



Proving Lower bounds using Information Theory

Frédéric Magniez
CNRS, Univ Paris Diderot

Paris, July 2013

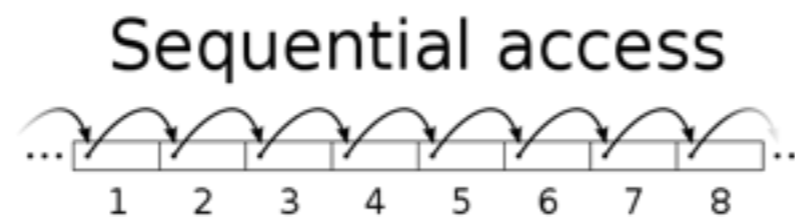
•Massive data sets

- Examples:
 - Network monitoring
 - Genome decoding
 - Web databases
- Impossible to store full input in memory
 - ▶ Random access model is not realistic



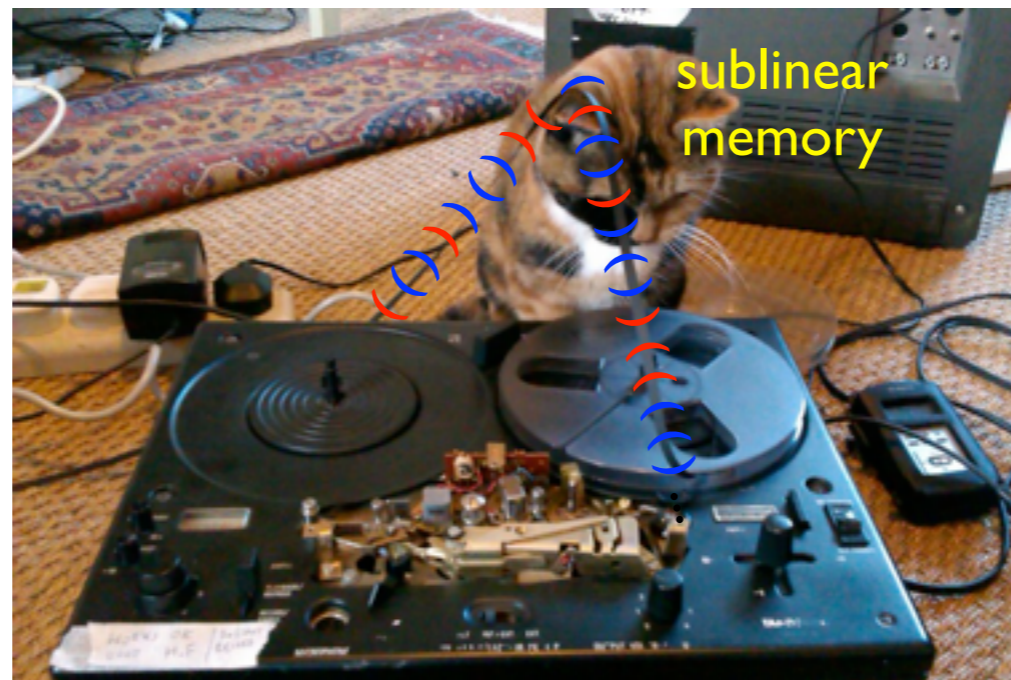
•Data streams

- Sequential access to input in **one pass**



- Prove lower bounds for streaming algorithms

1. Check if all elements are pairwise distinct
2. Decide if there is an element with frequency $> 50\%$
3. Check if a document is well-parenthesized

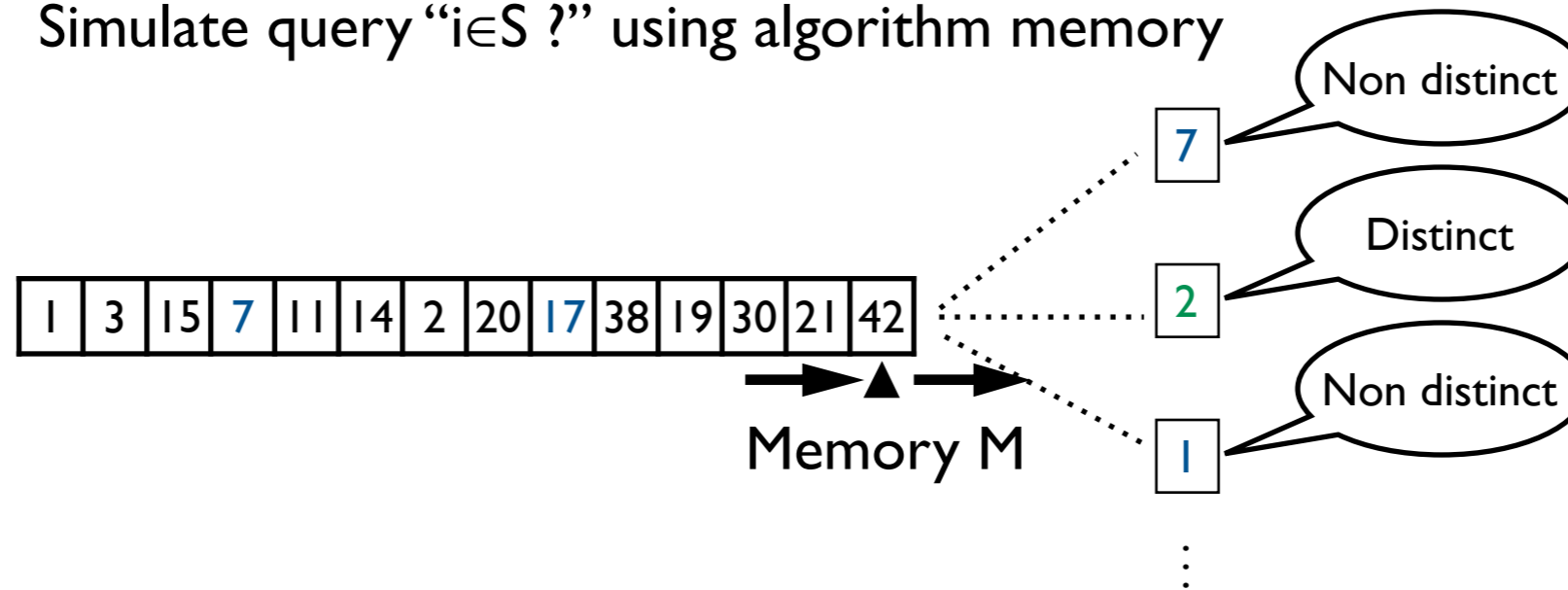


- Theorem

- For 1-pass randomized streaming algorithms
 - Problems 1 & 2 require memory $|\text{input}|$
 - Problem 3 requires memory $\sqrt{|\text{input}|}$

• Adversary argument

- Run streaming algorithm on a long sequence S of pairwise distinct integers
- Simulate query “ $i \in S$?” using algorithm memory



