Drzewa czwórkowe i kd-drzewa

Michał Dobranowski   Wiktor Perczak

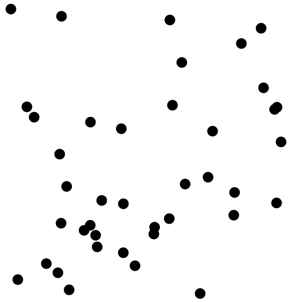
2 stycznia 2024

Plan prezentacji

1. Przedstawienie problemu
2. Drzewa czwórkowe
3. Kd-drzewa
4. Porównanie

Przedstawienie problemu

Dany jest zbiór n punktów P na płaszczyźnie. Chcemy odpowiadać na zapytania typu:

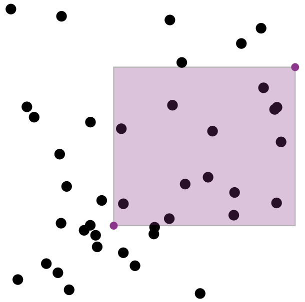


Przedstawienie problemu

Dany jest zbiór n punktów P na płaszczyźnie. Chcemy odpowiadać na zapytania typu:

dla zadanych x_1, x_2, y_1, y_2 znaleźć punkty $p \in P$ takie, że

$$x_1 \leq x_p \leq x_2, y_1 \leq y_p \leq y_2.$$



Rozwiązanie trywialne

Sprawdzić każdy punkt. Złożoność czasowa zapytania: $\mathcal{O}(n)$.

Rozwiązanie trywialne

Sprawdzić każdy punkt. Złożoność czasowa zapytania: $\mathcal{O}(n)$.

```
filter(lambda p: x_1 <= p[0] <= x_2 and y_1 <= p[1] <= y_2, points)
```

Rozwiązanie trywialne

Sprawdzić każdy punkt. Złożoność czasowa zapytania: $\mathcal{O}(n)$.

```
filter(lambda p: x_1 <= p[0] <= x_2 and y_1 <= p[1] <= y_2, points)
```

Nie uda nam się poprawić złożoności czasowej, jeśli miałaby zależeć tylko od n. Chcemy więc, żeby zależała od liczby punktów wynikowych, którą oznaczymy przez k.

Drzewa czwórkowe – opis struktury

Drzewo czwórkowe (ang. *quadtree*) do drzewiastą struktura danych, w której:

Drzewa czwórkowe – opis struktury

Drzewo czwórkowe (ang. *quadtree*) do drzewiastą struktura danych, w której:

1. każdy wierzchołek odpowiada za pewnien prostokąt na płaszczyźnie,

Drzewa czwórkowe – opis struktury

Drzewo czwórkowe (ang. *quadtree*) do drzewiastą struktura danych, w której:

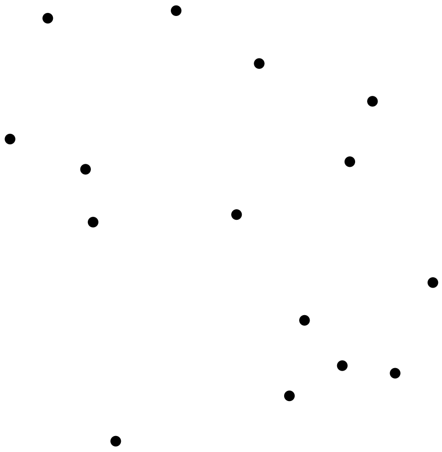
1. każdy wierzchołek odpowiada za pewnien prostokąt na płaszczyźnie,
2. każdy wierzchołek posiada maksymalnie czworo dzieci, z których każdy odpowiada za ćwiartkę prostokątu rodzica,

Drzewa czwórkowe – opis struktury

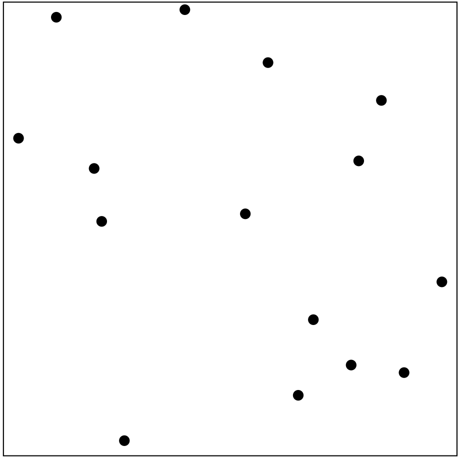
Drzewo czwórkowe (ang. *quadtree*) do drzewiastą struktura danych, w której:

1. każdy wierzchołek odpowiada za pewnien prostokąt na płaszczyźnie,
2. każdy wierzchołek posiada maksymalnie czworo dzieci, z których każdy odpowiada za ćwiartkę prostokątu rodzica,
3. każdy liść odpowiada za jeden punkt na płaszczyźnie.

