Cell-Probe Lower Bounds for Succinct Partial Sums

Mihai Pătrașcu

Emanuele Viola

Succinct Data Structures

N + o(N),
N + O(N/lg N),
N + O(√N), …

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 Σ

- Space $\lceil n \log_2 \Sigma \rceil$
- Constant time to read or write $A[i]$

[P, FOCS'08]

Store a vector $A[1..n]$ of bits

- Query time $O(t)$ for RANK$(k) := A[1] + \ldots + A[k]$
- Space $n + \boldsymbol{n / (w/t)^t}$

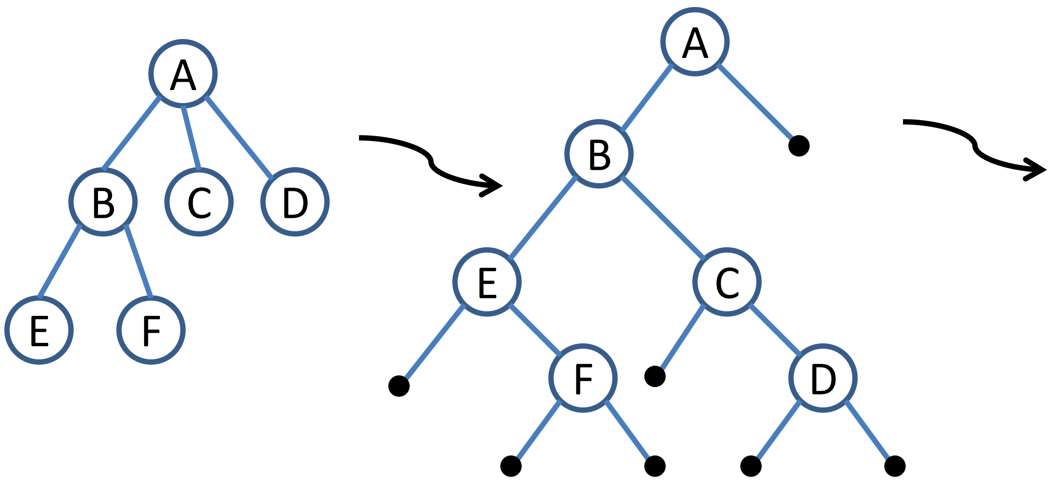
Rank / Select

RANK(k) = $A[1] + \ldots + A[k]$

SELECT(k) = index of k^{th} one in $A[1..n]$

A staple of succinct data structures.

Example: representing trees succinctly



A	B	.	E	C	.	F	.	D	.	.	.	.
1	1	0	1	1	0	1	0	1	0	0	0	0

LEFTCHILD(k) = $2 \cdot$ RANK(k)

RIGHTCHILD(k) = $2 \cdot$ RANK(k) + 1

PARENT(k) = SELECT ($\lfloor k/2 \rfloor$)

Interesting Lower Bounds

[Gál, Miltersen '03] polynomial evaluation

=> redundancy * query time $\geq \Omega(n)$

☹ nobody really expects a succinct solution

[Golynski SODA'09] store a permutation and query $\pi(\cdot)$, $\pi^{-1}(\cdot)$

With space $2n \lg n$, query time is 1

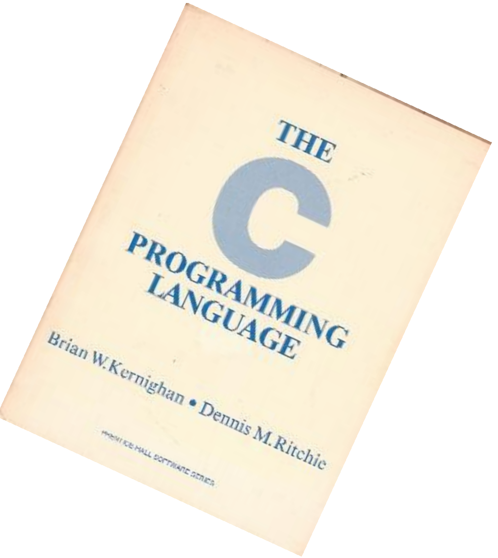
If space is $(1 + \varepsilon)\, n \lg n$ => query time is $\Omega(1 / \sqrt{\varepsilon})$

