M-strom

Andrei Babushkin
Georgii Korostii

Popis projektu

Cílem projektu je vytvoření vlastní implementace přístupové metody M-strom. Na vstupu spolu s nastaveními pro běh programu a vygenerováním databáze je výběr mezi rozsahovým dotazem a dotazem na k nejbližších sousedů. Na výstupu je seznam objektů databáze odpovídající dotazu.

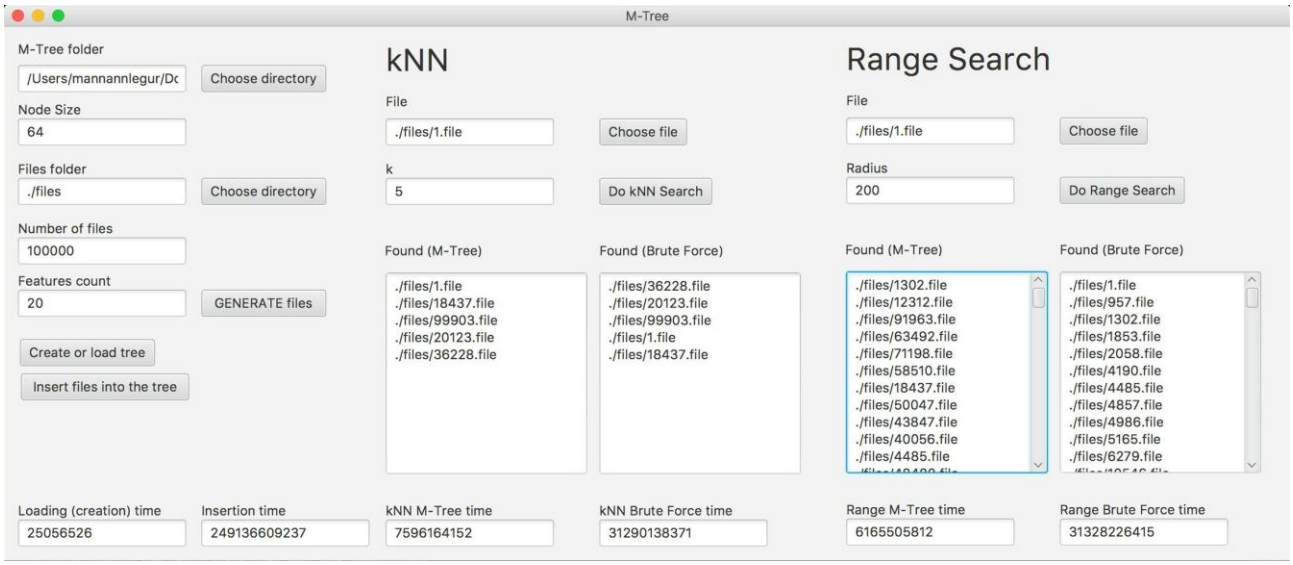
Způsob řešení

Byl použit klasický algoritmus M-strom ze článku "An Efficient Access Method for Similarity Search in Metric Spaces" s dotazem na k nejbližších sousedů a rozsahovým dotazem.

Implementace

Pro vytvoření projektu byly použity prostředky jazyka Java. Data pro práci byly vygenerovány jako n souborů (v našem případě 100 000) s 20 náhodně zvolenými čísly s pohyblivou řádovou čárkou.

