BI-VWM

M-Strom

Matúš Botek

Martin Ondejka

Projekt

Cieľom tohto projektu je overenie našich poznatkov o metrických prístupových metódach nadobudnutých počas behu predmetu BI-VWM . Našim konkrétnym cieľom bolo vytvorenie aplikácie, ktorá bude indexovať nami vybrané dáta pomocou M-Stromu a sledovať, ako sa vyhľadávanie v takomto indexe líši od sekvenčného priechodu databáze. Pri riešení projektu sme čerpali so študijných materiálov a odkazov na literatúru zaoberajúcu sa M-Stromom, ktoré boli uvedené v popise projektu.

Riešenie

Rozhodli sme sa projekt naimplementovať v jazyku Python. Naša implementácia využíva niekoľko knižníc (užívateľské rozhranie, priority queue), ktoré sú vymenované v inštalačnej príručke.

Ako dáta sme sa rozhodli používať náhodne generované vektory reálnych nezáporných čísel, ktoré by v realite mohli reprezentovať vlastnosti samotných databázových objektov. V našom jednoduchšom riešení sa využívajú len tieto vektory, neodkazujú na žiadne reálne dáta.

Algoritmy

Pri M-Strome je dôležitá voľba funkcií, ktoré sú zodpovedné za prácu so záznamami a môžu ovplyvniť vlastnosti vyhľadávania v M-Strome.

Najzákladnejšou a najdôležitejšou je voľba podobnostnej funkcie. Tá sa používa ako pri samotnom indexovaní objektov tak pri vyhľadávaní v indexe. Musí spĺňať axiómy metriky a mala by vhodne reprezentovať podobnosť indexovaných dát. Keďže naša implementácia používa číselné vektory, zvolili sme za funkciu euklidovskú vzdialenosť.

Pri vkladaní prvkov do stromu je potrebné zvoliť tzv. Promotion a Partition funkcie. Tie rozhodujú o tom, akým spôsobom budú rozdeľované záznamy v uzloch stromu v prípade, že pri vkladaní do uzlu by sa prekročila jeho maximálna kapacita záznamov a uzol je nutné rozdeliť.

Promotion

Promotion funkcia spomedzi záznamov zvolí dva, ktoré budú presunuté do vyššej úrovne a budú rodičmi uzlov vytvorených po delení.

V tomto prípade sme sa rozhodli použiť funkciu, ktorá je kombináciou dvoch bežných prístupov RANDOM a M_LB_DIST.

- RANDOM – tento algoritmus je veľmi jednoduchý, spomedzi všetkých záznamov vyberie náhodne dva záznamy, ktoré budú novými smerovacími objektami. Táto metóda je veľmi rýchla a pri náročných výpočtoch vzdialeností medzi objektami môže konkurovať ostatným metódam.
- M_LB_DIST (Maximum lower bound on distance) je algoritmus, ktorý vuyžíva už predpočítané vzdialenosti objektov k ich rodičovskému objektu, ktoré sú v strome uložené. Zvolili sme potvrdzujúcu variantu, ktorá ako jeden z nových smerovacích objektov použije

súčasný rodičovský objekt a ako druhý k nemu vyberie objekt s od neho najväčšou vzdialenosťou.

Naša metóda vykonáva promotion na základe algoritmu M_LB_DIST pokiaľ sú všetky vzdialenosti predpočítané. V opačnom prípade vyberá záznamy náhodne (to môže nastať pri delení koreňového uzlu)

Partition

Partition metóda rozdelí všetky záznami deleného uzlu do dvoch množín, ktoré budú predstavovať záznamy v novovzniknutých rozdelených uzloch.

