LOKALIZACJA PUNKTU W PRZESTRZENI 2D - METODA TRAPEZOWA

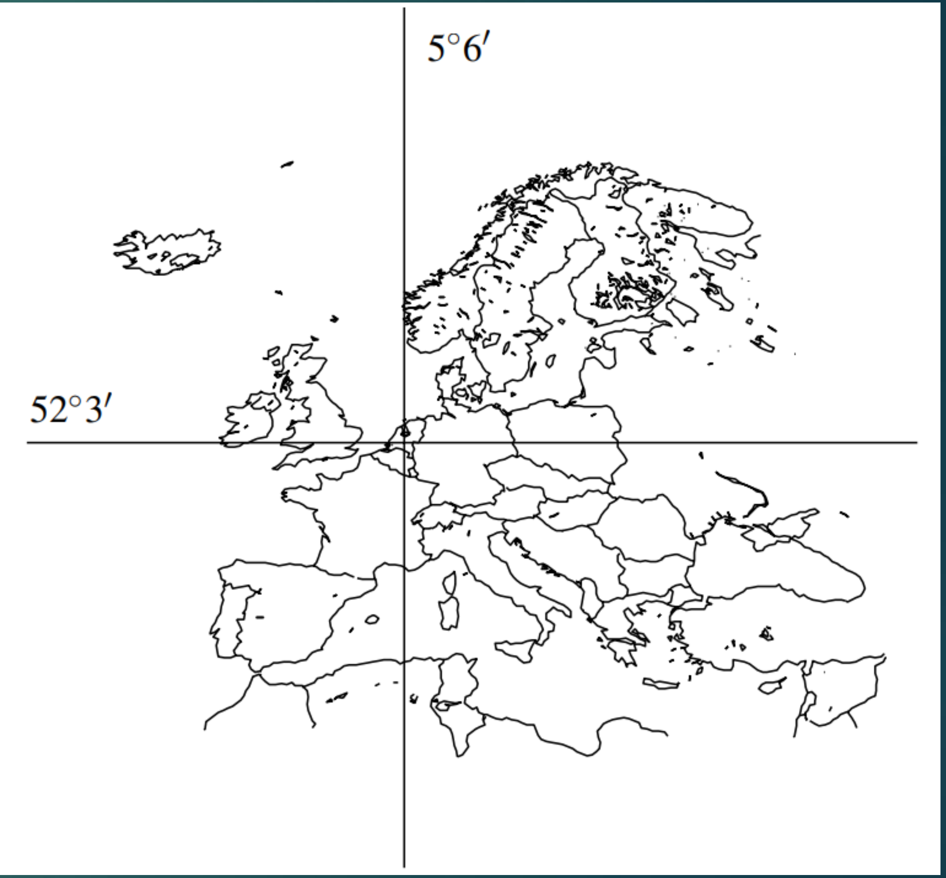
AUTORZY:

ŁUKASZ DRAGON, RAFAŁ BABSKI

Przedstawienie problemu

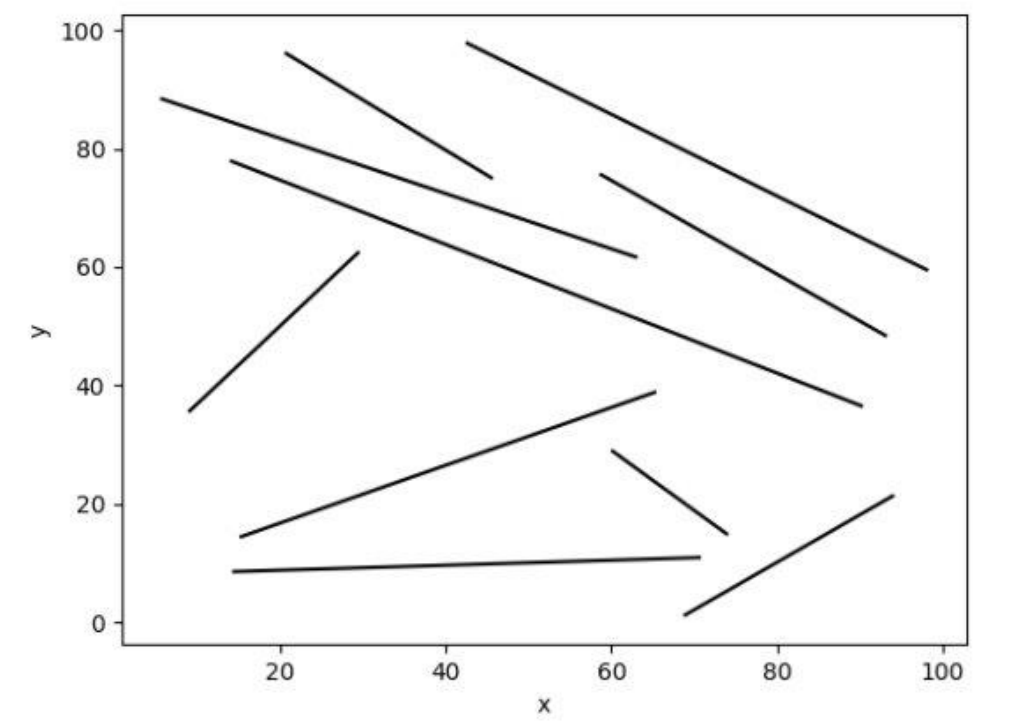
Mając zadany podział przestrzeni dwuwymiarowej **S,** zawierający n krawędzi, chcemy odnaleźć taki obszar (ścianę) **f** , w którym znajduje się zadany punkt **q**.