Příklad výstupu

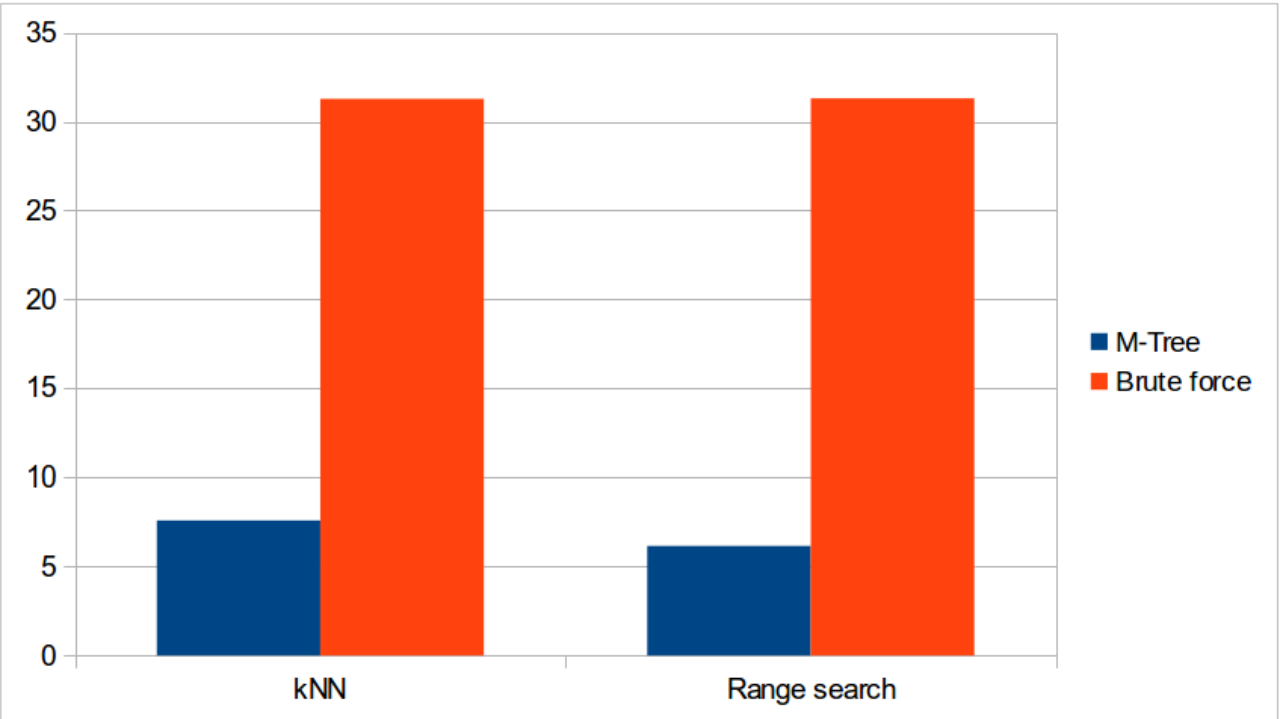


Obrázek 1. GUI programu.

Zadáváme velikost uzlu (může být libovolný) a cestu do adresáře ve kterém se vygenerují databázové objekty podle požadovaných nastavení, pak zadáváme data pro vyhledávání k nejbližších sousedů (data která chceme najít a počet sousedů) a/nebo rozsahový dotaz (znovu data a poloměr rozsahu).
Na výstupu máme seznam souborů odpovídající vybraným nastavením. Zobrazuje se taky čas provedení každé dotazované operace (v nanosekundách).

Experimentální sekce

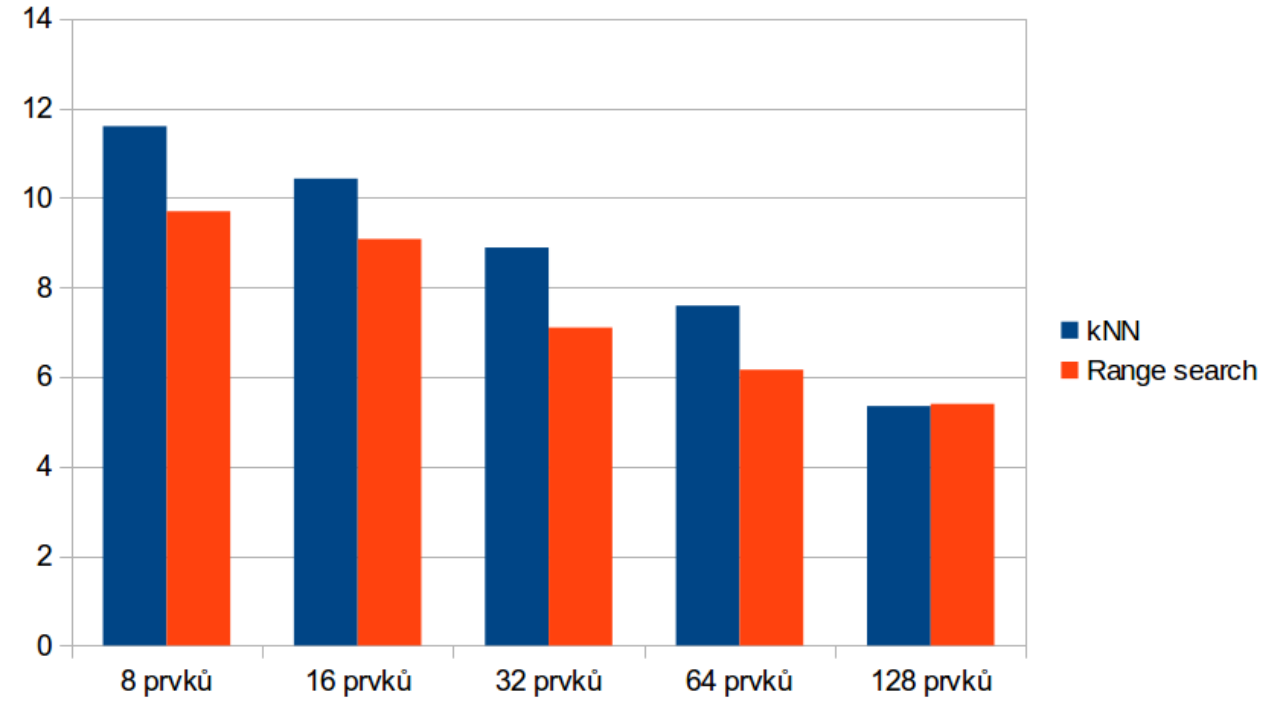
Porovnáme čas provedení dotazu na k nejbližších sousedu a rozsahového dotazu při použití algoritmu M-strom nebo metody hrubou sílou pro počet prvků v uzlu 64:



Graf 1. Čas (v sekundách) provedení dotazu na k nejbližších sousedů.

