Metrické stromy

6. prosince 2016

Metrickým stromem rozumíme jakoukoli stromovou datovou strukturu specializovanou na indexování dat v metrických prostorech. Metrické stromy využívají vlastnosti metrických prostoru, jako trojúhelníková nerovnost k efektivnějšímu přístupu k datům. Příklady jsou M-stromy, vp-stromy, cover stromy, MVP stromy, a bk stromy.

Metrika vzdálenosti $d(x, y)$ pro metrický prostor je definována následovně

- i) $d(x, y) = d(y, x)$
- ii) $0 < d(x, y) < \infty, x \neq y$
- iii) $d(x, x) = 0$
- iv) $d(x, y) \leq d(x, z) + d(z, y)$ (trojúhelníková nerovnost)

Metriky vzdálenosti

- Euklidovská vzdálenost

$$d(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

- Hammingova vzdálenost

$$d(\vec{x}, \vec{y}) = \sum_{i=1}^{n} |x_i - y_i|$$

- Čebyševova vzdálenost

$$d(\vec{x}, \vec{y}) = \max_{i=1}^{n} |x_i - y_i|$$

- Minkowského vzdálenost

$$d(\vec{x}, \vec{y}) = \sqrt[p]{\sum_{i=1}^{n} |x_i - y_i|^p} \quad p > 0$$

- Canberova vzdálenost

$$d(\vec{x}, \vec{y}) = \sum_{i=1}^{n} \frac{|x_i - y_i|}{x_i + y_i} \quad x_i \text{ a } y_i \text{ jsou kladné}$$

- Levenshteinova vzdálenost (editační vzdálenost)

  Vzdálenost sekvencí a a b je $\mathrm{lev}_{a,b}(|a|, |b|)$.

$$\mathrm{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{pokud } \min(i,j) = 0 \\ \min \begin{cases} \mathrm{lev}_{a,b}(i-1,j) + 1 \\ \mathrm{lev}_{a,b}(i,j-1) + 1 \\ \mathrm{lev}_{a,b}(i-1,j-1) + I(a_i \neq b_i) \end{cases} \end{cases}$$

  $I(a_i \neq b_i)$ – indikátor $a_i \neq b_i$.

1 VP-stromy

VP-stromy rozkládají množinu dat podle vzdáleností objektů vzhledem k referenčnímu bodu (vantage point). Medián těchto vzdáleností je použit jako oddělovač k rozkladu objektů do dvou vyvážených podmnožin, na které může být rekurzivně použita ta samá procedura.

Struktura binárního vp-stromu je velmi jednoduchá. Každý vnitřní uzel je ve tvaru $(S_\mathrm{v}, M, R_\mathrm{ptr}, L_\mathrm{ptr})$, kde Sv je vantage point, M mediánová vzdálenost všech bodů (od S_v) indexovaných pod tím uzlem, R_ptr a L_ptr jsou ukazatele na levý a pravý podstrom. Levý podstrom uzlu indexuje ty body, jejichž vzdálenost od S_v je menší nebo rovna M, a pravý podstrom uzlu indexuje ty body, jejichž vzdálenost od S_v je větší nebo rovna M. V listových uzlech jsou reference na datové body míst ukazatelů na podstromy.

Vytvoření VP stromu

Vstup: S – množina prvků z metrického prostoru.

1) Pokud $|S| = 0$, vytvoř prázdný strom.

2) Jinak, nechť S_v je libovolný objekt z S (vantage point).
   $M = \mathrm{median}(\{d(S_i, S_\mathrm{v}) \mid \forall S_i \in S\})$
   Nechť $S_\mathrm{l} = \{S_i \mid d(S_i, S_\mathrm{v}) \leq M \text{ kde } S_i \in S \text{ a } S_i \neq S_\mathrm{v}\ \}$
   $S_\mathrm{r} = \{S_j \mid d(S_j, S_\mathrm{v}) \geq M \text{ kde } S_j \in S\}$ (kardinalita S_l a S_r by měly byt zhruba rovny)
   Rekurzivně vytvoř vp-stromy na S_l a S_r jako levý a pravý podstrom.

