Předelira: Intervalové dotazy v pologrupách

$(X, \oplus)$, $\oplus$ je asociativní

příklady: min, +, ·, násobení matic ...

Chceme statickou DS, která pro $x_1 \ldots x_n \in X$ umí rychle vyhodnocovat $x_i \oplus x_{i+1} \oplus \ldots \oplus x_j$

Df: $f_i(n)$:= kolik prostoru stačí, abychom uměli dotaz vyhodnocovat pomocí i operací $\oplus$
↳ čas na předvýpočet bude vycházet ~ stejný

- $f_0(n) = \Theta(n^2)$ ... předpočítáme všechno
- $f_1(n)$ ... rekurzivní konstrukce [A | B], $n/2$, $n/2$
  - dotazy přes střed: px + sx součty (px pro B, sx pro A)
  - ostatní: rekurze na A nebo B

$$\left.\begin{array}{l} f_1(n) = n + 2\cdot f_1(n/2) \\ f_1(1) = 0 \end{array}\right\} f_1(n) = n \log n$$

Cvičení: dotaz v $O(1)$ [předpočítáme vhodné tabulky]

- $f_2(n)$ ... nic zajímavého
- $f_3(n)$ ... opět rekurzivně

bloky velikosti b → n/b bloků

- px + sx v každém bloku
- f_3 v každém bloku
- f_1 pro posloupnost součtů bloků

$$f_3(n) = \underbrace{2n}_{\text{px+sx součty}} + \underbrace{f_1(n/b)}_{\text{komplet. bloky}} + \underbrace{\frac{n}{b}\cdot f_3(b)}_{\text{vnitřky bloků}}$$

volbou $b := \log n$ bude $f_1\left(\frac{n}{b}\right) \leq n$

dotaz:
- uvnitř 1 bloku: f_3 v bloku, $3\times \oplus$
- $3\times \oplus$: sx $\oplus$ f_1 přes celé bloky $\oplus$ px

tedy $f_3(n) \leq 3n + \frac{n}{\log n}\cdot f_3(\log n)$

$f_3(1) = 0$

Σ přes každou hladinu $= n$

$f_3(n) \leq 3n \cdot \log^* n$

- obecně: $f_{2k+1}(n) \leq (2k+1)\, n \log^{\overbrace{*\ldots*}^{k}} n$

[dokážeme indukcí zobecněním postupu pro f_3]

- optimum:

$\alpha(n) := \min\{k \mid \log^{\overbrace{*\ldots*}^{k}} n \leq 2\}$

pak pro $k = \alpha(n)$ získáme:

Předzpracování v $O(n \cdot \alpha(n))$

Dotaz v $O(\alpha(n))$

↳ nutno domyslet, jak vše stihnout (zatím jsme dokázali, že stačí tolikrát vyhodnotit $\oplus$)

příklady:

g	g^*
$n-1$	$n-1$
$n-2$	$n/2$
$n-3$	$n/3$
$n/2$	$\log n$
n/c	$\log_c n$
$\sqrt{n}$	$\log\log n$
$\log n$	$\log^* n$ ☺

pokud pro i operací $\oplus$ stačí prostor $f'(n) = k' n \cdot g(n)$ (lib. neklesající fce), pak pro $i+2$ operací $\oplus$ použijeme $b := g(n)$, z čehož $(n/b)\cdot f'(n/b) \leq k'n$, takže:

Obecně:

$$f(n) = kn + \frac{n}{g(n)}\cdot f(g(n))$$

$k = k'+2$

⇓

$$f(n) = kn \cdot g^*(n)$$

kde $g^*(n) = 0$ pro $n \leq 1$

$g^*(n) = 1 + g^*(g(n))$ pro $n > 1$

tedy $g^*(n) = \min\{i \mid g^{(i)}(n) \leq 1\}$

(def. pro g: $\forall n\ g(n) < n$)

Speciální případy: $\oplus = + \Rightarrow$ prefixové součty $O(n) / O(1)$ (Build / Query)

$\oplus = \min \rightarrow$ RMQ $O(n) / O(1)$

dynamické $\rightarrow$ intervalové stromy $O(\log n) / O(\log n)$ (Query / Update), $O(n)$ ~~init~~ Build