- Therefore memory should be of same size than S

• Generalization

- Randomized algorithms
- Other problems

Decide if there is an element with frequency $> 50\%$

Check if a document is well-parenthesized

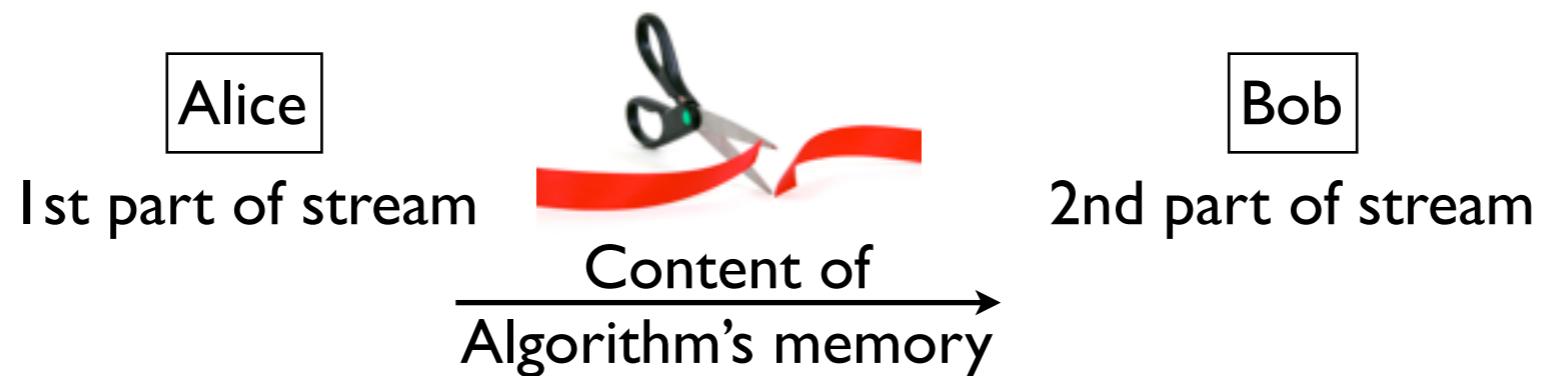
•From algorithm to protocol

- Build a communication protocol

Cut the stream in 2 or more pieces, one for each player

Simulate algorithm on distributed stream

Message size \leftrightarrow Memory space



•From protocol to information theory

- Lower bound the information in the message of some player

Correctness of protocol

→ Leakage of information by some player

→ There is a large message

• Definition

- Setting

Random variables (distributed according to fixed distributions)

Might be dependent

- Intuition

$H(X)$ = Amount of randomness in X

- Definition (from coding theory)

$H(X)$ = Average length of the best encoding for X

• Examples

- Uniform distribution on n bits: $H(X) = n$

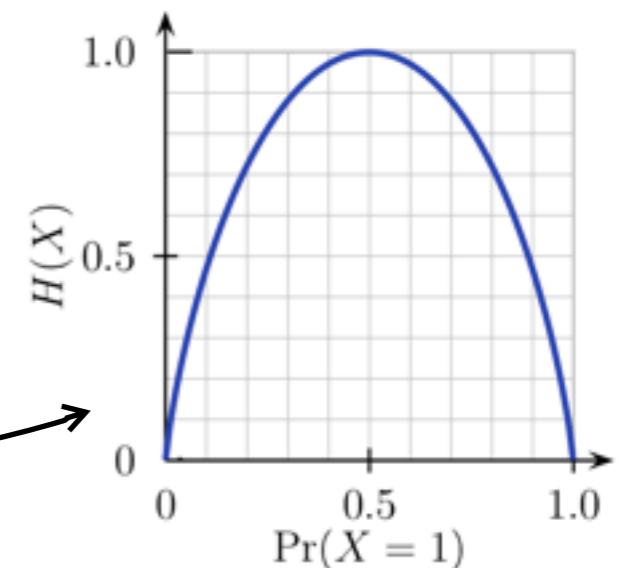
- Dirac distribution: $H(X) = 0$

- Bernoulli distribution

• First principles

- $H(X) \leq H(XY) \leq H(X) + H(Y)$

- $H(X) \leq n$ when X has n bits



• Definition

- Intuition

$H(X|Y)$ = Amount of randomness in X while Y is known

- Definition

$H(X|Y)$ = Average length of the best encoding for X , given Y

$H(X|Y) = E_y H(X|Y=y)$ where y is distributed according to Y

• Chain rule

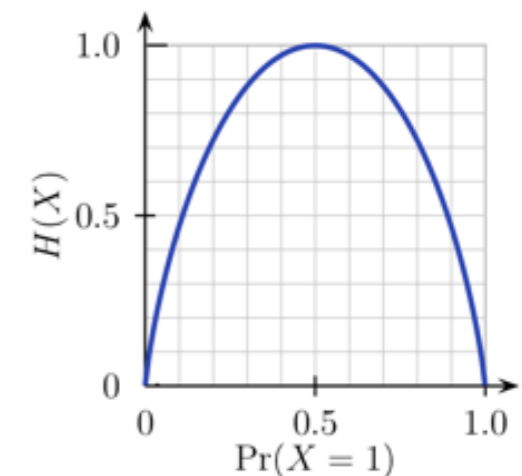
- $H(XY) = H(Y) + H(X|Y)$ $H(X_1X_2\dots X_n) = \sum_k H(X_k|X_1\dots X_{k-1})$

• First principles

- $H(X|Y) = 0$ when $X=Y$
- $H(X|Y) = H(X)$ when X & Y are independent
- $H(X|Y) \leq H(X)$

• Fano's inequality

- $H(X|Y) \leq H(\epsilon)$ when $\Pr(X \neq Y) \leq \epsilon$



• Definition

- Intuition

$I(X:Y)$ = Amount of information that Y carries on X

- Definition

$$I(X:Y) = H(X) - H(X|Y)$$

• Examples

- $I(X:Y) = H(X)$ when $X=Y$
- $I(X:Y) = 0$ when X&Y are independent

• First principles

- $I(X:Y) \leq H(X)$
- $I(X:Y) = H(X) + H(Y) - H(XY) = I(Y:X)$
- $I(X:YZ) \leq I(X:Y)$
- $I(X_1 X_2 \dots X_n : Y) = \sum_k I(X_k : Y | X_1 \dots X_{k-1})$

• Warning

- $I(X:Y|Z) \geq I(X:Y)$ when X&Z are independent
- $I(X:Y|Z) \leq I(X:Y)$ when $X=Z$

