How to Grow Your Lower Bounds

Mihai Pătrașcu

Tutorial, FOCS’10
MIT freshman, 2002

What problem could I work on?
P vs. NP

... half year and no solution later

How far did you guys get, anyway?
What lower bounds can we prove?

“Partial Sums” problem:

Maintain an array $A[n]$ under:
- $update(k, \Delta): A[k] = \Delta$
- $query(k): \text{return } A[1] + \ldots + A[k]$

(Augmented) Binary search trees: $t_u = t_q = O(lg n)$

Open problem: $\max \{ t_u, t_q \} = \Omega(lg n)$

$\Omega(lg n)$ not known for any dynamic problem
What kind of “lower bound”?

Memory: array of $S$ words

Word: \[ w = \Omega(\lg S) \text{ bits} \]

Unit-time operations:

- random access to memory
- $+, -, *, /, \%, <, >, ==, <<, >>, ^, \&, |, ~$

Internal state: $O(w)$ bits

Hardware: $\text{TC}^0$
What kind of “lower bound”? 

*Memory*: array of $S$ words  
*Word*: $w = \Omega(\lg S)$ bits  

Unit-time operations:  
- random access to memory  
- *any* function of two words (nonuniform)
Theorem: \[
\max \{ t_u, t_q \} = \Omega(\lg n)
\]

[Pătraşcu, Demaine  SODA’04]

I will give the full proof.

Maintain an array \(A[n]\) under:

- \(\text{update}(k, \Delta): A[k] = \Delta\)
- \(\text{query}(k): \text{return } A[1]+...+A[k]\)
The hard instance:

\[ \pi = \text{random permutation} \]

for \( t = 1 \) to \( n \):

query(\( \pi(t) \))

\( \Delta_t = \text{random()} \mod 2^w \)

update(\( \pi(t), \Delta_t \))

Maintain an array \( A[n] \) under:

\[ \text{update}(k, \Delta) : A[k] = \Delta \]

\[ \text{query}(k) : \text{return } A[1]+...+A[k] \]
Address and contents of cells $W \cap R$

How can Mac help PC run $t = 9, \ldots, 12$?
How much information needs to be transferred?

PC learns $\Delta_5$, $\Delta_5 + \Delta_7$, $\Delta_5 + \Delta_6 + \Delta_7$

$\Rightarrow$ entropy $\geq 3$ words
The general principle

Message entropy
\[ \geq w \cdot \# \text{ down arrows} \]

\[ E[\text{down arrows}] = (2k-1) \cdot \Pr[\square] \cdot \Pr[\square] \]

\[ = (2k-1) \cdot \frac{1}{2} \cdot \frac{1}{2} = \Omega(k) \]

# memory cells
\[ \begin{cases} \text{* read during mauve period} \\ \text{* written during beige period} \end{cases} \]

\[ = \Omega(k) \]
Every read instruction counted **once**

```latex
\texttt{lowest_common_ancestor}(\texttt{write time}, \texttt{read time})
```

**total** $\Omega(n/\log n)$
Q.E.D.

The optimal solution for maintaining partial sums = binary search trees
What were people trying before?

Fredman, Saks STOC’89
\[ \Omega(\lg n / \lg \lg n) \]

The hard instance:
\[ \pi = \text{random permutation} \]
for \( t = 1 \) to \( n \):
\[ \Delta_t = \text{random()} \mod 2^w \]
update\( (\pi(t), \Delta_t) \)
query\( (\text{random()} \mod n) \)
Build **epochs** of \((\lg n)^i\) updates

\(W_i = \) cells last written in epoch \(i\)

Claim: \(E[\#\text{cells read by query from } W_i] = \Omega(1)\)

