Úsporné datové struktury

- Chceme uložit prvek množiny C (třeba jeden ze stromů na n vrcholech) a použít téměř optimální prostor ... $OPT = \lceil \log |C| \rceil$ bitů } entropie při rovnoměrném rozdělení psti
- Přitom chceme rychlé operace
- Variandy:
  - implicitní DS – jen data sama ve vhodném pořadí (setříděné pole, halda)] $OPT + O(1)$ ↑ rounding &c.
  - úsporné (succinct) DS – $OPT + o(OPT)$
  - kompaktní DS – $O(OPT)$ ... ale pozor, třeba BVS nemusí být kompaktní kvůli pointerům – nepočítáme slova, ale bity! nebo trie

Problém: Reprezentace řetězce nad obecnou abecedou

Příklad: $[10] \to 4$ bity $(\log 10 \doteq 3.322)$
$[10]^2$ ... mohu uložit jako $2 \times 4b$ nebo jako $[100] \to 7$ bitů
$[10]^k$ ... uložím jako $[10^k] \to k \cdot \log 10 + 1$ namísto $k \cdot (\log 10 + 1)$
↳ limitně se blížím k $\log 10$ b/znak (redundance = délka kódu − entropie) ... ale ztrácím lokální dekódovatelnost] jde k 0

Praktické řešení: dělím na bloky o 9 znacích ... délka kódu = 30
entropie $9 \cdot \log 10 \doteq 29.897$
→ redundance $\doteq 0.103$
↳ pro string délky n: $30 \cdot \lceil n/9 \rceil \leq 30/9\, n + 30$
↳ redundance $\leq 0.103 \cdot \frac{n}{9} + 30$

Blok se vejde do 1 slova → počítám s ním v $O(1)$ času i prostoru

↳ ☺ $(a,b) \in X \times Y$ kóduji po složkách ⇒ délky kódů i entropie se sečtou → redundance také

Příklad #2: Posíláme proud bitů, ale dopředu nevíme, jak bude dlouhý. → instantní (prefixový) kód
[potřebujeme umět zakódovat značku pro konec, aniž by neustálé "ještě to pokračuje" zabralo dost místa]

Triviální řešení: rozdělíme na bloky o b bitech, za ∀ blok připíšeme 1 bitovou značku, do posledního bloku padding 10^k
↳ redundance $\frac{n}{b} + b + 1$

SOLE Encoding [Dodis, Pătraşcu, Thorup 2010]
(Short-Odd-Long-Even)

- vstup rozdělíme na bloky o b bitech → prvky abecedy $[B]$ pro $B = 2^b$
- poslední blok dopadujeme ($10 \text{---} 0$ na konec), ale potřebujeme přidat spec. blok označující EOF
  ↳ převod $[B+1]^* \to [B]^*$
- nastavíme $b \geq 2\log n + 2 \Rightarrow B \geq 4n^2$] bloky se vejdou do $O(1)$ slov RAMu

(2)

Číslo bloku	1	2	3	4	5	6	...	k
vstupní abeceda	B	B	B	B	B	B	...	EOF
+ EOF	B+1	B+1	B+1	B+1	B+1	B+1		B+1
1. průchod	B	B+3	B−3	B+6	B−6	B+9	...	0
2. průchod	B	B	B	B	B	B	...	

Sem si domyslíme nulový blok

1. průchod: $(B+1)(B+1) \leq (B-3i)(B+3i+3)$

$$\cancel{B^2} + 2B + 1 \leq \cancel{B^2} + \cancel{3Bi} + 3B - \cancel{3Bi} - 9i^2 - 9i$$

$$-B + 1 \leq -9i^2 - 9i$$

$$B - 1 \geq 9i^2 + 9i$$

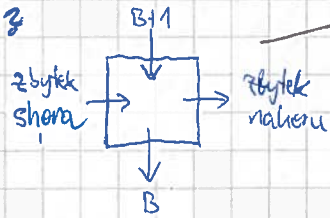
$$B \geq 9i^2 + 9i + 1, \quad \text{neboť } i \leq \frac{n+1}{2} \text{ a } B \geq 4n^2$$