[HERE] RANK / SELECT

For query time t => space is $\geq n + \boldsymbol{n / w^{O(t)}}$

NB: Upper bound was $n + \boldsymbol{n / (w/t)^{\Theta(t)}}$

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,
  +, -, *, /, %, <, >, ==, >>, <<, &, |, ^

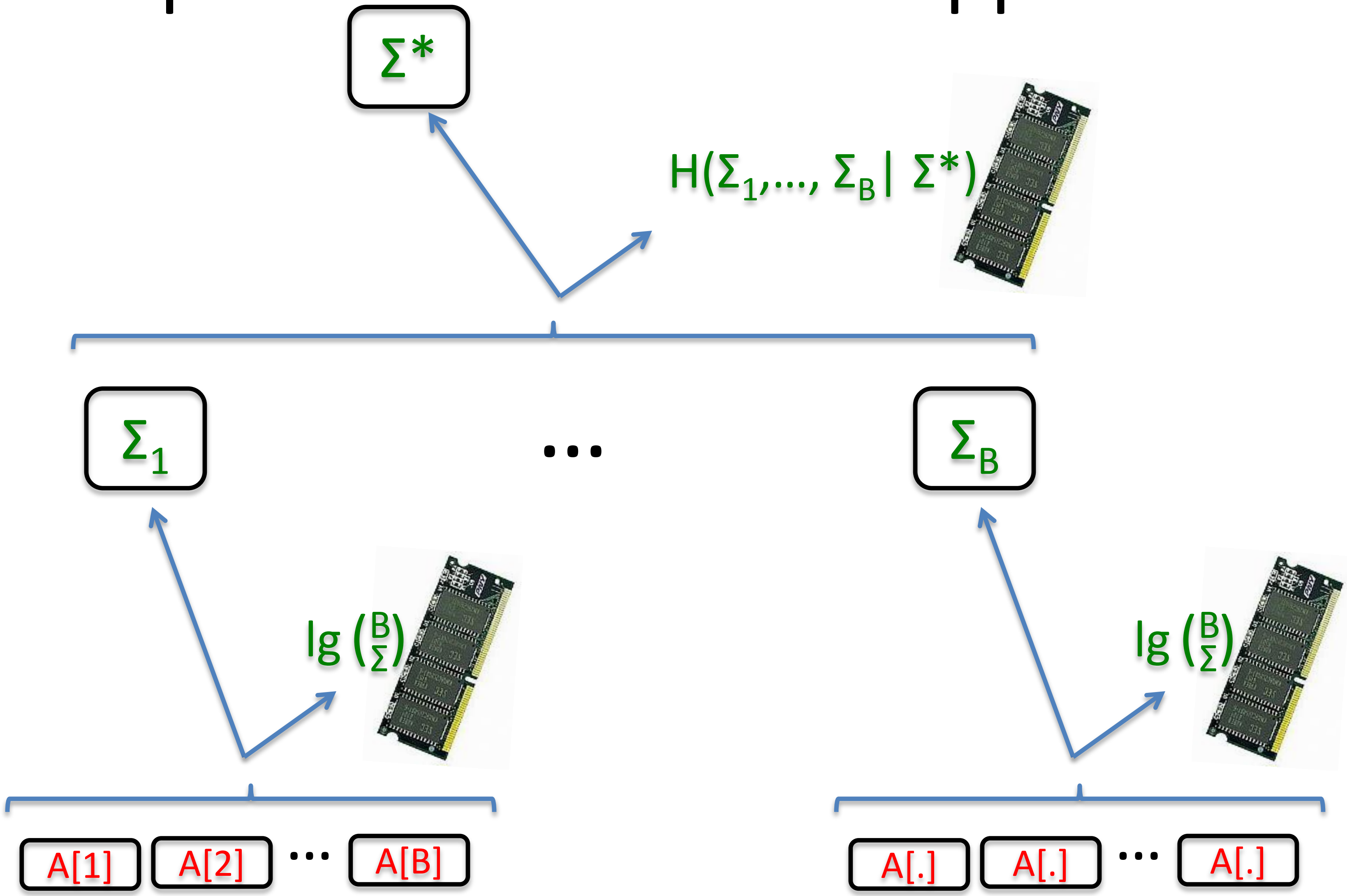
Lower bounds: **"Cell-Probe Model"**

- memory = words (cells) of w bits
- time = # cell reads/writes

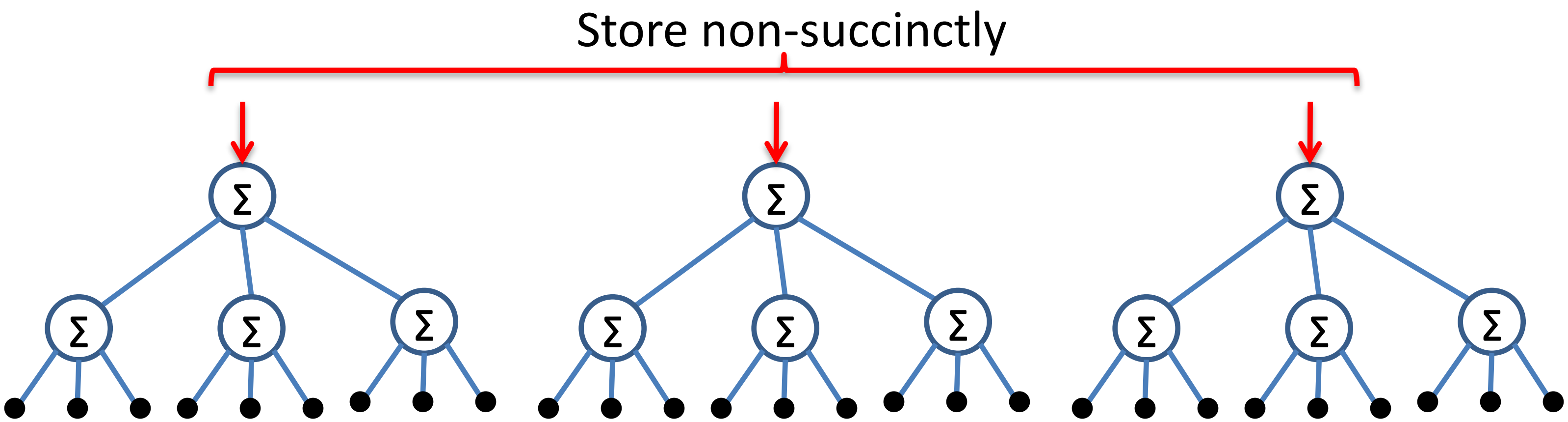
Nice: information theoretic, holds even with exotic instructions

The Lower Bound Proof

Step 1: Understand the Upper Bound



Recursion



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

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