\[ \Rightarrow E[t_q] = \Omega(\lg n / \lg \lg n) \]
Focus on some epoch $i$

$x = E[\#\text{cells read by query from } W_i]$ 

Generate $(\log n)^i$ random queries
Entropy = $\Omega(w \cdot \log^i n)$ bits

Possible message:
$W_0 \cup W_1 \cup \ldots \cup W_{i-1}$

$\sum_{j<i} (\log n)^j t_u = O(\log^{i-1} n \cdot t_u)$ cells

cells read by queries from $W_i$

$E[\#\text{cells}] = x \cdot \log^i n$

$\Rightarrow x = \Omega(1)$

Q.E.D.
# Dynamic Lower Bounds

<table>
<thead>
<tr>
<th>Year</th>
<th>Authors</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1989</td>
<td>[Fredman, Saks]</td>
<td>partial sums, union-find</td>
</tr>
<tr>
<td>1991</td>
<td>[Ben-Amram, Galil]</td>
<td></td>
</tr>
<tr>
<td>1993</td>
<td>[Miltersen, Subramanian, Vitter, Tamassia]</td>
<td></td>
</tr>
<tr>
<td>1996</td>
<td>[Husfeldt, Rauhe, Skyum]</td>
<td></td>
</tr>
<tr>
<td>1998</td>
<td>[Fredman, Henzinger]</td>
<td>dynamic connectivity</td>
</tr>
<tr>
<td></td>
<td>[Husfeldt, Rauhe]</td>
<td>nondeterminism</td>
</tr>
<tr>
<td></td>
<td><strong>[Alstrup, Husfeldt, Rauhe]</strong></td>
<td>marked ancestor</td>
</tr>
<tr>
<td>1999</td>
<td>[Alstrup, Ben-Amram, Rauhe]</td>
<td>union-find</td>
</tr>
<tr>
<td>2001</td>
<td>[Alstrup, Husfeldt, Rauhe]</td>
<td>dynamic 2D NN</td>
</tr>
<tr>
<td>2004</td>
<td>[Pătraşcu, Demaine]</td>
<td>partial sums $\Omega(lg n)$</td>
</tr>
<tr>
<td></td>
<td>[Pătraşcu, Demaine]</td>
<td>dynamic connectivity</td>
</tr>
<tr>
<td>2005</td>
<td>[Pătraşcu, Tarniţă]</td>
<td>$\Omega(lg n)$ by epochs</td>
</tr>
<tr>
<td>2010</td>
<td>[Pătraşcu]</td>
<td>proposal for $n^{\Omega(1)}$</td>
</tr>
<tr>
<td></td>
<td>[Verbin, Zhang]</td>
<td>buffer trees</td>
</tr>
<tr>
<td>2011</td>
<td>[Iacono, Pătraşcu]</td>
<td>buffer trees</td>
</tr>
<tr>
<td></td>
<td>[Pătraşcu, Thorup]</td>
<td>dynamic connectivity, union-find</td>
</tr>
</tbody>
</table>
Marked Ancestor

Maintain a perfect B-ary tree under:
mark(v) / unmark(v)
query(v): does v have a marked ancestor?
Marked Ancestor

\[ P[\text{marked}] \approx \frac{1}{\log_B n} \]

query

\[ W_i = \text{cells written by all versions} \]
Reductions from Marked Ancestor

Dynamic 1D stabbing:

Maintain a set of segments $S = \{ [a_1, b_1], [a_2, b_2], \ldots \}$
- insert / delete
- query($x$): is $x \in [a_i, b_i]$ for some $[a_i, b_i] \in S$?

Marked ancestor $\mapsto$ Dynamic 1D stabbing

Dynamic 1D stabbing $\mapsto$ Dynamic 2D range reporting
# Dynamic Lower Bounds

<table>
<thead>
<tr>
<th>Year</th>
<th>Reference</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1989</td>
<td>[Fredman, Saks]</td>
<td>partial sums, union-find</td>
</tr>
<tr>
<td>1991</td>
<td>[Ben-Amram, Galil]</td>
<td></td>
</tr>
<tr>
<td>1993</td>
<td>[Miltersen, Subramanian, Vitter, Tamassia]</td>
<td></td>
</tr>
<tr>
<td>1996</td>
<td>[Husfeldt, Rauhe, Skyum]</td>
<td></td>
</tr>
<tr>
<td>1998</td>
<td>[Fredman, Henzinger], [Husfeldt, Rauhe]</td>
<td>dynamic connectivity, nondeterminism</td>
</tr>
<tr>
<td>1999</td>
<td>[Alstrup, Husfeldt, Rauhe]</td>
<td>marked ancestor</td>
</tr>
<tr>
<td>1999</td>
<td>[Alstrup, Ben-Amram, Rauhe]</td>
<td>union-find</td>
</tr>
<tr>
<td>2001</td>
<td>[Alstrup, Husfeldt, Rauhe]</td>
<td>dynamic 2D NN</td>
</tr>
<tr>
<td>2004</td>
<td>[Pătraşcu, Demaine], [Pătraşcu, Demaine]</td>
<td>partial sums $\Omega(lg n)$</td>
</tr>
<tr>
<td>2005</td>
<td>[Pătraşcu, Tarniţă]</td>
<td>$\Omega(lg n)$ by epochs</td>
</tr>
<tr>
<td>2010</td>
<td>[Pătraşcu]</td>
<td>proposal for $n^{\Omega(1)}$</td>
</tr>
<tr>
<td>2011?</td>
<td>[Iacono, Pătraşcu]</td>
<td>buffer trees</td>
</tr>
<tr>
<td>2011?</td>
<td>[Pătraşcu, Thorup]</td>
<td>dynamic connectivity, union-find</td>
</tr>
</tbody>
</table>
Dynamic Lower Bounds

