

# Wyszukiwanie geometryczne K-DTree i QuadTree

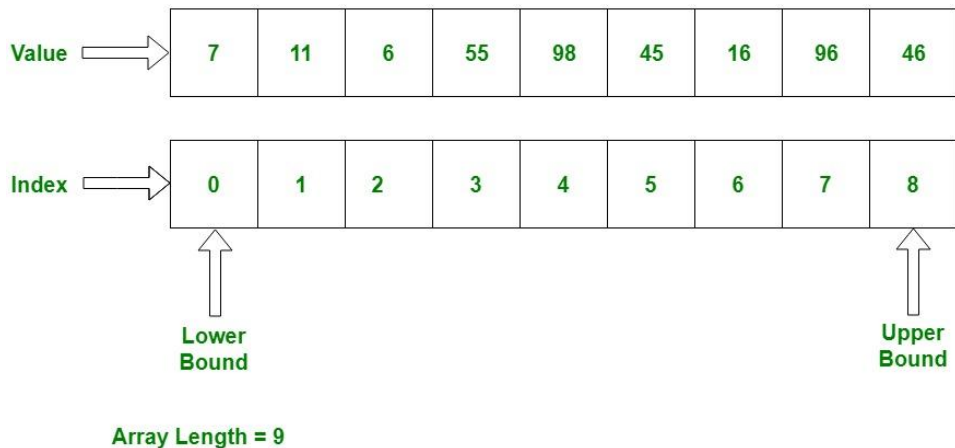
Wykonali: Cyprian Neugebauer, Bartłomiej Wiśniewski

# Opis projektu

Celem projektu było zaimplementowanie odpowiednich struktur danych - quadtree oraz kd-drzew, które pozwalają na szybkie przeszukiwanie obszarów ortogonalnych. Na wejściu otrzymujemy zbiór punktów  $P$  na płaszczyźnie i należy dla zadanych  $x_1, x_2, y_1, y_2$  znaleźć punkty  $q$  ze zbioru  $P$  takie, że  $x_1 \leq q_x \leq x_2, y_1 \leq q_y \leq y_2$ .

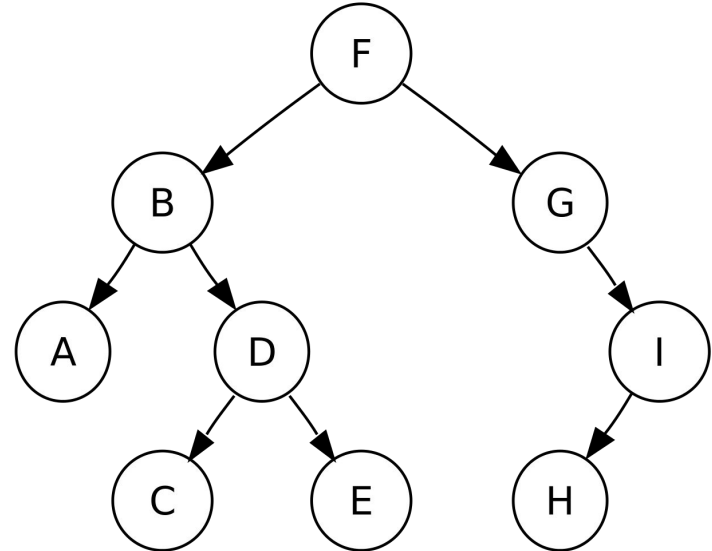
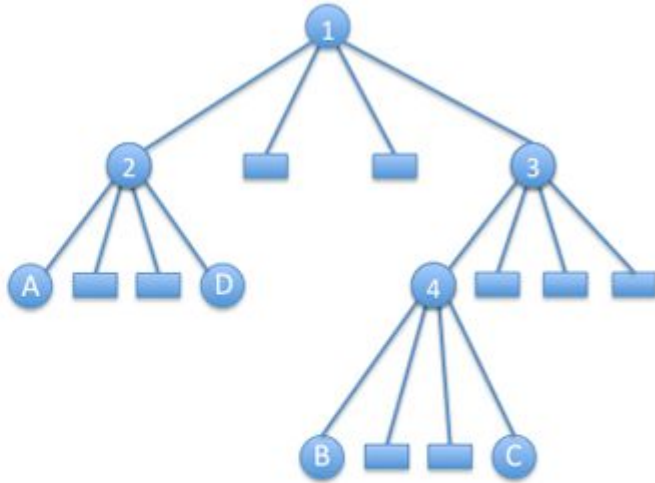
# Najprostsze rozwiązanie problemu

Najłatwiejszym i początkowo narzucającym się podejściem do rozwiązania tego problemu jest iteracja po punktach i sprawdzenie przynależności do danego obszaru dla każdego punktu indywidualnie.