Je zřejmé, že algoritmus M-strom pracuje skoro pětkrát rychleji než metoda hrubou sílou (čas v sekundách).

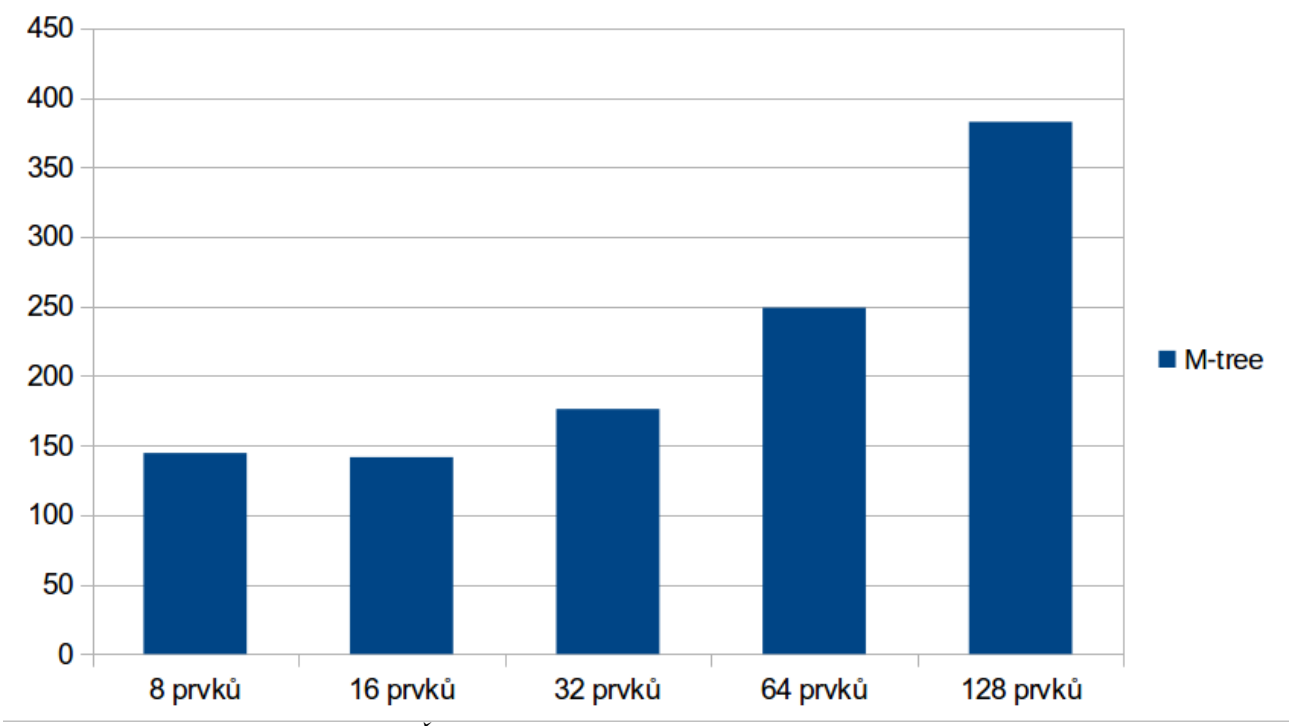
Teď prozkoumáme rychlost algoritmu M-strom v závislosti na počtu prvků v uzlu:



Graf 2. Rychlost zpracování dotazů v závislosti na počtu prvků v uzlu.

Je vidět, že čím víc prvků patří do uzlu, tím rychleji pracuje program (čas v sekundách).

Dál prozkoumáme vliv počtu prvků uzlů na rychlost vytváření stromu (čas v milisekundách):



Graf 3. Čas (v sekundách) vytváření struktury M-stromu v závislosti na počtu prvků v uzlech.

Můžeme si všimnout, že čím víc prvků je v uzlu, tím víc času je potřeba na vygenerování struktury stromu.

Diskuze

Funkce promotion je implementovaná na základě náhodného výběru dvou objektů. Možností vylepšení je implementace např. algoritmu m_RAD nebo mM_RAD. Taky nedostatkem je to, že na vytvoření struktury stromu je potřeba poměrně hodně času a ta zabírá určité množství paměti na disku. A na závěr to, že strom by byl víc vyvážený kdyby se použil Bulk-algoritmus stažení dat a proto vyhledávání by trvalo méně času.
Výhodou naše implementace je to, že M-strom jde použit s minimálním množstvím RAM-paměti.

Závěr

Vytvořili jsme vlastní implementaci metrické přístupové metody M-strom s jednoduchým GUI pro pohodlnost zadávání dat a vyhledávání. V našem projektu byly použity prostředky jazyka Java. Prozkoumali jsme taky rozdíl mezi časem běhu programu při použití algoritmu M-strom a při použití metody hrubou sílou. Taky jsme určili závislost času běhu programu na počtu prvků v uzlech. A po tomto výzkumu je jasné, že algoritmus M-strom je velmi efektivní co se tyče časové složitostí vyhledávání v metrickém prostoru, ale má taky své nevýhody (poměrně hodně času na vytváření struktury stromu).