[Pa˘tra¸cu, Demaine’04]

Partial sums:
\[
\max \{ t_u, t_q \} = B \cdot \lg n
\]
\[
\min \{ t_u, t_q \} = \log_B n
\]
Dynamic Lower Bounds

[Șt. Pătrașcu, Thorup’10]

Dynamic connectivity:
- \( t_u = B \cdot \log n \), \( t_q = \log_B n \)

Maintain an acyclic undirected graph under:
- insert / delete edges
- connected(\( u, v \)): is there a path from \( u \) to \( v \)?
$k$ updates $\geq$ Entropy lower bound $= \Omega(k \cdot w)$ bits

$R \cap W$ = cells read

$W$ = cells written

$k$ updates

$k$ queries
\[ R \cap W \geq \text{Entropy lower bound} \leq \frac{k}{n^{1-\epsilon}} \]

\[ t_u = o(\lg n), \quad t_q = n^{1-\epsilon} \]

W = cells written

k updates

k updates

k/n^{1-\epsilon} queries

R = cells read
Partial sums: Mac doesn’t care about PC’s updates
⇒ communication complexity ≈ \( k/n^{1-\varepsilon} \)

Dynamic connectivity:
nontrivial interaction between Mac’s and PC’s edges
⇒ communication complexity = \( \Omega(k \lg n) \)
communication complexity \geq \Omega(k \lg n) \geq k \text{ updates} \geq k/n^{1-\epsilon} \text{ queries} \\
W = \text{cells written} \\
R = \text{cells read}
Note: |R|, |W| = o(k \lg n)
Note: $|R|, |W| = o(k \lg n)$

$$|R \cap W| = \Omega(k)$$

$$|R \cap W| \cdot \lg n + |R| + |W| \geq \text{nondeterministic communication complexity} \geq \Omega(k \lg n)$$

$W = \text{cells written}$

$k$ updates

$R = \text{cells read}$

$k/n^{1-\varepsilon}$ queries
# Dynamic Lower Bounds

<table>
<thead>
<tr>
<th>Year</th>
<th>Authors</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1989</td>
<td>[Fredman, Saks]</td>
<td>partial sums, union-find</td>
</tr>
<tr>
<td>1991</td>
<td>[Ben-Amram, Galil]</td>
<td></td>
</tr>
<tr>
<td>1993</td>
<td>[Miltersen, Subramanian, Vitter, Tamassia]</td>
<td></td>
</tr>
<tr>
<td>1996</td>
<td>[Husfeldt, Rauhe, Skyum]</td>
<td></td>
</tr>
<tr>
<td>1998</td>
<td>[Fredman, Henzinger]</td>
<td>dynamic connectivity</td>
</tr>
<tr>
<td></td>
<td>[Husfeldt, Rauhe]</td>
<td>nondeterminism</td>
</tr>
<tr>
<td></td>
<td>[Alstrup, Husfeldt, Rauhe]</td>
<td>marked ancestor</td>
</tr>
<tr>
<td>1999</td>
<td>[Alstrup, Ben-Amram, Rauhe]</td>
<td>union-find</td>
</tr>
<tr>
<td>2001</td>
<td>[Alstrup, Husfeldt, Rauhe]</td>
<td>dynamic 2D NN</td>
</tr>
<tr>
<td>2004</td>
<td>[Pătrașcu, Demaine]</td>
<td>partial sums $\Omega(lg n)$</td>
</tr>
<tr>
<td></td>
<td>[Pătrașcu, Demaine]</td>
<td>dynamic connectivity</td>
</tr>
<tr>
<td>2005</td>
<td>[Pătrașcu, Tarniță]</td>
<td>$\Omega(lg n)$ by epochs</td>
</tr>
<tr>
<td>2010</td>
<td>[Pătrașcu]</td>
<td>proposal for $n^{\Omega(1)}$</td>
</tr>
<tr>
<td></td>
<td>[Verbin, Zhang]</td>
<td>buffer trees</td>
</tr>
<tr>
<td>2011?</td>
<td>[Iacono, Pătrașcu]</td>
<td>buffer trees</td>
</tr>
<tr>
<td></td>
<td>[Pătrașcu, Thorup]</td>
<td>dynamic connectivity, union-find</td>
</tr>
</tbody>
</table>
The Multiphase Problem