Union-Find Problem, [pův. Tarjan & van Leeuwen 1984, hezčí analýza od Seidela 200x]

- udržujeme ekvivalenci na $\{1 \ldots n\}$
  - začínáme se samými 1prvkovými třídami
  - Union(x,y) sloučí třídy obsahující x, y
  - Find(x) zjistí třídu obsahující x (vrátí reprezentanta)
- alternativně: udržujeme komponenty souvislosti grafu, Union přidává hranu

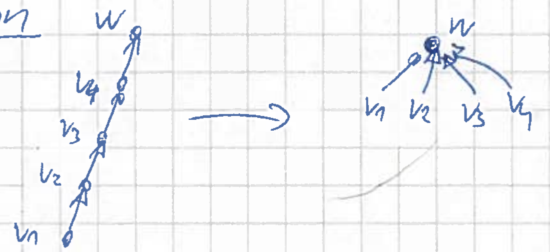
Reprezentace: ∀ třídu reprezentujeme stromem orientovaným do kořene (reprezentant třídy)

... vrchol si pamatuje svého otce.

Optimalizace: 1) Union by rank ... kořeny si pamatují rank $r(v)$
Pokud $r(u) < r(v)$, pověsíme u pod v,
jinak libovolně, ale novému kořeni zvýšíme rank o 1

Důsledky:
- rank = hloubka stromu
- strom ranku r obsahuje aspoň 2^r vrcholů
- všechny ranky jsou $\leq \log n$
- stromy mají hloubku $\leq \log n$
- Union i Find trvají $O(\log n)$

2) Path Compression



kdykoli nějakou cestu projdeme, zkomprimujeme si do ~~hv~~ hvězdy

Postupně ukážeme, že Path Compression zaručuje sama o sobě čas $O(\log n)$ amort.
a 1 + 2 společně budou daleko lepší.

Ceny operací měříme počtem přepojení pointerů cost(op): (nepočítáme první přiřazení)

Find trvá $O(1 + \text{délka cesty}) = O(1 + \text{cost})$

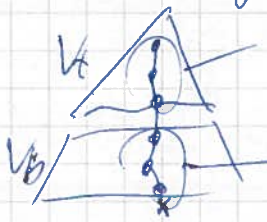
Union trvá $O(1)$ + 2x Find, tedy také $O(1 + \text{cost})$.

Trik: Nejprve provedeme všechna spojení stromů, (zde se pointery nepřepojují (pouze definují))
teprve pak všechny komprese cest (ale vždy se zastavíme ve vrcholu, který původně byl kořenem) (vše podstatné se děje zde)

Df: Rozklad ~~stromu~~ lesa na množině vrcholů V je (V_t, V_b) taková, že:

1. $V_t \cup V_b = V$, $V_t \cap V_b = \emptyset$ (rozklad v množinovém smyslu)
2. V_t je "uzavřená nahoru" – tedy otec vrcholu z V_t leží zase ve V_t.

Nápad: Uvážíme nějaký rozklad a sledujeme, jak ho protínají jednotlivé komprese:



komprese uvnitř V_t ⟶ normální komprese $comp(x,y)$
→ cost ≈ délka cesty

vrcholy přijdou uvnitř V_b o otce (cesta se rozpadne) ⟶ speciální komprese $comp(x,\infty)$
→ cost $= 0$ ... musíme účtovat jinak

Notace:
$\mathcal{F}$ = les na množině vrcholů X
C = posloupnost kompresí
$\|C\|$ = #normálních kompresí v C
$cost(C)$ = celková cena kompresí v C

} Obecně nás zajímá, kolik nejvýš může být $cost(C)$ vzhledem k $|X|$ a $\|C\|$.

Lemma: Nechť C je posloupnost kompresí v lese $\mathcal{F}$ na množině X a (X_t, X_b) je rozklad lesa $\mathcal{F}$ indukující lesy $\mathcal{F}_t$ a $\mathcal{F}_b$.

