Vyhledávání k přibližných nejbližších sousedů pomocí BBD stromů

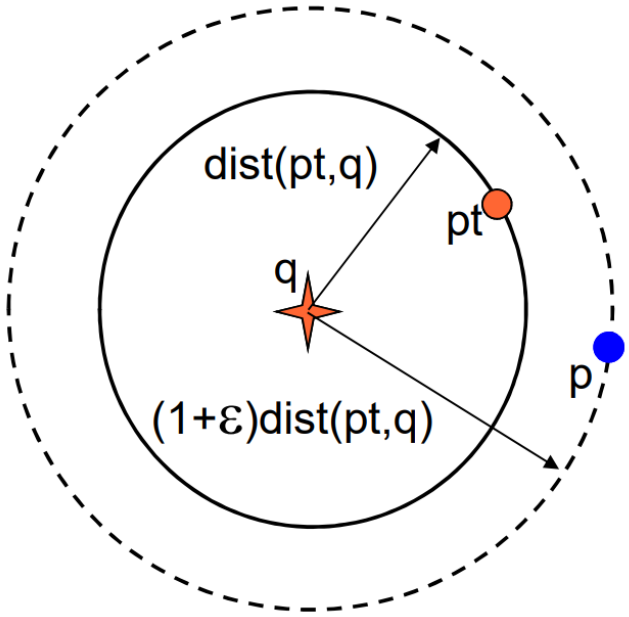
Approximate k-nearest neighbor search with BBD-trees

Petr Šádek

21.11.2022

Připomenutí – přibližný nejbližší soused

Pro $\varepsilon > 0$ definujeme bod **p** jako $(1 + \varepsilon)$–přibližného souseda bodu **q** pokud:
dist(**p**, **q**) < (1 + ε)*dist(**pt**, **q**), kde **pt** je nejbližší soused **q**

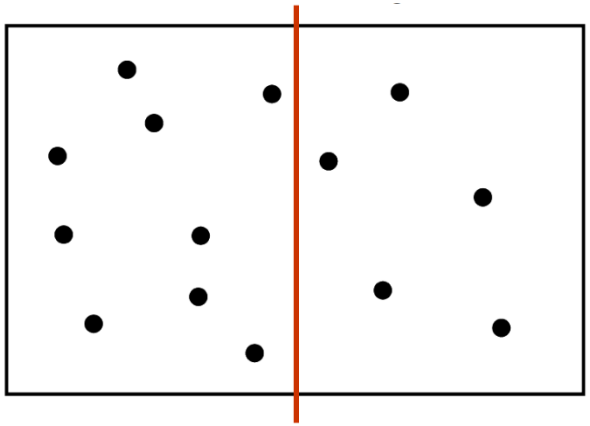


Vyhledávání k přibližných nejbližších sousedů

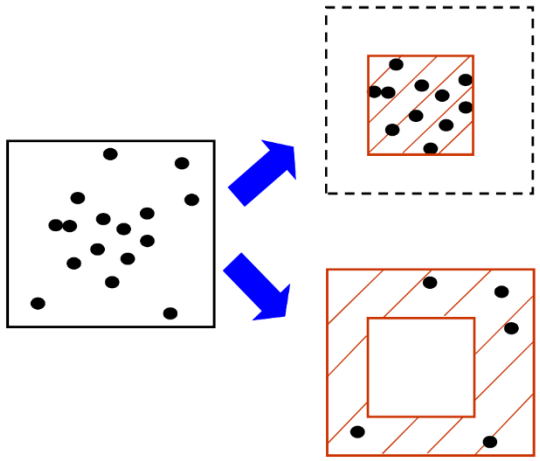
- Může být výrazně rychlejší než vyhledávání přesných nejbližších sousedů v závislosti na zvoleném ε
- Implementace využívá BBD strom
  - Pro některé typy dat stačí kd strom
- Použití
  - Fotonové mapy
  - Ve strojovém učení

BBD strom

- Balanced box-decomposition tree – vyvážený strom hierarchie obálek
- Podobný kd-stromu
  - Klasické dělení podle rovin kolmých na souřadnicové osy (fair split)



