Comparison-Based Time-Space Lower Bounds for Selection

Timothy Chan
School of CS
U of Waterloo

Beginning of Story: Sorting

- Standard alg'ms:

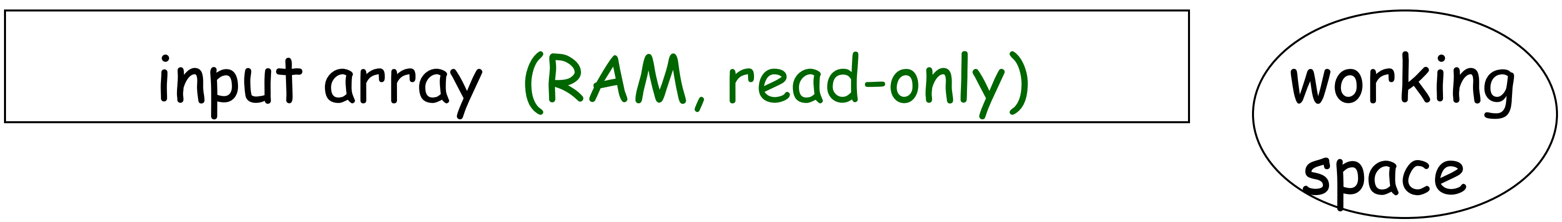
  $O(n \log n)$ time,   with $O(n)$ space

- Lower bd:

  $\Omega(n \log n)$ time

  for comparison-based alg'ms

What if Space is Limited?



(+ write-only output stream)

special case: multi-pass streaming alg'ms

Sorting (cont'd)

- Easy upper bd:

  with $O(s)$ words of space, $O(n^2/s + n \log n)$ time

  repeat:

  find the next s smallest elements, in $O(n + s \log s)$ time

  $\Rightarrow O(n/s)$ passes

Sorting (cont'd)

- Easy upper bd:

  with $O(s)$ words of space, $O(n^2/s + n \log n)$ time

- Lower bd: [Borodin,Fischer,Kirkpatrick,Lynch,Tompa, FOCS'79]

  with S bits of space, $\Omega(n^2/S + n \log n)$ time

  for comparison RAM alg'ms

Sorting (cont'd)

- Improved upper bd: [Pagter,Rauhe,FOCS'98]
  with $S \gg \log n$ bits of space, $O(n^2/S + n \log n)$ time

- Lower bd: [Borodin,Fischer,Kirkpatrick,Lynch,Tompa, FOCS'79]
  with S bits of space, $\Omega(n^2/S + n \log n)$ time

  for comparison RAM alg'ms

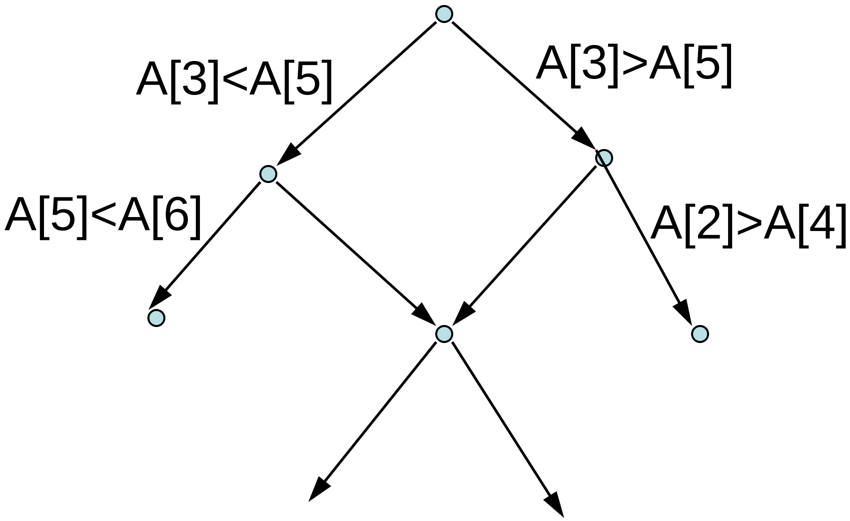
  OPTIMAL !