Vyhledávání

Pro daný objekt Q, množina datových objektů, které jsou ve vzdálenosti τ od Q je nalezena vyhledávacím algoritmem popsaným níže:

1) Pokud $d(Q, S_\mathrm{v}) \leq \tau$, pak S_v je ve výsledné množině.
2) Pokud $d(Q, S_\mathrm{v}) + \tau \geq \mu$, rekurzivně prohledej pravý podstrom.
3) Pokud $d(Q, S_\mathrm{v}) - \tau \leq \mu$, rekurzivně prohledej levý podstrom.

(všimněte si, že 2) a 3) mohou nastat současně, pak se prohledávají oba podstromy).

NN ve VP stromech

```
Data: vp strom T, bod q
τ = ∞;
nodestovisit= [T]  NN = NULL;
while not null nodestovisit do
    node = nodestovisit.popleft();
    d = d(q, node.vp);
    if d < τ then
        NN = node.vp;
        τ = d;
    end
    if d < node.μ then
        if d < node.μ + τ then
            nodestovisit.append(node.left)
        end
        if d ≥ node.μ − τ then
            nodestovisit.append(node.right)
        end
    else
        if d ≥ node.μ − τ then
            nodestovisit.append(node.right)
        end
        if d < node.μ + τ then
            nodestovisit.append(node.left)
        end
    end
end
```

Zobecnění binární vp-stromů na m-ární vp-stromy

Binární vp-strom může být snadno zobecněn na m-ární stromovou strukturu. Konstrukce vp-stromu řádu m je velmi podobná konstrukci binárního vp-stromu. Namísto hledání mediánu vzdáleností mezi vantage pointem a datovými body, jsou body seřazeny a rozloženy do m skupin o stejné kardinalitě. Hodnoty vzdáleností použité pro ten rozklad jsou zaznamenány v uzlu.

2 MVP-stromy (multiple vantage point trees)

MVP-stromy, rozkládají prostor do sférických řezů okolo vantage pointů (podobně jako vp-stromy), rozklady vytváří vzhledem k více než jednomu vantage pointu na každé úrovni a udržuje si informaci navíc v listech pro kvůli efektivnímu odfiltrování kandidátních bodů.

MVP-strom používá dva vantage pointy v každém uzlu. Na každý uzel v MVP-stromu může být nahlíženo jako na dvě úrovně VP-stromu (rodič a jeho přímí potomci) kde všechny uzly v potomcích na nižší úrovni používají ten samý vantage point. To umožňuje mít více odfiltrovaných dat v každém uzlu při vyhledávání a menší počet vantage pointů v nelistových úrovních.

MVP-strom má tři parametry:

- počet tříd rozkladu vytvořený každým vantage pointem (m),
- maximum dat pro listové uzly (k),
- a počet vzdáleností pro datové body v listech (p).

V binárních MVP-stromech rozděluje první vantage point (označme ho $S_{\mathrm{v}1}$) prostor na dvě části, a druhý vantage point (označme ho $S_{\mathrm{v}2}$) rozděluje každou z těchto částí na dvě. Máme tedy 4 potomky v binárním případě. Obecně počet potomku vnitřního uzlu je m^2.

V každém vnitřním uzlu je udržovány mediány M_1 a M_1 pro rozklad vzhledem k $S_{\mathrm{v}1}$ a $S_{\mathrm{v}2}$.