Can go t levels up => 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 = redundancy
-> publish some arbitrary$^{(!)}$ P bits

Intuitively, think of published bits ≈ sums at roots of the subtrees

Published Bits Die by Direct Sum

Do published bits trivialize the problem?

No. Consider T=100·P subproblems

Problem 1 Problem T

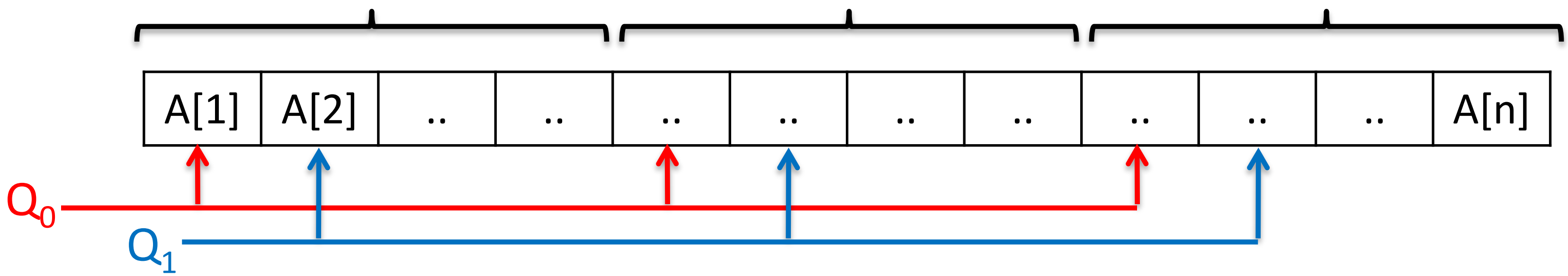
A[1]	A[2]	..	..	..	..	..	..	..	..	..	A[n]

For the average problem, published bits have 0.01 information

=> can't really help

Recursion Intuition

Upper Bound	Lower Bound
Remove top level of the trees => t -= 1 redundancy *= w	Main Lemma: Remove one cell-probe Increase P *= O(w)



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

Careful for Bad Algorithms!