# Lepsze rozwiązanie problemu

Lepszym podejściem jest użycie struktur drzewiastych takich jak QuadTree i KD-Tree, które pozwalają przeszukać zbiór punktów w krótszym czasie.

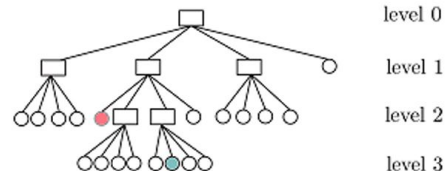
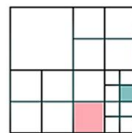
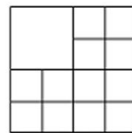
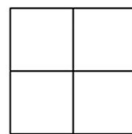


# QuadTree

**Drzewo czwórkowe** (ang. *quadtree*) – struktura danych będąca drzewem, używana do podziału dwuwymiarowej przestrzeni na mniejsze części, dzieląc ją na cztery równe ćwiartki, a następnie każdą z tych ćwiartek na cztery kolejne itd.

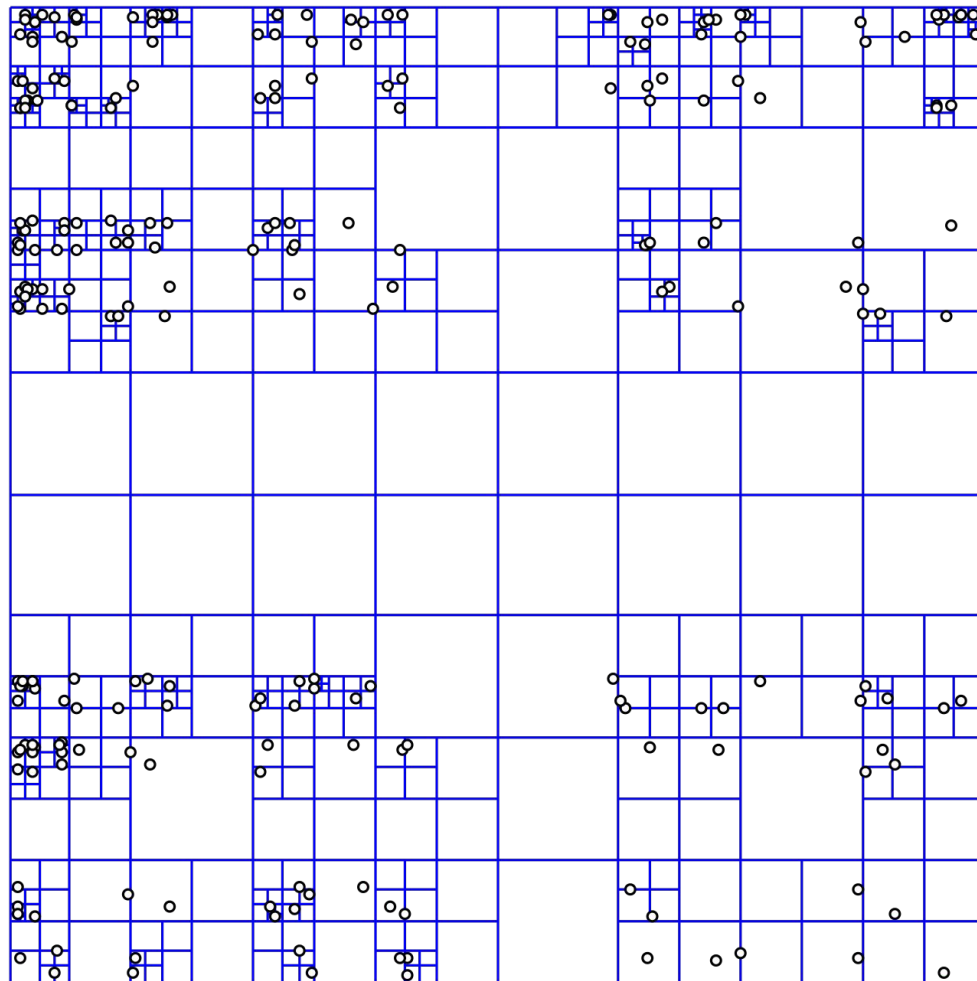
Najczęściej wykorzystuje się ją do:

