

PREDECESSOR

Y set of integers, "the keys". $|Y| = n$.

Find

$$\text{pred}(x, Y) = \max \{ y \in Y \mid y \leq x \}.$$

"key length" = length (in bits) of keys.

"word length" = w (in bits)

"space" S = number of words

Throughout these notes,
O means Ω .

$$a := \log \frac{S}{n} + \log w$$

KNOWN CONSTRUCTIONS:

$$P(n, l) = O(1) + P\left(\frac{n}{2}, l\right) \quad \text{binary search tree}$$

results in $O(\log n)$ algorithm

$$P(n, l) = O(1) + P\left(\frac{n}{w}, l\right) \quad \text{Fredman-Willard}$$

results in $O(\log_w n)$ algorithm

$$P(n, l) = O(1) + P\left(n, l/2\right) \quad \text{Van Emde Boas}$$

results in $O(\log w)$ algorithm

$$P(n, l) = \begin{cases} O(1) + P(n, l/2) \\ O(1) + P(n/S^{1/k}, l) \end{cases}, \text{ or} \quad \text{Beame-Fitch, using space } S.$$

no control over which case the reduction uses.

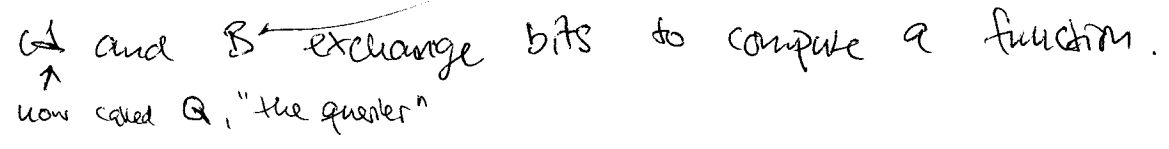
Also, tabulation is a good trick if you have enough space.

Mission statement:

want to prove that these tricks are essentially the only constructions that can be used for ~~the~~ the predecessor problem.

ROUND ELIMINATION

Communication problem: ^{now called D "the data structure"}



New restriction:

- Communication happens m rounds.



Original function:

$f: A \times B \rightarrow \{0,1\}$. From this we construct.

\uparrow Alice's input \leftarrow Bob's input

~~the~~ $f^{A(k)}: A^k \times B \times [k] \rightarrow \{0,1\}$

\uparrow Alice's input \uparrow Bob's input

$$((a_1, \dots, a_k), b, i) \mapsto f(a_i, b)$$

Intuition: If Alice speaks first, her first

"half-round" will be wasted (because she

does not know i)

THE ROUND ELIMINATION LEMMA

Claim: If $f^{(k)}$ can be solved in

$$[A; m_1, m_2, \dots, m_t]$$
 with annotations:

- who speaks first \downarrow A
- first message bits \swarrow m_1
- last message bits \searrow m_t

 , error ϵ

then f can be solved in

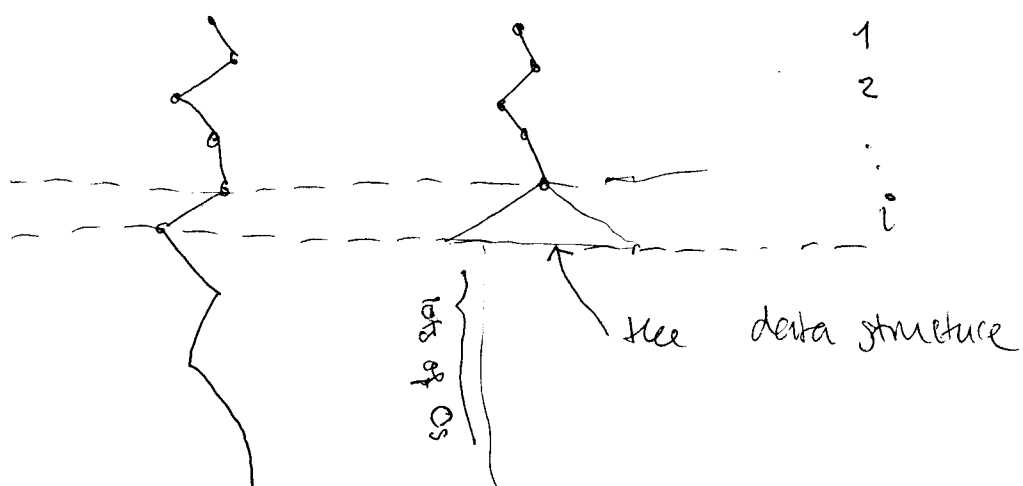
$$[B; m_2, \dots, m_t]$$
 , error $\epsilon + O\left(\sqrt{\frac{m_t}{t}}\right)$

Given $(q_1, d_1), \dots, (q_k, d_k)$ according to the distribution over $f: A \times B$.

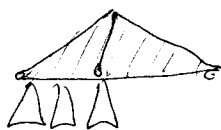
Construct the input

(q_1, \dots, q_k) as query

$(q_1, q_2, \dots, q_{i-1}, d_i, \bar{0})$ as data
 \uparrow
 lots of 0s.



