Vyhledávání k přibližných nejbližších sousedů pomocí BBD stromů

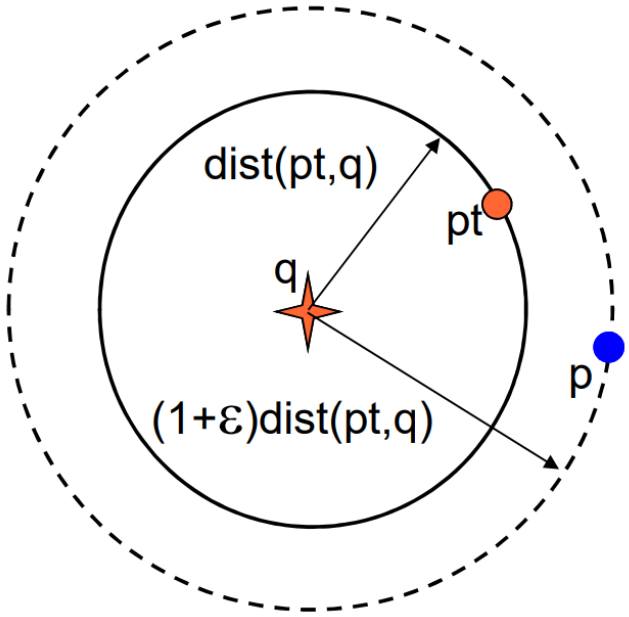
Approximate k-nearest neighbor search with BBD-trees

Petr Šádek

9.2.2023

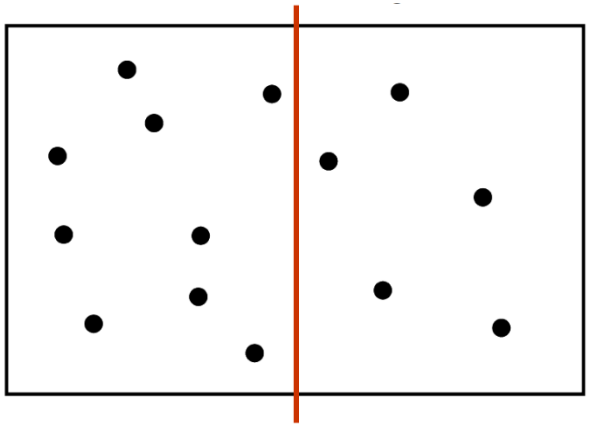
Připomenutí – přibližný nejbližší soused

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**

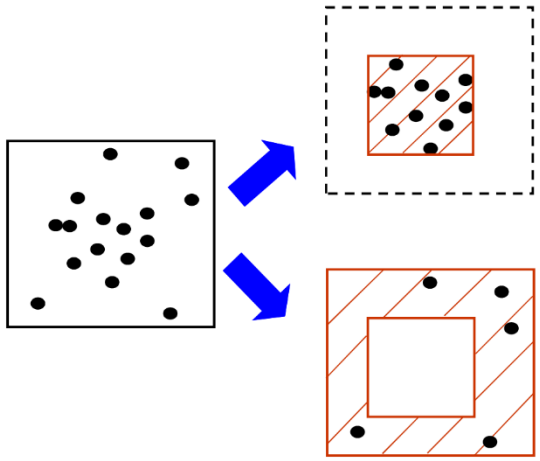


BBD strom

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

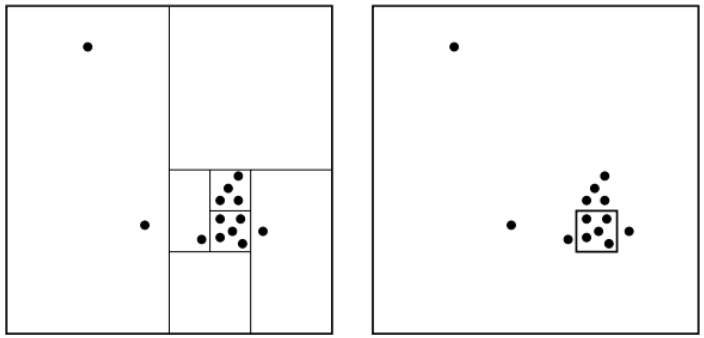


  - Obsahuje navíc „smršťující“ (shrinking) uzly, které dělí uzel na vnitřní a vnější část



