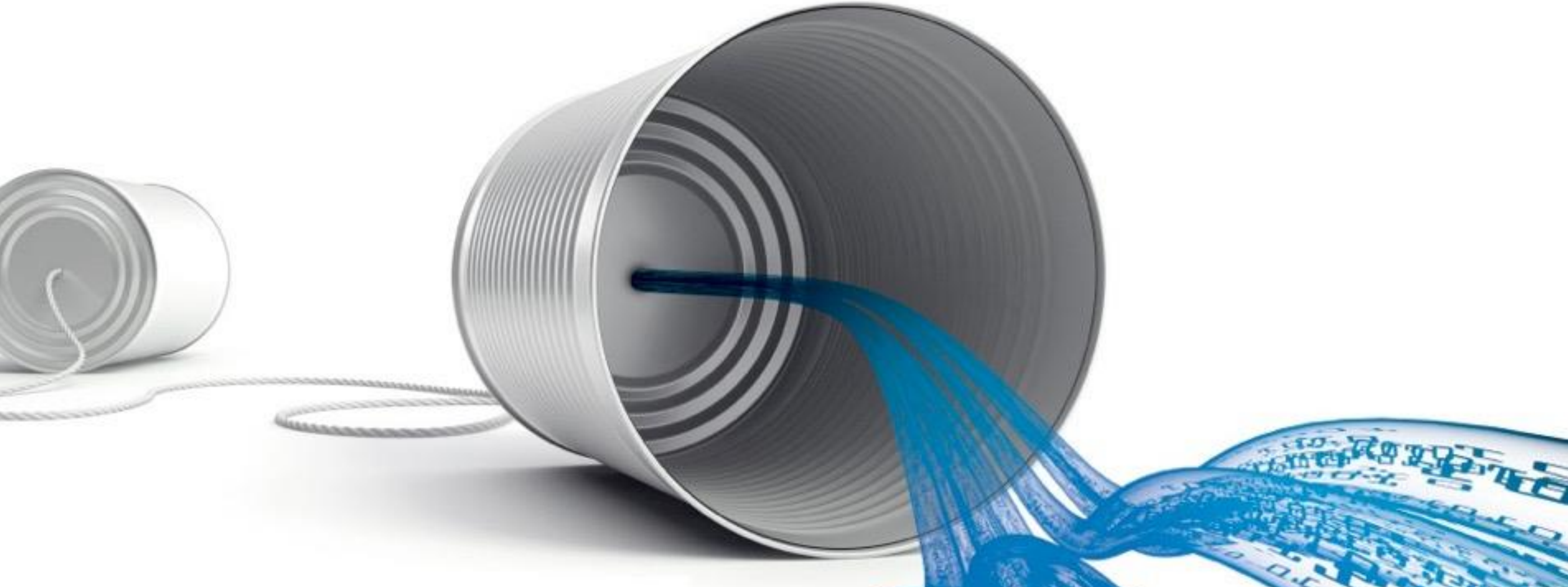
15-251
Great Theoretical Ideas in Computer Science

Lecture 25:
Communication Complexity



Cool Things About Communication Complexity

Many useful applications:

distributed computing, machine learning, proof complexity, quantum computation, pseudorandom generators, data structures, game theory,…

The setting is simple and neat.

Beautiful mathematics

combinatorics, information theory, algebra, analysis, …

One of few approaches to prove unconditional lower bounds about computational problems.

Motivating Example 1: Checking Equality

01001010101110101 $\stackrel{?}{=}$ 01001010100110101

$\longleftarrow$ n bits $\longrightarrow$ $\longleftarrow$ n bits $\longrightarrow$

How many bits need to be communicated?

Naively: n **Actually:** n

What if we allow 0.0000000001% probability of error?

Naively: $\Omega(n)$ **Actually:** $O(\log n)$

Defining the model a bit more formally

2 Player Model of Communication Complexity

$$F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$$

known to both players

(Alice)

$x \in \{0,1\}^n$

(Bob)

$y \in \{0,1\}^n$

Goal: Compute $F(x,y)$ (both players should know the value)

How: Sending bits back and forth according to a protocol.

Resource: Number of communicated bits.

(We assume players have unlimited computational power individually.)

Poll 1

$x, y \in \{0,1\}^n$, $PAR(x,y) =$ parity of the sum of all the bits.

(i.e. it's 1 if the parity is odd, 0 otherwise.)

How many bits do the players need to communicate?
Choose the tightest bound.