Upper Bound	Lower Bound
Remove top level of the trees => t -= 1 redundancy *= w	Main Lemma: Remove one cell-probe Increase P *= $O(w)$

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

Upper Bound	Lower Bound
Remove top level of the trees => t -= 1 redundancy *= w	Main Lemma: Remove one cell-probe Increase P *= O(poly(w))

footprint: Foot(Q) = { cells probes by Q }

Intuition: | Foot(Q_0) ∩ Foot(Q_k) | = Ω(P)

So just publish Foot(Q_0)

=> get rid of 1 probe for most queries

Why are Cells Reused?

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

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

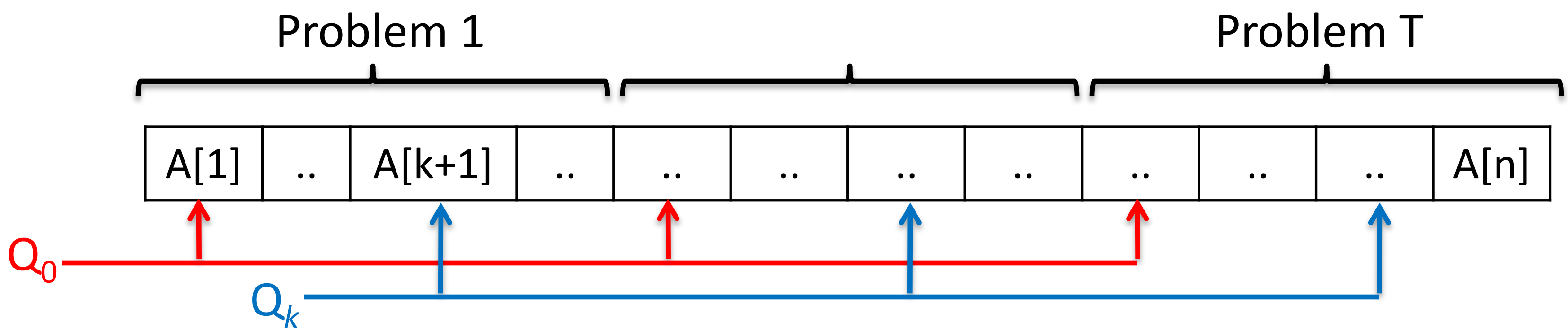
Ignored in this talk

But Answers(Q_0) and Answers(Q_k) are highly correlated:

$$H(\mathrm{Ans}(Q_0)) + H(\mathrm{Ans}(Q_k)) >> H(\mathrm{Ans}(Q_0), \mathrm{Ans}(Q_k))$$

So the data structure is an inefficient encoding (non-succinct).

Entropy Computation



$H(\text{Ans}(Q_0)) \approx H(\text{Ans}(Q_k)) \approx T \cdot H(\text{binomial on } n/T \text{ trials})$
$\quad \approx T \cdot c \lg(n/T)$ for some constant c

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

For k in "first half"…
Otherwise symmetric

Proof by Encoding

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

Write down…	Bits required
Published bits	P
Answers to Q_0, Q_k	$H(\mathrm{Ans}(Q_0), \mathrm{Ans}(Q_k))$
Cells in $\mathrm{Foot}(Q_0)$	$w \cdot \lvert \mathrm{Foot}(Q_0) \rvert - H(\mathrm{Ans}(Q_0))$
Other cells	$n - w \cdot \lvert \mathrm{Foot}(Q_0) \rvert - H(\mathrm{Ans}(Q_k))$
TOTAL	$\boldsymbol{n + P - \Delta H}$

$$\boxed{H(X \mid f(X)) = H(X) - H(f(X))}$$

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

Contradiction for $T \gg P$

The End