2. průchod: $(B+3i)(B-3i) = B^2 - 9i^2 \leq B^2$

- redundance $\leq 3b$ (1 blok zaokrouhlení + 1 EOF + 1 extra na lichý # bloků)
- proudové kódování + dekódování
- lokální dekódování i změny v $O(1)$ na RAMu [potřebuji $\Theta(\log n)$ bitů]
  ↳ $O(1)$ slov

zaokrouhlení na celé bity je příliš redundantní

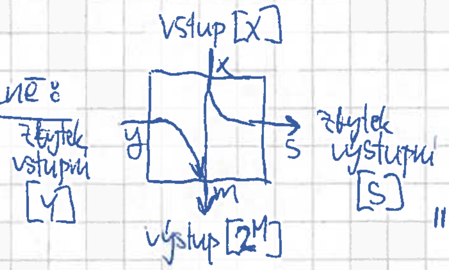
Postupně z toho odvodíme obecnou konverzi abeced...
- proč to vlastně fungovalo?



nechám zbýt malé množství informace a to v příštím kroku přimíchám k většímu, což snižuje redundanci

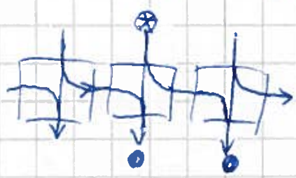
... zbytky se posílají jen do omezené vzdálenosti
→ lokálně dekódovatelné

Obecně:



"MIXÉR"

... při řetězení stále lok. dekódovatelné



pro dekódování ⊛ stačí znát oba ●

Lemma: Nechť $X, Y \leq 2^w$. Pak $\exists M, S$ a zobrazení $f: [X] \times [Y] \to [2^M] \times [S]$ t. ž.:

① $S = O(\sqrt{X})$, $2^M = O(Y \cdot \sqrt{X})$ a S, M závisí pouze na X, Y

② f lze vyhodnotit (na RAMu) v čase $O(1)$

③ x lze dekódovat z m, s v $O(1)$

④ y lze dekódovat ze samotného m v $O(1)$

⑤ Redundance je $O(1/\sqrt{X})$ bitů

$$\underbrace{(M + \log S)}_{\text{entropie výstupu}} - \underbrace{(\log X + \log Y)}_{\text{entropie vstupu}}$$

Dk: Zvolíme M (později)

↳ $[2^M]$ musí obsahovat celé y a nějakou část x' čísla x … jakou? $C := \lfloor 2^M / Y \rfloor$ možností

$$[X]\ x \to x'\ [C],\quad x \to s\ [S],\quad x', [Y]\,y \to [2^M],\quad [S] = \lceil X/C \rceil$$

dokud každý mezivýsledek používáme jen 1x, redundance můžeme sčítat (steleskopují se)

- redundance ~~…~~ kódování $x', y \to m$:

$$R_1 = M - \log(\underbrace{Y \cdot C}) \le M - \log(2^M - Y) = \log \frac{2^M}{2^M - Y} = \log \frac{1}{1 - \frac{Y}{2^M}} = O\left(\frac{Y}{2^M}\right) = O\left(\frac{1}{C}\right)$$

$$\llcorner \log\left(\underbrace{Y \cdot \left\lfloor \frac{2^M}{Y} \right\rfloor}_{\ge 2^M - Y}\right)$$

$e^x \ge 1 + x$

$x \ge \log(1+x)$

$-x \le \log \frac{1}{1+x}$

$x \ge$ ~~…~~ $\log \frac{1}{1-x}$

- redundance rozkladu $x \to x', s$:

$$R_2 = (\log C + \log S) - \log X = \log C + \log \left\lceil \frac{X}{C} \right\rceil - \log X = \log \frac{C \cdot \lceil \frac{X}{C} \rceil}{X} \le \log \frac{X + C}{X} =$$

$$= \log\left(1 + \frac{C}{X}\right) = O\left(\frac{C}{X}\right).$$

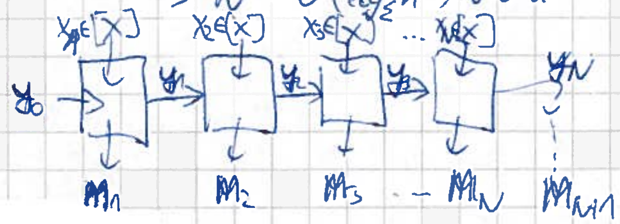
- celkem tedy minimalizujeme: $O\left(\frac{1}{C} + \frac{C}{X}\right)$ … $C \approx \sqrt{X} \to S = O(\sqrt{X}),\ 2^M = O(Y \cdot \sqrt{X})$