Dynamic reachability:

Maintain a directed graph under:
- insert / delete edges
- connected\((u,v)\): \(\exists\) path from \(u\) to \(v\)?

Hard-looking instance:

\[ S_1, \ldots, S_k \subseteq [m] \]

\[ T \subseteq [m] \]

\[ S_i \cap T? \]

\[ \text{time} \]
The Multiphase Problem

Conjecture: If $m \cdot t_u \ll k$, then $t_q = \Omega(m^\varepsilon)$
The Multiphase Problem

Conjecture: If $m \cdot t_u \ll k$, then $t_q = \Omega(m^\varepsilon)$

Follows from the 3SUM Conjecture:

**3SUM**: Given $S = \{ n \text{ numbers} \}$, $\exists a, b, c \in S$: $a+b+c = 0$?

Conjecture: 3SUM requires $\Omega^*(n^2)$ time
The Multiphase Problem

**Conjecture:** If $m \cdot t_u \ll k$, then $t_q = \Omega(m^\epsilon)$

Attack on unconditional proof:

3-party number-on-forehead communication

\[ S_1, \ldots, S_k \subseteq [m] \]

\[ T \subseteq [m] \]

\[ S_i \cap T? \]

\[ \text{time } O(k \cdot m \cdot t_u) \]

\[ \text{time } O(m \cdot t_u) \]

\[ \text{time } O(t_q) \]

\[ \text{time } \]
Static Data Structures
Asymmetric Communication Complexity

Input: $O(w)$ bits

Can’t look at total communication.
Theorem. Either Alice communicates \( a = \Omega(\lg m) \) bits, or Bob communicates \( b \geq m^{1-\varepsilon} \) bits.
Indexing

**Theorem.** Either Alice communicates $a = \Omega(\lg m)$ bits, or Bob communicates $b \geq m^{1-\varepsilon}$ bits.

$m/2^a$ positions of $v$ fixed to 1

$\Rightarrow b \geq m/2^a$
Lopsided Set Disjointness

Theorem. Either Alice communicates $\Omega(n \lg m)$ bits, or Bob communicates $\geq n \cdot m^{1-\varepsilon}$ bits

Direct sum on Indexing:
- deterministic: trivial
- randomized: [Pătraşcu’08]
A Data Structure Lower Bound

Partial match:

Preprocess a database $D = \text{strings in } \{0,1\}^d$

$\text{query}( x \in \{0,1,*\}^d ) : \text{does } x \text{ match anything in } D ?$

$C : [m] \rightarrow \{0,1\}^{3\lg m} \text{ constant-weight code}$

$A = \{ n \times \text{’s} = \{ (1,x_1), \ldots, (n,x_n) \} \}$

$\quad \rightarrow \text{query}( C(x_1) \circ \ldots \circ C(x_n) )$

$B = \{ \frac{1}{2}mn \bullet \text{’s} \} \mapsto D = \{ \frac{1}{2}mn \text{ strings} \}$

$(i,x_i) \mapsto 0 \circ \ldots \circ 0 \circ C(x_i) \circ 0 \circ \ldots$
A Data Structure Lower Bound

LSD \((n, m)\)

Alice sends \(\Omega(n \lg m)\) bits,
or Bob sends \(\geq n \cdot m^{1-\varepsilon}\) bits

\[ \leftrightarrow \text{Partial Match: } d = \Theta(n \lg m) \]