- wykrywania kolizji w dwóch wymiarach
- przetwarzania obrazu
- znajdowania punktów z zadanego przedziału
- generowania siatki



□ internal node

○ leaf node



Podział przestrzeni przez QuadTree

# QuadTree - budowanie drzewa

Drzewo czwórkowe budowane jest poprzez wstawianie punktów w kolejności w jakiej są one podane w liście. Każdy węzeł drzewa, odpowiadający pewnej płaszczyźnie może przechowywać 4 punkty. Gdy w danym węźle znajdują się już 4 punkty płaszczyzna odpowiadająca węzłowi zostaje podzielona na 4 równe części. Każda z nowych części (płaszczyzn) jest nowym węzłem w drzewie, którego rodzicem jest węzeł, w którym został dokonany podział. Punkty przypisane do węzła drzewa czwórkowego zawierają się w płaszczyźnie reprezentowanej przez dany węzeł.

# QuadTree - przeszukiwanie przestrzeni 2D

Aby znaleźć wszystkie punkty w zadanym obszarze A należy schodzić rekurencyjnie w głąb drzewa.

1. Jeżeli obszar reprezentowany przez aktualnie rozpatrywany węzeł nie przecina się z A, to rekurencyjnie cofamy się do rodzica.
2. Jeżeli obszar węzła przecina się z A, to punkty zawarte w aktualnie rozpatrywanym węźle, które należą do A dodajemy do listy wynikowej.
3. Następnie jeżeli aktualny węzeł został podzielony podczas budowania drzewa, to powtarzamy powyższe kroki dla każdej ćwiartki będącej dzieckiem aktualnie rozpatrywanego węzła.

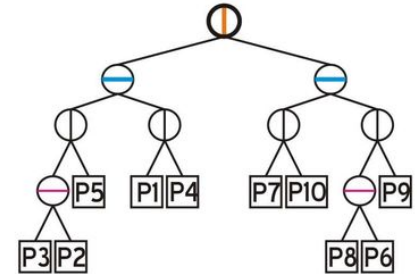
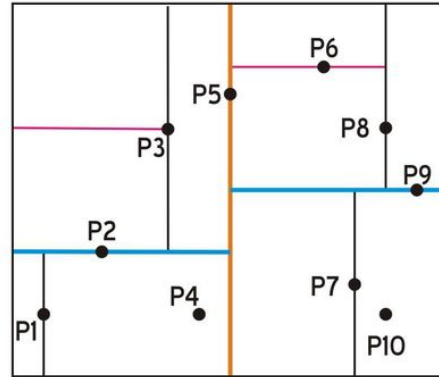