Stavba stromu

- Postup (midpoint algoritmus):
  - Dělíme podle osy kde má obálka největší rozměr, dělíme v půlce intervalu
  - Opakujeme dokud počet bodů neklesne pod 2/3 celkového počtu
  - Pokud stačilo pouze jedno dělení, tak vytvoříme uzel dělící rovinou
  - Jinak vytvoříme smršťující uzel
- Při stavbě zároveň dělíme vstupní body podobně jako v quick sortu

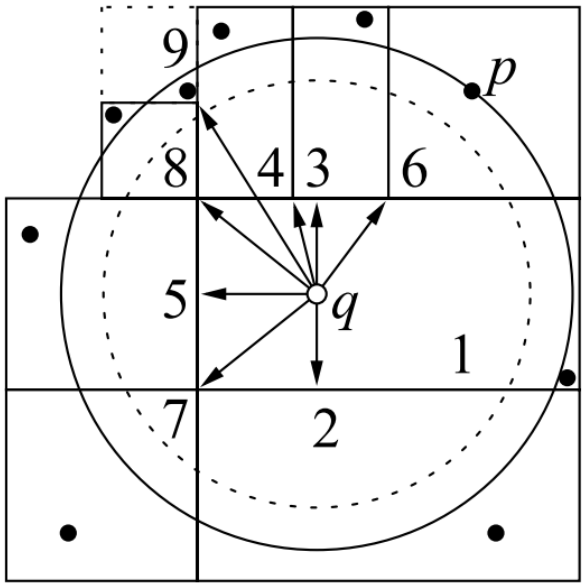


Typy uzlů

- Všechny uzly obsahují typ uzlu v prvních 2 bitech
- Uzel dělící rovinou – 4 B
  - 2 bity typ uzlu
  - 2 bity dimenze dělení
  - 1 bit indikuje zda má levého potomka
  - 27 bitů index pravého potomka
- Smršťující uzel – pro 3D body 28 B (4 + 2 * 3 * 4)
  - Podobné rozpoložení bitů v prvních 4 B jako u uzlu dělícího rovinou
  - Obsahuje 2 vektory reprezentující obálku
- List – 8 B
  - Obsahuje 2 indexy na začátek a konec rozsahu bodů na které se odkazuje

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** $(1+\varepsilon)$– přibližných nejbližších sousedů

- Stejné, jen máme navíc frontu k nejbližších
- Ukončovací podmínku vyhodnocujeme pro bod z prioritní fronty co je poslední
- Nesmíme zastavit dokud nenalezneme alespoň k sousedů
- Lze použít různé typy front, použil jsem:
  - Lineární frontu
  - Haldu – vlastní implementace
  - Haldu – c++ std implementace

Statistiky stavby stromu

Distrib.	N_P	T_B	M	N_I	N_L	$N_S(\%)$	N_{AP}	D_{MAX}	D_{AVG}
clusters	10^3	88.0 μs	3 kB	137	138	53.3	7.2	9	7.3
normal	10^3	89.0 μs	3 kB	138	139	44.2	7.2	10	7.4
uniform	10^3	62.0 μs	2 kB	136	137	5.9	7.3	8	7.2
clusters	10^5	11.0 ms	246 kB	14052	14053	24.7	7.1	17	14.1
normal	10^5	11.5 ms	218 kB	14085	14086	15.9	7.1	16	14.0
uniform	10^5	9.2 ms	184 kB	14156	14157	5.3	7.1	15	13.9
clusters	10^7	1.4 s	19 MB	1402959	1402960	8.0	7.1	25	20.8
normal	10^7	1.4 s	18 MB	1402984	1402985	6.7	7.1	24	20.7
uniform	10^7	1.2 s	18 MB	1398423	1398424	5.8	7.2	22	20.5

