Cell-Probe Lower Bounds for Succinct Partial Sums

Mihai Pătrașcu

Emanuele Viola
Succinct Data Structures

Given some input of $N$ bits
  => some data structure of close to $N$ bits
to answer useful queries

**Why?**

*Practice:* functional data compression
  You often want to query compressed data,
  ... so the data structure on top better be small, too

*Theory:* algorithmic ideas with nice information theory flavor
Interesting Upper Bounds

[Dodis, P, Thorup ’10]
Store a vector $A[1..n]$ from alphabet $\Sigma$
- Space $[n \log_2 \Sigma]$
- Constant time to read or write $A[i]$

[P, FOCS’08]
Store a vector $A[1..n]$ of bits
- Query time $O(t)$ for $\text{RANK}(k) := A[1] + ... + A[k]$
- Space $n + n / (w/t)^t$
Rank / Select

\[
\text{RANK}(k) = A[1] + \ldots + A[k]
\]
\[
\text{SELECT}(k) = \text{index of } k^{\text{th}} \text{ one in } A[1..n]
\]

A staple of succinct data structures.

Example: representing trees succinctly

**Diagram:**

- **Tree Representation:**
  - Tree with nodes A, B, C, D, E, F.
  - A is the root, B is the left child of A, C is the right child of A, D is the left child of C, E is the left child of D, and F is the right child of D.

- **Array Representation:**
  - Array indices: 1 1 0 1 1 0 1 0 1 0 0 0 0

- **Functions:**
  - \( \text{LEFTCHILD}(k) = 2 \cdot \text{RANK}(k) \)
  - \( \text{RIGHTCHILD}(k) = 2 \cdot \text{RANK}(k) + 1 \)
  - \( \text{PARENT}(k) = \text{SELECT}([k/2]) \)
Interesting Lower Bounds

[Gál, Miltersen ’03] polynomial evaluation
  => redundancy * query time ≥ Ω(n)
  ☹ nobody really expects a succinct solution

[Golynski SODA’09] store a permutation and query π(·), π⁻¹(·)
  With space 2n lg n, query time is 1
  If space is (1 + ε) n lg n => query time is Ω(1 / √ε)

[HERE] RANK / SELECT
  For query time t => space is ≥ n + n / w^{O(t)}
  NB: Upper bound was n + n / (w/t)^{Θ(t)}
Models

“Standard Model” (a.k.a. Word RAM)
- memory = words of $w$ bits
- $w \geq \lg n$ (store pointers, indices)
- constant time ops: read/write memory, +, -, *, /, %, <, >, ==, >>, <<, &, |, ^

Lower bounds: “Cell-Probe Model”
- memory = words (cells) of $w$ bits
- time = # cell reads/writes

Nice: information theoretic, holds even with exotic instructions
The Lower Bound Proof
Step 1: Understand the Upper Bound

\[ \Sigma^* \]

\[ H(\Sigma_1, \ldots, \Sigma_B | \Sigma^*) \]

\[ \lg \left( \frac{B}{\Sigma} \right) \]


A[.] A[.] \ldots A[.]
Recursion

Store non-succinctly

Spend constant time per node
  => decode $\Sigma_i$ from $\Sigma^*$ and the $H(\Sigma_1,\ldots, \Sigma_B | \Sigma^*)$ memory bits

Set parameters such that: $H(\Sigma_1,\ldots, \Sigma_B | \Sigma^*) = O(w) \Rightarrow B \approx w$

Can go $t$ levels up \Rightarrow redundancy $\approx n / w^t$
Published Bits

For induction, use a stronger model:
• memory = cells of $w$ bits; cost 1 to read a cell
• $P$ published bits, free to read (=cache)

Initially, set $P = \text{redundancy}$
→ publish some arbitrary(!) $P$ bits

Intuitively, think of published bits $\approx$ sums at roots of the subtrees
**Published Bits Die by Direct Sum**

Do published bits trivialize the problem?
No. Consider $T=100 \cdot P$ subproblems

<table>
<thead>
<tr>
<th>Problem 1</th>
<th>Problem $T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>.. .. .. ..</td>
<td>..</td>
</tr>
<tr>
<td>.. .. .. ..</td>
<td>..</td>
</tr>
<tr>
<td>.. .. .. ..</td>
<td>.. A[n]</td>
</tr>
</tbody>
</table>

For the average problem, published bits have 0.01 information
=> can’t really help
Recursion Intuition

<table>
<thead>
<tr>
<th>Upper Bound</th>
<th>Lower Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove top level of the trees</td>
<td>Main Lemma:</td>
</tr>
<tr>
<td>=&gt; $t = 1$</td>
<td>Remove one cell-probe</td>
</tr>
<tr>
<td>redundancy $*= w$</td>
<td>Increase $P = O(w)$</td>
</tr>
</tbody>
</table>

Intuition: $\{ \text{first cells probed by } Q_0 \} = \{ \text{first cells probes by } Q_k \}$
= the roots of the trees

So just publish $\{ \text{first cells probed by } Q_0 \}$
=> get rid of 1 probe for all queries
Careful for Bad Algorithms!