  - Obsahuje navíc „smršťující“ (shrinking) uzly, které dělí uzel na vnitřní a vnější část

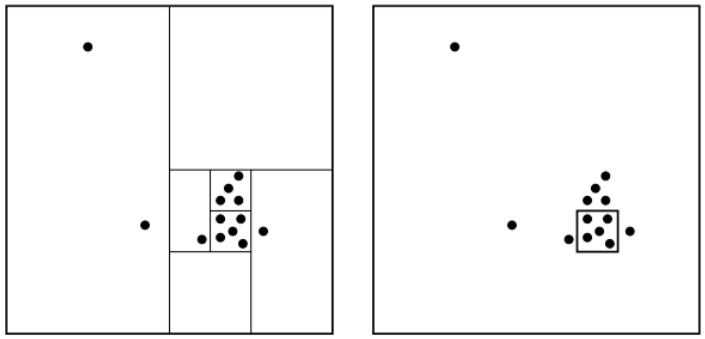


Stavba stromu

- Chceme aby většina uzlů dělila rovinou (fair split), smršťování (shrinking) děláme jen když to je nutno
  - zjištění vzdálenosti od obálky je mnohem jednodušší, také jednodušší reprezentace uzlu
- Postup (midpoint algoritmus):
  - Vždy nejprve dělíme podle osy kde má obálka největší rozměr, dělíme v půlce intervalu
  - Pokud po d/2 děleních nesnížíme počet bodů alespoň na polovinu, pak uděláme smršťování
- Po dokončení stavby stromu nahradíme posloupnosti triviálních uzlů jedním triviálním smršťováním
- Pro většinu dat je výsledný počet vzniklých smršťovacích uzlů malý (5-20%), často stačí použít klasický kd-strom

Smršťování

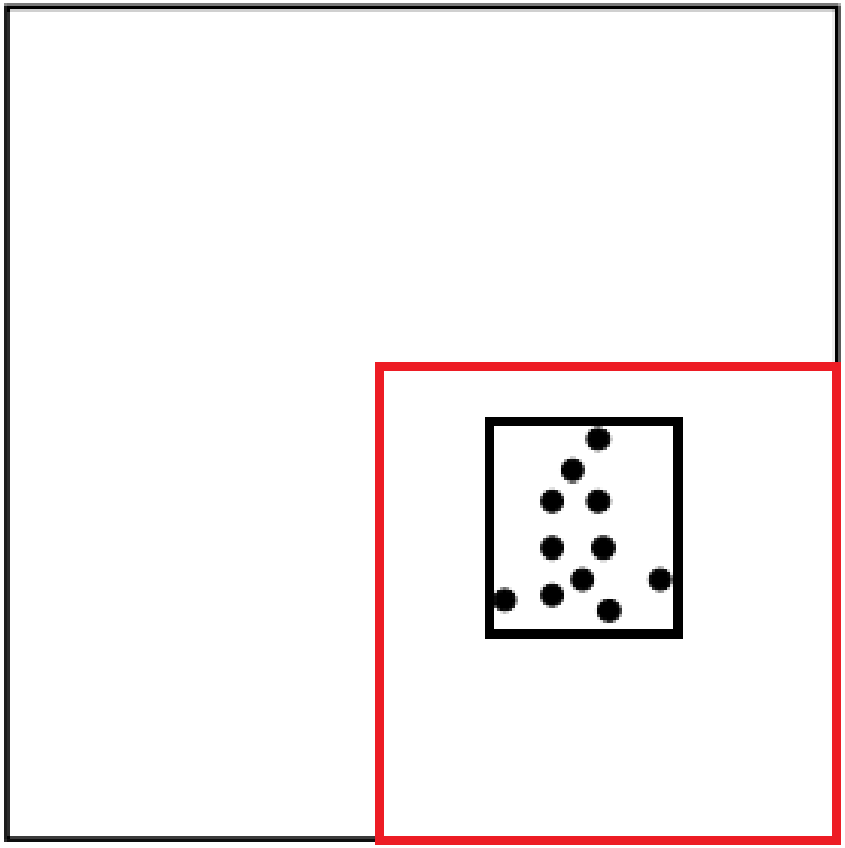
- Dělím prostor rovinami dokud mi počet bodů neklesne pod 2/3 celkového počtu
- Poté tuto posloupnost dělení nahradím jedním smršťovacím uzlem