CPU sends \(\Omega(d)\) bits,
or Memory sends \(\geq |D|^{1-\varepsilon}\) bits

\[ \text{Partial Match: } d = \Theta(n \lg m) \]

\[ \text{Alice sends } \Omega(n \lg m) \text{ bits,} \]

or Bob sends \(\geq n \cdot m^{1-\varepsilon}\) bits

\[ \leftrightarrow \text{query( } C(x_1) \circ \ldots \circ C(x_n) \text{ )} \]

\[ \leftrightarrow \text{query( } C(x_1) \circ \ldots \circ C(x_n) \text{ )} \]

\[ A=\{ n \times \text{’s} \} = \{ (1,x_1), \ldots, (n,x_n) \} \]

\[ \text{B=\{ } \frac{1}{2}mn \bullet \text{’s} \} \leftrightarrow D = \{ \frac{1}{2}mn \text{ strings } \} \]

\[ (i,x_i) \leftrightarrow 0 \circ \ldots \circ 0 \circ C(x_i) \circ 0 \circ \ldots \]
**A Data Structure Lower Bound**

LSD$(n, m)$

Alice sends $\Omega(n \lg m)$ bits, or Bob sends $\geq n \cdot m^{1-\varepsilon}$ bits

$\mapsto$ Partial Match: $d = \Theta(n \lg m)$

$\mapsto$ CPU sends $\Omega(d)$ bits, or Memory sends $\geq |D|^{1-\varepsilon}$ bits

$\Rightarrow t \lg S = \Omega(d)$ or $t \cdot w \geq |D|^{1-\varepsilon}$

$\Rightarrow S = 2^{\Omega(d/t)}$

upper bound $\approx$ either:
- exponential space
- near-linear query time

Diagram:

- $t_q$ axis
- $n^{1-o(1)}$ axis
- $S$ axis
- $O(n)$ axis
- $2^{O(d)}$ axis
- Graph curves showing $O(d/\lg n)$, $n^{1-o(1)}$, $O(n)$, and $2^{O(d)}$.
## Space Lower Bounds for $t_q = O(1)$

<table>
<thead>
<tr>
<th>Year</th>
<th>Reference</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1995</td>
<td>[Miltersen, Nisan, Safra, Wigderson]</td>
<td></td>
</tr>
<tr>
<td>1999</td>
<td>[Borodin, Ostrovsky, Rabani]</td>
<td>partial match</td>
</tr>
<tr>
<td>2000</td>
<td>[Barkol, Rabani]</td>
<td>randomized exact NN $S = 2^{\Omega(d)}$</td>
</tr>
<tr>
<td>2003</td>
<td>[Jayram, Khot, Kumar, Rabani]</td>
<td>partial match</td>
</tr>
<tr>
<td>2004</td>
<td>[Liu]</td>
<td>deterministic $O(1)$-apx NN $S = 2^{\Omega(d)}$</td>
</tr>
<tr>
<td>2006</td>
<td>[Andoni, Indyk, Pătraşcu]</td>
<td>randomized $(1+\varepsilon)$-apx NN $S = n^{\Omega(1/\varepsilon^2)}$</td>
</tr>
<tr>
<td></td>
<td>[Pătraşcu, Thorup]</td>
<td>direct sum for near-linear space partial match $S = 2^{\Omega(d)}$</td>
</tr>
<tr>
<td>2008</td>
<td>[Pătraşcu]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[Andoni, Croitoru, Pătraşcu]</td>
<td>$\ell_\infty$: $\text{apx} = \Omega(\log \rho \log d)$ if $S = n^\rho$</td>
</tr>
<tr>
<td></td>
<td>[Panigrahy, Talwar, Wieder]</td>
<td>$c$-apx NN $S \geq n^{1+\Omega(1/c)}$</td>
</tr>
<tr>
<td>2009</td>
<td>[Sommer, Verbin, Yu]</td>
<td>$c$-apx distance oracles $S \geq n^{1+\Omega(1/c)}$</td>
</tr>
<tr>
<td>2010</td>
<td>[Panigrahy, Talwar, Wieder]</td>
<td></td>
</tr>
</tbody>
</table>
Asymmetric Communication Complexity