•Problem

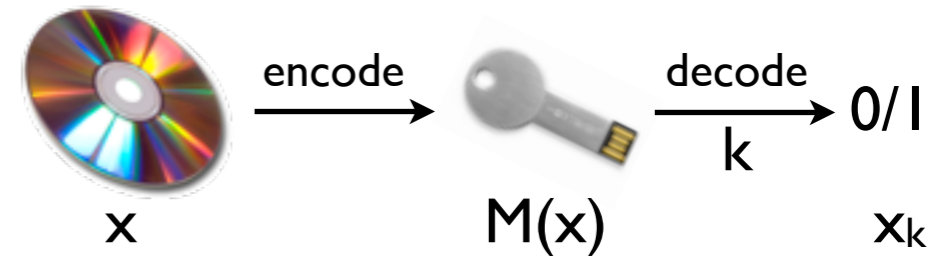
- Input: n -bit string x
- Goal: Encode x into $M(x)$ such that

Any bit of x can be decoded from $M(x)$ with probability $\geq 1-\epsilon$

For all x, k : $\Pr_{\text{Enc\&Dec}}(\text{Decoding}(M(x), k) = x_k) \geq 1-\epsilon$

→ **Worst-case error**

While minimizing the size of $M(x)$ (# of bits)



•Theorem

- At least $(1-H(\epsilon))n$ of bits are needed

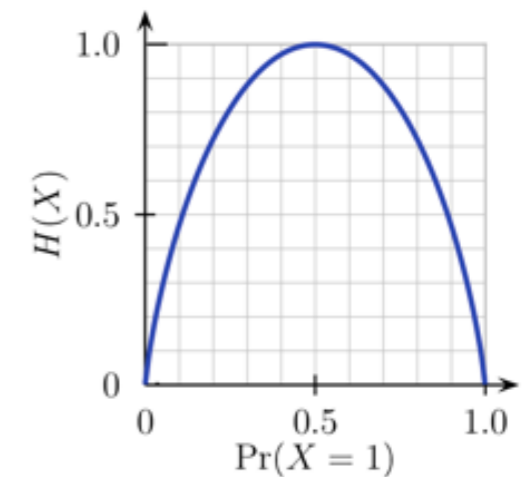
•Proof

- X : uniform distribution for x , M : resp. distribution for $M(x)$
- $I(M:X) \leq H(M) \leq$ size of the encoding
- **$I(X:M) \geq (1-H(\epsilon))n$**

$I(X:M) = \sum_k I(X_k:M|X_1 \dots X_{k-1}) \geq \sum_k I(X_k:M)$ chain rule & (X_k) independent

$H(X_k) = 1$ and $H(X_k|M) \leq H(\epsilon)$ Fano's inequality

- $n(1-H(\epsilon)) \leq I(X:M) = I(M:X) \leq$ size of the encoding

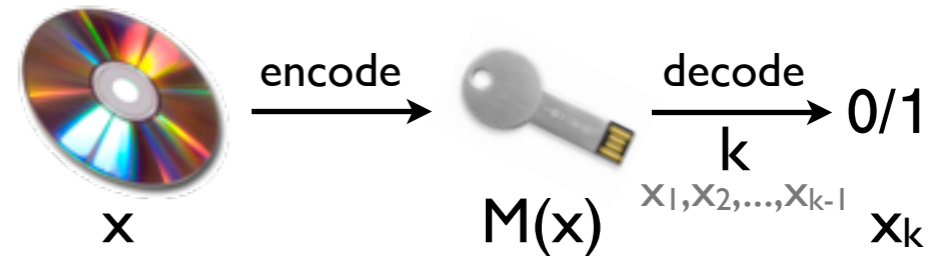


•Extension

- Random input: uniform n-bit string x
- Goal: Encode x into $M(x)$ such that

$$\Pr_{x,k, \text{Enc\&Dec}}(\text{Decoding}(M(x), k) = x_k) \geq 1 - \epsilon$$

→ **Distributed error** (over x)

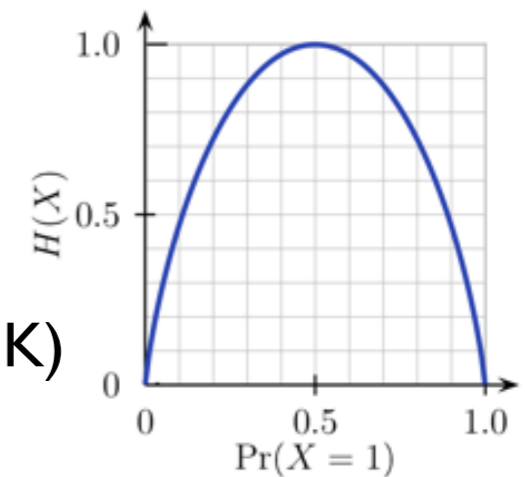


•Theorem

- At least $(1 - H(\epsilon))n$ of bits are needed

•Proof

- $I(X:M) = \sum_k I(X_k:M | X_1 \dots X_{k-1}) \leq \sum_k I(X_k:M) = n I(X_K:M | K)$
where K is uniformly distributed in $[n]$



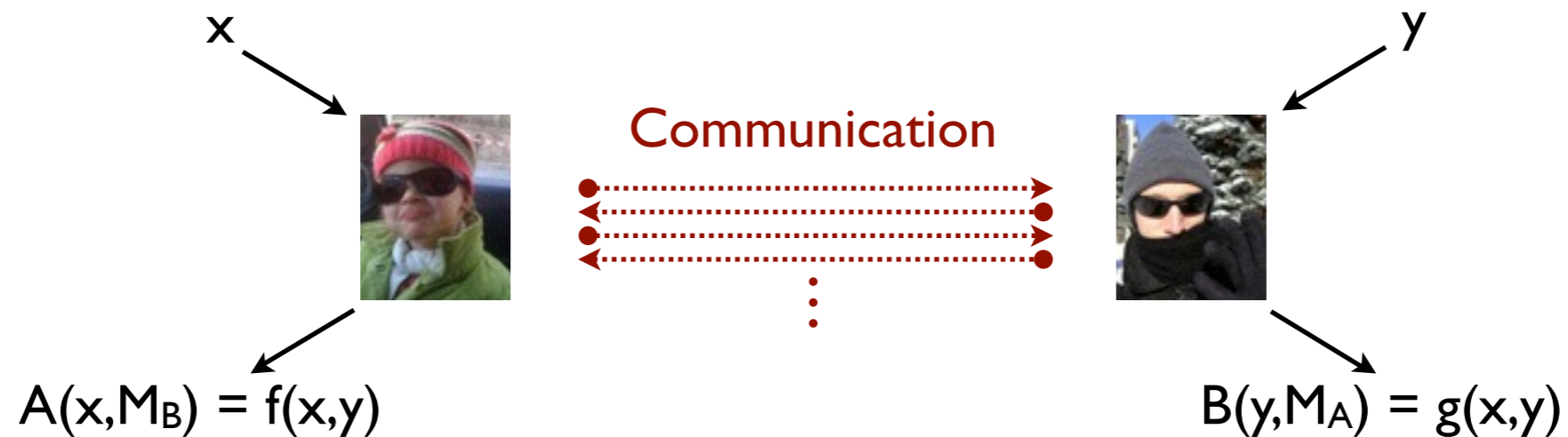
•Remarks

- Same lower bounds if $x_1 x_2 \dots x_{k-1}$ are given to the decoder
- For distributed error, best encoding/decoding is deterministic
- Yao's principle

$$\begin{aligned} &\text{Encoding size of (RAC with worst-case error)} \\ &\geq \text{Encoding size of (det-RAC with distributed error)} \end{aligned}$$

