Deterministic Dictionaries

6.851 Lecture 15
André Schulz notes

[Hagerup, Miltersen, Pagh '01]

<u>Goal</u> : build a ~~static~~ static dictionary for w-bit keys in $O(n \log n)$ time on a word-RAM using $O(n)$ space <u>deterministically</u> !

History of deterministic dictionaries

- [Tarjan, Yao '79] $O(n)$ construction if w is $O(\log n)$
  n, u polynomially related
  (Universe : $\{0, \ldots, 2^w - 1\}$)
  keys fit in a single word
  (for $n = 2^w$) Table can be stored as bitvector (many entries)
- [Fredman, Komlós, Szemerédi '84] : deterministic FKS needs $O(n^3 w)$
- [Raman '96] : Refinement of FKS idea : $O(n^2 w)$

- Remark : we can build $O(n)$ ~~table~~ ds in constant query time if $w \in O(n^{\Omega(1)})$ [Fusion Trees]
- [Alon, Naor '96] : $O(n w \log^4 n)$ but queries need $\Theta\left(\frac{w}{\log n}\right)$ few entries

today $O(n \log n)$ construction
$O(1)$ look-up } deterministic
$O(n)$ space

Abstract ~~...~~ construction

(1) universe reduction ($|\mathcal{U}| = n^{O(1)}$)
(Error correcting codes)

(2) Further universe reduction ($|\mathcal{U}| = n^2$)
(Trees)

(3) Solution for $u = n^2$
(Displacement method)

1. Universe reduction to $u = n^{O(1)}$

We want $\phi : \{0,1\}^w \to \{0,1\}^r$, such that

- $r = O(\log n)$
- ϕ is 1-1 on S

Let $\psi : \{0,1\}^w \to \{0,1\}^{4w}$ be an error correcting code with relative minimum distance $\delta > 0$.

This means: $\forall x, y (x \neq y)$: ~~x & y d~~
$\psi(x)$ & $\psi(y)$ differ at at least ~~most~~
$4\delta w$ bits

Remark: $\delta \leq \frac{1}{2}$ can be picked for $w \geq 2$

Lemma There exist a set $D \subseteq [4w]$ of distinguishable bit positions for $\psi(S)$ (i.e. $\psi(x)$ & $\psi(y)$, $x \neq y$, differ at one of the bits of D) with $|D| \leq 2 \log n / \cancel{\log 1/\delta}$

Proof: We construct D incrementally:

$D_0 \subseteq D_1 \subseteq D_2 \quad \ldots \quad D \subseteq D_k \subseteq \{1, \ldots 4w\}$

For D_j we have a natural partition of S: ~~$\{ C(D, v \in \{0,1\}^{|D|}$~~
$\{ C(D, v), v \in \{0,1\}^{|D|} \}$ with

$C(D, v) = \{ w \mid w \in S$, the bits of w at the positions of D match $v \}$

Cluster

-3-

$$B(D) = \sum'_{\nu \in \{0,1\}^{|D|}} \binom{C(D,\nu)}{2} = \text{Collisions}$$

coll(D_i) = pairs not separated by D_i

<u>Example</u>

$D = \{d_1, d_2, d_3\}$

		d_1			d_2		d_3	
$\psi(x)$	0	1	0	1	0	1	0	1
$\psi(y)$	1	1	1	0	0	1	0	0

$\Rightarrow x, y \in C(\{d_1, d_2, d_3\}, 100)$

Fact: $\psi(x)$ & $\psi(y)$ can match on at most $2\cancel{4}w \cancel{(1-\delta)}$ positions

$\Rightarrow$ For at most $\cancel{4}2w \cancel{(1-\delta)}$ new bits x & y stay in the same cluster

$d \in [4w]$

$\{x,y\} \in C(D_i, \nu),\ x \neq y \Rightarrow \sum'_d C(\{d\} \cup D_i, \nu 0) + C(\{d\} \cup D_i, \nu 1) \leq 2w$

$$\sum'_{d \in D} B(D_i \cup \{d\}) \leq 2w\, B(D_i)$$

$$\Rightarrow \frac{\sum_{d=1}^{4w} B(D_i \cup \{d\})}{\cancel{(1-\delta)}4w} \leq \frac{1}{2} B(D_i) \cancel{(1-\delta)}$$