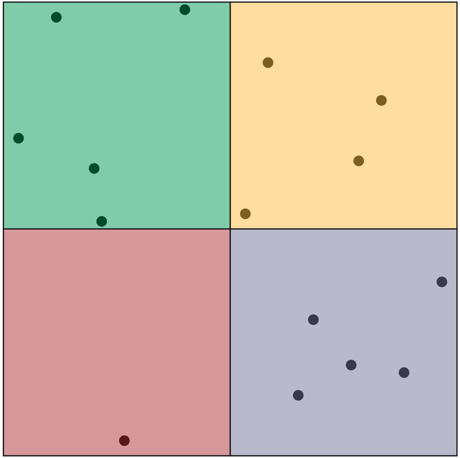
Drzewa czwórkowe – sposób podziału

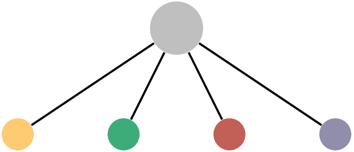


Drzewa czwórkowe – sposób podziału

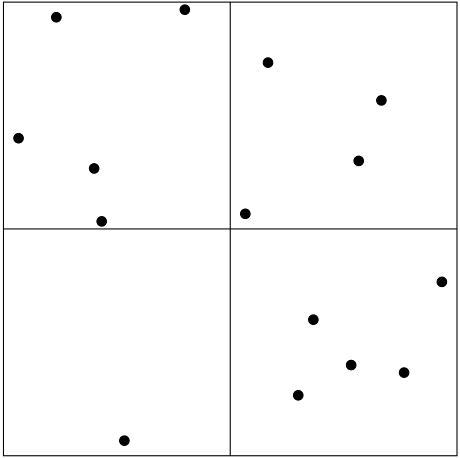


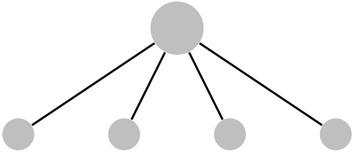
Drzewa czwórkowe – sposób podziału



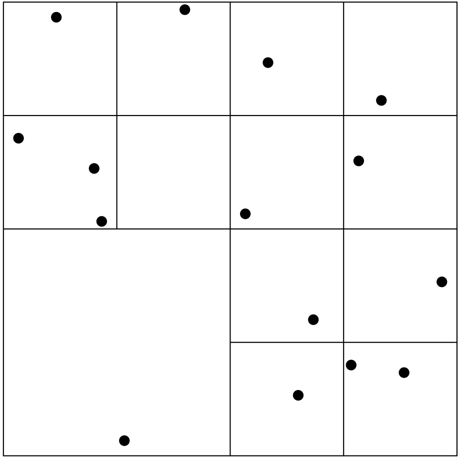


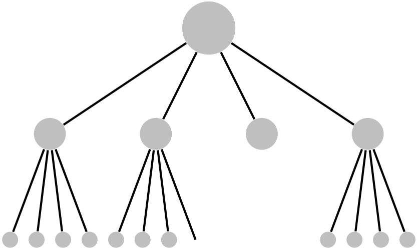
Drzewa czwórkowe – sposób podziału



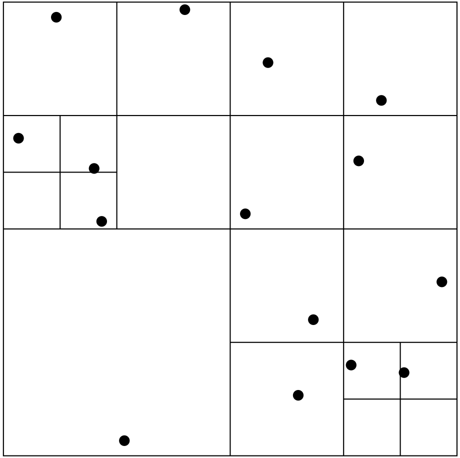


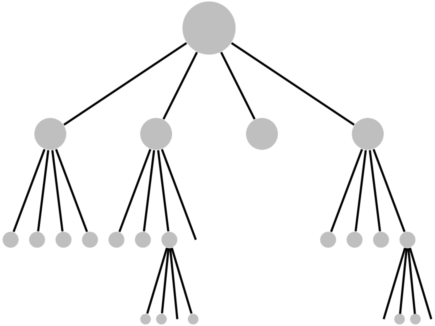
Drzewa czwórkowe – sposób podziału





Drzewa czwórkowe – sposób podziału





Drzewa czwórkowe – odpowiadanie na zapytania

Znajdowanie punktów, które należą do zadanego obszaru działa następująco:

1. przeszukujemy drzewo w dowolny sposób (DFS, BFS, ...), dla każdego niebędącego liściem wierzchołka sprawdzając, czy odpowiadający mu prostokąt ma część wspólną z szukanym prostokątem,

Drzewa czwórkowe – odpowiadanie na zapytania

Znajdowanie punktów, które należą do zadanego obszaru działa następująco:

1. przeszukujemy drzewo w dowolny sposób (DFS, BFS, ...), dla każdego niebędącego liściem wierzchołka sprawdzając, czy odpowiadający mu prostokąt ma część wspólną z szukanym prostokątem,
2. jeśli tak, to schodzimy wgłąb tego poddrzewa,

Drzewa czwórkowe – odpowiadanie na zapytania

Znajdowanie punktów, które należą do zadanego obszaru działa następująco:

1. przeszukujemy drzewo w dowolny sposób (DFS, BFS, ...), dla każdego niebędącego liściem wierzchołka sprawdzając, czy odpowiadający mu prostokąt ma część wspólną z szukanym prostokątem,
2. jeśli tak, to schodzimy wgłąb tego poddrzewa,
3. gdy dojdziemy do liścia, sprawdzamy, czy odpowiadający mu punkt należy do szukanego prostokąta.