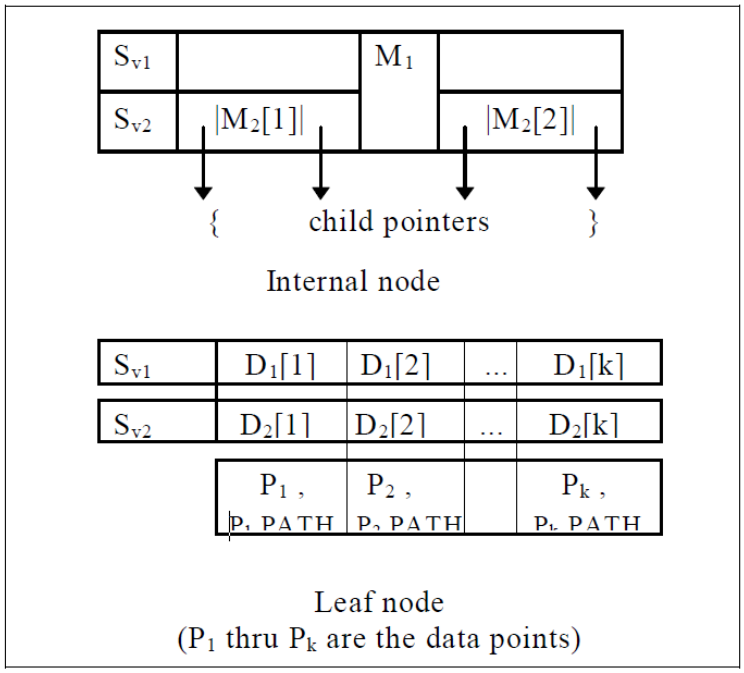
V listových uzlech uchováváme přesné vzdálenosti mezi datovými body a vantage pointy toho listu. $\mathrm{D}_1[i]$ a $\mathrm{D}_2[i]$ $(i = 1, 2, \ldots, k)$ jsou vzdálenosti z prvního a druhého vantage pointu, a k je počet dat v listech (může být zvoleno vyšší než m^2.

Pro každý datový bod x v listech uchovává pole $x.\mathrm{PATH}[p]$ vypočtené vzdálenosti mezi datovým bodem x a prvními p vantage pointy na cestě z rootu do listového uzlu. Obrázek 1 ukazuje strukturu vnitřního uzlu a a listových uzlů v binárním MVP-stromu.

2.1 Konstrukce MVP-stromu

Je-li dána konečná množina $S = S_1, S_2, \ldots, S_n$ objektů, a metrika $d()$, binární MVP-strom s parametry $m = 2, k$ a p je konstruován na S následovně.

1) Pokud $|S| = 0$, vytvoř prázdný strom a skonči.



Obrázek 1: Struktura uzlu binárního MVP-stromu

2) Pokud $|S| \leq k + 2$ pak

   2.1) Vyber libovolný objekt z S (S_{v1}, první vantage point)

   2.2) $S := S - \{S_{\text{v1}}\}$

   2.3) Vypočítej všechny $d(S_i, S_{\text{v1}})$ kde $S_i \in S$, a ulož je v poli D_1.

   2.4) Nechť S_{v2} je nejvzdálenější bod od S_{v1} (druhý vantage point).

   2.5) $S := S - \{S_{\text{v2}}\}$

   2.6) Vypočítej všechny $d(S_j, S_{\text{v1}})$ kde $S_j \in S$, a ulož je v poli D_2.

   2.7) skonči.

3) Jinak (pokud $|S| > k + 2$)

   3.1) Vyber libovolný objekt z S (S_{v1}, první vantage point)

   3.2) $S := S - \{S_{\text{v1}}\}$

   3.3) Vypočítej všechny $d(S_i, S_{\text{v1}})$
   Pokud $(level \leq p)$ $S_i.PATH[l] = d(S_i, S_{\text{v1}})$.

   3.4) Uspořádej objekty v S vzhledem k jejich vzdálenosti od S_{v1}.
   M_1 = medián z $\{d(S_i, S_{\text{v1}})\}$, rozlož objekty do dvou seznamů, SS_1 a SS_2, stejné délky (podle M_1).