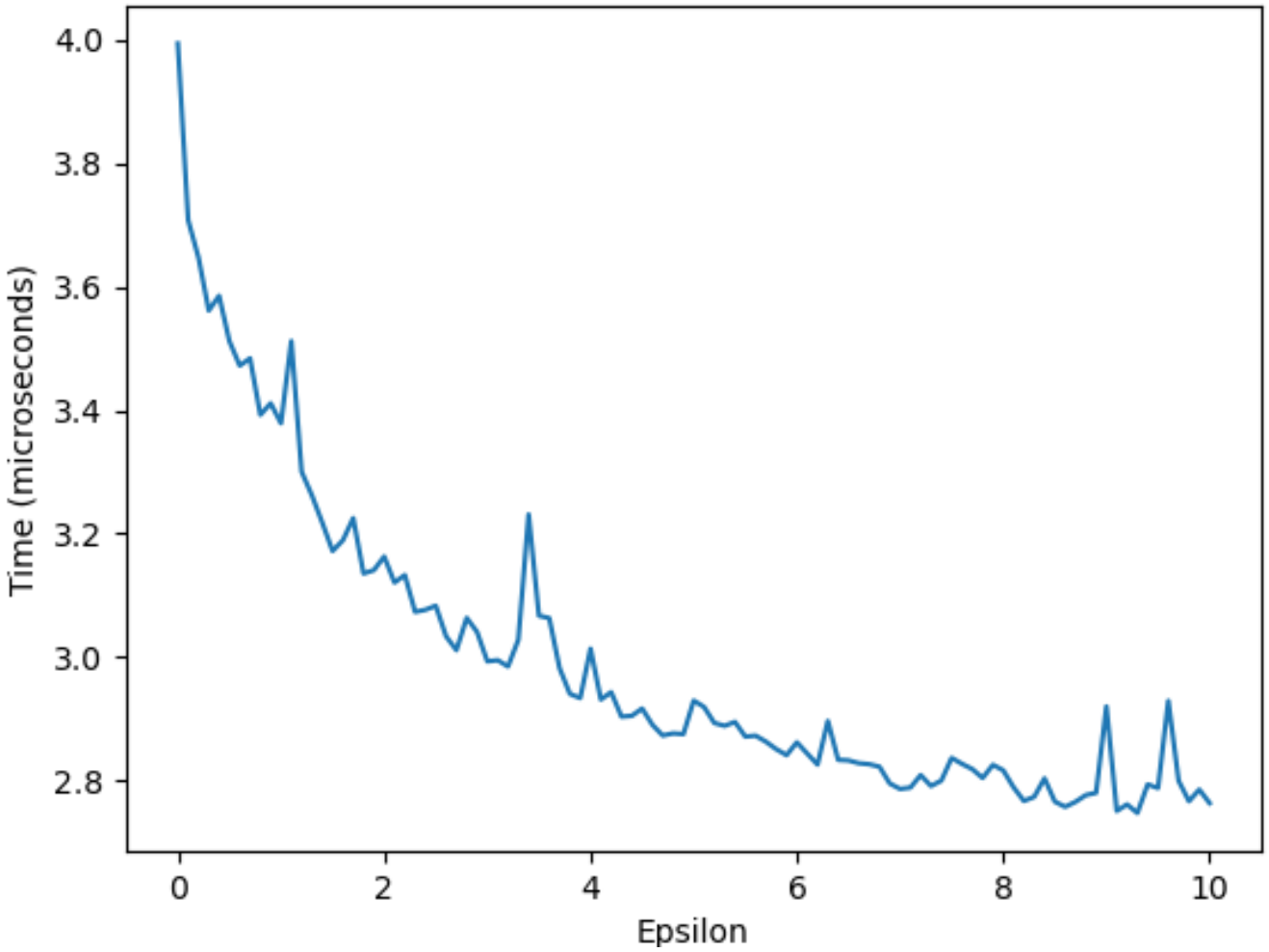
Měřeno pro dimenzi = 3, max velikost listu = 10

Statistiky dotazů

		$\epsilon = 0$			$\epsilon = 1$			$\epsilon = 10$		
D	N	$T_R(\mu s)$	N_{TR}	S	$T_R(\mu s)$	N_{TR}	S	$T_R(\mu s)$	N_{TR}	S
2	10^3	4.0	25.4	2.3	4.2	22.1	2.2	2.1	18.6	4.3
2	10^5	7.5	53.0	74.9	6.3	52.5	89.2	5.4	48.1	104.0
2	10^7	12.9	75.1	4391	7.5	69.4	7554	6.3	61.6	8993
3	10^3	6.2	43.7	1.5	3.8	31.4	2.4	2.2	19.0	4.1
3	10^5	11.9	90.2	49.2	8.7	72.9	67.3	5.7	46.5	102.6
3	10^7	19.4	102.1	3038	10.1	88.8	5836	7.9	70.8	7462
4	10^3	8.9	70.0	1.0	4.7	35.7	2.0	2.4	17.4	3.8
4	10^5	21.0	141.9	30.4	12.3	100.8	52.0	7.1	53.2	90.0
4	10^7	42.5	190.0	1535	13.9	115.6	4695	10.3	83.4	6337