$O(1)$

$O(\log n)$

$O(\log^2 n)$

$O(\sqrt{n})$

$O(n/\log n)$

$O(n)$

Poll 1 Answer

$x, y \in \{0,1\}^n$, $PAR(x,y) =$ **parity of the sum of all the bits.**

(i.e. it's 1 if the parity is odd, 0 otherwise.)

How many bits do the players need to communicate? Choose the tightest bound.

- $O(1)$ ✓
- $O(\log n)$
- $O(\log^2 n)$
- $O(\sqrt{n})$
- $O(n/\log n)$
- $O(n)$

Poll 1 Answer

$x, y \in \{0,1\}^n$, $PAR(x,y) =$ parity of the sum of all the bits.

(i.e. it's 1 if the parity is odd, 0 otherwise.)

How many bits do the players need to communicate?
Choose the tightest bound.

Once Bob knows the parity of x, he can compute $PAR(x,y)$.

- Alice sends $PAR(x)$ to Bob. 1 bit
- Bob computes $PAR(x,y)$ and sends it to Alice. 1 bit

2 bits in total

2 Player Model of Communication Complexity

Goal: Compute $F(x, y)$. (both players should know the value)

How: Sending bits back and forth according to a protocol.

Resource: Number of communicated bits.

A protocol P is the "strategy" players use to communicate.

It determines what bits the players send in each round.

$P(x, y)$ denotes the output of P.

2 Player Model of Communication Complexity

Goal: Compute $F(x, y)$. (both players should know the value)

How: Sending bits back and forth according to a protocol.

Resource: Number of communicated bits.

A (deterministic) protocol P computes F if

$$\forall (x, y) \in \{0,1\}^n \times \{0,1\}^n, \qquad P(x, y) = F(x, y)$$

Analogous to: algorithm (TM) — decision problem

$$\forall x \in \Sigma^* \quad A(x) = F(x)$$

2 Player Model of Communication Complexity

Goal: Compute $F(x, y)$. (both players should know the value)

How: Sending bits back and forth according to a protocol.

Resource: Number of communicated bits.

A randomized protocol P computes F with ϵ error if

$$\forall (x, y) \in \{0,1\}^n \times \{0,1\}^n, \quad \Pr[P(x, y) \neq F(x, y)] \leq \epsilon$$

Analogous to: Monte Carlo algorithms

2 Player Model of Communication Complexity

Goal: Compute $F(x, y)$ (both players should know the value)

How: Sending bits back and forth according to a protocol.

Resource: Number of communicated bits.

$$\text{cost}(P) = \max_{(x,y)} \# \text{ bits } P \text{ communicates for } (x, y)$$

if P is randomized, you take max over the random choices it makes.

Deterministic communication complexity

$\mathbf{D}(F)$ = min cost of a (deterministic) protocol computing F.

Randomized communication complexity

$\mathbf{R}^{\epsilon}(F)$ = min cost of a randomized protocol computing F with ϵ error.

2 Player Model of Communication Complexity

Goal: Compute $F(x, y)$ (both players should know the value)

How: Sending bits back and forth according to a protocol.

Resource: Number of communicated bits.

$\text{cost}(P) = \max_{(x}$ # bits P communicates for (x, y)

...xe max
...t makes.

Deterministic

$\mathbf{D}(F) =$ mi... ...uting F.

Randomized c...

$\mathbf{R}^{\epsilon}(F) =$ min ... ...ting F
with ϵ error.

We usually fix ϵ to some constant.

e.g. $\epsilon = 1/3$

We can always boost the success probability if we want.

What is considered hard or easy?

$$F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$$

$$0 \leq \quad \mathbf{R}_2^{\epsilon}(F) \leq \mathbf{D}_2(F) \quad \leq n+1$$

$$c \qquad \log^c(n) \qquad\qquad n^\delta \qquad \delta n$$

Example

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

$$\mathbf{D}(EQ) =$$

Poll 2

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

What is $\mathbf{R}^{1/3}(EQ)$?

$O(1)$

$O(\log n)$

$O(\log^2 n)$

$O(\sqrt{n})$

$O(n/\log n)$

$O(n)$

Poll 2 Answer

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

What is $\mathbf{R}^{1/3}(EQ)$?

- $O(1)$
- $O(\log n)$ ✓
- $O(\log^2 n)$
- $O(\sqrt{n})$
- $O(n/\log n)$
- $O(n)$

Poll 3

$MAJ(x, y) =$ 1 iff majority of all the bits in x and y are set to 1.

What is $\mathbf{D}(MAJ)$? Choose the tightest bound.