•2-player game

- Alice & Bob wants to solve a common task based on their respective inputs while minimizing # of exchanged bits



•Powerful tool

- For stressing limitations of **various** computational models
 - Complexity theory (Circuits)
 - VLSI design
 - Turing machines
 - Streaming algorithms

• Problem



- One-way communication
 - Only Alice sends a message M_A
- Goal
 - Bob wants to learn one bit of x

• Theorem

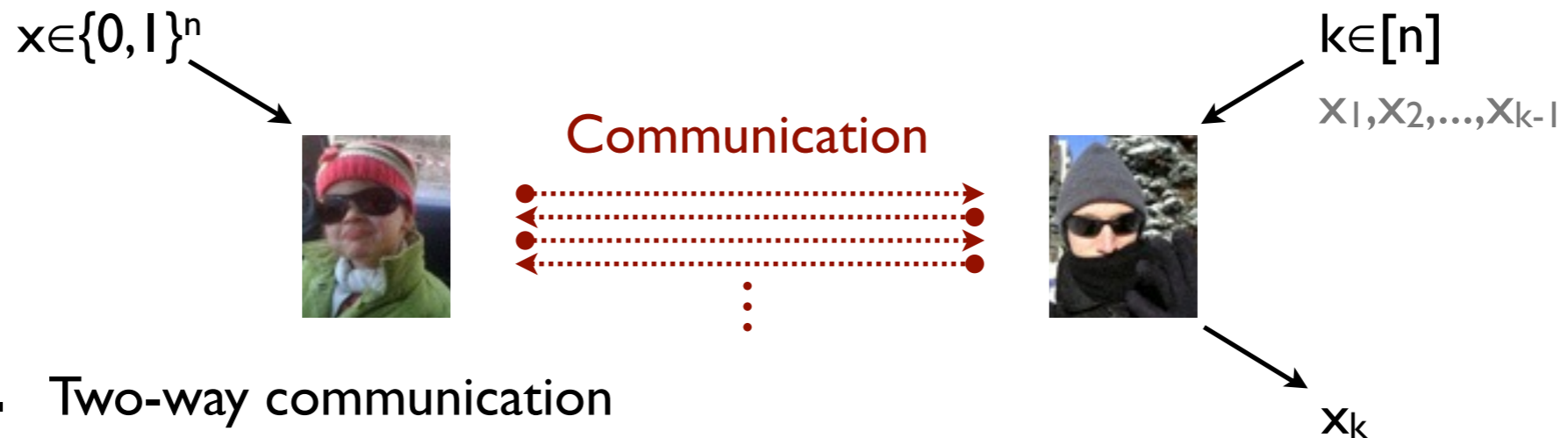
- M_A has size at least $(1-H(\epsilon))n$ when error probability $\leq \epsilon$

• Preuve

- M_A is a random access code for x

$$I(X:M_A|K) \geq (1-H(\epsilon))n$$
 when X, K are uniformly distributed

• Problem



- Two-way communication

• Theorem

- If M_B has size b then M_A has size $\Omega(n/2^b)$ (when error probability $\leq 1/3$)

• Proof by round elimination

- Special case: one round Alice \rightarrow Bob \rightarrow Alice, and Alice outputs x_i
- Alice guesses M_B by flipping b uniform random bits
 Error probability: $\varepsilon = 1 - \Omega(1/2^b) \rightarrow H(\varepsilon) = H(1-\varepsilon) = \Omega(1/2^b)$
 One-way protocol $\rightarrow M_A$ has size $\geq (1-H(\varepsilon))n = \Omega(n/2^b)$

• Theorem

- If $I(K:M_B|X) \leq b$ then $I(X:M_A|K) = \Omega(n/2^b)$ (when error probability $\leq 1/3$)

• Problem

$$x^1, x^2, \dots, x^t \in \{0, 1\}^n$$



Communication



$$k^1, k^2, \dots, k^t \in [n]$$

$$x^1[k^1], x^2[k^2], \dots, x^t[k^t]$$

- Two-way communication

• Theorem

- If M_B has size $t \times b$ then M_A has size $\Omega(t \times n / 2^b)$ (when error probability $\leq 1/3$)

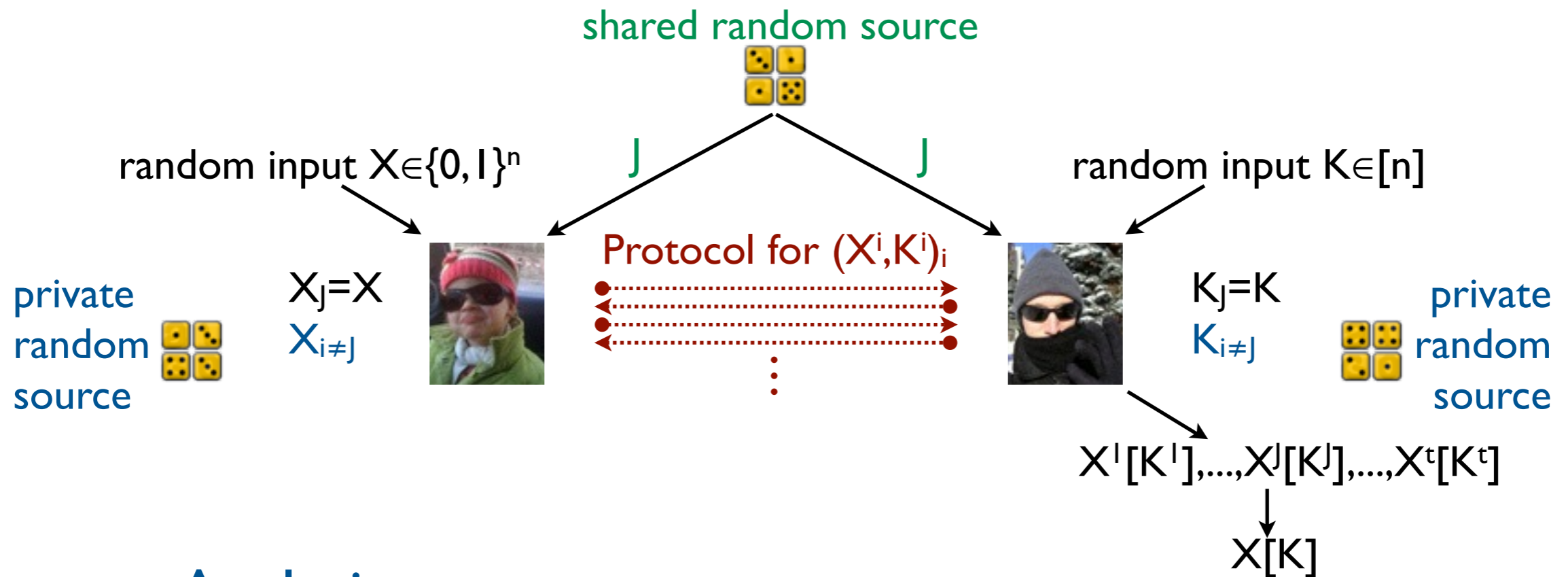
• Proof

- Using the protocol, A&B can build a protocol for INDEX such that

$$I(X: M_A | K) \leq |M_A|/t \text{ and } I(K: M_B | X) \leq |M_B|/t \leq b$$
- Single instance lower bound