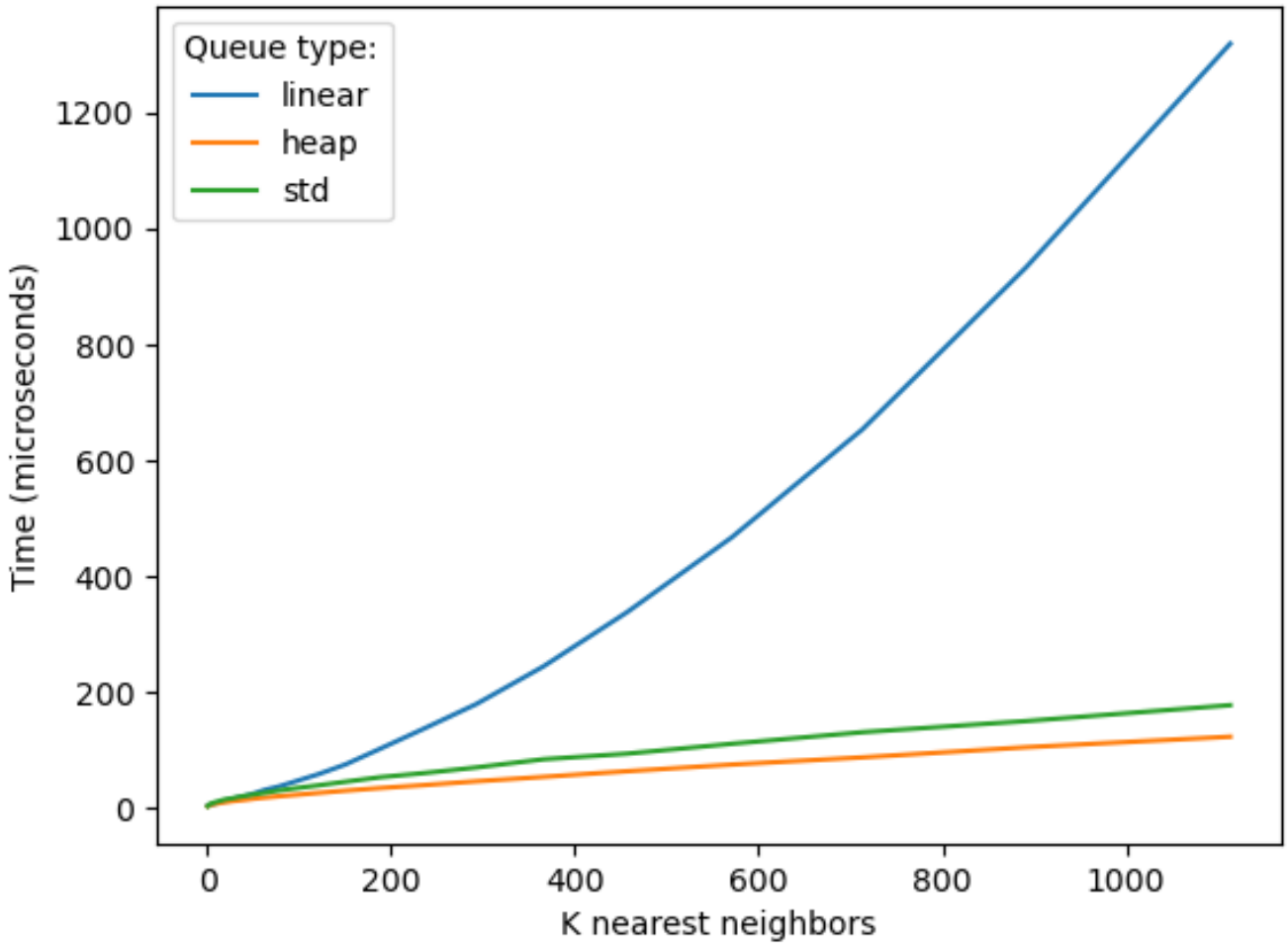
Měřeno pro k = 10, použita implementace fronty haldou, max velikost listu = 10