$O(1)$

$O(\log n)$

$O(\log^2 n)$

$O(\sqrt{n})$

$O(n/\log n)$

$O(n)$

Poll 3 Answer

$MAJ(x, y) =$ 1 iff majority of all the bits in x and y are set to 1.

What is $\mathbf{D}(MAJ)$? Choose the tightest bound.

- $O(1)$
- $O(\log n)$ ✓
- $O(\log^2 n)$
- $O(\sqrt{n})$
- $O(n/\log n)$
- $O(n)$

Poll 3 Answer

$MAJ(x, y) =$ 1 iff majority of all the bits in x and y are set to 1.

What is $\mathrm{D}(MAJ)$? Choose the tightest bound.

The result can be computed from

$$\sum_{i \in \{1,2,\ldots,n\}} x_i + \sum_{i \in \{1,2,\ldots,n\}} y_i$$

- Alice sends $\sum_i x_i$ to Bob. log n +1 bits
- Bob computes $MAJ(x, y)$ and sends it to Alice. 1 bit

log n + 2 in total

Another Important Example

Disjointness: $DISJ(x,y) = \begin{cases} 0 & \text{if } \exists i : x_i = y_i = 1 \\ 1 & \text{otherwise} \end{cases}$

$$\mathbf{R}^{1/3}(DISJ) = \Omega(n).$$

hard

The plan

1. Efficient randomized communication protocol for checking equality.

2. Several applications of communication complexity.

Efficient randomized communication protocol for checking equality

The Power of Randomization

$$\mathbf{R}^{1/3}(EQ) = O(\log n).$$

Alice gets $x \in \{0,1\}^n$, Bob gets $y \in \{0,1\}^n$.

We treat x and y as numbers: $0 \leq x, y \leq 2^n - 1$.

The Protocol:

- Let p_i be the i'th smallest prime number.

$$p_1 = 2,\ p_2 = 3,\ p_3 = 5,\ p_4 = 7,\ \ldots$$

- Alice picks a random $i \in \{1, 2, \ldots, n^2\}$.
- Alice sends Bob: $i, \quad x \bmod p_i$
- Bob outputs 1 iff $x \bmod p_i = y \bmod p_i$. ($x \equiv_{p_i} y$)

The Power of Randomization

$$\mathbf{R}^{1/3}(EQ) = O(\log n).$$

Correctness:

Want to show: For all (x, y), probability of error is $\leq 1/3$.

For all (x, y) with $x = y$:

$$\Pr[\text{error}] = \Pr_i[x \not\equiv_{p_i} y] = 0.$$

For all (x, y) with $x \neq y$:

$$\Pr[\text{error}] = \Pr_i[x \equiv_{p_i} y] = \Pr_i[p_i \text{ divides } x - y]$$

Claim: $x - y$ has at most n distinct prime factors.

$$\Pr[\text{error}] = \Pr[p_i \text{ is a prime factor of } x - y] \leq \frac{n}{n^2} = \frac{1}{n}.$$

The Power of Randomization

$$\mathbf{R}^{1/3}(EQ) = O(\log n).$$

Cost:

The only communication is:

- Alice sends Bob: i, $x \bmod p_i$

The first number i is such that $i \leq n^2$.

Can represent it using $\sim \log_2 n^2 = 2\log_2 n = O(\log n)$ bits.

The second number $x \bmod p_i$ is at most p_{n^2}.

By the Prime Number Theorem: $p_{n^2} \sim n^2 \log n^2 \leq 2n^3$

Can represent p_{n^2} using at most $\log(2n^3) = O(\log n)$ bits. $\square$

The Power of Randomization

$$\mathbf{R}^{1/3}(EQ) = O(\log n).$$

Alice gets $x \in \{0,1\}^n$, Bob gets $y \in \{0,1\}^n$.

We treat x and y as numbers: $0 \leq x, y \leq 2^n - 1.$

<u>The Protocol:</u>

- Let p_i be the i'th smallest prime number.

  $$p_1 = 2,\ p_2 = 3,\ p_3 = 5,\ p_4 = 7,\ \ldots$$

- Alice picks a random $i \in \{1, 2, \ldots, n^2\}$.
- Alice sends Bob: $i, \quad x \bmod p_i$
- Bob outputs 1 iff $x \bmod p_i = y \bmod p_i$. ($x \equiv_{p_i} y$)

The plan

1. Efficient randomized communication protocol for checking equality.

2. Several applications of communication complexity.

Applications of Communication Complexity

- circuit complexity
- time/space tradeoffs for Turing Machines
- VLSI chips
- machine learning
- game theory
- data structures
- proof complexity
- pseudorandom generators
- pseudorandomness
- branching programs
- data streaming algorithms
- quantum computation
- lower bounds for polytopes representing NP-complete problems

How Communication Complexity Comes In

Setting: Solve some task while minimizing some resource.

e.g. find a fast algorithm, design a small circuit, find a short proof of a theorem, …

Goal: Prove lower bounds on the resource needed.