Potom $\exists\, C_t, C_b$ posloupnosti kompresí pro $\mathcal{F}_t$ a $\mathcal{F}_b$ takové, že:

$$\|C_t\| + \|C_b\| \le \|C\| \quad (*)$$

& $$cost(C) \le cost(C_t) + cost(C_b) + |X_b| + \|C_t\| \quad (\#)$$

$- \#roots(\mathcal{F}_b)$ [silnější lemma, které se bude hodit později]

Dk: C_t, C_b získáme přímočaře:

1. komprese leží celá uvnitř $\mathcal{F}_t$ → jde do C_t
2. analogicky $\mathcal{F}_b$ → C_b
3. jde napříč → speciální komprese uvnitř $\mathcal{F}_b$, normální uvnitř $\mathcal{F}_t$

} dohromady přispěje k $\|\cdot\|$ 1× → z toho → $(*)$

Nyní dokážeme $(\#)$:

$cost(C)$:

T mění otce na T ------ platíme z $cost(C_t)$

B mění otce na B ------ platíme z $cost(C_b)$

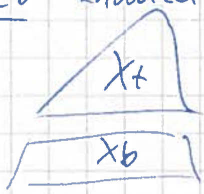
B mění otce na T → poprvé --- platíme z $|X_b|$ $(-\#roots(\mathcal{F}_b))$ pro silnější verzi

↘ znovu --- platíme z $\|C_t\|$ (v každé T-kompresi to nastane max. 1×)

Df: $f(m,n) :=$ max. cost libovolné posloupnosti kompresí C t.ž. $\|C\| = m$ na stromu s n vrcholy.

Věta: $f(m,n) \le (m+n)\log n$.

Dk: Indukcí ... rozdělíme X na X_t, X_b velikosti $n/2$



$\|C\| = m$

z lemmatu: $\exists C_t, C_b \quad \|C_t\| + \|C_b\| \leq \|C\|$
$$m_t + m_b \leq m$$
$$\& \quad cost(C) \leq cost(C_t) + cost(C_b) + |X_b| + \|C_t\|$$

Z indukce: $cost(C) \leq (m_t + n/2)\log n/2 + (m_b + n/2)\log n/2 + n/2 + m_t$
$$\leq m(\log n/2 + 1) + n(\log n/2 + 1) = (m+n)\log n.$$

Důsledek: Union i Find provedený m-krát na n-prvkové množině trvá celkem $O((m+n)\log n)$.

Nyní přidáme Union by rank

Df: Rankový les je les s funkcí $r: V \to \mathbb{N}$ t.ž.

① $r(v)$ = výška podstromu s kořenem v (měřená v hranách)

② $\forall v \ \forall i = 0, \ldots, r(v)-1 \ \exists w$ syn vrcholu v t.ž. $r(w) = i$.

Union by rank bez komprese cest produkuje rankové lesy ... a kompresi provádíme až dodatečně, takže nepřekáží.

Vrchol ranku r má alespoň r synů, jeho podstrom obsahuje alespoň 2^r vrcholů.

Lemma: Nechť $\mathcal{F}$ je rankový les ~~s kořenem~~ ranku r (na množině X), s číslo $(0 \leq s < r)$.

rankem lesa myslíme max. rank kořenů

Pak množiny $X_t := \{x \in X \mid r(x) > s\}$ a indukované lesy $\mathcal{F}_t$
a $X_b := \{x \in X \mid r(x) \leq s\}$ $\mathcal{F}_b$

splňují:

① (X_t, X_b) je rozklad lesa $\mathcal{F}$.

② $\mathcal{F}_b$ je rankový les s rankem $\leq s$

③ $\mathcal{F}_t$ je rankový les s rankem $\leq r-s-1$

[④ $|X_t| \leq |X| / 2^{s}$] nebudeme potřebovat

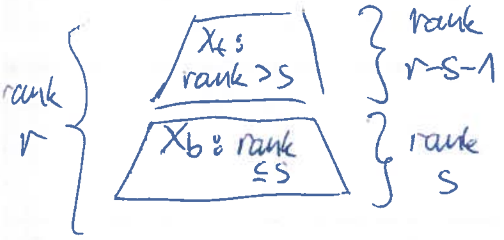
Počítejme $f(m,n,r)$:= max. cost posloupnosti m kompresí v lese ranku r s n vrcholy

i po (částečné) kompresi platí, že směrem nahoru ranky ostře rostou

$f(m,n,r) \le (r-1)n$ … přepojením vrcholu vzroste rank otce

$\le (r-1)m$ … cesta má max. $r+1$ vrcholů, takže přepojím max. $r-1$ z nich

Z rozkladu stromu, dostaneme:



$n_t := |X_t|$ $\qquad C \to C_t \qquad m_t := \|C_t\|$

$n_b := |X_b|$ $\qquad\quad\;\; C_b \qquad m_b := \|C_b\|$

Podle lemmatu o rozkladu: $n_t + n_b = n,\ m_t + m_b \le m$

& $\mathrm{cost}(C) \le \mathrm{cost}(C_t) + \mathrm{cost}(C_b) + |X_b| - \#\mathrm{roots}(F_b) + \|C_t\|$

$\le f(m_t, n_t, r-s-1) + f(m_b, n_b, s) + n - n_t - (s+1)n_t + m_t$

$\forall v \in X_t$ má aspoň $s+1$ synů v X_b a jsou to navzájem různé kořeny stromů v F_b

Protože: $\boxed{f(m,n,r) \le f(m_t,n_t,r-s-1) + f(m_b,n_b,s) + n - (s+2)n_t + m_t}$

Zkusíme rekurzivní krok: Předpokládejme, že $f(\mu,\nu,\varrho) \le k\cdot\mu + \nu\cdot g(\varrho)$

Potom: $f(m,n,r) \le k\cdot m_t + \underbrace{n_t\cdot g(r-s-1)}_{\le n_t\cdot g(r)} + \underbrace{f(m_b,n_b,s)}_{\le f(m_b,n,s)} + n \underbrace{-(s+2)n_t}_{\le -s\cdot n_t} + m_t$

volbou $s := g(r)$: $f(m,n,r) \le (k+1)m_t + f(m_b, n, g(r)) + n \qquad / -(k+1)(\underbrace{m_b+m_t}_{\le m})$

$$\underbrace{f(m,n,r) - (k+1)m}_{\text{označíme } \varphi(m,n,r)} \le \underbrace{f(m_b,n,g(r)) - (k+1)m_b}_{\text{toto tedy je } \varphi(m_b,n,g(r))} + n$$

Tedy: $\varphi(m,n,r) \le \varphi(m_b,n,g(r)) + n$

$\downarrow$

$\varphi(m,n,r) \le n\cdot g^*(r)$

$\downarrow$

$f(m,n,r) \le n\cdot g^*(r) + (k+1)m$

$\boxed{\text{Posouvací lemma: Pokud } f(m,n,r) \le km + n\cdot g(r), \text{ pak také } f(m,n,r) \le (k+1)m + n\cdot g^*(r)}$

Důsledek: $f(m,n,r) \le (k+i)m + n\cdot g^{\overbrace{*\cdots*}^{i}}(r)$

Kde ale začít? Z triv. odhadu máme $f(m,n,r) \le (r-1)n$

... tedy $k=0$, $g(r)=r-1$ – jenže $g^*(r)$ je také $r-1$ ↯

První krok tedy uděláme trochu jinak. Vyjdeme z ⊛8 (dosadíme triv. mez)

$$f(m,n,r) \le n_t\cdot(r-s-2) + f(m_b,n_b,s) + n - (s+2)n_t + m_t$$

$$\le n_t\cdot(r-2s-4) + f(m_b,n_b,s) + n + m_t$$

$s:=\lfloor r/2 \rfloor$

$$\le f(m_b,n_b,r/2) + n + m_t$$

Znovu: $f(m,n,r) - m \le f(m_b,n_b,r/2) - m_b + n$

Tedy: $\boxed{f(m,n,r) \le m + n\cdot \log r}$ ← to již můžeme iterovat

Iterováním: $f(m,n,r) \le (i+1)m + n\cdot \log^{\overbrace{*\ldots*}^{i}} r$

Volba i: (1) $\alpha(r) := \min\{i \mid \log^{\overbrace{*\ldots*}^{i}}(r) \le i\}$

$$\Rightarrow f(m,n) \le (1+\alpha(\log n))(m+n)$$

(2) $\alpha(m,n,r) := \min\{i \mid \log^{*\ldots*}(r) \le m/n\}$

$$\Rightarrow f(m,n) \le (2+\alpha(m,n,\log n))m$$