(Averaging principle) $\Rightarrow$

$$\exists d : \quad B(D_i \cup \{d\}) \leq \frac{B(D_i)}{2} \cancel{(1-\delta)}$$

Notice: it might be that $D_i \cup \{d\} = D_i$

We use $D_{i+1} = D_i \cup \{d\}$

Since $D_0 = \emptyset \Rightarrow B(D_0) \le \binom{n}{2} \le n^2$

we reach $B(D_k) < 1$ for

$k = \lfloor 2 \log_{1/(1-\delta/2)} n \rfloor$

$\Rightarrow |D| < \cancel{O(n^2)}\ O(\log n)$ $\square$

Lemma A bit position d:

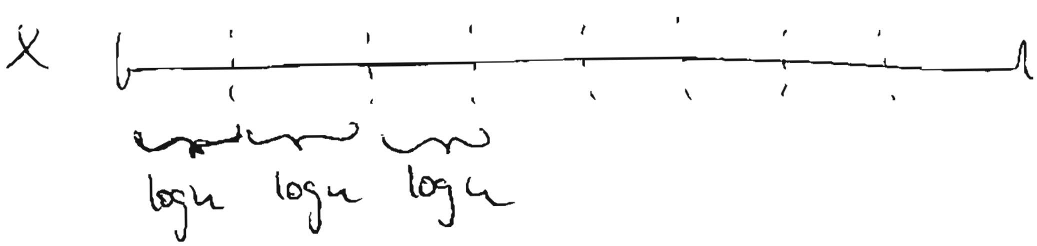
$$B(D \cup \{d\}) \le (1 - \tfrac{\delta}{2}) B(D)$$

can be found in $O(n)$ time & space deterministically

Without proof (Bit-Tricks)

2. Universe reduction to $\mathcal{U} = [n^2]$

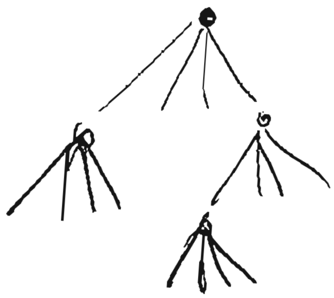
split $x \in \mathcal{U}$ into $\log n$ chunks



$\Theta(1)$ many chunks

"# of possible $\log n$ Blocks" $= n \Rightarrow$

Store ~~$x \in$~~ S in a trie with $|\Sigma_i| = n$



- Fan-out $\leq n$
- nodes : n

↳ Every node gets an ID

To find a node we build a table that can answer queries of the form

$(ID, a) \longrightarrow$ which node can I reach from node ID following $a \in \Sigma_i$

$O(n^2)$ tuples to store in Hash-table

We also store in every node of the trie if it is terminal

$\Rightarrow$ We can find $x \in S$ (and observe if $x \notin S$) using the queries

$\Rightarrow$ Search time is $O(1)$ because the height of the trie is $O(1)$

3. Solution for a quadratic universe

We search a perfect hash function

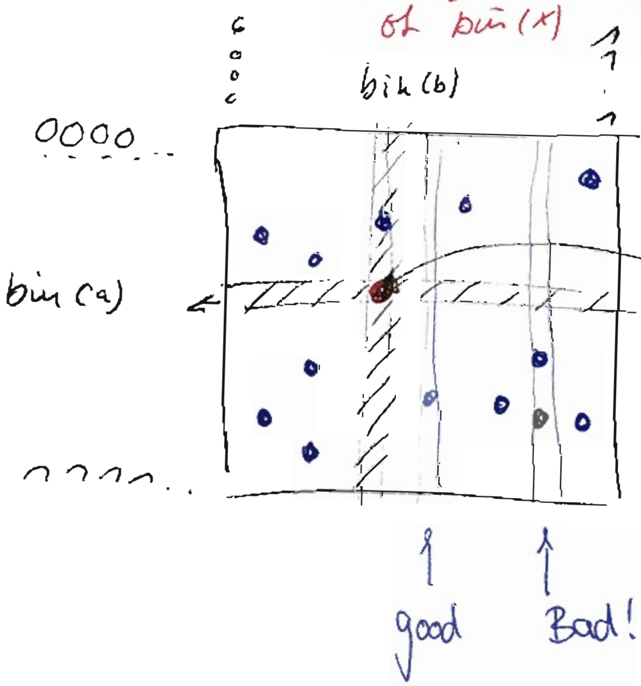
$$h : [n^2] \longrightarrow [2^r], \quad r = \log n + O(1)$$