Remark

- Standard comparison-based lower bds:
  Decision trees

- Time-space comparison-based lower bds:
  "Decision DAGs" (comparison branching programs)

nodes ⇔ states
time ≥ height
space (in bits) ≥ log # nodes



Other Problems?
Element Distinctness

- Lower bd: [Borodin,Fich,Meyer auf der Heide,Upfal, Wigderson '86]

  with S bits of space, $\Omega(n^{3/2}/S^{1/2})$ time

- Lower bd: [Yao, FOCS'88]

  with S bits of space, $\Omega(n^{2-\varepsilon}/S)$ time

for comparison RAM alg'ms

(open: $\Omega(n^2/S)$??)

Other Problems?
Median (i.e. Selection)

- Standard alg'ms:

  $O(n)$ time, with $O(n)$ space

  ($< 2.95n$ comparisons [Dor,Zwick'95])

- Lower bd:

  $\Omega(n)$ time

  ($> (2+\varepsilon)n$ comparisons [Dor,Zwick'96])

Median (cont'd)

- First time-space upper bd: [Munro,Paterson,FOCS'78]

  with $O(s)$ words of space ($s \gg \log^2 n$),

  $O(n \log_s n + n \log s)$ time

for i = 0,1,...:

compute $O(s)$ approx. quantiles

to reduce "problem size" to $n_{i+1} \le n_i / s$

in $O(n + n_i \log s)$ time

$\Rightarrow O(\log_s n)$ passes

Median (cont'd)

- Improved upper bd: [Frederickson '86]
  with $O(s)$ words of space $(s \gg \log^2 n)$,
  $O(n \log_s n + n \log^* n)$ time

for $i = 0,1,\ldots$:

compute $O(s)$ approx. quantiles
to reduce "problem size" to $n_{i+1} \le n_i / s_i$
in $O(n + n_i \log s_i)$ time $(s_i \le s)$

set $s_i = 2^{n/n_i} \Rightarrow n_{i+1} \le n_i / 2^{n/n_i} \Rightarrow n/n_{i+1} \ge 2^{n/n_i}$

$\Rightarrow O(\log^* n + \log_s n)$ passes

Median (cont'd)

- Rand. upper bd: [Munro,Raman '92]
  with $O(s)$ words of space ($s \gg 1$),
  $O(n \log \log_s n)$ expected time

for i = 0,1,...:
  use rand. sampling* (as in Floyd,Rivest's alg'm)
  to reduce "problem size" to $n_{i+1} \le n_i^{1/2+\varepsilon}$
  in $O(n)$ time

$\Rightarrow O(\log \log_s n)$ passes

*2-point sampling uses only $O(1)$ words of space on RAM

Median (cont'd)

- Upper bd summary:

  with $O(s)$ words of space ($s \gg \log^2 n$),
  $O(n \log_s n + n \log^*(n/s))$ det. time or
  $O(n \log \log_s n)$ rand. time

- Lower bd:

  NOTHING, TILL NOW...

Median (cont'd)

- Upper bd summary:

  with $O(s)$ words of space ($s \gg \log^2 n$),
  $O(n \log_s n + n \log^*(n/s))$ det. time or
  $O(n \log \log_s n)$ rand. time

- Lower bd:

  TODAY: $\Omega(n \log \log_s n)$ rand. time

  for comparison RAM alg'ms

  OPTIMAL !

Median (cont'd)

- Upper bd summary:

  with $O(s)$ words of space ($s \gg \log^2 n$),
  $O(n \log_s n + n \log^*(n/s))$ det. time or
  $O(n \log \log_s n)$ rand. time

- Lower bd for special case:

  $\Omega(n \log_s n)$ det. time [Munro,Paterson,FOCS'78]
  TODAY: $\Omega(n \log^*(n/s))$ det. time
  for comparison-based multi-pass streaming alg'ms