   3.5) Nechť S_{v2} je libovolný bod z SS_2 (druhý vantage point).

   3.6) $S := S - \{S_{\text{v2}}\}$

   3.7) Vypočítej všechny $d(S_j, S_{\text{v2}})$ kde $S_j \in SS_1 \cup SS_2$.
   Pokud $(level \leq p)$ $S_j.PATH[level + 1] = d(S_j, S_{\text{v2}})$.

   3.8) $M_2[1]$ = medián z $\{d(S_j, S_{\text{v2}}) \mid \forall S_j \in SS_1\}$
   $M_2[2]$ = medián z $\{d(S_j, S_{\text{v2}}) \mid \forall S_j \in SS_2\}$

   3.9) Rozpul seznam SS_1 do dvou seznamů stejné délky podle $M_2[1]$. Podobně pro SS_2. $level+ = 2$, rekurzivně zpracuj všechny čtyři množiny.

2.2 Vyhledávání v MVP-stromech

Pro daný objekt Q, množina datových objektů, které jsou ve vzdálenosti r od Q je nalezena vyhledávacím algoritmem popsaným níže:

1) Vypočítej vzdálenosti $d(Q, S_{\text{v1}})$ a $d(Q, S_{\text{v2}})$.

   – pokud $d(Q, S_{\text{v1}}) \leq r$, pak S_{v1} je ve výsledné množině;

   – pokud $d(Q, S_{\text{v2}}) \leq r$, pak S_{v2} je ve výsledné množině.

2) pokud je aktuální uzel list,
   Pro všechny datové body (S_i) v uzlu,

   2.1) Najdi $d(S_i, S_{\text{v1}})$ a $d(S_i, S_{\text{v2}})$ v polích D_1 a D_2.

2.2) pokud $[d(Q,S_{v1})-r \leq (S_i,S_{v1}) \leq d(Q,S_{v1})+r]$ a současně $[d(Q,S_{v2})-r \leq (S_i,S_{v2}) \leq d(Q,S_{v2})+r]$, pak pokud pro všechna $i = 1 \ldots p$ platí $(\mathrm{PATH}[i] - r \leq S_i.\mathrm{PATH}[i] \leq \mathrm{PATH}[i] - r)$, vypočítej $d(Q,S_i)$. Pokud $d(Q,S_i) \leq r$, pak S_i je ve výsledné množině.

3) jinak, (pokud je aktuální uzel list interní)

3.1) pokud $(l \leq p)$, $PATH[l] = d(Q,S_{v1})$, pokud $(l < p)$, $PATH[l+1] = d(Q,S_{v2})$.

3.2) pokud $d(Q,S_{v1} + r \leq M_1$, pak

- pokud $d(Q,S_{v2}) + r \leq M_2[1]$, rekurzivně prohledej první podstrom s $l = l + 2$.
- pokud $d(Q,S_{v2}) - r \geq M_2[1]$, rekurzivně prohledej druhý podstrom s $l = l + 2$.

3.3) pokud $d(Q,S_{v1} - r \geq M_1$, pak

- pokud $d(Q,S_{v2}) + r \leq M_2[1]$, rekurzivně prohledej třetí podstrom s $l = l + 2$.
- pokud $d(Q,S_{v2}) - r \geq M_2[1]$, rekurzivně prohledej čtvrtý podstrom s $l = l + 2$.

3 BK-stromy

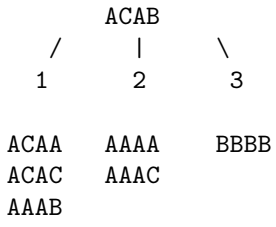
BK-strom (Burkhard & Keller) je metrický strom s adaptovaný pro diskrétní metrické prostory. Uvažujme, že $d(x,y) \in \mathbb{N}_0$.

Každý uzel má jeden klíč x libovolný počet potomků, každý ukazatel na potomka je ohodnocen vzdáleností z $\mathbb{N}_0$; klíče v potomku mají vzdálenost odpovídající tomuto ohodnocení.