Sometimes:

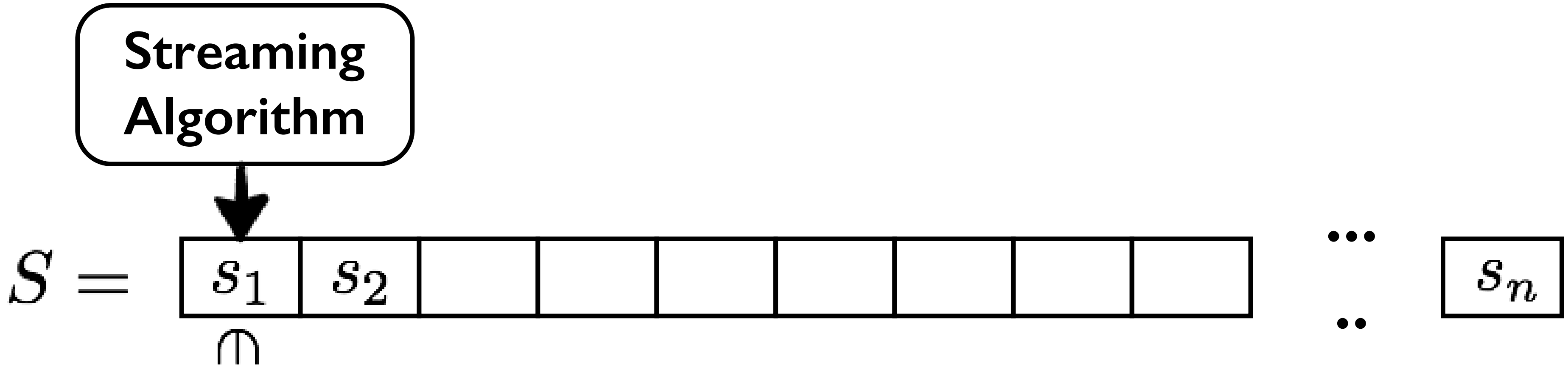
efficient solution to our problem →

efficient communication protocol for a certain function.

i.e. no efficient protocol for the function →
no efficient solution to our problem.