Input: $O(w)$ bits

$w$ bits

$\lg S$ bits

$\lg S$ bits

Input: $n$ bits

No difference between $S=O(n)$ and $S=n^{O(1)}$
Separation $S = n \log^{O(1)} n$ vs. $S = n^{O(1)}$

<table>
<thead>
<tr>
<th>Year</th>
<th>Reference</th>
<th>Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>2006</td>
<td>[Pătraşcu, Thorup]</td>
<td>predecessor search, exact near neighbor</td>
</tr>
<tr>
<td></td>
<td>[Pătraşcu, Thorup]</td>
<td></td>
</tr>
<tr>
<td>2007</td>
<td>[Pătraşcu]</td>
<td>2D range counting</td>
</tr>
<tr>
<td>2008</td>
<td>[Pătraşcu]</td>
<td>2D stabbing, etc.</td>
</tr>
<tr>
<td></td>
<td>[Panigrahya, Talwar, Wieder]</td>
<td>c-apx. near neighbor</td>
</tr>
<tr>
<td>2009</td>
<td>[Sommer, Verbin, Yu]</td>
<td>c-apx. distance oracles</td>
</tr>
<tr>
<td>2010</td>
<td>[Panigrahya, Talwar, Wieder]</td>
<td>c-apx. near neighbor</td>
</tr>
<tr>
<td></td>
<td>[Greve, Jørgensen, Larsen, Truelsen]</td>
<td>range mode</td>
</tr>
<tr>
<td>2011</td>
<td>[Jørgensen, Larsen]</td>
<td>range median</td>
</tr>
</tbody>
</table>

$\times$ = Tight bounds for space $n \log^{O(1)} n$
2D Stabbing

Static 2D Stabbing:
Preprocess $D = \{ n \text{ axis-aligned rectangles} \}$

query($x,y$): is $(x,y) \in R$, for some $R \in D$?

Goal: If $S = n \lg^{O(1)} n$, the query time must be $t = \Omega(\lg n / \lg \lg n)$

Remember: Dynamic 1D Stabbing

Maintain a set of segments $S = \{ [a_1,b_1], [a_2,b_2], \ldots \}$

insert / delete

query($x$): is $x \in [a_i, b_i]$ for some $[a_i, b_i] \in S$?

We showed: If $t_u = \lg^{O(1)} n$, then $t_q = \Omega(\lg n / \lg \lg n)$
Persistence

**Persistent** : \{ dynamic problems \} \mapsto \{ static problems \}

Given dynamic problem \( \mathcal{P} \) with \( \text{update}_\mathcal{P}(x) \), \( \text{query}_\mathcal{P}(y) \)

**Persistent(\mathcal{P})** = the static problem

Preprocess \((x_1, x_2, \ldots, x_T)\) to support:

\( \text{query}(y, t) = \text{the answer of} \ \text{query}_\mathcal{P}(y) \ \text{after} \ \text{update}_\mathcal{P}(x_1), \ldots, \text{update}_\mathcal{P}(x_t) \)
Persistence

Persistent : \{ \text{dynamic problems} \} \mapsto \{ \text{static problems} \}

Given dynamic problem $\mathcal{P}$ with $\text{update}_\mathcal{P}(x)$, $\text{query}_\mathcal{P}(y)$

$\text{Persistent}(\mathcal{P}) = \text{the static problem}$

Preprocess $(x_1, x_2, \ldots, x_T)$ to support:

$\text{query}(y, t) = \text{the answer of} \text{query}_\mathcal{P}(y) \text{after} \text{update}_\mathcal{P}(x_1), \ldots, \text{update}_\mathcal{P}(x_t)$

Static 2D stabbing $\leq$ Persistent(dynamic 1D stabbing)
Recap: Marked Ancestor
Persistent (Marked Ancestor)
Persistent (Marked Ancestor)
Persistent (Marked Ancestor)
Persistent (Marked Ancestor)

Butterfly graph!
Butterfly Reachability

Let $G =$ butterfly of degree $B$ with $n$ wires

Preprocess a subgraph of $G$

query($u, v$) = is there a path from source $u$ to sink $v$?