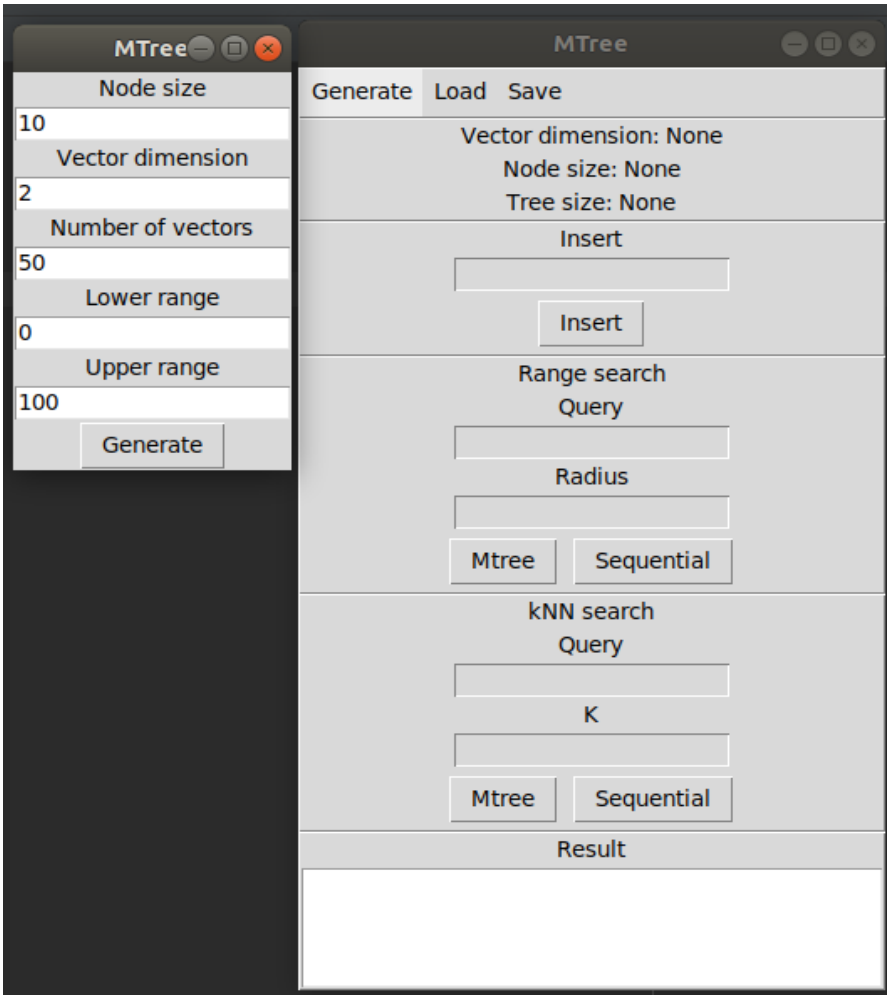
Tu sme zvolili tzv. Vyvážené delenie (Balanced distribution). Táto funkcia v každom kroku striedavo prideľuje k smerovacím objektov ten objekt, ktorý je k nim v danom momente najbližšie. Týmto prístupom sa dosiahne to, že počet záznamov v nových uzloch bude vyvážený, líšiť sa bude maximálne o jeden záznam.

Dotazovanie

Pri dotazovaní sú možné dva druhy požiadavok na vyhľadávanie a to buď rozsahová alebo kNN varianta. Rozsahová varianta vyhľadá všetky objekty vzdialené od hľadaného objektu (query) o maximálnu hodnotu (radius). kNN varianta k hľadanému objektu nájde k najbližších susedov. Pri implementácii týchto metód sme postupovali podľa popisov a pseudokódov nájdených v literatúre o M-Stromoch.