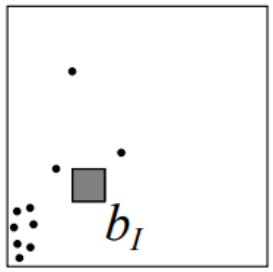
Problémy:

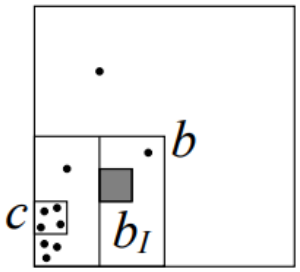
- Dělení může být hodně
- Smršťující uzly se mi v hierarchii můžou překrývat

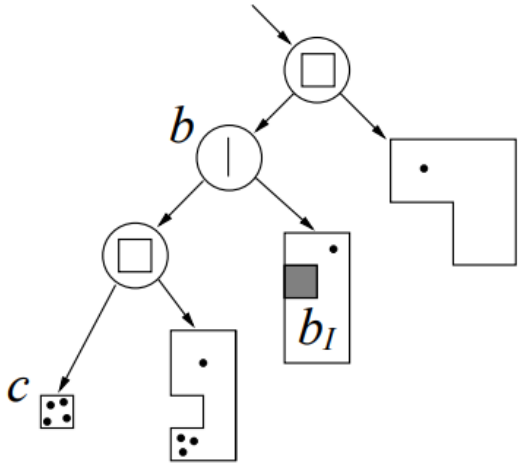
Řešení 1. problému

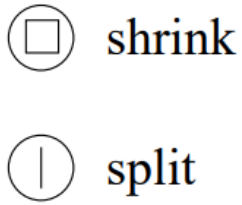


Řešení 2. problému



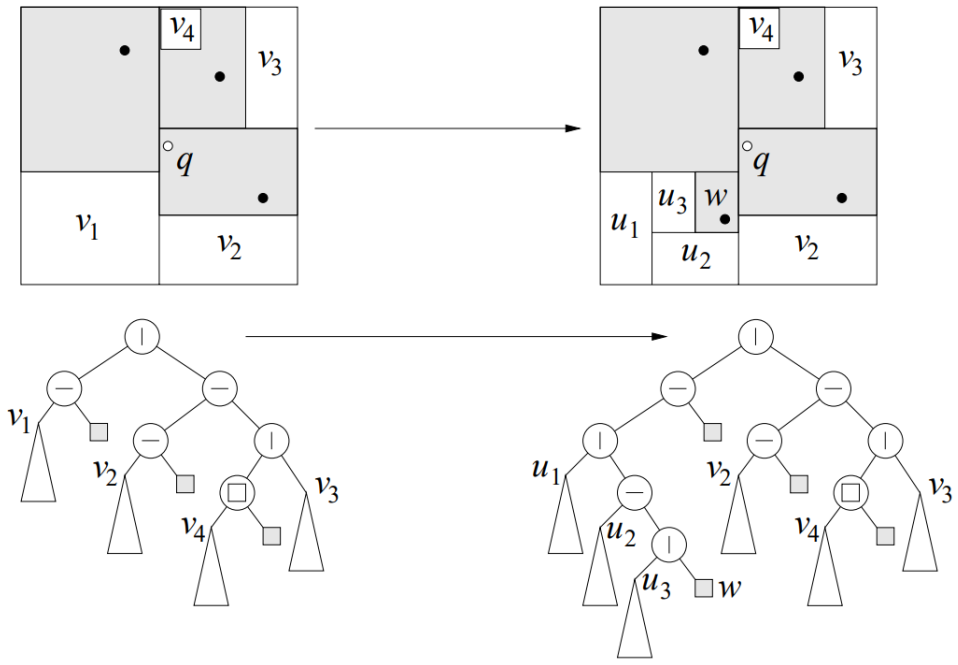




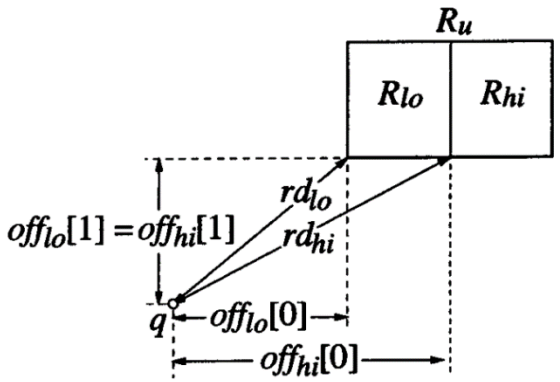


Hledání $(1 + \varepsilon)$– přibližného nejbližšího souseda

Průchod prioritní frontou podle vzdálenosti obálky

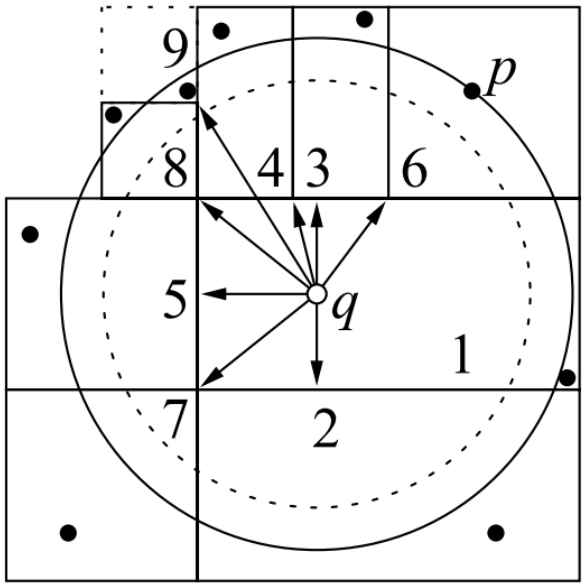


Při průchodu uzly dělícími rovinou:



Hledání $(1+\varepsilon)$– přibližného nejbližšího souseda

Zastavíme když je vzdálenost aktuální obálky větší než $r/(1+\varepsilon)$, kde r je vzdálenost k nejbližšímu dosud nalezenému bodu.




```
Distance ‘d’ = infinity, approximate nearest neighbor ‘N’ = none