Viewed from the ~~the~~ queries



Every time we shave off ~~the~~ Q 's round:

$$\text{key length: } l \rightarrow l / \log^2 s.$$

Every time we shave off one of D 's rounds:

$$\text{problem size: } n \rightarrow n / w^2.$$

How often can we do this?

$$t = \Omega \left(\min \left\{ \frac{\log w}{\log(\log^2 s)}, \frac{\log n}{\log(w^2)} \right\} \right)$$

For example: $s = n^{O(1)}$, bound becomes

$$\min \left\{ \frac{\log w}{\log \log n}, \frac{\log n}{\log(w^2)} \right\} \quad \text{so}$$

$$t = \Omega \left(\min \left\{ \frac{\log w}{\log \log w}, \sqrt{\frac{\log n}{\log \log n}} \right\} \right).$$

This is the Beame-Fickl bound

what if k is much smaller than n ?

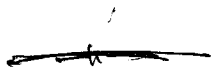
MESSAGE COMPRESSION LEMMA:

If $f^{*(k)}$ can be computed in

$$[A; m_1, m_2, \dots, m_t] \text{ and error } \epsilon$$

then f can be computed in

$$[A; \frac{O(1 + m_1/k)}{\delta^2}, m_2, \dots, m_t] \text{ and error } \epsilon + \delta.$$



OBSERVATION

If f can be computed in

$$[A; m_1, \dots, m_t]$$

then f can be computed in

$$[B; \underbrace{2^{m_1} m_2, m_1 + m_3, m_4, m_5, \dots, m_t}_{t-1 \text{ rounds}}]$$

(Easy: Bob starts by sending all his ~~answer~~ end-round answer)

Using both these reductions once, we reduce

$$l \rightarrow l/k, \quad n \rightarrow \cancel{n/m_2}$$

$$\text{so } m_1' \rightsquigarrow m_1/k \quad O(\log s/k)$$

$$m_2' \rightsquigarrow 2^{m_1} m_2 \quad s^{O(1/k)} w$$

$$\text{so } n \rightarrow n/m_2' \quad n/s^{O(1/k)} w$$

Gives matching upper and ~~the~~ lower bounds for space $n^{1+\epsilon}$.

To be concrete,

we can shave off rounds $\frac{\log w}{\log k}$ times until

the keys disappear, or $\frac{\log n}{\log(s^{1/k} \cdot w)}$ times until there

are no more elements, so for

$$\min \left(\frac{\log w}{\log k}, \frac{\log n}{\log(s^{1/k})}, \frac{\log n}{\log w} \right) \text{ rounds.}$$

This is maximised at

$$k = \log_n \frac{\log w \cdot \log s}{\log n} \quad \text{which means}$$

PREDECESSOR SEARCH = IP LOOKUP.

In practice, solved by B-trees, using lots of space. E.g., tabulate the first 20 bits of a 32 bit IP number.

Message: dirty RAM tricks are used in practice, and comparison based models do not suffice.

Rest of talk:
Small space, i.e. $n^{1+o(1)}$ space

Back to the cell probe model.

$Q \leftrightarrow T$

Q : intelligent queries

DS : ~~data~~ data structure, $T(Y)$

For the sake of the induction,

~~DS~~ publishes bits ~~DS~~ $PB(Y)$

Queries look at his input x and $PB(Y)$ and then either
rejects (x) or

adds x to $T(Y)$
returns $\text{pred}(x, Y)$.

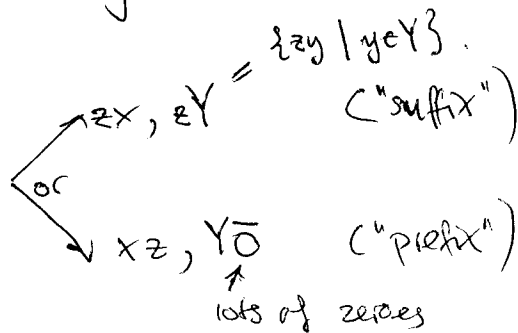
IDEA:

Eliminate probe from full solution with space s , $PB(Y) = \emptyset$, no rejects.

Construct solution to $P(n, \frac{w}{2})$ with space s , $|PB(Y)| = O(\sqrt{n})$, rejects (x, Y) with prob $\frac{1}{2}$ according to some distribution $\mathcal{D} : z^n \times z^{w/2}$.

Given (x, Y) according to \mathcal{D} .

Pick (z, \cdot) according to \mathcal{D} and construct



To the queries, zx and xz look the same (they are sampled from the same distributions).

How does the data structure look? Two cases

suffix case:
 if $|probe_on_input(zx)| \leq \sqrt{s}$ then publish $T(Y)_{probe_on_input(zx)}$ in the first round.