# vertices: $N = n \cdot \log_B n$
# edges: $N \cdot B$

Database = \{x’s\}
= vector of $N \cdot B$ bits
Butterfly Reachability $\mapsto$ 2D Stabbing

Let $G$ = butterfly of degree $B$ with $n$ wires

Preprocess a subgraph of $G$

query($u,v$) = is there a path from source $u$ to sink $v$?

Preprocess $D$ = \{ axis-aligned rectangles \}

query($x,y$): is $(x,y) \in R$, for some $R \in D$?

Which sources & sinks care about this edge?

\[ \text{low bits}(u) = \text{low bits of } a \]
\[ \text{high bits}(v) = \text{high bits of } b \]
Butterfly Reachability $\mapsto$ 2D Stabbing

Let $G =$ butterfly of degree $B$ with $n$ wires

Preprocess a subgraph of $G$

$\text{query}(u,v) =$ is there a path from source $u$ to sink $v$?

Preprocess $D = \{ \text{axis-aligned rectangles} \}$

$\text{query}(x,y):$ is $(x,y) \in R$, for some $R \in D$?

query($u$, $v$) $\mapsto$ (u, reverse-bits($v$))
Hardness of Butterfly Reachability

query\((u,v)\)

Input: 2 \(\lg n\) bits

\(\Omega(\lg n)\) bits

\(\lg S\) bits

\(O(\lg n)\) bits

Input: \(N \cdot B\) bits

Either Alice sends \(\Omega(\lg n)\) bits
or Bob sends \(B^{1-\epsilon}\) bits
query(\(u_1, v_1\))

... 

query(\(u_n, v_n\))

\[\lg(S) \approx n \lg \frac{S}{n} = O(n \lg \lg n)\]

Either Alice sends \(n \times \Omega(\lg n)\) bits \(\Rightarrow t = \Omega(\lg n / \lg \lg n)\)

or Bob sends \(n \times B^{1-\varepsilon}\) bits
Hardness of Butterfly Reachability

Either Alice sends $n \times \Omega(\lg n)$ bits
or Bob sends $n \times B^{1-\varepsilon}$ bits
Either Alice sends $n \times \Omega(\lg n)$ bits
or Bob sends $n \times B^{1-\varepsilon}$ bits
A \cap B = \emptyset \iff \text{query}(1,8) = \text{true} \land \text{query}(2,5) = \text{true} \land \ldots

LSD lower bound:
Either Alice sends \( n \times \Omega(\log n) \) bits
or Bob sends \( n \times B^{1-\varepsilon} \) bits

\[ A = \{ \log_B n \text{ matchings} \} = \{ N \uparrow \text{’s}\} \]
\[ \rightarrow n \text{ queries} \]

B = \{ \frac{1}{2} NB \xmark \text{’s} \} \rightarrow \text{database}
Bibliography: Predecessor Search

Preprocess a set $S = \{ n \text{ integers} \}$

$\text{predecessor}(x) : \max \{ y \in S \mid y \leq x \}$

1988  [Ajtai]  $1^{\text{st}}$ static lower bound
1992  [Xiao]
1994  [Miltersen]
1999  [Beame, Fich]  optimal bound for space $n^{O(1)}$
  [Chakrabarti, Chazelle, Gum, Lvov]**
2001  [Sen]  randomized
2004  [Chakrabarti, Regev]**
2006  [Pătraşcu, Thorup]  optimal bound for space $n \lg^{O(1)}n$
  $1^{\text{st}}$ separation between polynomial and linear space
2007  [Pătraşcu, Thorup]  randomized

**) Work on approx. nearest neighbor
Bibliography: Succinct Data Structures

On input of \( n \) bits, use \( n + o(n) \) bits of space.

[Gál, Miltersen ’03] polynomial evaluation
\[ \Rightarrow \text{redundancy} \times \text{query time} \geq \Omega(n) \]

[Golynski ’09] store a permutation and query \( \pi(\cdot), \pi^{-1}(\cdot) \)
If space is \( (1+\varepsilon) \cdot n \lg n \) bits \( \Rightarrow \) query time is \( \Omega(1/ \sqrt{\varepsilon}) \)

[Pătrașcu, Viola ’10] prefix sums in bit vector
For query time \( t \) \( \Rightarrow \) redundancy \( \geq n / \lg^{O(t)} n \)

NB: Also many lower bounds under the indexing assumption.
The End