$$\rightarrow |M_A|/t = \Omega(n/2^b)$$

•Reduction



•Analysis

- $|M_A| \geq I((X^i)_i; M_A | (K^i)_i) = \sum_j I(X^j; M_A | X^1 \dots X^{j-1} (K^i)_i)$
 $\geq \sum_j I(X^j; M_A | K^j) = t \times I(X^J; M_A | K^J, J) = t \times I(X; M_A | K, J)$
 where J is the shared randomness
- $|M_B| \geq t \times I(K; M_B | X, J)$ similarly

• Streaming algorithm for problem P

- Read a stream w of length n
- Solve P on w
- **Complexity**
 - # of passes (1 , constant, \log)
 - memory space (polylog, sublinear)
 - processing time per symbol (polylog)



sublinear space
polylog time



• 1-pass deterministic algorithms

- Generalization of automata
- ▶ Recognize Regular languages with constant space!

• 1-pass algorithm

- $c \leftarrow 0$
- While stream non empty
 - $a \leftarrow$ next item
 - If $c = 0$ then $v \leftarrow a$
 - If $v = a$ then $c \leftarrow c + 1$
 - else $c \leftarrow c - 1$

• Fact

- If u has frequency $> 50\%$, then $v=u$

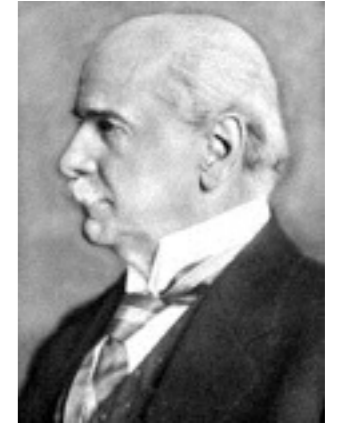
• 2-pass algorithm

- Run previous algorithm on 1st pass
- On 2nd pass check that v has frequency $> 50\%$

•Dyck(s)

- Well-formed expressions over s types of parentheses
- Example:

((()) ()) : well-formed ((()) ()) : ill-formed



Walther von Dyck

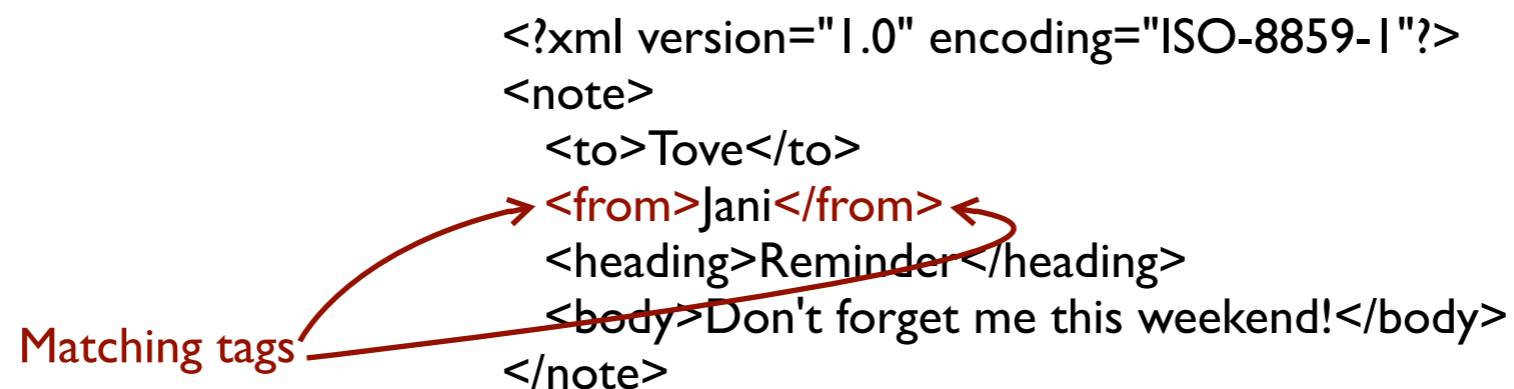
•Database application

- Well-formedness of large XML file

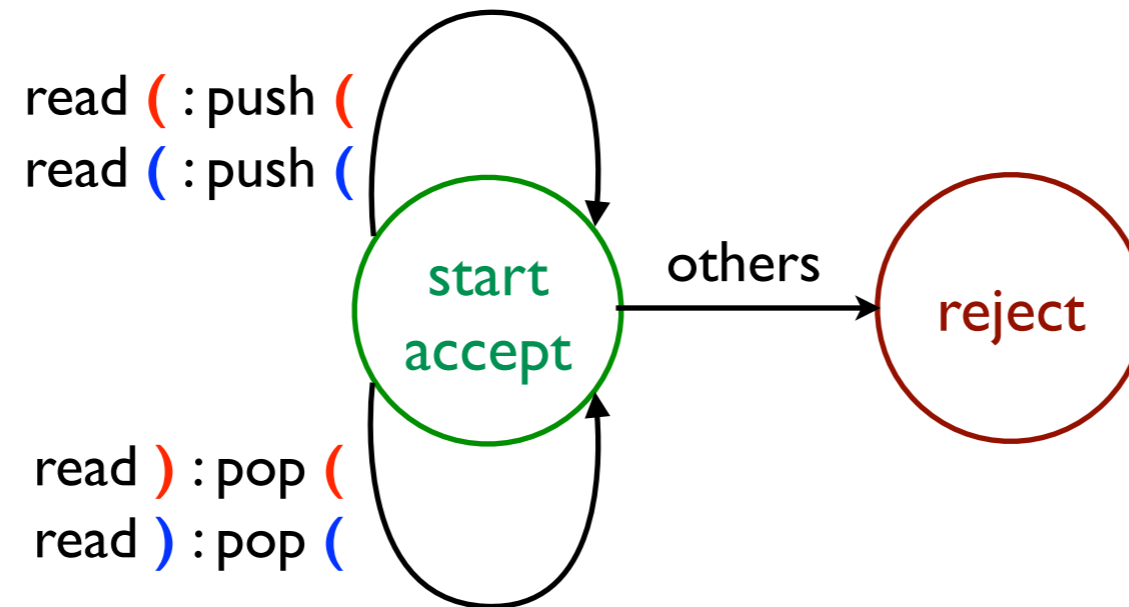
Check if document is well-parenthesized (opening tags are correctly closed)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Matching tags

A diagram illustrating matching tags in an XML document. The XML code is shown above. A red label 'Matching tags' is positioned to the left of the code. Two red arrows originate from this label: one points to the opening tag '<from>Jani</from>' and the other points to its corresponding closing tag '</from>'. This visualizes the concept of well-parenthesized expressions in the context of XML tags.

