VÍCEROZMĚRNÉ DS ← především geometrické

- Reprezentujeme množinu bodů $X \subseteq \mathbb{R}^k$ ← Obecněji by to mohly být třeba úsečky, kvádry…
- Typy dotazů: – hledání konkrétního bodu
  - – intervalový dotaz: $X \cap \underbrace{[a_1,b_1]\times \_\_ \times [a_k,b_k]}_{\text{k-rozměrný interval (hyperkvádr)}}$
    - enumerační (chceme najít všechny body v intervalu)
    - počítací (zajímá nás jen jejich počet)
  - – částečná shoda (některé souřadnice známe přesně, jiné vůbec)
    - → spec. případ int. dotazu: buď $[a_i,a_i]$ nebo $\mathbb{R}$ v každé souřadnici
  - – polygonální dotazy
  - – nejbližší bod k danému

    } zatím neřešíme
- Zatím vše staticky (dynamizace později)

Na zahřátí: 1-D intervalové dotazy

- Staticky stačí setříděné pole a vyhledat krajní body intervalu binárně
  - konstrukce v čase $O(n \log n)$
  - zabere $O(n)$ prostoru
  - dotaz: enumerační v $O(\log n + |\text{výstup}|)$, počítací v $O(\log n)$ – stačí odečíst indexy
- Též lze použít BVS
  - každému vrcholu v přiřadíme interval $int(v)$: ← množina všech x, při jejichž hledání navštívíme v
    - $int(kořen) = \mathbb{R}$
    - je-li ve vrcholu v klíč x, pak: $int(\ell(v)) := int(v) \cap (-\infty, x)$
      $int(p(v)) := int(v) \cap (x, +\infty)$

    } rekurzivní dělení přímky na intervaly
  - všechny hodnoty v podstromu s kořenem v leží v $int(v)$.
  - ~~intervalový dotaz~~:

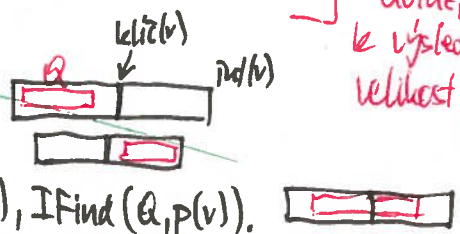
    $IFind(Q, v)$: ← Q: dotazový interval, na nějž se ptáme; v: kořen podstromu

    1. Pokud $int(v) \subseteq Q$: vypíšeme všechny klíče v $T(v)$ a skončíme. ← $T(v)$: podstrom zakořeněný ve v
       ] je-li to počítací dotaz, přičteme k výsledku předpoč. velikost podstromu
    2. ~~Jinak:~~ $x \leftarrow klíč(v)$

       ~~3. Je-li max(Q) <~~
    3. Je-li $x \in Q$: vypíšeme x
    4. • Je-li $\max(Q) \le x$: $IFind(Q, \ell(v))$

       ~~5.~~ • Je-li $\min(Q) \ge x$: $IFind(Q, p(v))$

       • Jinak (x je uvnitř Q): $IFind(Q, \ell(v))$, $IFind(Q, p(v))$.



- Alternativně:

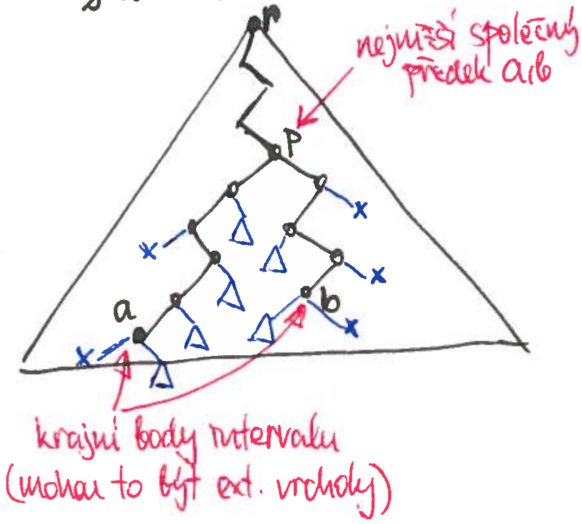
  NA PŘEDNÁŠCE NAKONEC POUŽIT JINÝ PSEUDOKÓD, VIZ STR. 4 DOLE