Insert root node to PQ, dist 0
DO
  ‘node’ = take closest one from PQ
  IF ‘node’ is a leaf THEN
    Compute distance ‘dL’ to the closest point in the leaf
    Record the nearest neighbor ‘N’ so far and update ‘d’ by ‘dL’      (if ‘dL’ < ‘d’)
  ELSE /* interior node */
     Insert the farther child to PQ (distance d1)
     Insert the closer child to PQ (distance d2)
  ENDIF
  ‘Dclosest’ = the closest node to query in PQ
UNTIL  (d / (1+eps) > Dclosest)
Report the approximate nearest neighbor ‘N’
```

Hledání **k** $(1+\boldsymbol{\varepsilon})$– přibližných nejbližších sousedů

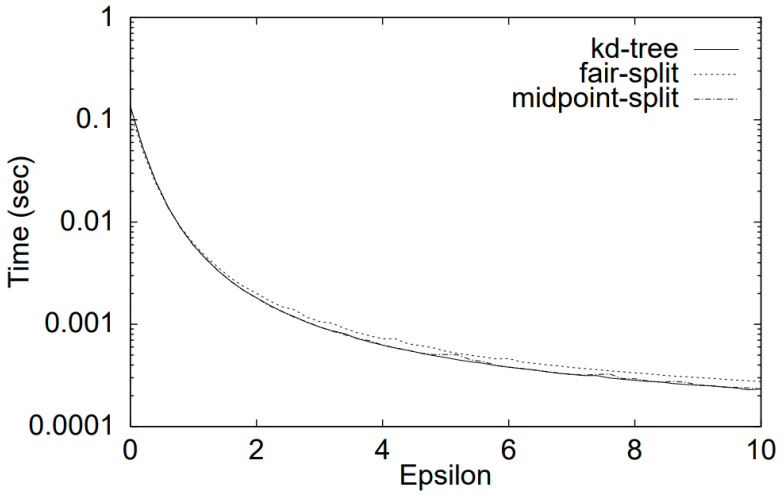
- Stejné, jen máme navíc frontu k nejbližších
- Ukončovací podmínku vyhodnocujeme pro bod z prioritní fronty co je nejblíže
- Pouze nesmíme zastavit dokud nenalezneme alespoň k sousedů