Lower bounds for data streaming algorithms

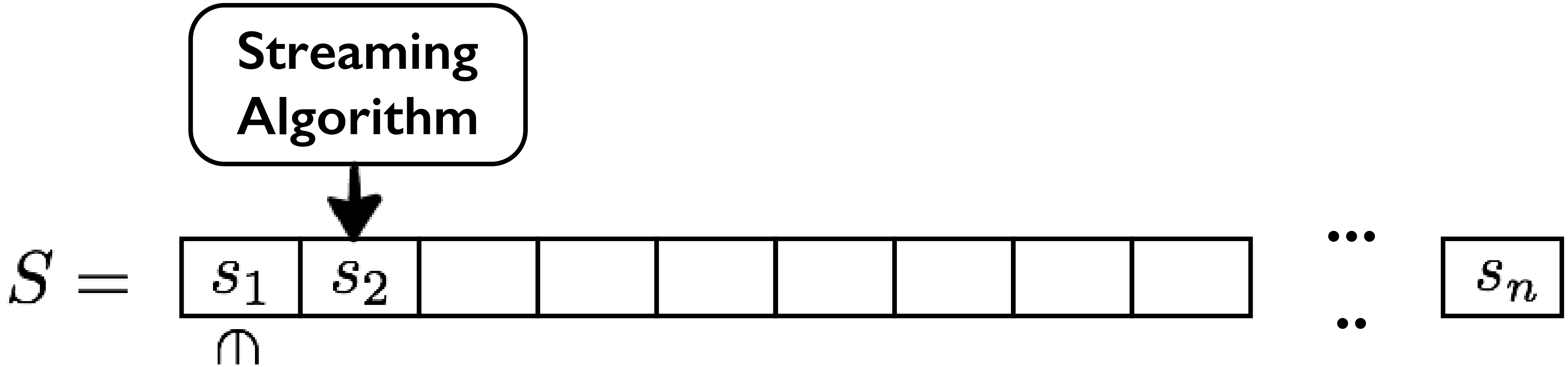
Data Streaming Algorithms



$$[n] = \{1, 2, \ldots, n\}$$

$$S \in [n]^n$$

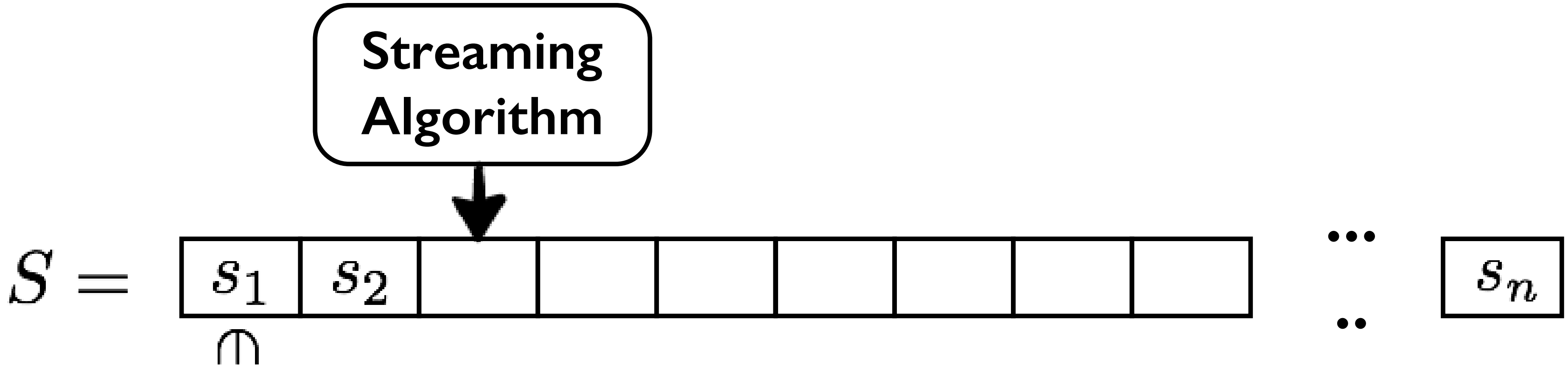
Data Streaming Algorithms



$$[n] = \{1, 2, \ldots, n\}$$

$$S \in [n]^n$$

Data Streaming Algorithms

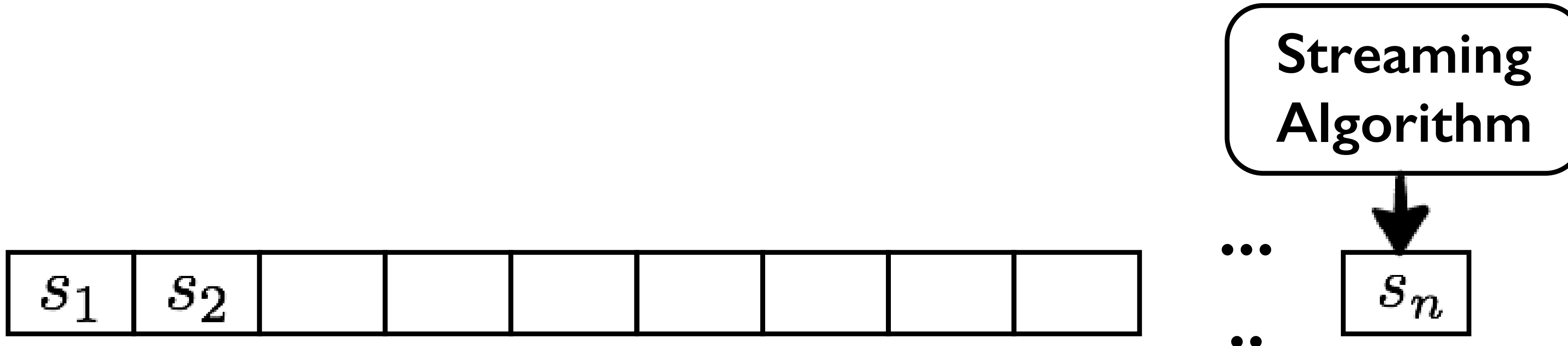


$$[n] = \{1, 2, \ldots, n\}$$

$$S \in [n]^n$$

Data Streaming Algorithms

$S =$



$$\cap\!\!\!|$$

$$[n] = \{1, 2, \ldots, n\}$$

$$S \in [n]^n$$

Fix some function $f : [n]^n \to \mathbb{Z}.$

e.g. $f(S) = \#\ \text{most frequent symbol in } S$

Goal: **On input** S **, compute (or approximate)** $f(S)$ **while minimizing space usage.**