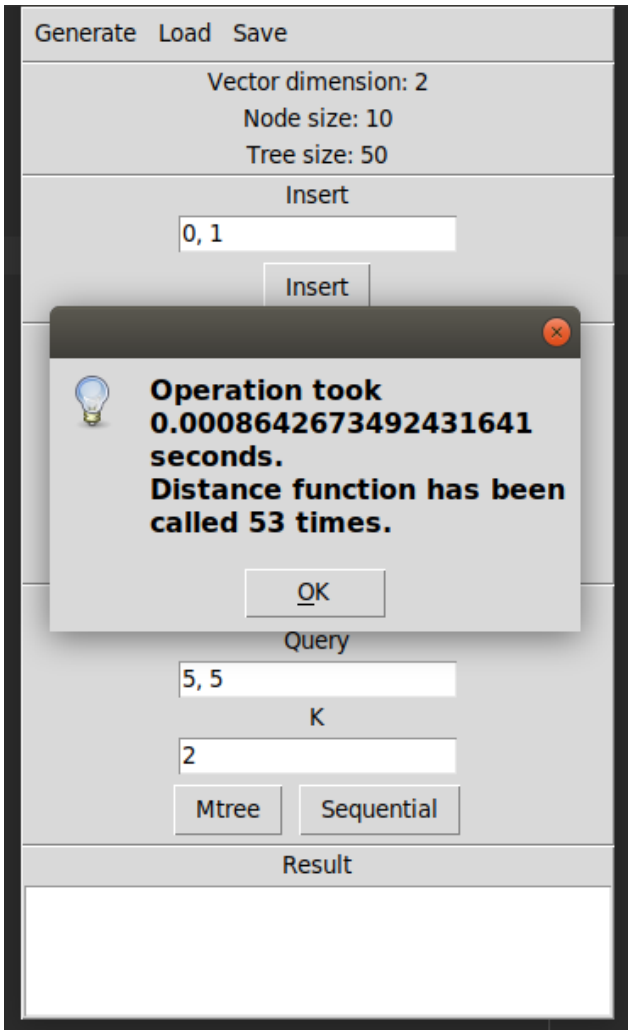
Výstup

Nižie sú pre priblíženie fungotania aplikácie zobrazené niektoré časti užívateľského rozhrania

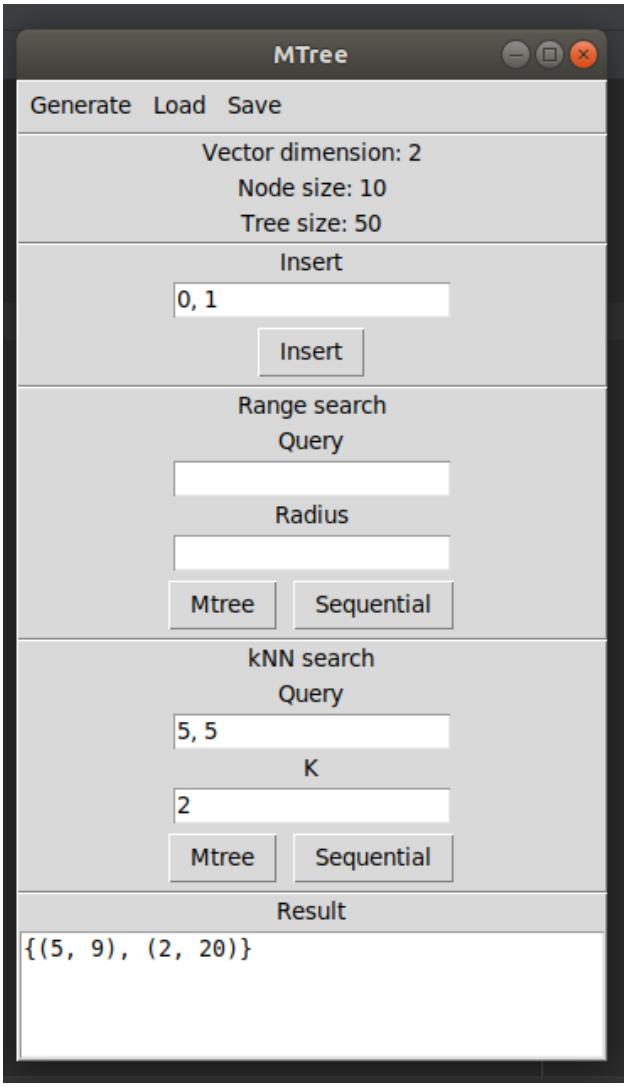


Obr. 1 - Menu zobrazujúce možnosti nastavení pri generovaní nového M-Stromu

Na obrázku 1 je vidno, že pri tvorbe M-Stromu je možné nastaviť kapacitu uzlov, dimenziu vektora, počet vygenerovaných dát, rozmedzie, spomedzi ktorého majú byť jednotlivé hodnoty vektora.