Vlastnosti BBD stromu

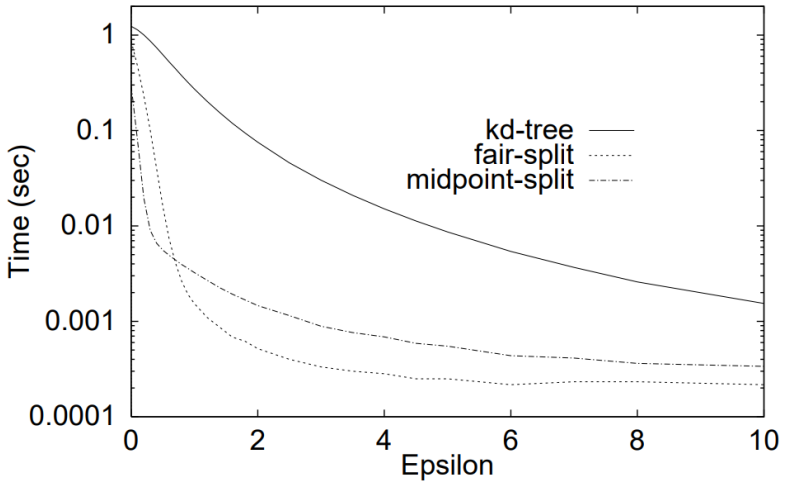
- Čas stavby: $O(dn \log n)$
- Paměťová náročnost: $O(dn)$
- Nalezení $(1+\boldsymbol{\varepsilon})$– přibližného nejbližšího souseda:
  - $\mathrm{O}\left(\left\lceil 1+\frac{6d}{\varepsilon}\right\rceil^{d} d \log n\right)$
  - Jen horní odhad, ve skutečnosti je v průměru mnohem lepší
  - U kd-stromu: $\mathrm{O}\left(2^{d} \log n\right)$
- Nalezení k $(1+\boldsymbol{\varepsilon})$– přibližných nejbližších sousedů:
  - $\mathrm{O}\left(\left(k+\left\lceil 1+\frac{6d}{\varepsilon}\right\rceil^{d}\right) d \log n\right)$