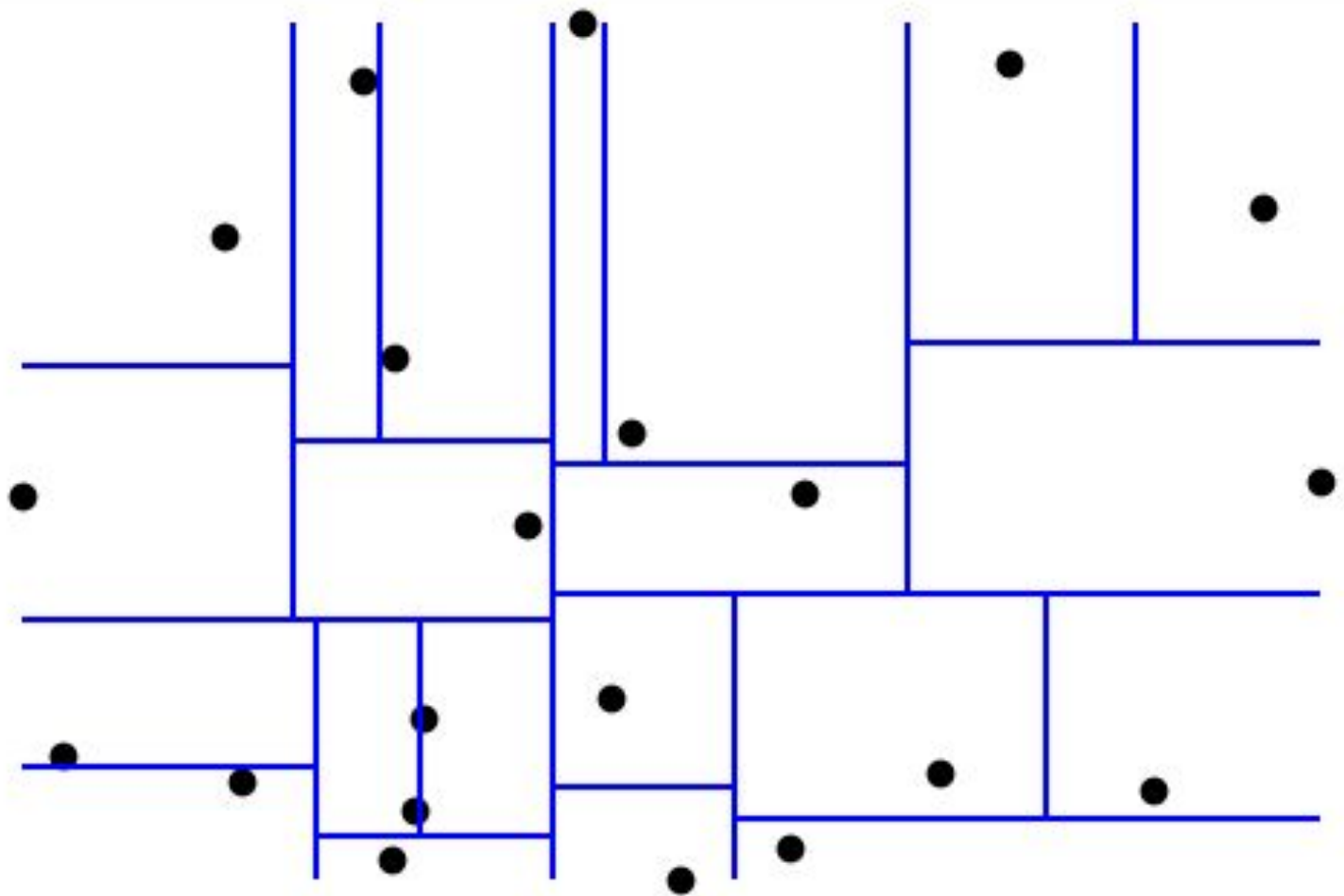
# KD-Tree

**Drzewo k-wymiarowe** (ang. kd-tree) – struktura danych, będąca wariantem drzew binarnych, używana do dzielenia przestrzeni. W każdym węźle następuje podział przestrzeni na dwie podprzestrzenie według jednej ze współrzędnych.

Najczęściej wykorzystuje się ją do:

- wyszukiwania najbliższych sąsiadów
- znajdowanie punktów w prostokątnych obszarach
- tworzenie chmury punktów





Podział przestrzeni przez KD-Tree

# KD-tree - budowanie drzewa

Algorytm budujący drzewo polega na podziale zbioru punktów na dwa podzbiory względem hiperpłaszczyzny (w 2D prostej). Zwykle położenie takiej hiperpłaszczyzny ustala się za pomocą mediany  $i$ -tej współrzędnej punktów.

