sign}(r.z) = +] \quad \text{and} \quad y_i = I[\text{sign}(r_j) = +]$$

Hamming Distance Lower Bound

- Reduction from INDEX problem: Alice knows $z \in \{0,1\}^t$ and Bob knows $j \in [t]$. Let's assume $|z| = t/2$ and this is odd.
- Alice and Bob pick $r \in_R \{-1,1\}^t$ using public random bits.
- Alice computes $\text{sign}(r.z)$ and Bob computes $\text{sign}(r_j)$
- *Lemma:* For some constant $c > 0$,

$$\mathbb{P}\left[\text{sign}(r.z) = \text{sign}(r_j)\right] = \begin{cases} 1/2 & \text{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \text{if } z_j = 1 \end{cases}$$

- Repeat $n = 25t/c^2$ times to construct

$$x_i = I[\text{sign}(r.z) = +] \quad \text{and} \quad y_i = I[\text{sign}(r_j) = +]$$

- Note that

$$z_j = 0 \Rightarrow \mathbb{E}\left[\Delta(x,y)\right] = n/2$$
$$z_j = 1 \Rightarrow \mathbb{E}\left[\Delta(x,y)\right] = n/2 - 5\sqrt{n}$$

and by Chernoff bounds $\mathbb{P}\left[|\Delta(x,y) - \mathbb{E}\left[\Delta(x,y)\right]| \geq 2\sqrt{n}\right] < 1/10$.

Hamming Distance Lower Bound

- Reduction from INDEX problem: Alice knows $z \in \{0,1\}^t$ and Bob knows $j \in [t]$. Let's assume $|z| = t/2$ and this is odd.
- Alice and Bob pick $r \in_R \{-1,1\}^t$ using public random bits.
- Alice computes $\text{sign}(r.z)$ and Bob computes $\text{sign}(r_j)$
- *Lemma:* For some constant $c > 0$,

$$\mathbb{P}\left[\text{sign}(r.z) = \text{sign}(r_j)\right] = \begin{cases} 1/2 & \text{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \text{if } z_j = 1 \end{cases}$$

- Repeat $n = 25t/c^2$ times to construct

$$x_i = I[\text{sign}(r.z) = +] \quad \text{and} \quad y_i = I[\text{sign}(r_j) = +]$$

- Note that

$$z_j = 0 \Rightarrow \mathbb{E}\left[\Delta(x,y)\right] = n/2$$
$$z_j = 1 \Rightarrow \mathbb{E}\left[\Delta(x,y)\right] = n/2 - 5\sqrt{n}$$

  and by Chernoff bounds $\mathbb{P}\left[|\Delta(x,y) - \mathbb{E}\left[\Delta(x,y)\right]| \geq 2\sqrt{n}\right] < 1/10$.
- Hence, a $\pm\sqrt{n}$ approx. of $\Delta(x,y)$ determines z_j with prob. $> 9/10$.

Proof of Lemma

Claim

Let A be the event $A = \{\text{sign}(r.z) = r_j\}$. For some constant $c > 0$,

$$\mathbb{P}[A] = \begin{cases} 1/2 & \text{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \text{if } z_j = 1 \end{cases}$$

Proof of Lemma

Claim

Let A be the event $A = \{\mathrm{sign}(r.z) = r_j\}$. For some constant $c > 0$,

$$\mathbb{P}[A] = \begin{cases} 1/2 & \text{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \text{if } z_j = 1 \end{cases}$$

- If $z_j = 0$: $\mathrm{sign}(r.z)$ and r_j are independent. So $\mathbb{P}[A] = 1/2$.

Proof of Lemma

Claim

Let A be the event $A = \{\text{sign}(r.z) = r_j\}$. For some constant $c > 0$,

$$\mathbb{P}[A] = \begin{cases} 1/2 & \text{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \text{if } z_j = 1 \end{cases}$$

- If $z_j = 0$: $\text{sign}(r.z)$ and r_j are independent. So $\mathbb{P}[A] = 1/2$.
- If $z_j = 1$: Let $s = r.z - r_j$ which is the sum of an even number ($\ell = t/2 - 1$) of independent $\{-1, 1\}$ values. Then,

Proof of Lemma

Claim

Let A be the event $A = \{\text{sign}(r.z) = r_j\}$. For some constant $c > 0$,

$$\mathbb{P}[A] = \begin{cases} 1/2 & \textit{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \textit{if } z_j = 1 \end{cases}$$

- If $z_j = 0$: $\text{sign}(r.z)$ and r_j are independent. So $\mathbb{P}[A] = 1/2$.
- If $z_j = 1$: Let $s = r.z - r_j$ which is the sum of an even number ($\ell = t/2 - 1$) of independent $\{-1, 1\}$ values. Then,
  - $\mathbb{P}[A] = \mathbb{P}[A|s = 0]\,\mathbb{P}[s = 0] + \mathbb{P}[A|s \neq 0]\,\mathbb{P}[s \neq 0]$

Proof of Lemma

Claim

Let A be the event $A = \{\text{sign}(r.z) = r_j\}$. For some constant $c > 0$,

$$\mathbb{P}[A] = \begin{cases} 1/2 & \text{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \text{if } z_j = 1 \end{cases}$$