Čas dotazu

Uniformní data

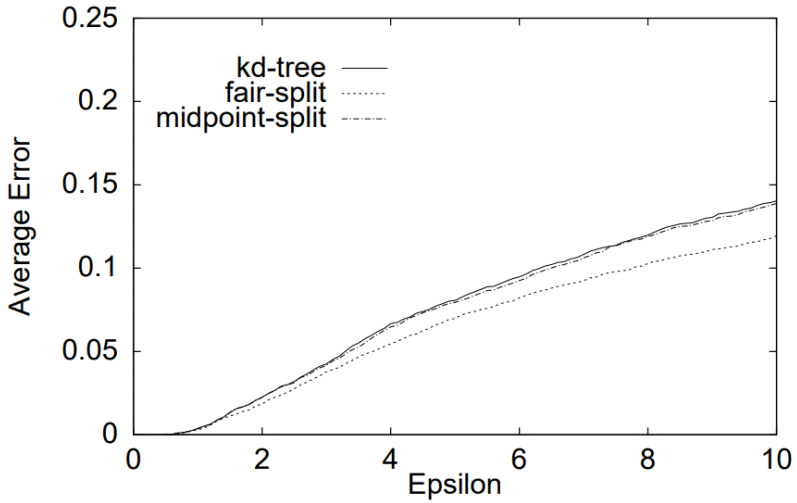


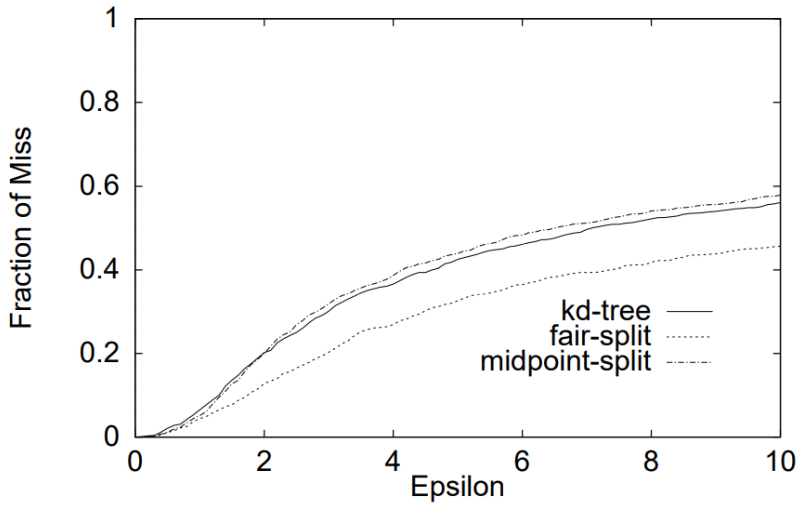
Shluky



- Měřeno pro 100 000 bodů, dimenze 16
- Pro uniformní rozdělení dat stačí klasický kd-strom
  - Podobně u normálního rozdělení
- Pro shluková data se BBD-strom vyplácí

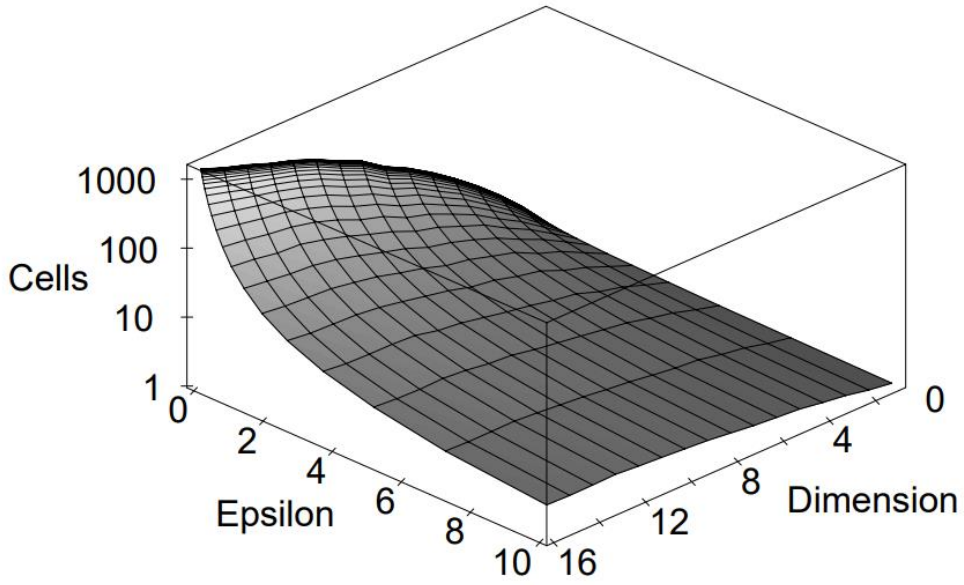
Chyba





- Grafy jsou pro uniformní rozdělení dat
  - Podobné výsledky jsou pro normální rozdělení
- I pro velké ε je poměrně velká šance že se trefíme do nejbližšího souseda

Počet navštívených listů na dotaz



- Horní odhad v článku je: $\lceil 1 + 6d/\varepsilon \rceil^d$
- Pro $\varepsilon = 1$ a dimenzi 16 by tento odhad byl 10^{31}
- Ve skutečnosti je ale navštíveno jenom zhruba 100 listů

Měření

- T_B – čas postavení datové struktury
- T_R – průměrný čas na dotaz pro hodně dotazů
- N_TR – průměrný počet kroků při průchodu datovou strukturou
- N_Q – počet dotazů pro konkrétní testy
- M – spotřeba paměti datový struktur
- PERF – počet dotazů za sekundu
- N_I – počet vnitřních uzlů
- N_L – počet listů
- D_MAX – maximální hloubka hierarchie
- N_AP – průměrný počet bodů v listech

Děkuji za pozornost