Drzewa czwórkowe – analiza złożoności

Lemat (wysokość drzewa czwórkowego)

Niech

$$\beta = \frac{\max_{p,q \in P} \|p - q\|}{\min_{p,q \in P, p \neq q} \|p - q\|}.$$

Wysokość h drzewa czwórkowego dla zbioru P jest $\Theta(\log \beta)$.

Drzewa czwórkowe – analiza złożoności

Lemat (wysokość drzewa czwórkowego)

Niech

$$\beta = \frac{\max_{p,q \in P} \|p - q\|}{\min_{p,q \in P, p \neq q} \|p - q\|}.$$

Wysokość h drzewa czwórkowego dla zbioru P jest $\Theta(\log \beta)$.

Warto zauważyć, że h nie jest ograniczone od góry przez funkcję n. Łatwo pokazać, że h jest $\Omega(\log n)$ (a jeśli punkty w P są rozłożone równomiernie, to $\Theta(\log n)$).

Drzewa czwórkowe – analiza złożoności

Lemat (wysokość drzewa czwórkowego)

Niech

$$\beta = \frac{\max_{p,q \in P} \|p - q\|}{\min_{p,q \in P, p \neq q} \|p - q\|}.$$

Wysokość h drzewa czwórkowego dla zbioru P jest $\Theta(\log \beta)$.

Warto zauważyć, że h nie jest ograniczone od góry przez funkcję n. Łatwo pokazać, że h jest $\Omega(\log n)$ (a jeśli punkty w P są rozłożone równomiernie, to $\Theta(\log n)$).
Złożoność

1. konstrukcji drzewa:
2. zapytania:
3. pamięciowa:

Drzewa czwórkowe – analiza złożoności

Lemat (wysokość drzewa czwórkowego)

Niech

$$\beta = \frac{\max_{p,q \in P} \|p - q\|}{\min_{p,q \in P, p \neq q} \|p - q\|}.$$

Wysokość h drzewa czwórkowego dla zbioru P jest $\Theta(\log \beta)$.

Warto zauważyć, że h nie jest ograniczone od góry przez funkcję n. Łatwo pokazać, że h jest $\Omega(\log n)$ (a jeśli punkty w P są rozłożone równomiernie, to $\Theta(\log n)$).
Złożoność

1. konstrukcji drzewa: $\mathcal{O}(hn)$,
2. zapytania:
3. pamięciowa:

Drzewa czwórkowe – analiza złożoności

Lemat (wysokość drzewa czwórkowego)

Niech

$$\beta = \frac{\max_{p,q \in P} \|p - q\|}{\min_{p,q \in P, p \neq q} \|p - q\|}.$$

Wysokość h drzewa czwórkowego dla zbioru P jest $\Theta(\log \beta)$.

Warto zauważyć, że h nie jest ograniczone od góry przez funkcję n. Łatwo pokazać, że h jest $\Omega(\log n)$ (a jeśli punkty w P są rozłożone równomiernie, to $\Theta(\log n)$).

Złożoność

1. konstrukcji drzewa: $\mathcal{O}(hn)$,
2. zapytania:
3. pamięciowa: $\mathcal{O}(hn)$.

Drzewa czwórkowe – analiza złożoności

Lemat (wysokość drzewa czwórkowego)

Niech

$$\beta = \frac{\max_{p,q \in P} \|p - q\|}{\min_{p,q \in P, p \neq q} \|p - q\|}.$$

Wysokość h drzewa czwórkowego dla zbioru P jest $\Theta(\log \beta)$.

Warto zauważyć, że h nie jest ograniczone od góry przez funkcję n. Łatwo pokazać, że h jest $\Omega(\log n)$ (a jeśli punkty w P są rozłożone równomiernie, to $\Theta(\log n)$).
Złożoność

1. konstrukcji drzewa: $\mathcal{O}(hn)$,
2. zapytania: $\mathcal{O}(hk)$,
3. pamięciowa: $\mathcal{O}(hn)$.

kd-drzewo – opis struktury

Kd-drzewo (ang. *kd-tree*) to drzewo binarne, w którym:

kd-drzewo – opis struktury

Kd-drzewo (ang. *kd-tree*) to drzewo binarne, w którym:

1. każdy poziom drzewa odpowiada za wymiar, względem którego dzieli się punkty,

kd-drzewo – opis struktury

Kd-drzewo (ang. *kd-tree*) to drzewo binarne, w którym:

1. każdy poziom drzewa odpowiada za wymiar, względem którego dzieli się punkty,
2. każdy wierzchołek odpowiada za wartość współrzędnej podziału w danym wymiarze,

kd-drzewo – opis struktury

Kd-drzewo (ang. *kd-tree*) to drzewo binarne, w którym:

1. każdy poziom drzewa odpowiada za wymiar, względem którego dzieli się punkty,
2. każdy wierzchołek odpowiada za wartość współrzędnej podziału w danym wymiarze,
3. wartość współrzędnej podziału jest przyjmowana jako mediana zbioru (znajdowana za pomocą algorytmu quickselect),

kd-drzewo – opis struktury

Kd-drzewo (ang. *kd-tree*) to drzewo binarne, w którym:

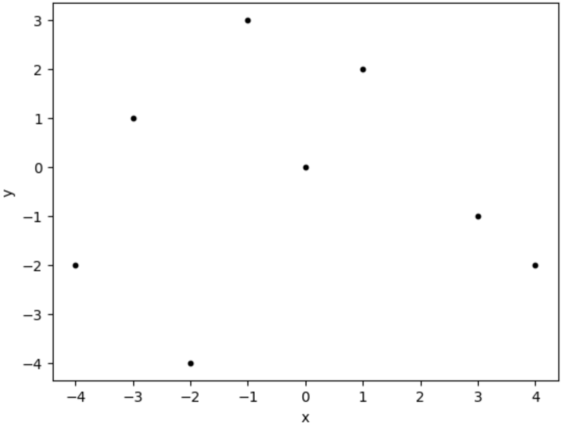
1. każdy poziom drzewa odpowiada za wymiar, względem którego dzieli się punkty,
2. każdy wierzchołek odpowiada za wartość współrzędnej podziału w danym wymiarze,
3. wartość współrzędnej podziału jest przyjmowana jako mediana zbioru (znajdowana za pomocą algorytmu quickselect),
4. dla każdego wierzchołka obiekty mniejsze trafiają do lewego potomka, większe do prawego, a równe albo do lewego albo do prawego (dla zrównoważenia drzewa),

kd-drzewo – opis struktury

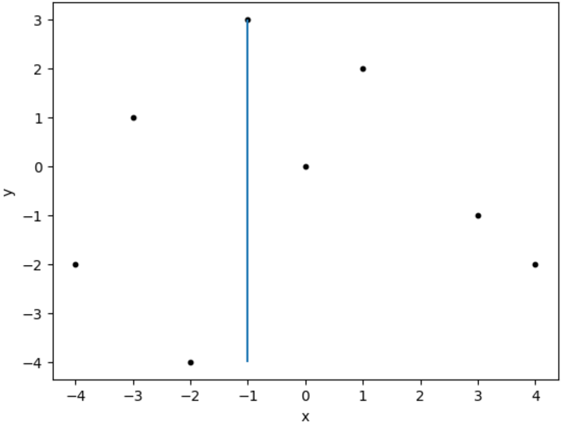
Kd-drzewo (ang. *kd-tree*) to drzewo binarne, w którym:

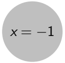
1. każdy poziom drzewa odpowiada za wymiar, względem którego dzieli się punkty,
2. każdy wierzchołek odpowiada za wartość współrzędnej podziału w danym wymiarze,
3. wartość współrzędnej podziału jest przyjmowana jako mediana zbioru (znajdowana za pomocą algorytmu quickselect),
4. dla każdego wierzchołka obiekty mniejsze trafiają do lewego potomka, większe do prawego, a równe albo do lewego albo do prawego (dla zrównoważenia drzewa),
5. każdy liść odpowiada za jeden punkt na płaszczyźnie.

