Lower Bounds in Streaming

Piotr Indyk
MIT
Streaming Algorithms

- Norm estimation, heavy hitters/sparse approximations
- Question: are these algorithms (close to) optimal?
Lower Bounds in Streaming

- Two techniques:
  - Pigeonhole principle: need enough space to distinguish “different” inputs
  - Communication complexity
- PP is really a special case of CC (but is often easier to apply)
- Today:
  - Randomness and approximation are both necessary for estimating $\|x\|_0$ in polylog $(n+m)$ space (even in the insertions-only case)
  - Need $\Omega(1/\epsilon^2)$ bits to $(1+\epsilon)$-approximate $\|x\|_2$
Estimating $||x||_0$

- Warmup theorem: any deterministic exact algorithm for computing $||x||_0$ needs $\Omega(m)$ bits of space

- Proof:
  - Assume there is an algorithm $A$ using $M=o(m)$ bits of space
  - Take any vector $y\in\{0,1\}^m$, $||y||_0=m/2$
  - Feed the coordinates of $y$ to $A$
  - Let $A[y]$ be the state of $A$ at the end of this process, and $E$ be the estimation of $||y||_0$
  - We can decode $y$ from $A[y]$:
    - For any $z\in\{0,1\}^m$, $||z||_0=m/2$, feed $z$ to $A$ in state $A[y]$, obtaining $A[y+z]$
    - The algorithm computes an estimation $E'$ of $||y+z||_0$
    - We have $y=z$ iff $E=E'$
  - The number of distinct states $A[y]$ is $\exp(\Omega(m))$ - contradiction
Estimating $||x||_0$, ctd.

- Upgraded theorem: any deterministic $c$-approximate algorithm for computing $||x||_0$ needs $\Omega(m)$ bits of space, for $c=1+\varepsilon<2$
  - Estimation $E$ such that $||x||_0 \leq E < c||x||_0$

- Proof:
  - For any $y\in\{0,1\}^m$, let $ECC(y)\in\{0,1\}^{m'}$, $m'=O(m)$ be such that:
    - $||ECC(y)||_0=m'/a$, $a=\Theta(1)$
    - For any $y\neq z$, the distance $||ECC(y)-ECC(z)||_0 \geq 2\varepsilon \cdot m'/a$
      (which implies that $||ECC(y+z)||_0 \geq m'/a + m'\varepsilon/a = m'c/a$)
  - Take any $y\in\{0,1\}^m$
  - Feed the coordinates of $ECC(y)$ to $A$
  - The remainder of the argument essentially as before
    (except that $y=z$ iff $E' < m'c/a$)
Estimating $\|x\|_0$, ctd.

- Upgraded theorem 2: any randomized exact algorithm for computing $\|x\|_0$ needs $\Omega(m)$ bits of space.
- Proof:
  - Assume $o(m)$ space, and the probability of error < 1/8.
  - Take any ECC of length $m'$ with minimum distance $m'/4$.
  - Take any $y$.
  - Feed the coordinates of ECC($y$) to $A$.
  - With prob. 1/2 we can recover $z$ such that $\|z-ECC(y)\|_0 < m'/4$.
    (and therefore can recover $y$):
      - In parallel, for any $i=1..m'$, feed $e_i$ to $A$ with state $A[ECC(y)]$, obtaining estimate $E_i$.
      - Set $z_i=0$ iff $E_i > m/a'$ (fails with probability < 1/8).
      - Markov inequality implies that the fraction of errors is < 1/4 with prob. 1/2.
    (ctd on the next slide)
Recap: for any $y \in \{0,1\}^m$, by feeding $ECC(y)$ to $A$ and then recovering a vector in $\{0,1\}^m$, we correctly recover $y$ with prob. $1/2$.

Formally: we have the following setup:
- A mapping $F_r(y)$ that, given $y \in \{0,1\}^m$ and a sequence $r$ of random bits used by the algorithm, returns a state of the algorithm (obtained by feeding $ECC(y)$ to $A$).
- A mapping $G(S)$ that maps a state $S$ of the algorithm to a vector in $\{0,1\}^m$ (the mapping is defined by the recovery process).
- The mappings satisfy the following property: for each $y \in \{0,1\}^m$ the probability $\Pr_r[ G(F_r(y)) = y ]$ is at least $1/2$.

This implies that there exists $r$ such that $G(F_r(y)) = y$ holds for at least $1/2$ fraction of $y \in \{0,1\}^m$.

And this implies that the number of the states of the algorithms is at least $2^m/2$. 

Proof ctd.
Communication Complexity

Alice: $x \in \{0,1\}^m$

Bob: $y \in \{0,1\}^m$

- Resources:
  - # bits
  - # rounds
    - Today, we will be only interested in one-round protocols
- Probability of error: some constant $\delta > 0$
- See [Kushilevitz-Nisan] for more
  (and there is much more)
C.c. vs streaming

Alice: $x \in \{0,1\}^m$

Bob: $y \in \{0,1\}^m$

• Fact: If there is a streaming algorithm $A$ that computes $F$ given $x \cdot y$ as an input using space $M$, then there is a one-round c.c. protocol solving $F$ using communication $M$.

• Proof:
  – Alice feeds $A$ with $x$ obtaining state $A[x]$
  – Send $A[x]$ to Bob
  – Bob feeds $A[x]$ with $y$, outputs the answer

$F(x,y)$
Indexing

• (Balanced) indexing problem:
  – Alice: a vector $x \in \{0,1\}^m$, $\|x\|_0 = m/2$
  – Bob: an index $i = 1 \ldots m$
  – Goal: compute $f(x,i) = x_i$

• Theorem: any randomized one-round protocol for indexing has $\Omega(m)$ bit complexity

• Proof: pigeonhole principle as applied earlier
Gap Dot Product

• (Gap) parameter $\Delta$
• Alice: a vector $u \in \mathbb{R}^m$, $\|u\|_2 = 1$ (with $O(\log m)$ bits)
• Bob: a vector $v \in \mathbb{R}^m$, $\|v\|_2 = 1$
• Goal:
  – If $u^*v = 0$, return 0
  – If $u^*v \geq \Delta$, return 1

• Theorem: the randomized one-round CC of GDP with gap $\Delta = 1/(m/2)^{1/2}$ is $\Omega(m)$

• Proof: via reduction from indexing:
  – Alice: computes $u = \Delta x$
  – Bob: computes $v = e_i$
  – Fact: $u^*v = \Delta x_i$
Space complexity of \( L_2 \) norm estimation

- Theorem: any streaming algorithm for estimating the \( L_2 \) norm of an \( m \)-dimensional vector \( x \) up to a factor of \( 1 \pm \Delta \), \( \Delta = c/m^{1/2} \), requires \( \Omega(m) \) bits for some constant \( c > 0 \) (even if coordinates of \( x \) have \( O(\log m) \) bits)

- Proof:
  - Assume we have an \( M \)-space streaming algorithm that computes \( (1 \pm \Delta) \|x\|_2 \)
  - Then we have an \( M \)-space streaming algorithm that, given a stream \( u \circ v \), \( \|u\|_2 = \|v\|_2 = 1 \), computes \( u^*v \pm O(\Delta) \)
    - Using the equality \( \|u-v\|_2^2 = \|u\|_2^2 + \|v\|_2^2 - 2u^*v \)
  - Then we have an \( M \)-bit one-round protocol that solves GDP with gap \( 1/(m/2)^{1/2} \) (assuming \( c \) small enough)
  - Ergo, \( M = \Omega(m) \)