Lower Bounds via Communication Complexity

$$f(S) = \#\ \text{most frequent symbol in}\ S$$

Space efficient streaming algorithm computing f $\longrightarrow$

communication efficient protocol computing $DISJ$**.**

Disjointness: $DISJ(S_x, S_y) = \begin{cases} 1 & S_x \cap S_y = \emptyset \\ 0 & \text{otherwise} \end{cases}$

Lower Bounds via Communication Complexity

$$f(S) = \#\text{ most frequent symbol in } S$$

Space efficient streaming algorithm computing f ⟶ communication efficient protocol computing $DISJ$.

$$S_x = \{2, 4, 5\} \qquad S_y = \{1, 5, 7, 8\}$$

$x =$

0	1	0	1	1	0	0	0
1	*2*	*3*	*4*	*5*	*6*	*7*	*8*

$y =$

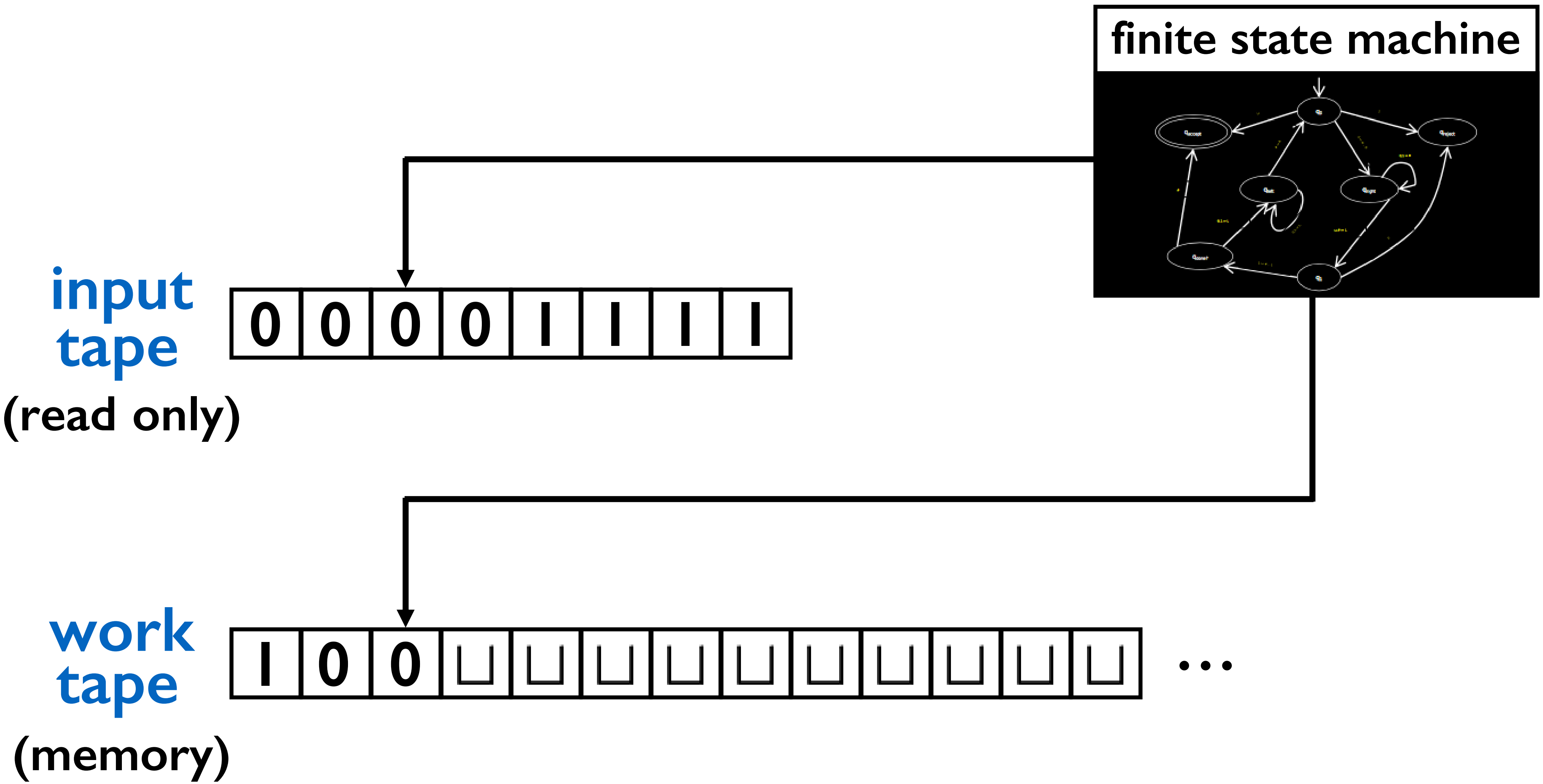
1	0	0	0	1	0	1	1
1	*2*	*3*	*4*	*5*	*6*	*7*	*8*

Protocol: Alice runs streaming algorithm on S_x.
She sends the state and memory contents to Bob.
Bob continues to run the algorithm on S_y.
If $f(S_x \cdot S_y) = 2$, Bob outputs 0, otherwise 1.