We first obtain a function

$$\bar{h} : [n^2] \to [2^r]^2$$

$$\bar{h}(x) := (\mathrm{low}(x), \mathrm{high}(x)), \quad \bar{h}(x) \text{ is 1-1 on } S$$

1st $\frac{n}{2}$ Bits of bin(x) 2nd $\frac{n}{2}$ Bits of bin(x)

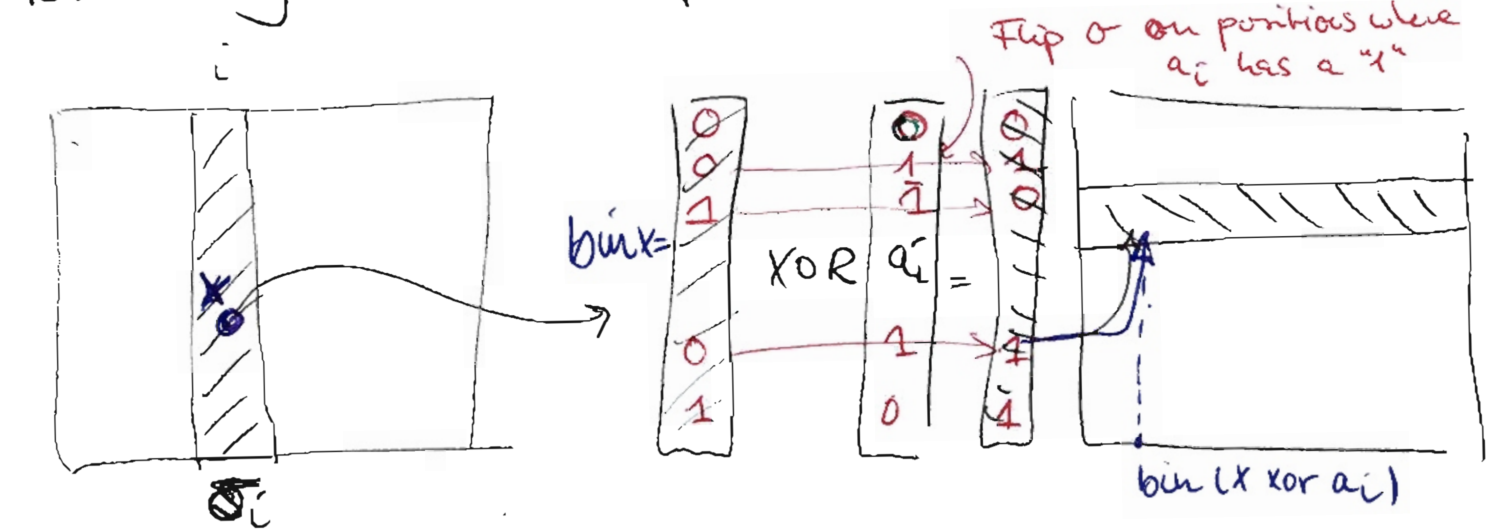


$\bar{h}(x) = (a, b)$

if every column would contain ~~exa~~ only 1 "entry" of $\{\bar{h}(x) \mid x \in S\}$ we can use the column# as hash value

<u>Idea</u> : Scramble the table

For every column σ_i pick some $a_i \in \{0,1\}^r$



→ compute σ_i xor a_i and add the vector as row i in the new table

Formal: $(x,y) \in \text{Table} \longrightarrow (y \text{ XOR } a_x, x) \in \text{NewTable}$

q = collisions in the columns of Table

q' = collisions in the columns of NewTable

Lemma: There is a set of $\{a_i\}_{i \le r}$ such that $q' \le \min\{n, \lfloor \frac{q}{2^{r-3}} \rfloor n\}$

Proof: (1) Reorder columns, such that

$|S_1| \ge |S_2| \ge \dots |S_{2^r}|$

↑ S_i = row nrs. of elements in the ith column of Table

$S_i = \{x \mid (i,x) \in \bar{h}(S)\}$

- Notice: $q = \sum_i \binom{|S_i|}{2}$

- (2) Pick $a_1, a_2, a_3 \dots a_{2^r}$ in increasing order

Assume we ~~insert~~ want to find a_i

→ pick a_i uniformly at random from $\{0,1\}^r$

↳ Let m_j be the collisions on NewTable at column j so far (Def.)