Part 1:
Det. Streaming Lower Bd

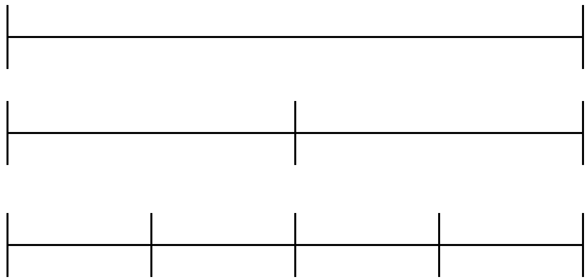
- To show: any det. streaming alg'm using cn comparisons & s space needs $\Omega(\log^*(n/s))$ passes
- Pf Idea: use a simple adversary argument to construct bad input

"Bit Revealing" Idea

- Suppose alg'm wants to compare p & q
- Adversary can set just 1 bit of p and/or q to resolve comparison

  ex 1: p = 10XXXXX, q = 100XXXX

  ex 2: p = 011XXXX, q = 011XXXX
- Note: at any time, an element p's range of possible values is a dyadic interval



The Proof

At start of i-th pass, assume median is in interval I_i, & I_i has n_i elements, all of which are free ($n_i \gg s$)

- Simulate alg'm until $n_i/2$ elements in I_i are encountered. Whenever $8cn/n_i$ bits of an element p has been revealed, fix p's value.
- $\leq 2cn$ bits revealed $\Rightarrow$ $\leq n_i/4$ fixed elements
  Some subinterval I_{i+1} of length $|I_i|/2^{8cn/n_i}$ has $n_{i+1} \geq (n_i/4) / 2^{8cn/n_i}$ free elements
- Adversary sets remaining $n_i/2$ elements in I_i to force median in I_{i+1}

The Proof

$n_{i+1} \geq (n_i/4) / 2^{8cn/n_i} \Rightarrow n/n_{i+1} \leq 2^{O(n/n_i)}$

$\Rightarrow \Omega(\log^*(n/s))$ passes

Part 2:
Rand. RAM Lower Bd

- To show: any comparison RAM alg'm using S bits of space needs $\Omega(n \log \log_S n)$ rand. time
- Pf Idea: use random input (unif. distributed) as bad input

Rand. "Bit Revealing" Idea

- Suppose alg'm wants to compare p & q
- "Adversary" sets next bit of p and/or q at random, until comparison is resolved

  ex: p = 10XXXXX, q = 101XXXX
- # bits revealed ~ geometric distrib. with mean 2

  ~ O(1) amortized w.h.p.
- Note: each element p will then be unif. distributed

Encounter Lemma

- Given interval I_i containing n_i elements, εn steps of alg'm can encounter at most $\sim \varepsilon n_i$ elements in I_i w.h.p.

- Pf:
  - from each fixed state, the sequence of elements in the order they are first encountered by the alg'm is unif. distributed

  - # states $\leq 2^S$

Proof Sketch

At start of i-th round, assume median is in interval I_i, & I_i has n_i elements, many of which are free ($n_i \gg s$)

- Simulate alg'm until ~ εn_i elements in I_i are encountered.

$\Rightarrow$ many elements still free in I_i

$\Rightarrow$ median is likely to be anywhere in a subinterval J_i with ~ $n_i^{1/2}$ elements (without forcing)

- # bits revealed is small

$\Rightarrow$ many subintervals of J_i with ~ $n_{i+1} := n_i^{1/2-\varepsilon}$ elements have many free elements…

Proof Sketch

$n_{i+1} := n_i^{1/2-\varepsilon} \Rightarrow \Omega(\log \log_s n)$ rounds

Closing Remarks

- Time-space lower bds beyond comparison model
  - Sorting: Borodin,Cook [STOC'80], Beame [STOC'89]
  - Others: Ajtai [STOC'99,FOCS'99], Beame,Jayram,Saks [FOCS'98], Beame,Saks,Sun,Vee [FOCS'00], …
  - Median: Chakrabarti,Jayram,Patrascu [SODA'09] $\Omega(\log \log_s n)$ passes for streaming alg'ms for randomly ordered streams

Closing Remarks

- Open problems for median
  - $\Omega(n \log \log_s n)$ rand. lower bd beyond comparison RAM model ??
  - $\Omega(n \log^*(n/s))$ or $\Omega(n \log_s n)$ det. lower bd for comparison RAM model ??