kd-drzewo – sposób podziału

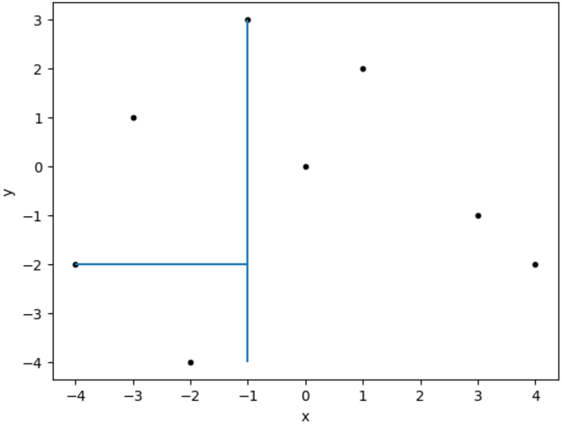


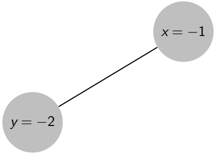
kd-drzewo – sposób podziału



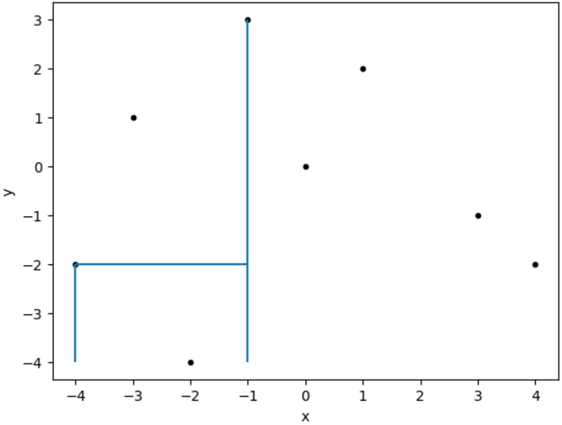


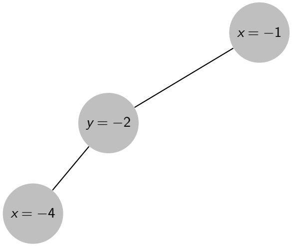
kd-drzewo – sposób podziału



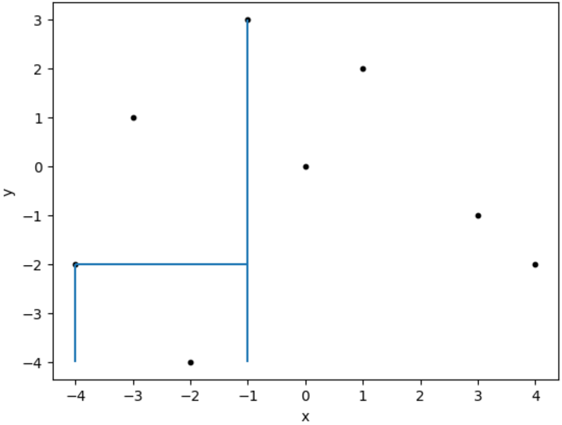


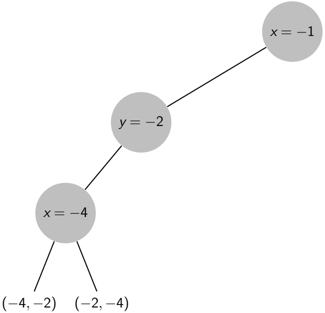
kd-drzewo – sposób podziału



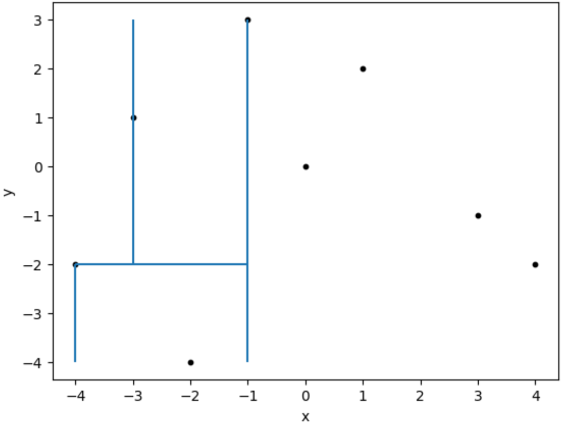


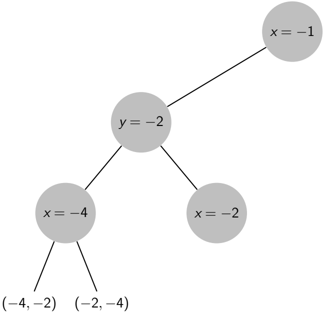
kd-drzewo – sposób podziału



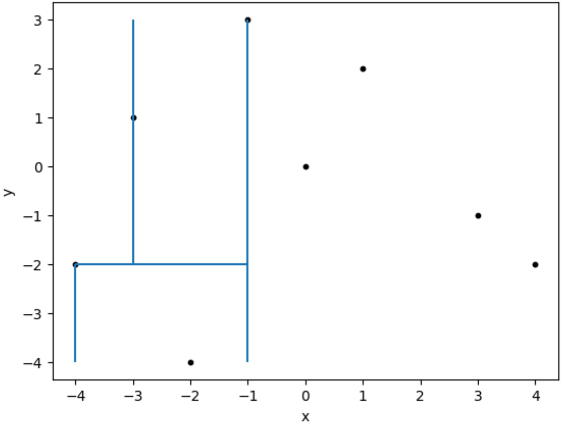


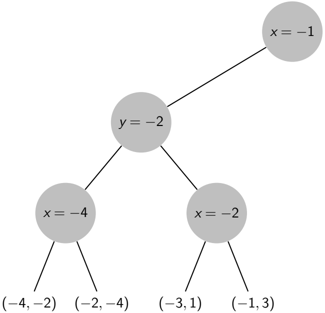
kd-drzewo – sposób podziału



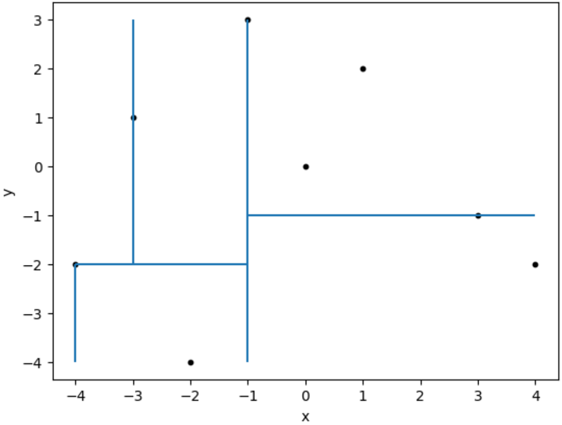


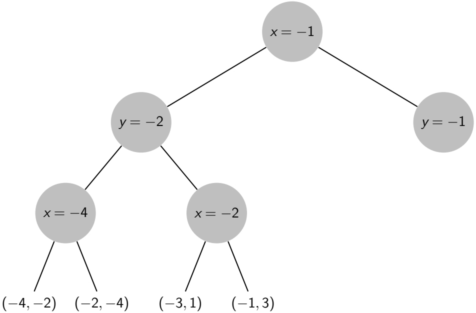
kd-drzewo – sposób podziału