→ redundance $O(1/\sqrt{X})$, jak jsme slíbili. [nezávisí na Y]

První pokus o reprezentaci stringu

$A \in [\Sigma]^n$ … rozdělíme na bloky o velikosti $\Theta(\log_\Sigma n)$ tak, aby $X \approx n^2$

$\Rightarrow N = O\left(\frac{n}{\log_\Sigma n}\right)$ bloků



$y_i \in [O(\sqrt{X})] = [O(n)]$

$m_i \in [O(\sqrt{X} \cdot \sqrt{X})] = [O(n^2)]$

… redundance $O\left(\frac{1}{n}\right)$ v každém kroku

↳ celkově $O(1)$, což je milé

… lokální kódování / dekódování

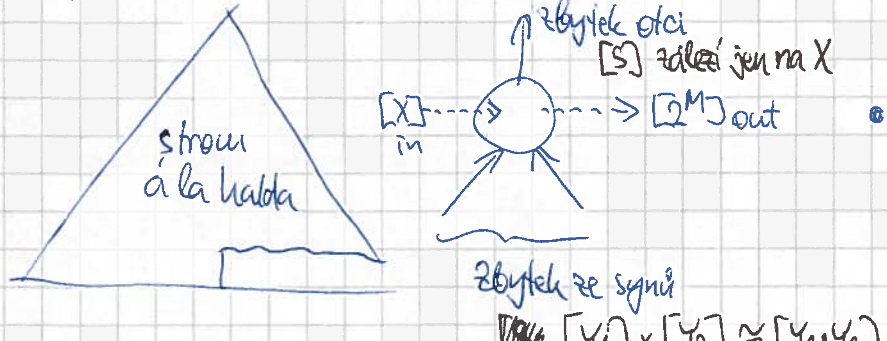
OOPS! Každý blok má jiné parametry!

→ potřebujeme tabulku $O\left(\frac{n}{\log n}\right)$ konstant!

↳ tohle v SOLE nevadilo, protože $X = 2^b + 1$, takže jsme jednotlivá Y_i uměli aproximovat aritmetickou posloupností … ale obecně to nebude fungovat

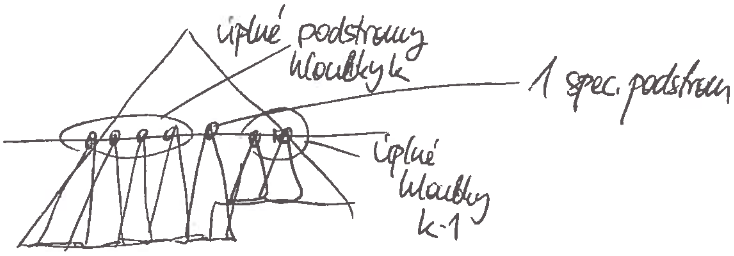
Záchrana: stromové kódování

Opět volíme $X := \Theta(n^2)$, $N = O\left(\frac{n}{\log_\Sigma n}\right)$



- dekódování je stále lokální (z otce stačí out, ten určuje zbytek)

~~…~~ $[Y_1] \times [Y_2] \cong [Y_1 \cdot Y_2]$



na každé hladině 3 typy vrcholů, pro ně si pamatujeme:

- # vrcholů tohoto typu
- adresu dat 1. vrcholu
- parametry mixéru

celkem $O(\log n)$ konstant (slov)

↓

+ potřebujeme tabulku $O(\log n)$ mocnin $|\Sigma|$ na extrakci symbolů z bloku

Dekódování: pozice ve streamu → číslo bloku = pozice ve stromu

$\log i$ (v čase $O(1)$) → hladina → typ → adresa v paměti

Vše $O(1)$

Lokální změna též v $O(1)$.

Celková redundance $O(1)$ [jako u předch. pokusu] + $O(1)$

↑ za kódování bloků

↑ zaokrouhlení posledního (neúplného) bloku (ten kódujeme atypicky)

Věta: Na Word-RAMu lze reprezentovat prvek $[\Sigma]^n$ v prostoru $\lceil n \log \Sigma \rceil + O(1)$ bitů se čtením/zápisem prvku v čase $O(1)$ s použitím $O(\log n)$ konstant závislých na n a Σ.