W naszej implementacji algorytm do wyznaczenia pozycji hiperpłaszczyzny podziału obszaru reprezentowanego przez węzeł " $v$ " nie używa mediany współrzędnych punktów należących do tego obszaru, a średniej arytmetycznej  $i$ -tych współrzędnych na pozycjach  $(n \div 2 - 1)$  i  $(n \div 2)$  w posortowanej po  $i$ -tej współrzędnej tablicy punktów gdzie  $n$  to liczba punktów należąca do obecnego podobzaru. Dla każdego węzła " $i$ " obliczane jest jako " $i = \text{depth} \bmod k$ " gdzie " $\text{depth}$ " to głębokość na jakiej znajduje się węzeł " $v$ ", a " $k$ " to liczba wymiarów.

Tak więc algorytm rekurencyjnie dzieli zbiór punktów na podzbiory według  $i$ -tej współrzędnej punktów, a gdy zbiór zawiera tylko jeden punkt tworzony jest liść drzewa który zawiera ten punkt.

# KD-tree - Przeszukiwanie przestrzeni 2D

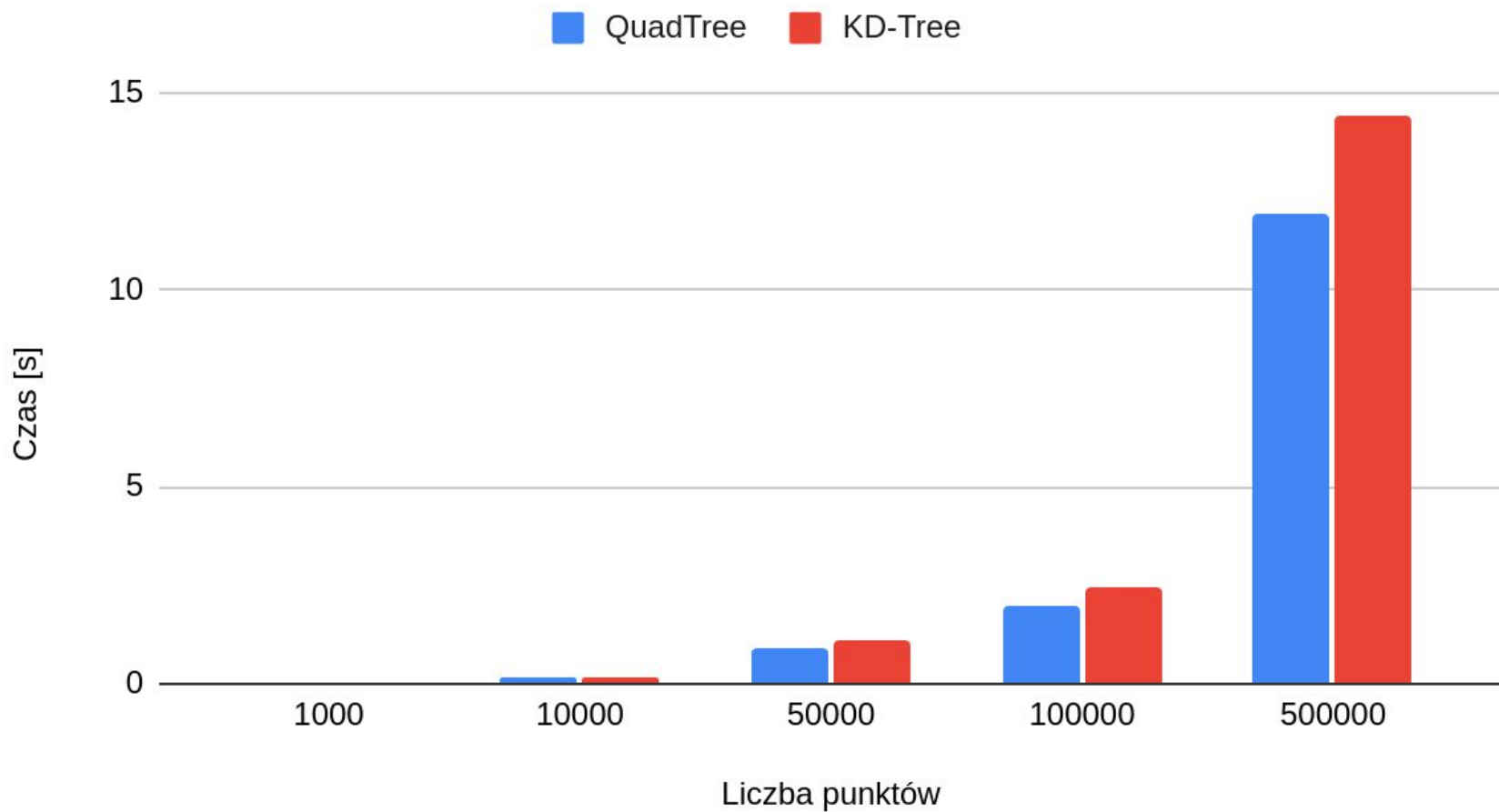
Algorytm przeszukiwania przestrzeni służy do znalezienia wszystkich punktów znajdujących się w pewnym zadanym obszarze “A”.