~~Alt~~

$$\#\text{new collisions by picking } a_i = \sum_{y \in S_i} m_{\underbrace{y \text{ xor } a_i}_{\text{def.}}}$$

$$\hookrightarrow \mathbb{E}\Big(\sum_{y \in S_i} m_{y \text{ xor } a_i}\Big) = \sum_{y \in S_i} \mathbb{E}(m_{y \text{ xor } a_i})$$

$$= \sum_{y \in S_i} \sum_{u \in \{0,1\}^r} \underbrace{\mathbb{P}[(y \text{ xor } a_i) = u]}_{\frac{1}{2^r}} \cdot m_u$$

$$= \sum_{y \in S_i} \frac{1}{2^r} \underbrace{\sum_{u \in \{0,1\}^r} m_u}_{= \sum_{j<i} S_j}$$

$$= \frac{|S_i| \sum_{j<i} |S_j|}{2^r} \quad (*)$$

By Markov's inequality:

$\mathbb{P}[\text{new collisions} \leq 2 \cdot (*)] \geq \frac{1}{2}$

↳ After 2 tries we found a a_i vector:

$\#$ new collisions $\leq \lfloor 2^{-r+1} |S_i| \sum_{j<i}' |S_j| \rfloor = M_i$

↳ Floor, because $\#$ is $\in \mathbb{Z}$

$$\Rightarrow \quad q' \leq \sum_i' \left\lfloor |S_i| \sum_{j<i}' |S_j| \cdot \cancel{2} \cdot 2^{-r+1} \right\rfloor$$

<u>Consequence</u> for $r = \log n + 4$

(A) $q' \leq \sum_i' |S_i| \, n \cdot 2^{-r+1} \leq n^2 2^{-r+1} \leq n$

(B) $q' \leq \sum_i' M_i$

(B.1) $|S_i| = 1 \Rightarrow M_i = \left\lfloor 2 \cdot \sum_{j<i}' \frac{|S_i|}{2^r} \right\rfloor \leq \left\lfloor 2 \cdot \frac{n}{2^r} \right\rfloor = 0$

(B.2.) $|S_i| > 1 \Rightarrow \forall j < i : |S_j|^2 \leq 4 \binom{|S_j|}{2}$

$$\Rightarrow M_i \leq \left\lfloor 2 \cdot \sum_{j<i}' \frac{|S_j|^2}{2^r} \right\rfloor \leq \left\lfloor 2^{-r+3} \sum_{j<i}' \binom{|S_j|}{2} \right.$$

$\leq q$

$$\hookrightarrow q' \leq \sum_i \lfloor 2^{-r+3} q \rfloor \leq n \cdot \lfloor 2^{-r+3} q \rfloor$$

□

$\Rightarrow$ 1st ~~shuffeling~~ shuffle

lemma
$\hookrightarrow \quad q' \le n$

2nd ~~shuffeling~~ shuffle

lemma
$\hookrightarrow \quad q'' \le \left\lfloor \frac{q}{2^{r+3}} \right\rfloor \cdot n = \underbrace{\left\lfloor \frac{n}{2^{\log n + 1}} \right\rfloor}_{0} \cdot n = \underline{\underline{0}}$

