**INF561: Algorithmes de streaming** **Hiver 2012**

Lecture 6 — 8 février

*Lecturer: Frédéric Magniez* *Scribe: Raphael BOST*

# 6.1 Model

## 6.1.1 Definitions

Two players Alice and Bob are separated. They have access to unlimited computational resources.

At the beginning of the protocol, A gets $x \in X$ and B gets $y \in Y$. A starts the protocol. The two players successively exchange messages $M_i$, functions of the previous received messages and $x$ for A, $y$ for B. The last message is named the output of the protocol and we call the transcript $P(x, y)$ of the protocol the sequence of the messages ($P(x, y) = (M_1, M_2, ...)$).

The complexity of $P$ on input $(x, y)$ is the number of bits in the transcript ($|P(x, y)|$). The complexity of P is $C(P) = \max_{x \in X, y \in Y} |P(x, y)|$

We say that $P$ computes $f$ if $\forall x, y$ output of $P(x, y) = f(x, y)$.

The protocol $P$ can be :

- deterministic
- randomized by bounded error ($\mathbb{P}(\text{output } P(x, y) \neq f(x, y)) \leq \epsilon$)
- using public coins : A and B can access to the same sequence of random bits
- using private coins : A and B have access to independent sequence of random bits

We can define for a function f the following quantities :

$$D(f) = \min_{P \textit{ deterministic}} C(P)$$

$$R_\epsilon(f) = \min_{P \textit{ random. bounded error, private coins}} C(P)$$

$$R_\epsilon^{pub}(f) = \min_{P \textit{ random. bounded error, public coins}} C(P)$$

We have

$$R_\epsilon^{pub}(f) \leq R_\epsilon(f) \leq D(f)$$

### 6.1.2 Example

Lets consider the problem $EQ_n : X = Y = \{0,1\}^n$ and determine if $x = y$.

We can prove $D(EQ_n) \leq n$(check bit by bit) and $R_{\frac{1}{2}}(EQ_n) \leq O(\log n)$.

If we have access to public randomized bits we have indeed $R^{pub}_{\frac{1}{2}}(EQ_n) = 2$. To prove it we can use the following protocol : let $r \in \{0,1\}^n$; Alice computes $a = \sum_i r_i x_i \mod 2 = \oplus r_i x_i$ and sends $a$ to Bob (1 bit). Bob computes $b = \oplus r_i y_i$. If $a = b$, Bob outputs 1, else he outputs 0 (1 bit).
If $x = y$, then $a = b$, the protocol always outputs 1, if $x \neq y$, then $\mathbb{P}(a = b) = \frac{1}{2}$.

We are here in the special case of one-way protocol (only one message exchanged for Alice to Bob plus the output). In those cases, we introduce the quantities $\overrightarrow{D}(f)$, $\overrightarrow{R}_\epsilon(f)$, $\overrightarrow{R}^{pub}_\epsilon(f)$ which are the size of Alice's message.

## 6.2 General principle for lower bounds of $D(f)$

Let $M_f$ be :

$$M_f = (f(x,y))_{(x,y)\in X\times Y}.$$

**Theorem 6.1.** $\overrightarrow{D}(f) \geq \log_2(|$*distinct rows of* $M_f|)$

**Proof:** As the protocol is one-way, the output is only a function of $y$ and $M_1(x)$. So, if $M_1(x) = M_1(x')$ then output$(x,y) =$ output$(x',y)$.

So, $\forall x, x'$, if $\exists y$ such that $f(x,y) \neq f(x',y)$, then $M_1(x) \neq M_1(x')$. And in this case, the rows of $x$ and $x'$ in $M_f$ are different. So Alice must be able to send |distinct rows of $M_f$| different messages, using $\log_2$(|distinct rows of $M_f$|) bits.