- If $z_j = 0$: $\text{sign}(r.z)$ and r_j are independent. So $\mathbb{P}[A] = 1/2$.
- If $z_j = 1$: Let $s = r.z - r_j$ which is the sum of an even number ($\ell = t/2 - 1$) of independent $\{-1, 1\}$ values. Then,
  - $\mathbb{P}[A] = \mathbb{P}[A|s = 0]\,\mathbb{P}[s = 0] + \mathbb{P}[A|s \neq 0]\,\mathbb{P}[s \neq 0]$
  - $\mathbb{P}[A|s = 0] = 1$ since $s = 0 \Rightarrow r.z = r_j \Rightarrow A$

Proof of Lemma

Claim

Let A be the event $A = \{\text{sign}(r.z) = r_j\}$. For some constant $c > 0$,

$$\mathbb{P}[A] = \begin{cases} 1/2 & \textit{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \textit{if } z_j = 1 \end{cases}$$

- If $z_j = 0$: $\text{sign}(r.z)$ and r_j are independent. So $\mathbb{P}[A] = 1/2$.
- If $z_j = 1$: Let $s = r.z - r_j$ which is the sum of an even number ($\ell = t/2 - 1$) of independent $\{-1, 1\}$ values. Then,
  - $\mathbb{P}[A] = \mathbb{P}[A|s = 0]\,\mathbb{P}[s = 0] + \mathbb{P}[A|s \neq 0]\,\mathbb{P}[s \neq 0]$
  - $\mathbb{P}[A|s = 0] = 1$ since $s = 0 \Rightarrow r.z = r_j \Rightarrow A$
  - $\mathbb{P}[A|s \neq 0] = 1/2$ since $s \neq 0 \Rightarrow s = \{\ldots, -4, -2, 2, 4, \ldots\}$. Hence, $\text{sign}(r.z) = \text{sign}(s)$ which is independent of r_j.

Proof of Lemma

Claim

Let A be the event $A = \{\mathrm{sign}(r.z) = r_j\}$. For some constant $c > 0$,

$$\mathbb{P}[A] = \begin{cases} 1/2 & \text{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \text{if } z_j = 1 \end{cases}$$

- If $z_j = 0$: $\mathrm{sign}(r.z)$ and r_j are independent. So $\mathbb{P}[A] = 1/2$.
- If $z_j = 1$: Let $s = r.z - r_j$ which is the sum of an even number ($\ell = t/2 - 1$) of independent $\{-1, 1\}$ values. Then,
  - $\mathbb{P}[A] = \mathbb{P}[A|s = 0]\,\mathbb{P}[s = 0] + \mathbb{P}[A|s \neq 0]\,\mathbb{P}[s \neq 0]$
  - $\mathbb{P}[A|s = 0] = 1$ since $s = 0 \Rightarrow r.z = r_j \Rightarrow A$
  - $\mathbb{P}[A|s \neq 0] = 1/2$ since $s \neq 0 \Rightarrow s = \{\ldots, -4, -2, 2, 4, \ldots\}$. Hence, $\mathrm{sign}(r.z) = \mathrm{sign}(s)$ which is independent of r_j.
  - $\mathbb{P}[s = 0] = \binom{\ell}{\ell/2}/2^{\ell} = 2c/\sqrt{t}$ for some constant $c > 0$

Proof of Lemma

Claim

Let A be the event $A = \{\text{sign}(r.z) = r_j\}$. For some constant $c > 0$,

$$\mathbb{P}[A] = \begin{cases} 1/2 & \text{if } z_j = 0 \\ 1/2 + c/\sqrt{t} & \text{if } z_j = 1 \end{cases}$$

- If $z_j = 0$: $\text{sign}(r.z)$ and r_j are independent. So $\mathbb{P}[A] = 1/2$.
- If $z_j = 1$: Let $s = r.z - r_j$ which is the sum of an even number ($\ell = t/2 - 1$) of independent $\{-1, 1\}$ values. Then,
  - $\mathbb{P}[A] = \mathbb{P}[A|s = 0]\,\mathbb{P}[s = 0] + \mathbb{P}[A|s \neq 0]\,\mathbb{P}[s \neq 0]$
  - $\mathbb{P}[A|s = 0] = 1$ since $s = 0 \Rightarrow r.z = r_j \Rightarrow A$
  - $\mathbb{P}[A|s \neq 0] = 1/2$ since $s \neq 0 \Rightarrow s = \{\ldots, -4, -2, 2, 4, \ldots\}$. Hence, $\text{sign}(r.z) = \text{sign}(s)$ which is independent of r_j.
  - $\mathbb{P}[s = 0] = \binom{\ell}{\ell/2}/2^\ell = 2c/\sqrt{t}$ for some constant $c > 0$
- So $\mathbb{P}[A] = \mathbb{P}[s = 0] + \frac{\mathbb{P}[s \neq 0]}{2} = \frac{1}{2} + \frac{\mathbb{P}[s=0]}{2} = \frac{1}{2} + \frac{c}{\sqrt{t}}$.