  $IFind(Q, v)$:
  1. Je-li $int(v) \subseteq Q$: vypíšeme celý $T(v)$
  2. Jinak je-li $int(v) \cap Q \neq \emptyset$: $IFind(Q, \ell(v))$, $IFind(Q, p(v))$
     a pokud $klíč(v) \in Q$, vypíšeme $klíč(v)$.

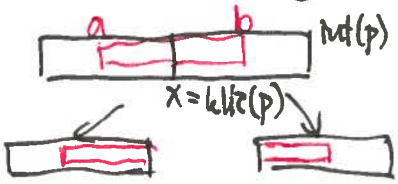
→ Možná lépe: $IFind([a,b], v)$: ← invariant: $[a,b] \subseteq$ uzávěr$(int(v))$

1. Pokud $[a,b] = int(v)$: vypíšeme $T(v)$ a skončíme.
2. Je-li $a \le x \le b$: vypíšeme x
3. Je-li $b \le x$: $IFind([a,b], \ell(v))$
4. Je-li $x \le a$: $IFind([a,b], p(v))$
5. Je-li $a < x < b$:
   $IFind([a,x], \ell(v))$
   $IFind([x,b], p(v))$

- Analýza IFind:



- od r do p je interval buď celý nalevo, nebo celý napravo → nevětvíme se
- ve vrcholu p:



- od p do a: interval "přiřazen k pravému okraji Int(v)"

buď  krok doprava, vše v $T(\ell(v))$ mimo

nebo  pokračujeme doleva, volání na $p(v)$ hned skončí (vše je uvnitř)

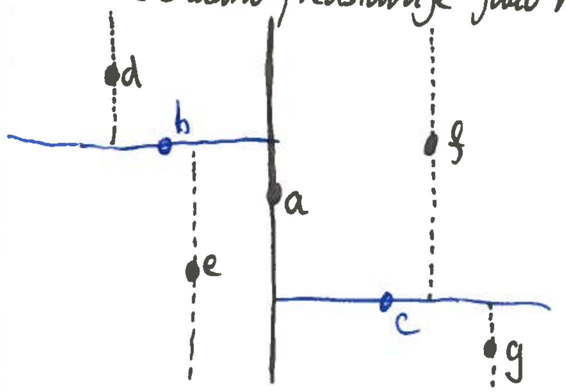
- od p do b: totéž stranově převrácené

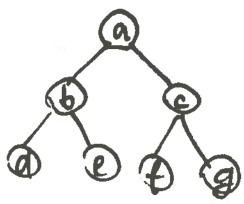
⇒ celkem navštívíme $O(\log n)$ vrcholů: 2 cesty + na každý vrchol max. 1 syn

⇒ Enumerace v $O(\log n + |\text{výstup}|)$, počítání v $O(\log n)$.

k-d stromy

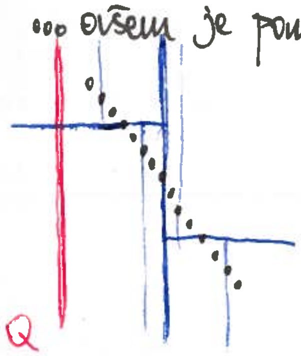
- přímočaré k-dim. zobecnění
- na i-té hladině porovnáváme ($i \bmod k$)-tou souřadnici
  - vybereme bod s mediánovou souřadnicí
  - vše < nalevo, vše > napravo
- zatím předpokládáme, že není více bodů s toutéž hodnotou souřadnice (projekce jsou prosté)
- Ve 2D se snadno představuje jako rekurzivní dělení roviny:





- Obecně: hloubka $\leq \lceil \log n \rceil$, dotaz na přesnou shodu prochází cestu shora dolů ⇒ čas $O(\log n)$
- Konstrukce v čase $O(n \log n)$ – mediány hledáme v čase $O(n)$ ⇒ součet časů přes hladinu je $O(n)$.
- Strom zabírá prostor $O(n)$.
- Intervalové dotazy: vrcholům mohu přidělovat $Int(v)$ ← k-rozměrný interval, funguje stejný rekurzivní algoritmus

... ovšem je pomalý: příklad v $\mathbb{R}^2$:



$n = 2^k - 1$

$X = \{(x,x) \mid x \in \{1 \ldots n\}\}$

Q = dotaz = $\{0\} \times \mathbb{R}$

- na "vodorovných" hladinách jdu doleva
- na "svislých" volám na oba syny

⇒ dojdu do $2^{\frac{1}{2}\log n} = \sqrt{n}$ listů, celkem navštívím $\Theta(\sqrt{n})$ vrcholů (ač je výsledkem prázdná množina)

... ale horší to už není:

Věta: Intervalový dotaz v k-d stromu má složitost $\Theta(n^{1-\frac{1}{k}} + |\text{výstup}|)$ při enumeraci,
$\Theta(n^{1-\frac{1}{k}})$ při počítání.

Dk: Pro 2-d: Při enumeraci navštívíme vrcholy, jejichž int má neprázdný průnik s Q. ~~vrcholy, jejichž~~

- $int(v) \subseteq Q$: vypíši bod → účtuji velikosti výstupu
- ostatní: tyto intervaly protíná některá ze stran obdélníka Q

celkem $O(\sqrt{n})$ {
... jejich # odhadnu počtem ~~vrcholů~~ intervalů protnutých přímkami proloženými stranami Q
... úvaha z příkladu ukazuje, že 1 přímka protne $O(\sqrt{n})$ intervalů.
}

Při počítání: intervaly ležící celé v Q navštívíme jen tehdy, pokud jejich otec neleží celý v Q
⇒ účtujeme je otcům.

Pro obecnou dimenzi: dolní odhad stejnou množinou, $Q := \{0\} \times \mathbb{R}^{k-1}$

- v každých k hladinách stromu 1× jdu doleva, (k-1)× rekurze na oba syny

⇒ dojdu do $2^{\frac{k-1}{k}\log n}$ listů $= n^{1-\frac{1}{k}}$, navštívím $\Omega(n^{1-\frac{1}{k}})$ vrcholů.
↑ konst. závisí na d

Zobecnění: je-li Q nadrovina kolmá na 1 souřadnici, protne $O(n^{1-\frac{1}{k}})$ intervalů.
Proto # intervalů protnutých hranicí kvádru je $O(n^{1-\frac{1}{k}})$.

Dotaz na částečnou shodu: z souřadnic známých, k-z neznámých:
za každých k hladin jdu z× jen do 1 syna, (k-z)× do obou
⇒ dojdeme do $2^{\frac{k-z}{k}\log n} = n^{1-\frac{z}{k}}$ listů.

! Zatím jsme předpokládali prosté projekce. Co když nejsou?
Všechny body se souřadnicí rovnou té dělící uložíme do samostatného (k-1)-d stromu.
→ ∀ vrchol má 3 syny: <, =, >

Analýza se zkomplikuje, int. dotazy stále běží v $\Theta(n^{1-\frac{1}{k}})$, částečnou shodu jsme zpomalili.