Operace nad BK-stromy jsou přímočaré.

Insert:
vstup: BK-strom T, vkládaný klíč x
výstup: upravený strom T.

- Pokud je strom prázdný, vytvoř nový kořen s klíčem x.
- Nechť k je klíč v kořeni. Rekurzivně vyvolej insert pro podstrom s ohodnocením $d(x,k)$.



Vyhledávání:

Range search:

vstup: BK-strom T, klíč x, vzdálenost d
výstup: množina klíčů se vzdáleností $\leq d$.

- pokud je strom prázdný, skonči.
- Nechť k je klíč v kořeni. Pokud $d(x,k) \leq d$

  Rekurzivně vyvolej Range search pro každý podstrom s ohodnocením $d - d(x,k) and d + d(x,k)$ včetně.

NN search:

vstup: BK-strom T, klíč x
výstup: klíč z T s nejmenší vzdáleností od x.

- inicializace $d = \infty; NN = NULL$.
- pokud je strom prázdný, skonči.
- Nechť k je klíč v kořeni. Pokud $d(x,k) < d$, pak NN =kořen, $d = d(x,k)$.

  Rekurzivně vyvolej NN search pro každý podstrom s ohodnocením $d - d(x,k) and d + d(x,k)$ včetně.

4 Cover tree

Cover tree T nad daty S je strom, ve kterém každá úroveň je pokrytí (cover) pro úroveň pod ní. Každá úroveň je indexovaná přirozeným číslem i, který se směrem k listům snižuje. Každý uzel je asociován s jedním bodem v S. Každý bod v S může být asociován s více uzly ve stromu, ale požaduje se, aby se každý bod vyskytoval nejvýše jednou v každé úrovní. Nechť C_i označuje množinu bodů v S asociovaných s uzly na úrovni i. Cover tree splňuje následující pravidla pro všechna i.

1. (vnoření) $C_i \subset C_{i-1}$. Z toho vyplývá, že pokud se bod $p \in S$ vyskytuje v C_i, pak každá nižší úroveň má uzel asociovaný s p.
2. (pokrytí) Pro všechna $p \in C_{i-1}$ existuje $q \in C_i$ t.ž. $d(p,q) < 2^i$ a uzel v úrovni i asociovaný s q je rodičem uzlu v úrovni $i-1$ asociovaným s p.
3. (separace) pro všechna různá $p, q \in C_i$, $d(p,q) > 2^i$.

Pro jednoduchost budeme sjednocovat uzly s jejich asociovanými body.

Hledání nejbližšího souseda. Pro nalezení nejbližšího souseda bodu p v cover tree sestupujeme skrze strom úroveň po úrovni, a pamatuje si podmnožinu $Q_i \subseteq C_i$ uzlů, které mohou obsahovat nejbližšího souseda p jako potomka. Algoritmus iterativně konstruuje Q_{i-1} rozšířením Q_i na jeho potomky v C_{i-1} a následným ořezáním potomků, kteří nemohou vést k nejbližšímu sousedu

p. Pro jednoduchost je snažnější uvažovat strom tak, že má nekonečný počet úrovní (C_∞ je pouze kořen, $C_{-\infty} = S$). Označme množinu potomků uzlu p jako Children(p) a označme $d(p, Q) = \min_{q \in Q} d(p, q)$.

Algorithm 1 Find-Nearest (cover tree T, query point p)

1. Set $Q_\infty = C_\infty$, where C_∞ is the root level of T.
2. for i from ∞ down to $-\infty$
   (a) Set $Q = \{$ Children$(q) : q \in Q_i\}$.
   (b) Form cover set $Q_{i-1} = \{q \in Q : d(p, q) \leq d(p, Q) + 2^i\}$.
3. return $\arg\min_{q \in Q_{-\infty}} d(p, q)$.