Derandomization of the selection of a_i

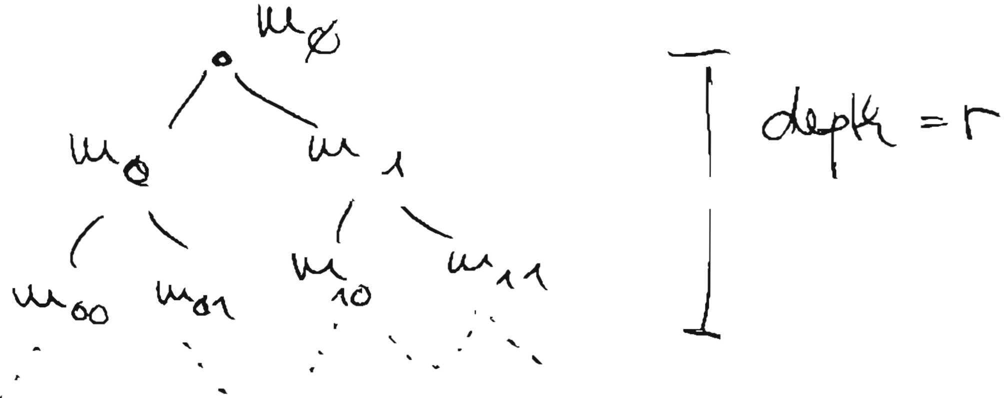
- ~~to~~ Extension of notation

  $\forall z \in \{0,1\}^{k} \qquad m_z = \sum_{v \in \{0,1\}^{r-k}} m_{z \circ v}$

  $k < r$

- We want to keep track of all m_* during the ~~Stuf~~ search for a_i

  $\hookrightarrow$ Store current m_* values inside a binary trie



→ ~~Adding~~ New a_i value → update trie

↳ insert all $y \in S_i$ in the trie by computing $a_i \circ y$ and increase all m_x's on the associated root → leaf path

↳ $O(|S_i| \cdot r)$ for every a_i

$\Rightarrow O(r \cdot n) = O(n \log n)$ total

Derandomize by the technique of

CONDITIONAL EXPECTATIONS

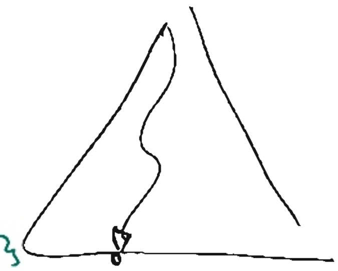
IDEA · We know that $\mathbb{E}(q')$ is "good" $(\leq *)$ for some a_i we have to pick

→ ~~Either~~

$\mathbb{E}(q' / a_i \text{ starts with } z)$ is also good for either $z=0$ or $z=1$

→ We use the trie to determine ~~the~~ the value of z and determine a_i bit by bit

$$\mathbb{E}(q') = \sum_{y\in S_i} \sum_{u\in\{0,1\}^r} m_u \frac{1}{2^r} = \sum_{y\in S_i} \frac{1}{2^r} m_y$$



Walk to reach a leaf

$\frac{1}{2^r}$

maybe use $S_i^{(z)} = \{x \in S_i \mid x = z\ldots\}$ notation

$$\mathbb{E}(q' \mid a_i \text{ starts with } 0) = \sum_{\substack{y\in S_i\\ y=1..}} \sum_{u\in\{0,1\}^{r-1}} \frac{1}{2^{r-1}} m_{1u} + \sum_{\substack{y\in S_i\\ y=0..}} \sum_{u\in\{0,1\}^{r-1}} \frac{1}{2^{r-1}} m_{0u}$$

$|S_i^{(1)}|$ $\quad$ $|S_i^{(0)}|$

$$= \frac{1}{2^{r-1}}\Big(|\{y\in S_i \mid y=1..\}| \cdot m_1 + |\{y \in S_i \mid y=0..\}| \cdot$$

$$\mathbb{E}(q' \mid a_i \text{ starts with } 1) = \frac{1}{2^{r-1}}\Big(|\{y\in S_i \mid y=0..\}|\Big) m_0 + \frac{1}{2^{r-1}} |\{y\in S_i \mid y=1\ldots\}| m_1$$

Check which one is ~~larger~~ smaller and use the corresponding bit in a_i

↳ iterate

↳ in the end

no randomness here

$$\mathbb{E}(q' \mid a_i = \underbrace{01001\ldots01}_{\text{something like that}}) \leq (*)$$

$\Rightarrow q'$ for $a_i = 01\ldots\ldots \leq (*)$

Idea dynamic perfect Hashing

[Dietzfelbinger et al '94]

- ~~Every~~ After $n' = c \cdot n$ Insert/Delete Opn rebuild

- Delete: mark a node as deleted if it is stored

- Insert: Lookup node: If marked as deleted, unmark

  If not there:
  - Add in table
  - If collision, rehash 2nd level table
  - If too many elements in the 2nd level table ↳ rehash everything

$\Rightarrow O(n)$ space, $O(1)$ per Opn in expectation & amortized

-/-