Zrychlení v závislosti na epsilon



Měřeno pro 100 000 bodů, dimenze 3, k = 1, clusterová data

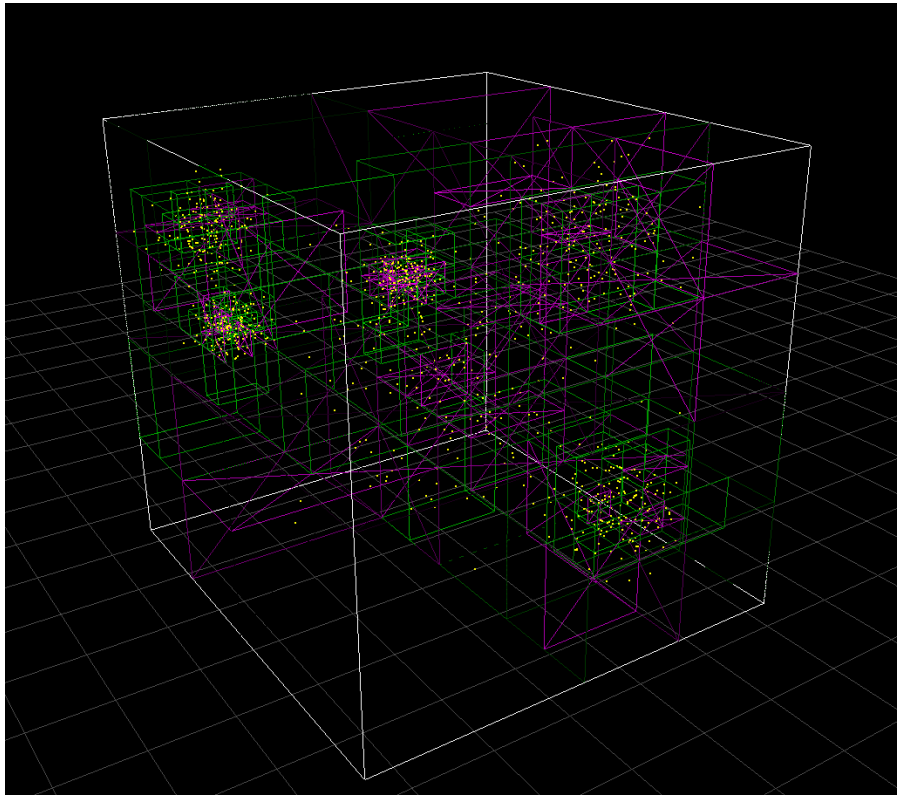
Porovnání prioritních front

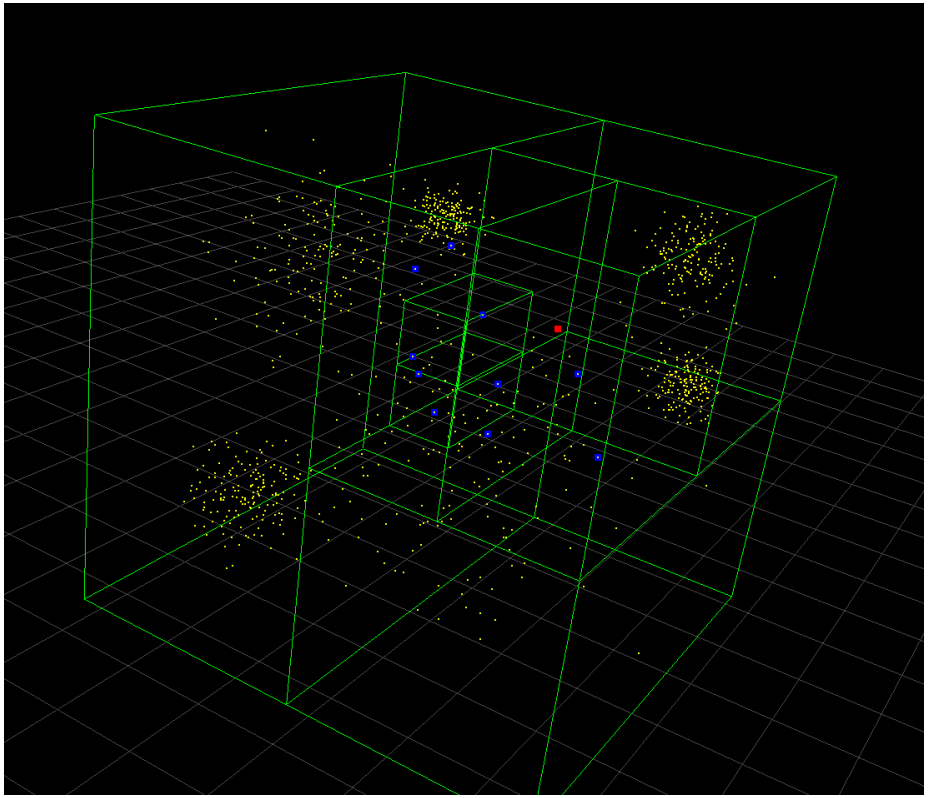


Měřeno pro 100 000 bodů, dimenze 3, epsilon = 0, clusterová data

ČVUT
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Vizualizace





Děkuji za pozornost