Algorytm przeszukuje drzewo zakorzenione w “w” w następujący sposób: jeżeli obszar reprezentowany przez węzeł “v” (będący dzieckiem “w”) w całości zawiera się w obszarze “A” to zwraca wszystkie punkty znajdujące się w poddrzewie zakorzenionym w v. Jeżeli natomiast obszar reprezentowany przez “v” ma niepustą część wspólną z “A” to rekurencyjnie przeszukuje drzewo zakorzenione w “v”.

# Pomiary wydajności

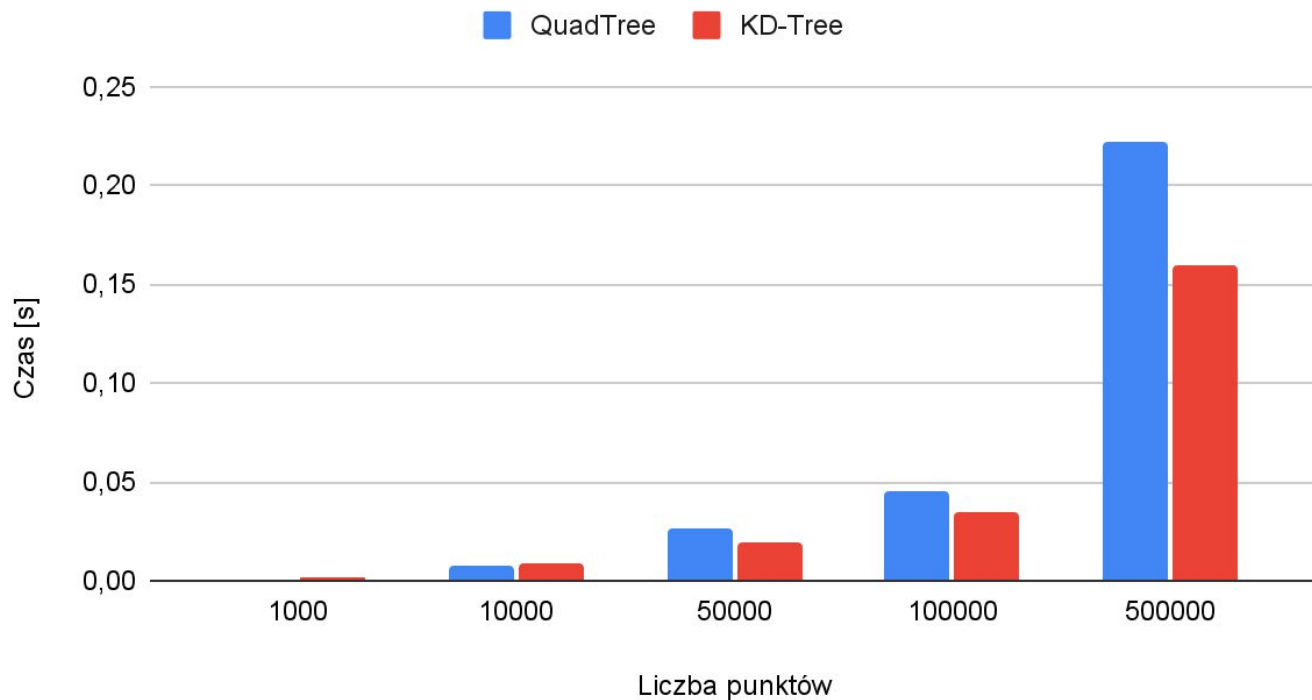
Zmierzyliśmy czas wykonania zaimplementowanych operacji dla każdego z drzew i porównaliśmy te czasy ze sobą. Pomiary wykonane były dla zbiorów punktów o różnych rozmiarach.