Correctness ✓ Cost ✓

Time/space tradeoffs for TMs

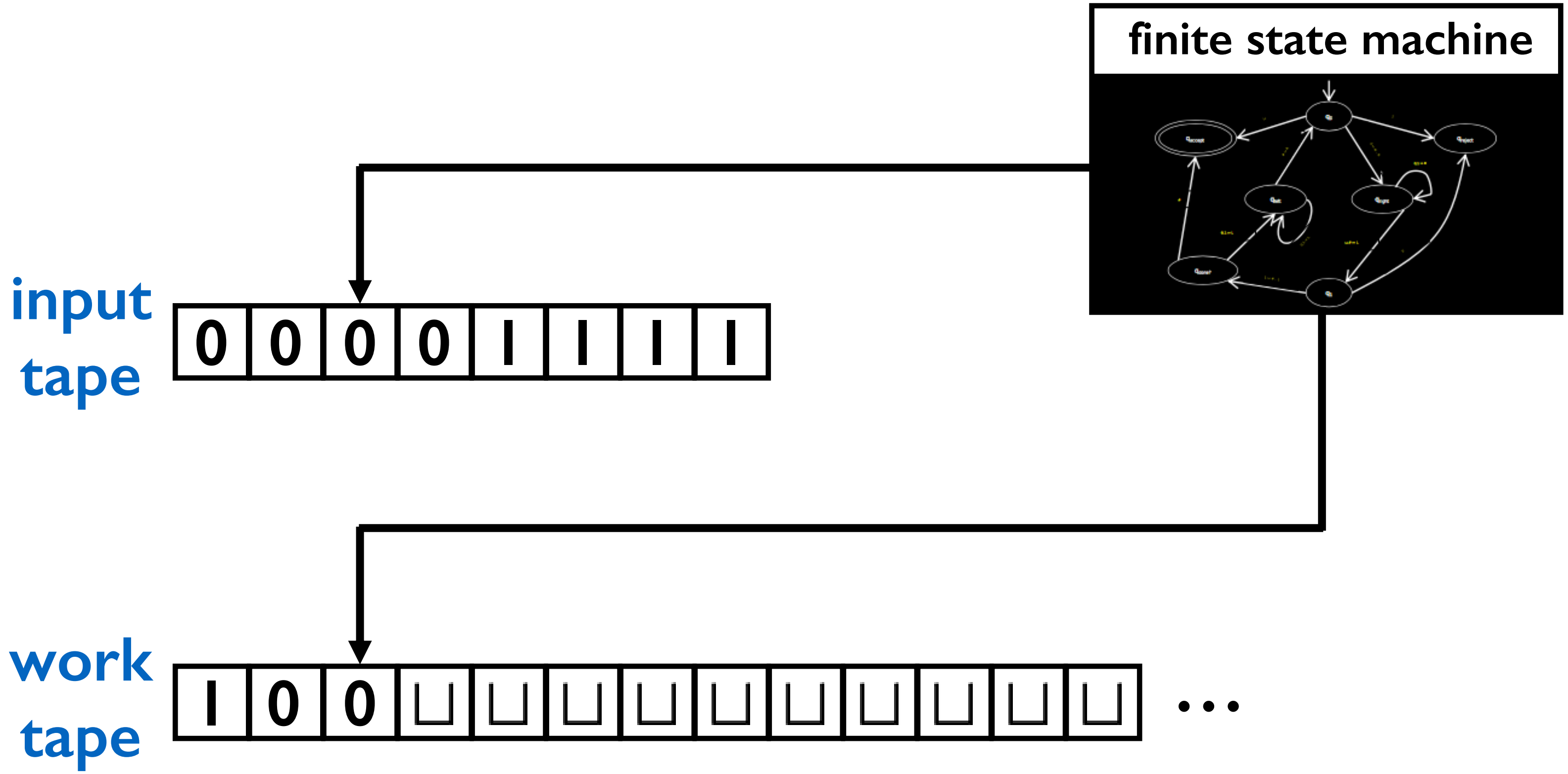
Recall Turing Machines



$T(n)$ **time:** # steps the machine takes

$S(n)$ **space:** # work tape cells the machine uses

An observation



Suppose we both know the input and the TM.