$\square$

**Example:** $M_{EQ_2} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ and more generally $M_{EQ_n} = I_{2^n}$ so $\overrightarrow{EQ_n}(f) \geq n$

**Theorem 6.2.** *Let $f$ be a boolean function : $f : X \times Y \to \{0,1\}$. $D(f) \geq \log_2(\text{rank}(M_f))$*

To prove the theorem, we will use a small lemma : we introduce the rectangles.
$X \times Y$ is a rectangle if we have

$$P(x,y) = P(x',y') \Rightarrow P(x',y) = P(x,y') = P(x,y)$$

**Lemma 6.3.** *Rectangle Principle :*
*For every fixed transcript $\tau$, $\{(x,y)|P(x,y) = \tau\}$ is a rectangle.*

**Proof (Proof of th.):** For each transcript $\tau$ such that the output is 1, we associate the rectangle $R_\tau$ of inputs and $(M_\tau)_{x,y} = 1$ if $(x,y) \in R_\tau, 0$ otherwise.

Then we have

$$M_f = \sum_{\tau} M_\tau$$

We have also $\mathrm{rank}(M_\tau) = 1$ (in the matrix, there is only copies of two different columns according to the rectangle principle). Thus we have

$$\mathrm{rank}(M_f) \leq \sum_{\tau} \mathrm{rank}(M_\tau) \leq 2^{|P|}$$

□

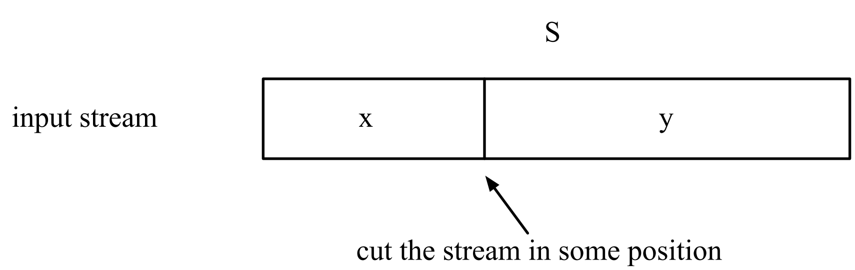
**Example:** consider the two problems :

| INDEX | | TRANSMIT | |
|---|---|---|---|
| A | B | A | B |
| $x \in \{0,1\}^n$ | $i \in \{1, 2, ..., n\}$ | $x \in \{0,1\}^n$ | $\emptyset$ |
| | output $x_i$ | | output $x$ |

When we work in one-way, we have TRANSMIT $\leq$ INDEX and as $\overrightarrow{D}(\text{TRANSMIT}) \geq n$, $\overrightarrow{D}(\text{INDEX}) \geq n$

## 6.3 Application to streaming

*Idea :* Given a 1-pass streaming algorithm with memory $M$ calculating $f$, we want to construct a one-way protocol with communication complexity $M$




The protocol computes $F(x, y) = f(x||y)$ ($x||y$ is the stream obtained by concatenating $x$ and $y$). Alice simulates the streaming algorithm for the beginning of the stream ($x$), sends the memory to Bob and Bob ends the simulation and outputs the result.

**Lemma 6.4.** *Memory of 1-pass deterministic (resp. randomized) algorithm for $f$ is greater than $\max \overrightarrow{D}(F)$ (resp. $\overrightarrow{R}(F)$) where the maximum is taken over all the possible cuts of the stream.*

We can extend this result to $k$-pass streaming algorithms : maximum memory of $k$-pass deterministic streaming algorithm for $f \geq \frac{1}{2k} \max D(F)$ (Bob sends its memory at the end of each pass so Alice can begin a new one).

**Theorem 6.5.**

$$\overrightarrow{R}_\delta(TRANSMIT) \geq \overrightarrow{D}(TRANSMIT)$$

*for* $\delta > \frac{1}{2}$

**Proof:** Let $P$ be a one-way probabilistic protocol for transmit with bounded error $\delta$, $r_A$ the random bits sequence of A and $r_B$ the random bits sequence of B. In the protocol, the message $M_1$ of Alice depends on $x$ and $r_A$ and Bob computes the output, depending on $M_1$ and $r_B$

Define $p(x, r_A)$ by $p(x, r_A) = \mathbb{P}_{r_B}(\text{output}(M_1(x, r_A), r_B) \neq f(x))$
Because $\mathbb{E}_{r_A} p(x, r_A) = \mathbb{P}_{r_A, r_B}(\text{output}(M_1(x, r_A), r_B) \neq f(x)) \leq \delta$, $\forall x \exists r_A(x)$ such that $p(x, r_A(x)) \leq \delta$