•Stack automaton

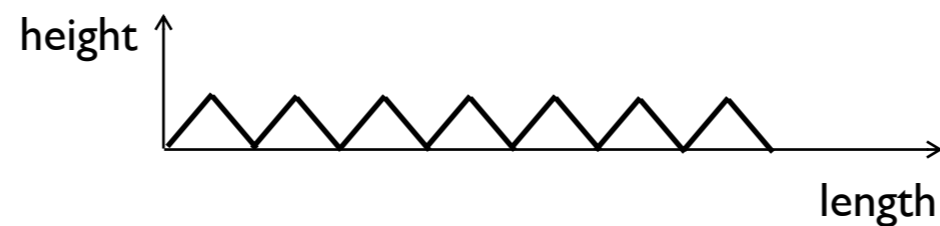


•Simple algorithm

- One-pass algorithm for Dyck(2) with space max height

•Observation

- Small height \Rightarrow small space



Fact

n: input size

- Dyck(1): 1-pass deterministic algorithm with space $\log n$
 - Dyck(2): p-passes deterministic algorithms need space $\Omega(n/p)$
- Dyck(s) is (streaming-)reducible to Dyck(2)
- ▶ From now $s=2$ and alphabet is $(,), (,)$

•Results for Dyck(2)

- **One pass**

Randomized algorithm with space $\sqrt{(n \log n)}$ and one-sided error $1/\text{poly}(n)$

Matching lower bound

- **More passes in same direction do not help**

Requires space $\Omega(\sqrt{n} / p)$ for p passes

- **Two passes: Forward and backward**

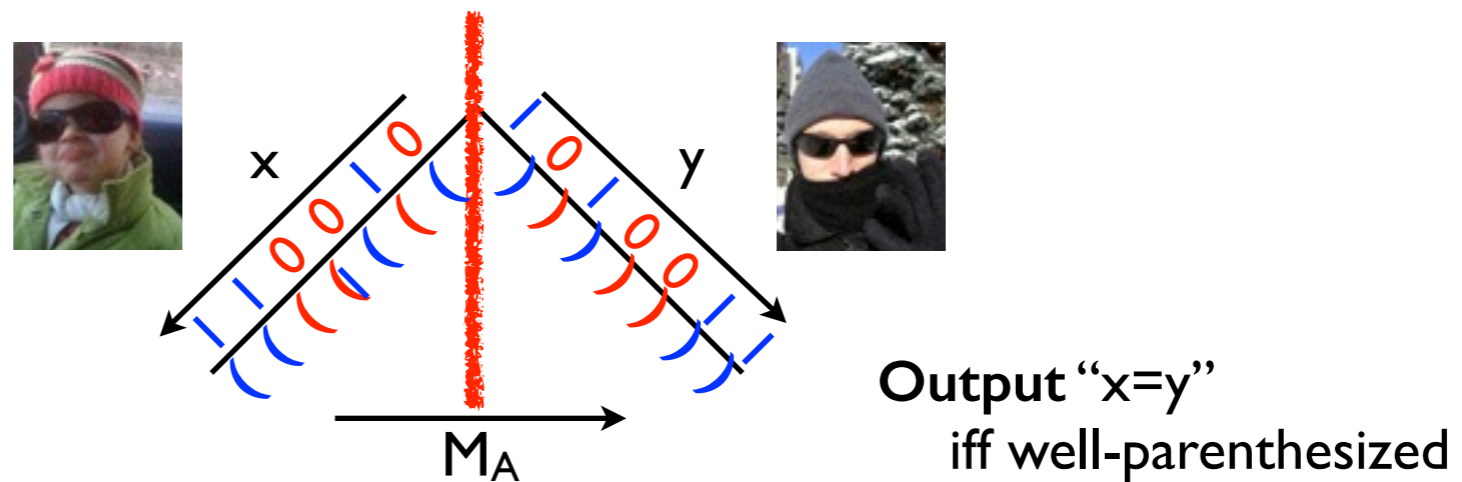
Randomized algorithm with space $(\log n)^2$ and one-sided error $1/\text{poly}(n)$

•Theorem

- Any 1-pass deterministic streaming algorithm requires memory $\Omega(n)$

•Proof

- Idea: Use the algorithm to construct a one-way protocol for **EQUALITY**



•Lemma

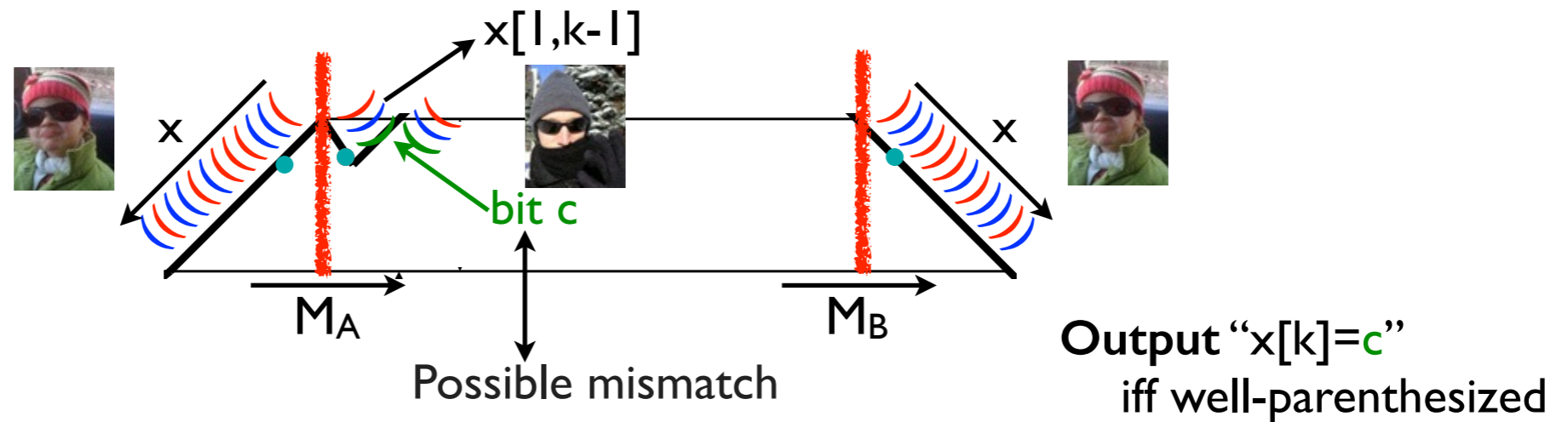
- A deterministic protocol for EQUALITY requires a message of linear size