You start running the TM with the input.

You pause after a certain number of steps.

What information do I need to be able to continue the computation from where you left it?

1. current state
2. positions of tape heads
3. contents of work tape

Time/space tradeoff for a simple language

Let $L = \{x\#^{|x|}x : x \in \{0,1\}^*\}$

$000\#\#\#000 \in L$

$1010\#\#\#\#1010 \in L$

$001\#\#\#000 \notin L$

$000\#\#000 \notin L$

Theorem:

If a TM M decides L **in** $T(n)$ **time and** $S(n)$ **space on inputs of size** $3n$, **then** $T(n) \cdot S(n) = \Omega(n^2)$.

Time/space tradeoff for a simple language

Let $L = \{x\#^{|x|}x : x \in \{0,1\}^*\}$

Theorem:

If a TM **M** decides L in $T(n)$ time and $S(n)$ space on inputs of size $3n$, then $T(n) \cdot S(n) = \Omega(n^2)$.

Strategy:

Using **M**, we design a communication protocol for EQ of cost $\leq c\,T(n)S(n)/n$ for some constant c.

We know EQ requires $\geq n$ bits of communication.

$$\implies c\,T(n)S(n)/n \geq n \implies c\,T(n)S(n) \geq n^2$$

Time/space tradeoff for a simple language

Let $L = \{x\#^{|x|}x : x \in \{0,1\}^*\}$. M decides L .

Protocol for EQ :

Given input $x \in \{0,1\}^n$ to Alice, and $y \in \{0,1\}^n$ to Bob.

They want to decide if $x = y$. They will make use of M.

Let $w = x\#^n y$.

They simulate M(w).

If M(w) accepts, they output 1.

If M(w) rejects, they output 0. A correct protocol.

Time/space tradeoff for a simple language

Let $L = \{x\#^{|x|}x : x \in \{0,1\}^*\}$. M decides L .

Protocol for EQ :

Given input $x \in \{0,1\}^n$ to Alice, and $y \in \{0,1\}^n$ to Bob.

They want to decide if $x = y$. They will make use of M.

Let $w = x\#^n y$.

They simulate M(w).

If M(w) accepts, they output 1 .

If M(w) rejects, they output 0.

How do they simulate M?

What is the cost?

A correct protocol.

Time/space tradeoff for a simple language

Let $L = \{x\#^{|x|}x : x \in \{0,1\}^*\}$.   M decides L .

Protocol for EQ :

They simulate M($x\#^n y$).

Alice starts the simulation.

When input tape head reaches a y symbol, she sends

1. current state
2. position of work tape head
3. contents of work tape

Time/space tradeoff for a simple language

Let $L = \{x\#^{|x|}x : x \in \{0,1\}^*\}$.   M decides L .

Protocol for EQ :

They simulate M($x\#^n y$).

Bob continues the simulation.

When input tape head reaches an x symbol, he sends

1. current state
2. position of work tape head
3. contents of work tape

This continues until M halts.

Time/space tradeoff for a simple language

Analysis:

It is clear the protocol is correct. What is the cost?

In each transmission, players send

1. current state $\longrightarrow$ $O(1)$
2. position of work tape head $\longrightarrow$ $O(\log S(n))$
3. contents of work tape $\longrightarrow$ + $O(S(n))$

$$O(S(n))$$

What is the number of transmissions?

For each transmission, M takes $\geq n$ steps.

So $T(n) \geq (\#\ \text{transmissions}) \cdot n$.

$\Longrightarrow\ \#\ \text{transmissions} \leq T(n)/n.$

Total cost: $O(S(n)T(n)/n)$. $\square$

Time/space tradeoff for a simple language

Let $L = \{x\#^{|x|}x : x \in \{0,1\}^*\}$

Theorem:

If a TM **M** decides L in $T(n)$ time and $S(n)$ space on inputs of size $3n$, then $T(n) \cdot S(n) = \Omega(n^2)$.

Strategy:

Using **M**, we design a communication protocol for EQ of cost $\leq c\,T(n)S(n)/n$ for some constant c.

We know EQ requires $\geq n$ bits of communication.

$$\implies c\,T(n)S(n)/n \geq n \implies c\,T(n)S(n) \geq n^2$$