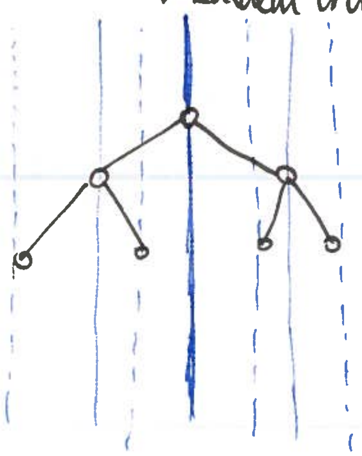
$O(n^{\max(\frac{1}{2}, \frac{1}{k-z+1})})$ [bez důkazu]

Pozn.: Intervalové dotazy jsou pomalé, ale pro struktury v prostoru O(n) to ani lépe nejde – dolní odhad viz Mehlhorn.

Vícerozměrné intervalové stromy

← angl. Range trees (Interval trees jsou něco jiného!)

2D strom: • 1D strom podle x, jeho vrcholy odpovídají pásům roviny
• v každém vrcholu x-stromu odkaz na y-strom: 1D strom y-ových souřadnic bodů v pásu



Konstrukce: 1. body setřídíme zvlášť podle x a podle y
2. rekurzivně budujeme x-strom, předáváme seznamy bodů v pásu setříděné podle x, y
3. pro ∀ vrchol x-stromu sestrojíme jeho y-strom.

↳ $O(n \log n)$

Pokud se x-ové souřadnice mohou opakovat, nutno rozmyslet otevřenost/uzavřenost intervalů a případně udržet i pásy šířky 0 („prostřední syn“ v k-d stromech)

↳ nebo dát body jen do listů a nechat vnitřní vrcholy odpovídat intervalům.

Prostor: $O(n \log n)$ [x-strom zabírá O(n), každý bod leží v $O(\log n)$ y-stromech

Dotaz $[x_1, x_2] \times [y_1, y_2]$: interval $[x_1, x_2]$ pokryjí $O(\log n)$ vrcholy x-stromu, jejich y-stromů se zeptám na $[y_1, y_2]$
→ $O(\log^2 n + |\text{výstup}|)$ pro enumeraci, $O(\log^2 n)$ pro počítání.

k-dim. strom: primární strom je 1D podle 1. souřadnice, v jeho vrcholech bydlí (k−1)-D stromy podle zbylých souřadnic.

- $O(n \log^{k-1} n)$ prostoru
- konstrukce v čase $O(n \cdot \log^{k-1} n)$
- obdélníkový dotaz v čase $O(\log^k n$ + velikost výstupu).

Dynamické intervalové stromy

- problém: rotace v x-stromu vyžaduje přebudovat y-stromy?

→ použijeme líné vyvažování z 1. přednášky (pomocí přebudovávání podstromů)

2D → Kdykoliv přebudujeme podstrom x-stromu s m vrcholy, strávíme tím $O(m)$ času, což se zamortizuje příspěvkem $O(\log n)$ na Insert / Delete.

→ Navíc trávíme čas $O(m \log m)$ přebudováním y-stromů → stačí příspěvek $O(\log^2 n)$

→ konečně musíme spořit na samostatné přestavby y-stromů → též $O(\log^2 n)$ na prvek.

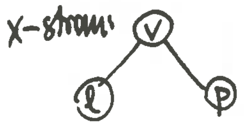
k-D $O(\log^k n)$ amort.

Rychlejší statická 2D-struktura

... bude odpovídat v $O(\log n + out)$ → k-D případ zlepšíme na $O(\log^{k-1} n + out)$

- y-ové struktury budou setříděná pole
- dotazy na jednotlivé y-struktury jsou závislé:

x-strom:



$Y(l) \subseteq Y(v)$
$Y(p) \subseteq Y(v)$ } pro $\forall$ prvek v $Y(v)$ si pamatuji jeho následníka (nejbližší $\geq$, případně $+\infty$) v $Y(l)$ a $Y(p)$

↓

znám-li výsledek interval. dotazu $[a,b]$ v $Y(v)$, umím v $O(1)$ vyhodnotit týž dotaz v $Y(l)$ a $Y(p)$

Důsledky:
- konstrukci nezpomalíme, prostor vzroste jen konst-krát
- dotaz na x-strom trvá $O(\log n)$, 1. dotaz na y-strukt. také $O(\log n)$, všechny ostatní už $O(1)$ } celkem $O(\log n)$
- rozbili jsme dynamizaci

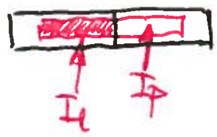
Poznámka: Umí se i 3D dotazy v $O(\log n)$, viz zápisky z DS2.