•Fact

- There is a randomized one-way protocol for EQUALITY with a message of logarithmic size

• Idea

- Use the algorithm to construct a one-way protocol for Augmented-INDEX



• Lemma

- Any 1-pass streaming algorithm for Dyck(2) (for input size n) induces a two-way protocol for INDEX s.t.
 $|M_A|, |M_B| \leq \text{memory space (of algorithm)}$

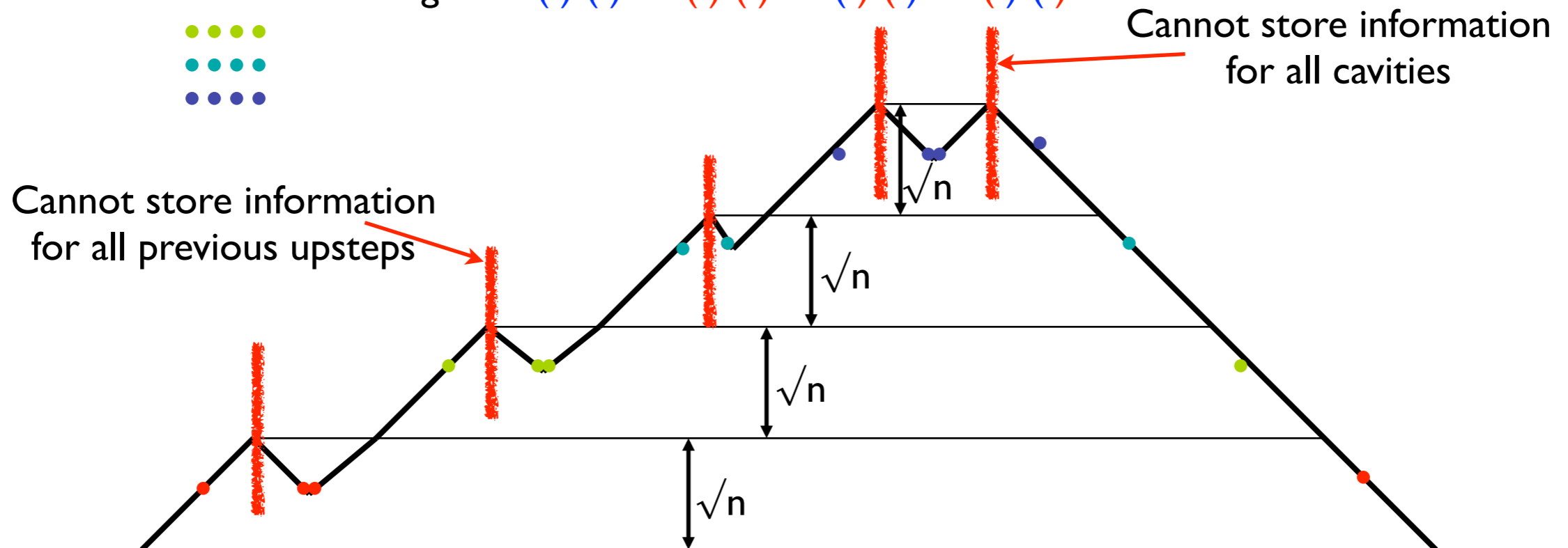
• Issue

- M_B is too large

• Hard instance

- String of size $\Theta(n)$: \sqrt{n} interleaved slices of size $\Theta(\sqrt{n})$
- Maximize the stack size of the 1-pass algorithm
- Forbid any possible simplifications on the stack

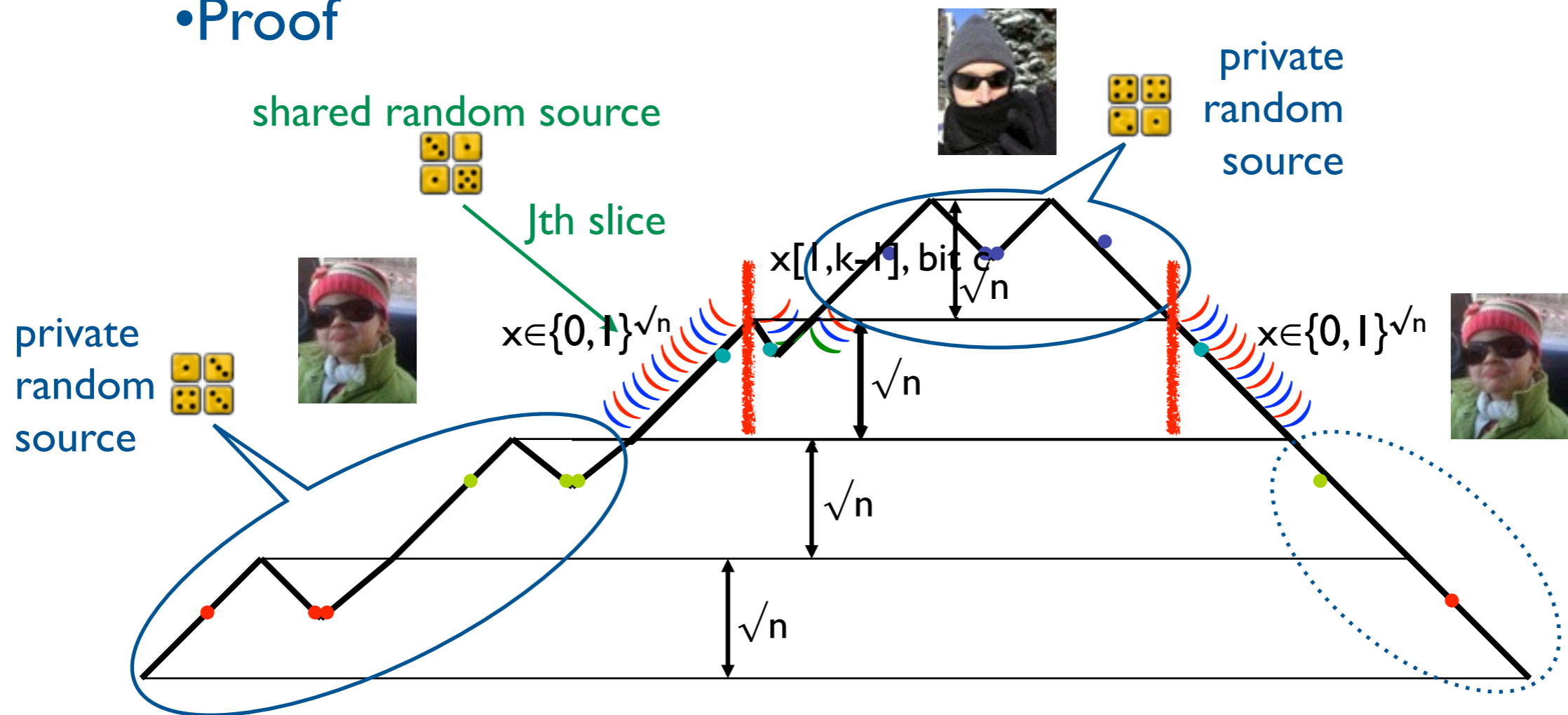
●●●● might be () () or () () or () () or () ()
●●●●
●●●●
●●●●