Obr. 2 – Aplikácia zobrazuje aktuálne nastavenia M-Stromu, po akejkoľvek operácii sa vypíše čas behu danej operácie a počet spustení podobnostnej funkcie



Obr. 3 – V spodnej časti je vidieť výsledok kNN požiadavku

Obrázky 2 a 3 vznikli pri zavolaní požiadavku na kNN vyhľadávanie pre dvojprvkový vektor (5, 5) pre dvoch najbližších susedov.

Experimenty

Rozhodli sme sa porovnať počet volaní podobnostnej funkcie a dĺžku spracovania požiadavku pre rozsahový, kNN požiadavok oproti sekvenčnému.

Pri testoch sme zvolili kapacitu uzlov na 10 záznamov, jednalo sa o 10.000 4-prvkových vektorov s reálnymi číslami z rozsahu 0 – 1000.

V oboch prípadoch bola pri sekvenčnom priechode zavolaná podobnostná funkcia výrazne častejšie a čas behu bol takisto väčší. Samotná indexácia databáze bola v porovnaní so získaním výsledkov sekvenčným priechodom dlhšia, no opakovaným volaním požiadavkov je tento čas zanedbateľný a dôležité sú rýchlejšie behy samotných požiadavkov.

Zdrojové kódy testov sú dostupné v súbore **tests.py.**

```
Indexing database
Time elapsed: 1.10740733146667 48s
Distance function called: 281707 times
Random range query with radius 30
Time elapsed: 0.0037257671356201 17s
Distance function called: 1008 times
Sequential
Time elapsed: 0.027928352355957 03s
Distance function called: 10000 times
Random kNN query with k 15
Time elapsed: 0.015097141265869 14s
Distance function called: 2802 times
Sequential
Time elapsed: 0.034379959106445 31s
Distance function called: 10000 times

Process finished with exit code 0
```

Diskusia a záver

Naša aplikácia nám dokázala, že metrické indexovacie metódy sú oproti bežnému sekvenčnému priechodu efektívnejšie, no túto skutočnosť sme testovali len na veľmi jednoduchej variante. Keby mal projekt širšie spracovanie, určite by bolo zaujímavé vyskúšať rôzne kombinácie funkcií použitých pri stavbe M-Stromu a sledovať, ako ovplyvňujú trvanie požiadavkov.

Ďalej by prinieslo iný pohľad aj použitie reálnych dát, ktoré by boli reprezentované na základe ich vlastností. Pri náhodnom generovaní sa nemusí vždy dosiahnuť situácia dostatočne podobná reálnej.

Pri experimentovaní by bolo možné sledovať viac parametrov, ktoré ovplyvňujú efektívnosť M-Stromu ako napríklad vzdialenosť a počet susedov pri požiadavkoch, dimenzie dátových vektorov.

V implementácii je pri hromadnom vytváraní stromu využitá jednoduchá varianta, ktorá pužíva opakované volania metódy insert. Bolo by možné tento proces zefektívniť vuyžitím tzv. Bulk-loadingu.

Týmto skutočnostiam sa náš projekt nevenoval, pretože našim cieľom bolo vyskúšať naše poznatky o metrickom indexovaní v malom rozsahu dostačujúcom na overenie funkčnosti spomínaných metód.

Použitá literatúra

Aplikace metrických indexovacích metod na data získanáhmotnostní spektrometri
https://moodle-vyuka.cvut.cz/pluginfile.php/383628/mod_page/content/23/novakj4_2008dipl.pdf

Similarity Search in Mass SpectraDatabases
https://moodle-vyuka.cvut.cz/pluginfile.php/383628/mod_page/content/23/thesis.pdf

M-tree: An Efficient Access Method for Similarity Search in Metric Spaces
https://www.cs.bu.edu/faculty/gkollios/ada17/LectNotes/Mtree.PDF

Indexing Metric Spaces with M-tree
http://www-db.deis.unibo.it/research/papers/SEBD97.pdf

Prednáška: Indexování metrické podobnosti pro rychlé vyhledávání v multimediálních databázích
Materiály predmetu BI-VWM

Zadanie projektu M-Strom z materiálov predmetu BI-VWM