ŠIKOVNĚJŠÍ PSEUDOKÓD INT. DOTAZŮ:

vrchol (kořen podstromu) — interval, na nějž se ptáme ... invariant: $I \subseteq$ uzávěr$(int(v))$

IQ(v, I):

1. Je-li $int(v) \subseteq I$: vypíšeme celý $T(v)$ ← podstrom zakořeněný ve v, příp. k celkovému počtu přičteme předpočítané $|T(v)|$.
2. Pokud klíč$(v) \in I$: vypíšeme klíč(v)
3. $I_l \leftarrow I \cap int(l(v))$, pokud $I_l \neq \emptyset$: $IQ(l(v), I_l)$
4. $I_p \leftarrow I \cap int(p(v))$, pokud $I_p \neq \emptyset$: $IQ(p(v), I_p)$



(takto formulováno to funguje pro jakýkoli strom popisující hierarchii do sebe zanořených intervalů [či obecněji hyperkvádrů].)

Zrychlení 2D dotazů na O(log n) je speciálním případem obecnější techniky:

ZLOMKOVÉ KASKÁDOVÁNÍ (fractional cascading)

- Mějme k setříděných polí o n prvcích $X_1 - X_k$
- Chceme pro dané x zjistit, kam patří v jednotlivých polích
  → triviálně v $O(k \log n)$, ukážeme $O(k + \log n)$.
- Definujeme: $X'_k := X_k$
  pro $i<k$ $X'_i := X_i \cup$ (každý druhý prvek X'_{i+1}) } Indukcí: $\forall i\ |X'_i| \leq 2n$
  "stíny" prvků z násl. seznamu
- Pro všechna i si zapamatujeme X'_i s násl. pomocnými informacemi:
  - $\forall$ prvek si pamatuje pozici předchozího a následujícího stejného druhu (stín/nestín)
  - $\forall$ stín si pamatuje svou pozici v X'_{i+1}
  - $\forall$ nestín si pamatuje pozici v X_i

pozici prvku v X'_i umíme v $O(1)$ přepočítat na pozici v X'_{i+1}
→ stačí bin. vyhledávání v X'_1 a pak $(k-1)$krát "skočit" o seznam dál.

OKÉNKOVÉ DOTAZY NA ÚSEČKÁCH (windowing queries)

- je dána množina n neprotínajících se úseček ~~[illegible]~~ v $\mathbb{R}^2$
  doteky dovoleny
- dotaz: vyjmenovat úsečky ležící aspoň z části v daném 2D intervalu

} vše statický

1D problém: dány intervaly v $\mathbb{R}$, okénko je interval

~~[illegible]~~ aspoň 1 krajní bod uvnitř
→ intervalový dotaz na množině bodů (už umíme)

intervaly, které začínají i končí mimo
→ musí obsahovat aspoň 1 krajní bod ~~[illegible]~~ okénka } detekce oba → BinO log

↓

jednodušší problém: dána mna intervalů, dotaz: které obsahují zadaný bod? ("propichovací dotaz – stabbing query")