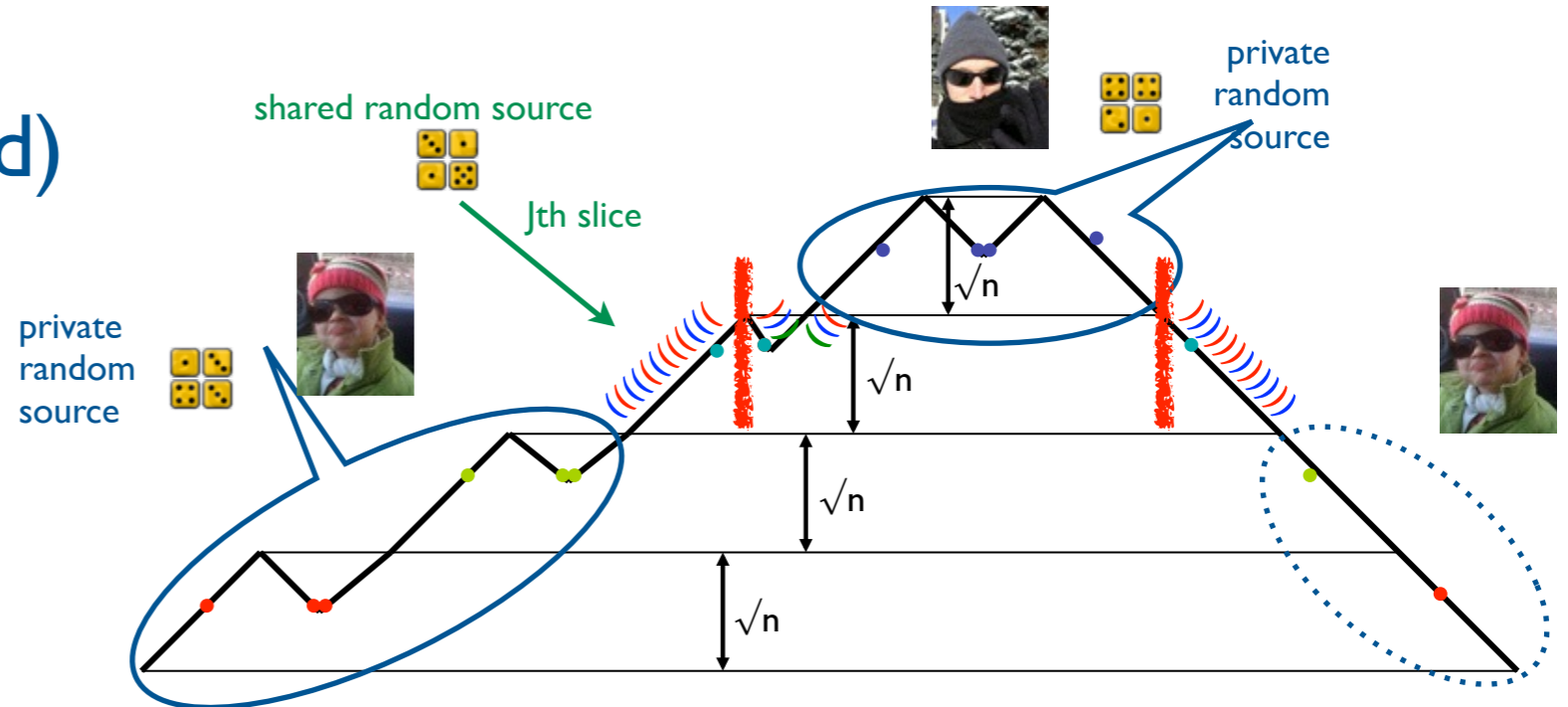
•Theorem

- Any streaming algorithm for Dyck(2) (for input size n) induces a protocol for Augmented-INDEX (for input size \sqrt{n}) s.t.
 - $|M_A|, |M_B| \leq$ memory space (of algorithm)
 - $I(K:M_B|X) \leq$ (memory space) / \sqrt{n}

•Proof



•Proof (continued)



- Requirement:

FULL expression is well-parenthesized iff SLICE is well-parenthesized

→ No error for other slices

→ **Collapsing distribution** (for Aug-INDEX): uniform (x,k) with $c=x[k]$

- $|M_A|, |M_B| \leq$ memory space (of algorithm)

- On the collapsing distribution

$$|M_B| \geq I((K^i)_i; M_B | (X^i)_i) = \sum_j I(K^j; M_B | K^1 \dots K^{j-1} (X^i)_i)$$

$$\geq \sum_j I(K^j; M_B | X^j) = t \times I(K^J; M_B | X^J, J) = t \times I(K; M_B | X, J)$$

Issue

- Need to extend INDEX lower bound for collapsing distribution

•Theorem

- Every streaming algorithm for Dyck(2) (for input size n) induces a protocol for Augmented-INDEX (for input size \sqrt{n}) s.t.
 - $|M_A|, |M_B| \leq$ memory space (of algorithm)
 - $I(K:M_B|X) \leq$ (memory space) / \sqrt{n}

for the collapsing distribution

•Lemma

- Any protocol for Augmented-INDEX satisfies
 - If $I(K:M_B|X) \leq b$ then $I(X:M_A|K) = \Omega(|X|/2^b)$ (when error probability $\leq 1/3$)for the collapsing distribution

•Corollary

- Every streaming algorithm for Dyck(2) (for input size n) requires memory space $\Omega(\sqrt{n})$

•Index

- Rahul Jain and Ashwin Nayak. [The Space Complexity of Recognizing Well-Parentthesized Expressions in the Streaming Model: the Index Function Revisited](#). ECCCC'10.
- A. Chakrabarti and R. Kondapally. [Everywhere-tight information cost tradeoffs for Augmented Index](#). APPROX'11/RANDOM'11 .

•Streaming

- M. Chu, S. Kannan, and A. McGregor. [Checking and spot-checking the correctness of priority queues](#). ICALP'07
- F. Magniez, C. Mathieu, and A. Nayak. [Recognizing well-parenthesized expressions in the streaming model](#). STOC'10

•Information and Communication Complexity

- Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. [An information statistics approach to data stream and communication complexity](#). FOCS'02.
- T. S. Jayram, Ravi Kumar, and D. Sivakumar. [Two applications of information complexity](#). STOC'03.
- Mark Braverman. [Interactive information complexity](#). STOC'12.