# Czas budowania drzewa



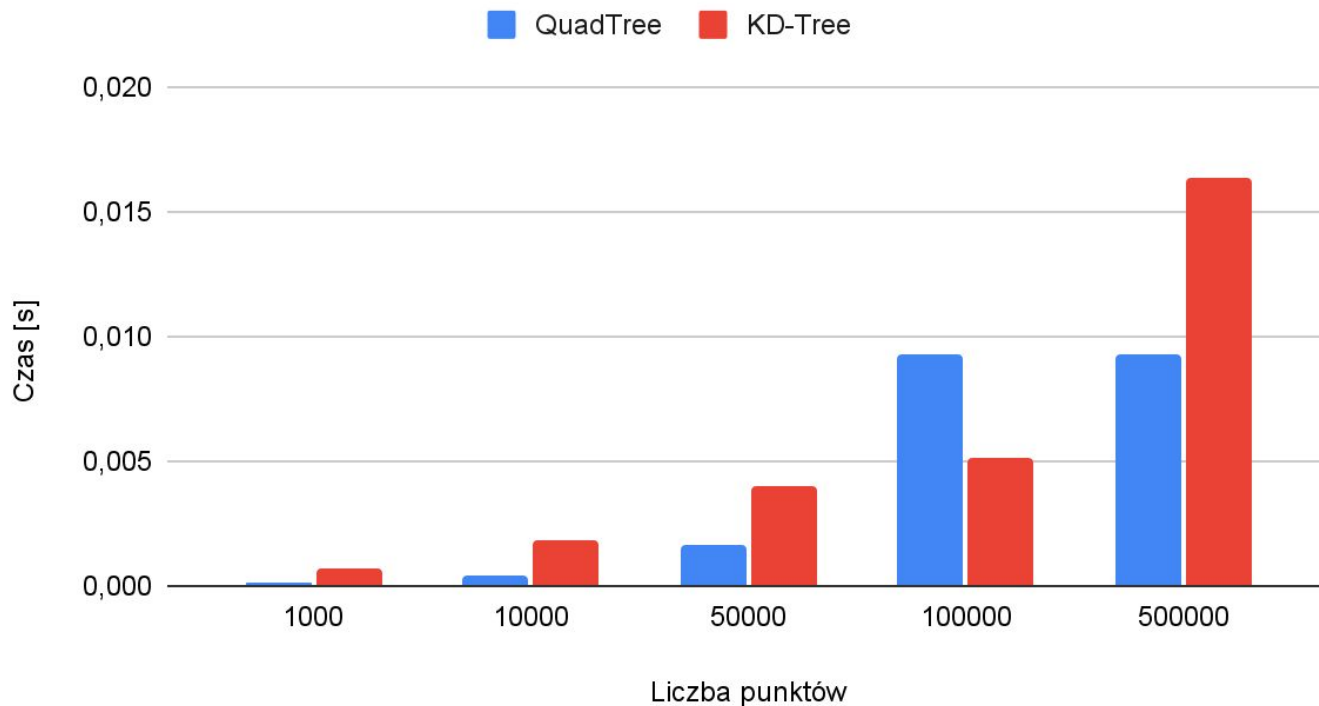
Punkty z przedziału (0 - 200, 0 - 200), zapytanie z przedziału (50, 50) - (150, 150)

### Czas przeszukiwania drzewa



Punkty z przedziału (0 - 200, 0 - 200), zapytanie z przedziału (40, 40) - (60, 60)

### Czas przeszukiwania drzewa





# Bibliografia

- <https://en.wikipedia.org/wiki/Quadtree>
- [https://en.wikipedia.org/wiki/K-d\\_tree](https://en.wikipedia.org/wiki/K-d_tree)
- Wykład - wyszukiwanie geometryczne

**Dziękujemy za uwagę :)**