Česky se i tomuto někdy říká intervalový strom

celý 1D problém:
prostor $O(n)$
Build $O(n \log n)$
Query $O(\log n + |out|)$

Stromová struktura (interval tree)

medián množiny začátků a konců
↓
strom má výšku $O(\log n)$

strom pro intervaly ležící celé nalevo od x

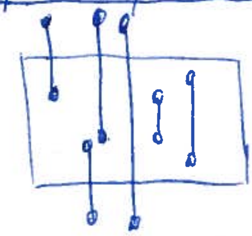
celé napravo od x

zde si pamatujeme intervaly obsahující x:
- jednou setříděné podle začátků
- podruhé dle konců

pro libovolné t najdeme všechny, které obsahují t, v čase $O(1 + |výstup|)$

Prostor $O(n)$, konstrukce v $O(n \log n)$, dotaz v $O(\log n + |výstup|)$

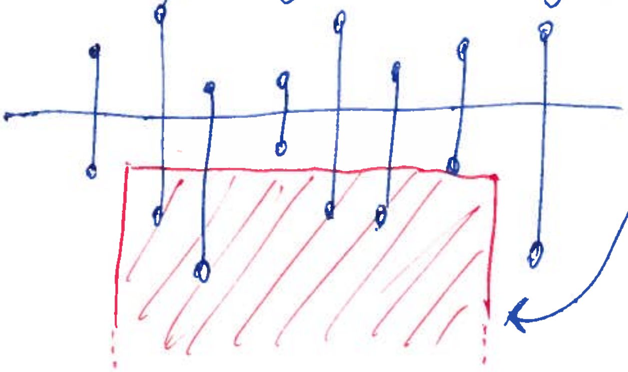
2D problém pro svislé úsečky (nebo vodorovné)



rozložím: — krajní bod uvnitř → 2D intervalový dotaz na množině bodů (range tree) → 2D int. strom s dotazy v $O(\log n)$

— oba kraj. body vně ↓ propichuji svislé úsečky vodorovnou úsečkou (hranicí obdélníka)

- kdyby to byla vodorovná přímka, byl by to vlastně 1D problém v y-ové souřadnici
- pro úsečku upravíme konstrukci stromu: větvím se podle y, u úseček, v nichž dané y leží, si zapamatuji množinu krajních bodů nad/pod úsečkou:



potřebuji 2D dotaz (byť degenerovaný) na množině bodů → 2D range tree
[tento typ dotazů lze řešit i jednodušším "prioritním stromem" ve stejném čase a prostoru $O(n)$]

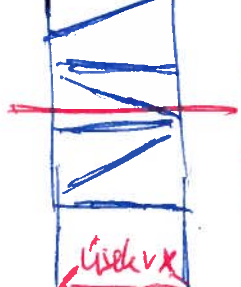
Celkově: prostor $O(n \log n)$, konstrukce v čase $O(n \log n)$, dotaz v $O(\log^2 n + |out|)$

Intermezzo: Jiné řešení 1D problému → "segment tree"

- krajní body ~~úseček~~ intervalů dělí $\mathbb{R}$ na elementární intervaly (∀ zadaný interval je sjednocením nějakých elementárních)
- vytvoříme strom: listy jsou elem. intervaly, vnitřní vrcholy jsou krajní body, ale také souvislé úseky listů
- ∀ zadaný interval rozložíme na $O(\log n)$ úseků listů odpovídajících vrcholům, do těchto vrcholů ho uložíme (∀ vnitř. vrchol si pamatuje seznam intervalů obsahujících celý jeho úsek)
- prostor: $O(n \log n)$, konstrukce v čase $O(n \log n)$
- dotaz: hledáme x, vysbíráme zapamatované intervaly na cestě → $O(\log n + |out|)$

2D problém pro obecné úsečky

- opět redukuji na propichování svislou/vodorovnou úsečkou — BÚNO vodorovnou
- segment tree podle průmětu do osy x, ve vnitřních vrcholech úsečky v pásu setříděné shora dolů: to jde, neb se neprotínají



- $O(\log n + |out|)$ v každém navštíveném vrcholu → celkem $O(\log^2 n + |out|)$
- prostor $O(n \log n)$, konstrukce v čase $O(n \log^2 n)$, lze i $O(n \log n)$.