(i.e., all the probe table ~~ent~~ entries and their contents).

otherwise it does not publish anything

Prefix case =

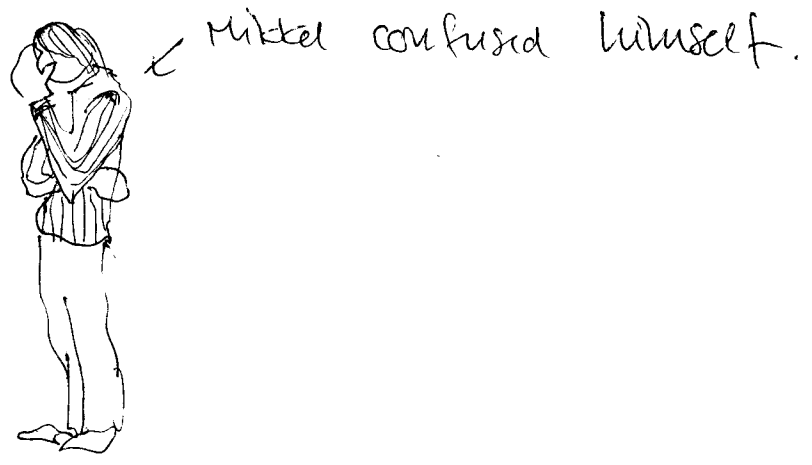
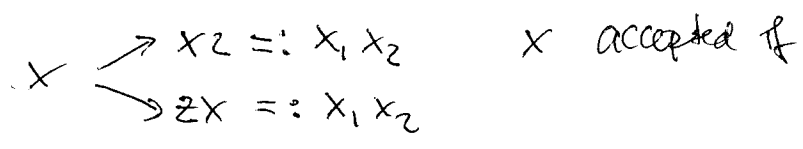
Data structure publishes $\tau_S \cdot w \in$ random cells.

In both cases, the querier ~~simulates~~ eliminates its first rand.

- ① either the answer is in PB
in which case we are happy
- ② or it isn't, in which case Q rejects

Lemma Reject at most $\frac{1}{2}$ of the inputs.

PF: Notation:



$\Theta(10)$

After deconfusion:

In the prefix case, the queries actually simulate all

first-probe ($x*$)

and sees if the answer is in PB.
Reject if no such $*$ exists.

(Intuition: This works because the datastructure stores yo anyway, so the second half does not matter.)

==

"

$xz =: x_1 x_2$

x accepted if $|\text{probe}(x_1*)| > \sqrt{s}$

$zx =: x_1 x_2$

x accepted if $|\text{probe}(x_1*)| \leq \sqrt{s}$.

□

Key Lemma: $\oplus^k P(n, l)$ accept rate δ .

Assume $|PB| = O(k\delta^3)$

Can eliminate probe for

$$\oplus^k P(n, l/2)$$

accept rate $\delta/8$

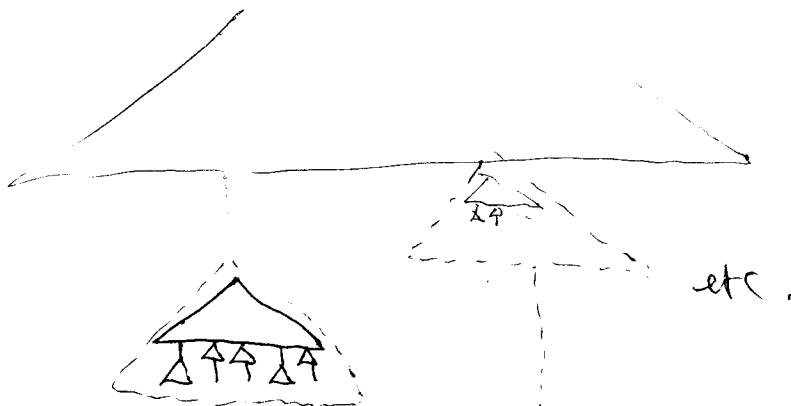
$$|PB| = \tilde{O}(\sqrt{ks}) = \tilde{O}(k \cdot \sqrt{s/k})$$

root of # bits per subproblem

number of subproblems

To reapply the key lemma repeatedly transform

$$\oplus^k P(n, l/2) \text{ into } \oplus^{\tilde{O}(\sqrt{ks}/\delta^3)} P\left(\frac{n}{\sqrt{ks}/\delta^3}, l/2\right)$$



Application: VEB is optimal, i.e.

$$t = \underline{O}(\log \log n) \text{ for}$$

$$d = 3 \log n, \quad s = n^{1+o(1)}$$

//

$u := u_0$, $l := l_0$ initial # elements, key length

$$u_{i+1} = u_i^{1/4}$$

$$k_i = n / u_i$$

$$l_i = l / 2^i, \text{ "almost"}$$

repeat until

$$u_t = 2^{\sqrt{\log n}}$$

By arguing that before that, we still have a nontrivial problem,

$$t = \underline{O}(\log \log n) \text{ follows.}$$

Concrete open problem: dynamic predecessor

$$N = 3 \lg u$$

$$\tilde{O}(u) \text{ space} \xrightarrow{\substack{\uparrow \\ \text{known}}} t_q \Theta(\log \log u)$$

can we have

$$t_u = o(\log \log u) \quad ???$$

or prove this can't be done