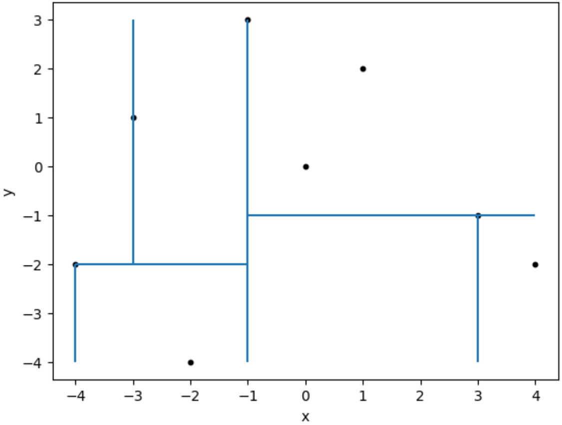


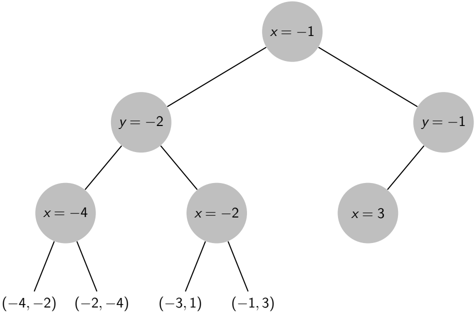
kd-drzewo – sposób podziału



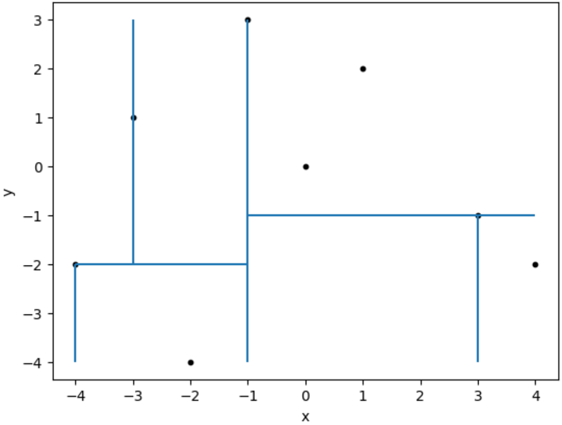


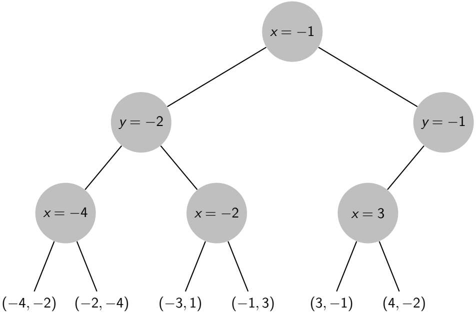
kd-drzewo – sposób podziału



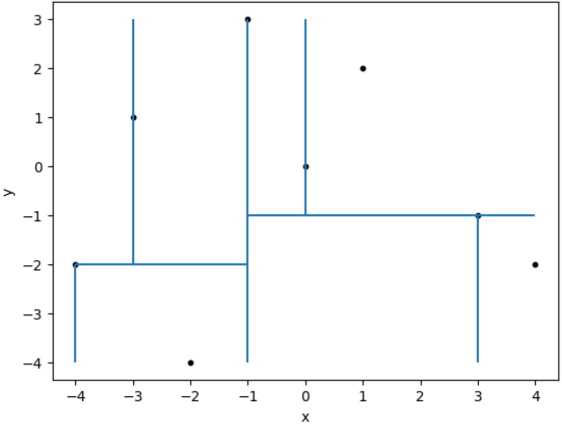


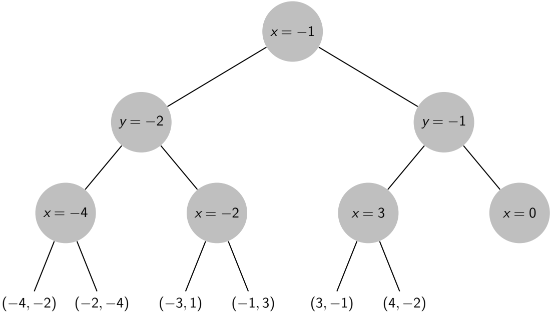
kd-drzewo – sposób podziału



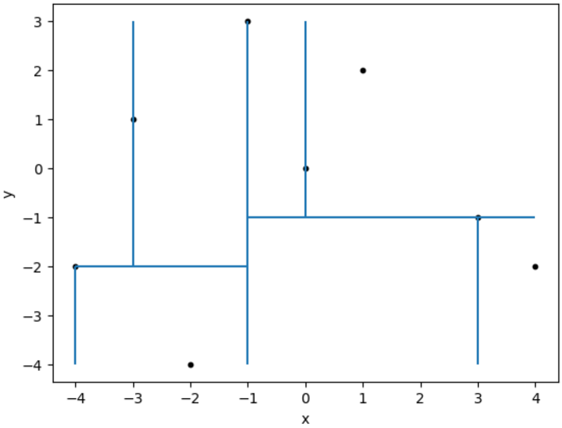


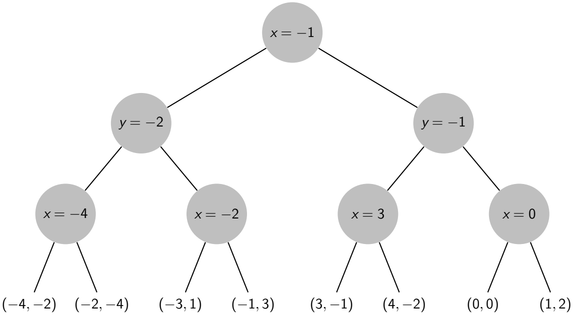
kd-drzewo – sposób podziału





kd-drzewo – sposób podziału





kd-drzewo – odpowiadanie na zapytania

Znajdowanie punktów, które należą do zadanego obszaru działa następująco:

kd-drzewo – odpowiadanie na zapytania

Znajdowanie punktów, które należą do zadanego obszaru działa następująco:

1. każdy wierzchołek w drzewie (poza liśćmi) odpowiada za konkretny obszar w przestrzeni,

kd-drzewo – odpowiadanie na zapytania

Znajdowanie punktów, które należą do zadanego obszaru działa następująco:

1. każdy wierzchołek w drzewie (poza liśćmi) odpowiada za konkretny obszar w przestrzeni,
2. przeszukuje się drzewo od korzenia – jeśli obszar z wierzchołka w pełni znajduje się w zadanym obszarze, to uwzględniamy wszystkie jego liście w odpowiedzi,

kd-drzewo – odpowiadanie na zapytania

Znajdowanie punktów, które należą do zadanego obszaru działa następująco:

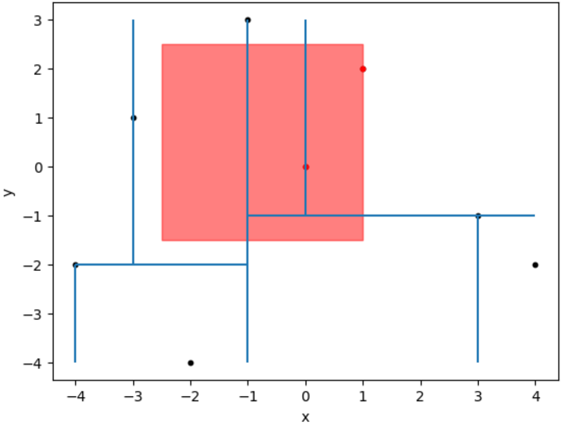
1. każdy wierzchołek w drzewie (poza liśćmi) odpowiada za konkretny obszar w przestrzeni,
2. przeszukuje się drzewo od korzenia – jeśli obszar z wierzchołka w pełni znajduje się w zadanym obszarze, to uwzględniamy wszystkie jego liście w odpowiedzi,
3. jeśli obszar tylko częściowo pokrywa się z zadanym obszarem, to sprawdzamy dalej jego dzieci,

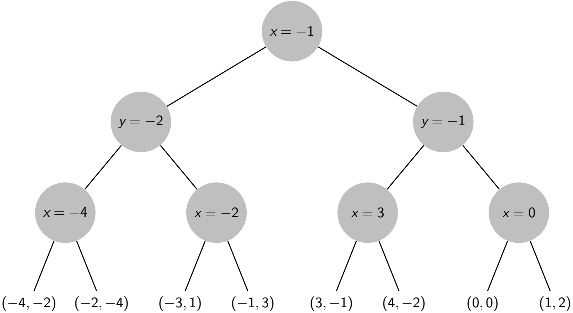
kd-drzewo – odpowiadanie na zapytania

Znajdowanie punktów, które należą do zadanego obszaru działa następująco:

1. każdy wierzchołek w drzewie (poza liśćmi) odpowiada za konkretny obszar w przestrzeni,
2. przeszukuje się drzewo od korzenia – jeśli obszar z wierzchołka w pełni znajduje się w zadanym obszarze, to uwzględniamy wszystkie jego liście w odpowiedzi,
3. jeśli obszar tylko częściowo pokrywa się z zadanym obszarem, to sprawdzamy dalej jego dzieci,
4. jesli obszar w ogóle nie pokrywa się z zadanym obszarem to kończymy przeszukiwanie w tym poddrzewie.

kd-drzewo – odpowiadanie na zapytania





kd-drzewo – analiza złożoności

Budując drzewo, trzeba w każdym kroku liczyć medianę, co odbywa się w czasie $\mathcal{O}(n)$. Drzewo jest zrównoważone, więc z każdym podziałem zbiór zmniejsza się dwukrotnie, więc ostateczna złożoność wynosi $\mathcal{O}(n \log n)$.

kd-drzewo – analiza złożoności

Budując drzewo, trzeba w każdym kroku liczyć medianę, co odbywa się w czasie $\mathcal{O}(n)$. Drzewo jest zrównoważone, więc z każdym podziałem zbiór zmniejsza się dwukrotnie, więc ostateczna złożoność wynosi $\mathcal{O}(n \log n)$.

Lemat (o kd-drzewie)

Dla zrównoważonego kd-drzewa o zamiennym podziale (czyli takim, gdzie punkt dzielący jest wybierany według różnych wymiarów dla różnych poziomów drzewa), dowolna pionowa lub pozioma prosta przecina $\mathcal{O}(\sqrt{n})$ komórek.