<table>
<thead>
<tr>
<th>Upper Bound</th>
<th>Lower Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove top level of the trees</td>
<td>Main Lemma:</td>
</tr>
<tr>
<td>=&gt; $t = 1$</td>
<td>Remove one cell-probe</td>
</tr>
<tr>
<td>redundancy $*= w$</td>
<td>Increase $P *= O(w)$</td>
</tr>
</tbody>
</table>

Can’t make an argument based on first probe!

Intuition: \{**first** cells probed by $Q_0$ \} = \{**first** cells probes by $Q_k$ \} = the roots of the trees

So just publish \{ first cells probed by $Q_0$ \} => get rid of 1 probe for all queries
## Intuition Fix

<table>
<thead>
<tr>
<th>Upper Bound</th>
<th>Lower Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove top level of the trees =&gt; $t = 1$</td>
<td>Main Lemma: Remove one cell-probe</td>
</tr>
<tr>
<td>redundancy $= w$</td>
<td>Increase $P = O(poly(w))$</td>
</tr>
</tbody>
</table>

### footprint:
$\text{Foot}(Q) = \{ \text{cells probes by } Q \}$

Intuition: $| \text{Foot}(Q_0) \cap \text{Foot}(Q_k) | = \Omega(P)$

So just publish $\text{Foot}(Q_0)$

$=>$ get rid of 1 probe for most queries
Why are Cells Reused?

Suppose, for contradiction, that $| \text{Foot}(Q_0) \cap \text{Foot}(Q_k) | = o(P)$

So the answers to most of $Q_k$ can be decoded from $\text{Foot}(Q_0)$

But $\text{Answers}(Q_0)$ and $\text{Answers}(Q_k)$ are highly correlated:

$$H(\text{Ans}(Q_0)) + H(\text{Ans}(Q_k)) \gg H(\text{Ans}(Q_0), \text{Ans}(Q_k))$$

So the data structure is an inefficient encoding (non-succinct).
\[ H(\text{Ans}(Q_0)) \approx H(\text{Ans}(Q_k)) \approx T \cdot H(\text{binomial on } n/T \text{ trials}) \]
\[ \approx T \cdot c \log(n/T) \quad \text{for some constant } c \]

\[ H(\text{Ans}(Q_0), \text{Ans}(Q_k)) = H(\text{Ans}(Q_0)) + H(\text{Ans}(Q_k) \mid \text{Ans}(Q_0)) \]
\[ \approx T \cdot c \log(n/T) + T \cdot H(\text{binomial on } k \text{ trials}) \]
\[ \approx T \cdot c \log(n/T) + T \cdot c \log k \]
\[ < T \cdot c \log(n/T) + T \cdot c \log n/(2T) \]
\[ = H(\text{Ans}(Q_0)) + H(\text{Ans}(Q_k)) - \Omega(T) \]

For k in “first half”...
Otherwise symmetric
Proof by Encoding

Claim: We will encode $A$ with less than $n$ bits (impossible)

<table>
<thead>
<tr>
<th>Write down...</th>
<th>Bits required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Published bits</td>
<td>$P$</td>
</tr>
<tr>
<td>Answers to $Q_0$, $Q_k$</td>
<td>$H(\text{Ans}(Q_0), \text{Ans}(Q_k))$</td>
</tr>
<tr>
<td>Cells in $\text{Foot}(Q_0)$</td>
<td>$w \cdot</td>
</tr>
<tr>
<td>Other cells</td>
<td>$n - w \cdot</td>
</tr>
<tr>
<td>TOTAL</td>
<td>$n + P - \Delta H$</td>
</tr>
</tbody>
</table>

$\Delta H = H(\text{Ans}(Q_0), \text{Ans}(Q_k)) - [H(\text{Ans}(Q_0)) + H(\text{Ans}(Q_k))] = \Omega(T)$

Contradiction for $T >> P$
The End