For the deterministic algorithm, Alice computes such a random bits sequence $r_A$ and sends the corresponding message $M_1(x, r_A)$. Then, Bob computes output$(M_1, r_B)$ for all possible sequence $r_B$ and outputs the most frequent one ($f(x)$ should appear at least with a fraction $\frac{2}{3}$). □

We can also prove complementary results for the communicational complexity of $TRANSMIT$ :

$$\overrightarrow{R}_\delta(TRANSMIT) \leq \overrightarrow{R}_{\frac{\delta}{n}}(INDEX_n)$$

Indeed,

$$\begin{aligned}
\mathbb{P}_{r_A, r_B}(\text{output}_{TRANSMIT} \neq x) &= \mathbb{P}_{r_A, r_B}(\exists i | \, \text{output}_{INDEX}(M_1(x, r_A), i, r_B) \neq x_i) \\
&\leq \sum_{i=1}^{n} \mathbb{P}_{r_A, r_B}(\text{output}_{INDEX}(M_1(x, r_A), i, r_B) \neq x_i) \\
&\leq n\frac{\delta}{n} = \delta
\end{aligned}$$

Finally, $\overrightarrow{R}_{\frac{\delta}{n}}(INDEX_n) \leq O(\log n)\overrightarrow{R}_\delta(INDEX_n)$ This can be see as a result of the parallel repetition paradigm : if $P$ is a random protocol for $INDEX$ with bounded error $\delta$, we construct the following protocol :

$$\begin{array}{ccc}
\text{A} & & \text{B} \\
r_A^1 & \xrightarrow{M_1(x, r_A^1)} & \text{Bob computes the majority} \\
r_A^2 & \xrightarrow{M_2(x, r_A^2)} & \text{of output}(i, M_i(x, r_A^i)) \\
\ldots & & \text{for } i = 1, \ldots, k \\
r_A^k & \xrightarrow{M_k(x, r_A^k)} &
\end{array}$$

If $k \sim \log n$ then the new protocol has bounded error $\frac{\delta}{n}$

**Example:** Let a stream be $a_1 a_2 ... a_n$, with $a_i \in \{1, 2, ..., n\}$. Define $f_j = |\{i | a_i = j\}|$ and $F_k = \sum f_j^k$, $F_\infty = \max f_j$.

Any randomized, 1-pass streaming algorithm $A$ that computes $z$ such that $\mathbb{P}[|z - F_\infty| \geq \frac{F_\infty}{3}] \leq \frac{1}{3}$ requires memory $\Omega(n)$

Assume $A$ is given and $A$ has memory $M$. We will construct a protocol for $INDEX$ using $A$. We will find a stream such that for this stream $F_\infty = x_i$ and the first part depends on $x$ and the second part on $i$.

| A | B |
|---|---|
| $x \in \{0,1\}^n$ | $i \in \{1, 2, ..., n\}$ |
| | output $x_i$ |

Imagine, we have a stream such that $f_j = 1$ if $x_j = 1$, $f_j = 0$ otherwise. Then for non-zero $x$, we will have $F_\infty = 1$. Append to such a stream the element $x$, the stream will be $\{j | x_j = 1\} || \{x\}$. Then, we have :

- if $x_i = 0$, then all the element in the stream appear only once and $F_\infty = 1$ and $\mathbb{P}[|z - 1| \geq \frac{1}{3}] \leq \frac{1}{3} \Rightarrow \mathbb{P}(z \geq \frac{4}{3}) \leq \frac{1}{3}$
- if $x_i = 1$, the $F_\infty = 2$ ($i$ appears twice in the stream) and $\mathbb{P}[|z - 2| \geq \frac{2}{3}] \leq \frac{1}{3} \Rightarrow \mathbb{P}(z \leq \frac{4}{3}) \leq \frac{1}{3}$

We end, by simulating $A$ on the stream. If the output is $< \frac{4}{3}$, then output 0, else output 1.

This is a protocol for $INDEX$ with bounded error $\frac{1}{3}$ and communicational complexity $M$. But $\overrightarrow{R}_{\frac{1}{3}}(INDEX_n) = \Omega(n)$ so $M = \Omega(n)$