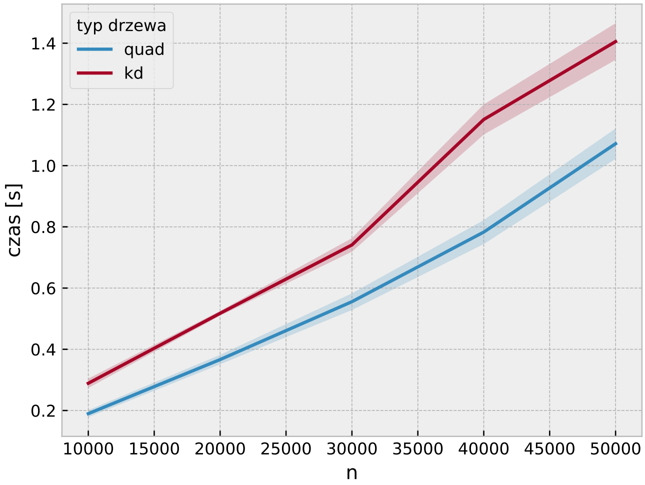
Złożoność

1. konstrukcji drzewa: $\mathcal{O}(n \log n)$,
2. zapytania: $\mathcal{O}(\sqrt{n} + k)$,
3. pamięciowa: $\mathcal{O}(n)$.

Porównanie

	quadtree	kd-tree
konstrukcja	$\mathcal{O}(n \log \beta)$	$\mathcal{O}(n \log n)$
zapytania	$\mathcal{O}(k \log \beta)$	$\mathcal{O}(\sqrt{n} + k)$
pamięć	$\mathcal{O}(n \log \beta)$	$\mathcal{O}(n)$

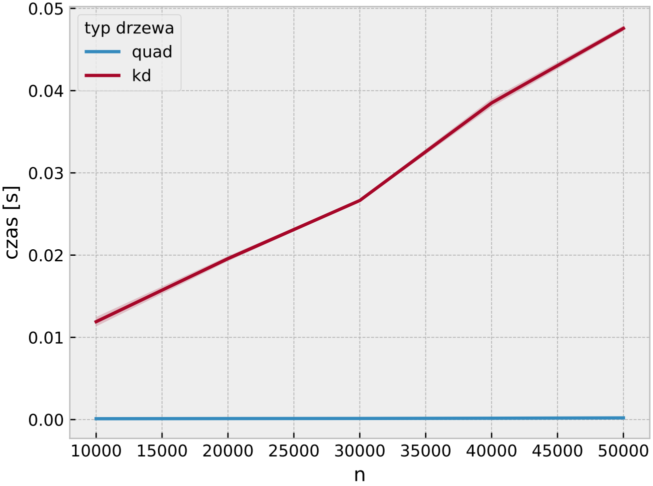
Porównanie


typ drzewa
quad
kd
1.4
1.2
1.0
0.8
0.6
0.4
0.2
czas [s]
10000
15000
20000
25000
30000
35000
40000
45000
50000
n

Porównanie

	quadtree	kd-tree
konstrukcja	$\mathcal{O}(n \log \beta)$	$\mathcal{O}(n \log n)$
zapytania	$\mathcal{O}(k \log \beta)$	$\mathcal{O}(\sqrt{n} + k)$
pamięć	$\mathcal{O}(n \log \beta)$	$\mathcal{O}(n)$

Porównanie

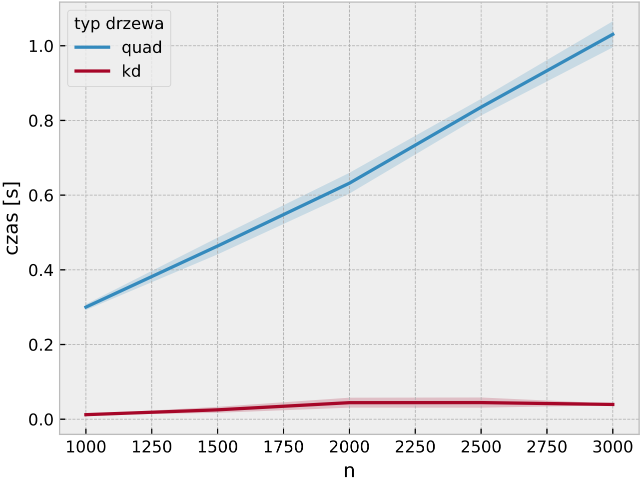


typ drzewa
quad
kd
0.05
0.04
0.03
0.02
0.01
0.00
czas [s]
10000
15000
20000
25000
30000
35000
40000
45000
50000
n

Porównanie

	quadtree	kd-tree
konstrukcja	$\mathcal{O}(n \log \beta)$	$\mathcal{O}(n \log n)$
zapytania	$\mathcal{O}(k \log \beta)$	$\mathcal{O}(\sqrt{n} + k)$
pamięć	$\mathcal{O}(n \log \beta)$	$\mathcal{O}(n)$

Porównanie


typ drzewa
quad
kd
1.0
0.8
0.6
0.4
0.2
0.0
czas [s]
1000
1250
1500
1750
2000
2250
2500
2750
3000
n

Bibliografia

[1] **Algorytmy geometryczne – wykład**, dr inż. Barbara Głut.

[2] **Quadtrees, Geometric Approximation Algorithms**, prof. Kevin Buchin, TU Dortmund. `https://ls11-www.cs.tu-dortmund.de/_media/buchin/teaching/akda_ws21/quadtrees.pdf`

[3] **Wprowadzenie do algorytmów**, Thomas H. Cormen et al.