Naszym celem jest stworzenie takiej struktury, która pozwoli odpowiadać na zapytania o położenie punktu w jak najkrótszym czasie i będzie również efektywna pamięciowo.



Punkt o zadanych współrzędnych geograficznych zaznaczony na mapie świata

Położenie ogólne odcinków



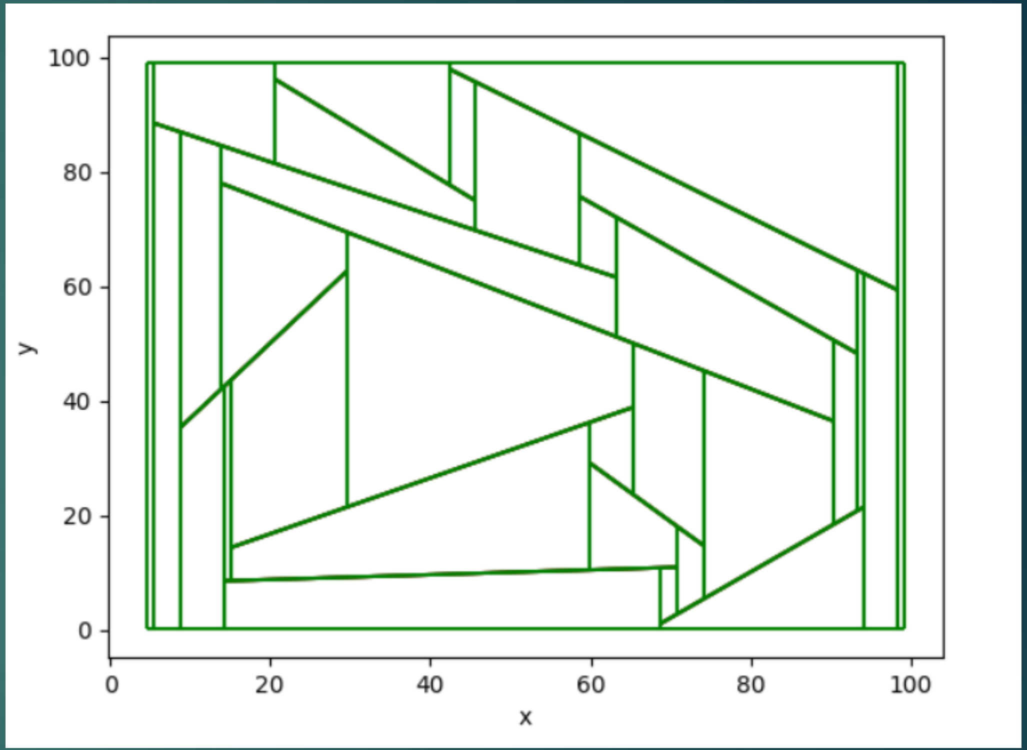
Przykładowy zbiór odcinków w położeniu ogólnym

Aby rozwiązać zadany problem metodą trapezową, będziemy reprezentować podział płaszczyzny przez zbiór odcinków w położeniu ogólnym, tzn.:

- Żaden z odcinków nie jest pionowy
- Odcinki nie przecinają się (mają co najwyżej wspólny wierzchołek)
- Z wyjątkiem powyższej sytuacji, żadne wierzchołki dwóch różnych odcinków nie mają takiej samej współrzędnej x-owej

Mapa trapezowa

Mapa trapezowa **T(S)** jest podziałem S na trapezy oraz trójkąty (które traktujemy jak trapezy o zdegenerowanym boku). Otrzymujemy ją poprzez ograniczenie całego obszaru odpowiednio dużym prostokątem **R**, a następnie poprowadzenie pionowych rozszerzeń z każdego wierzchołka, czyli pionowych odcinków łączących wierzchołki i pierwsze napotkane odcinki lub brzeg prostokąta **R**.

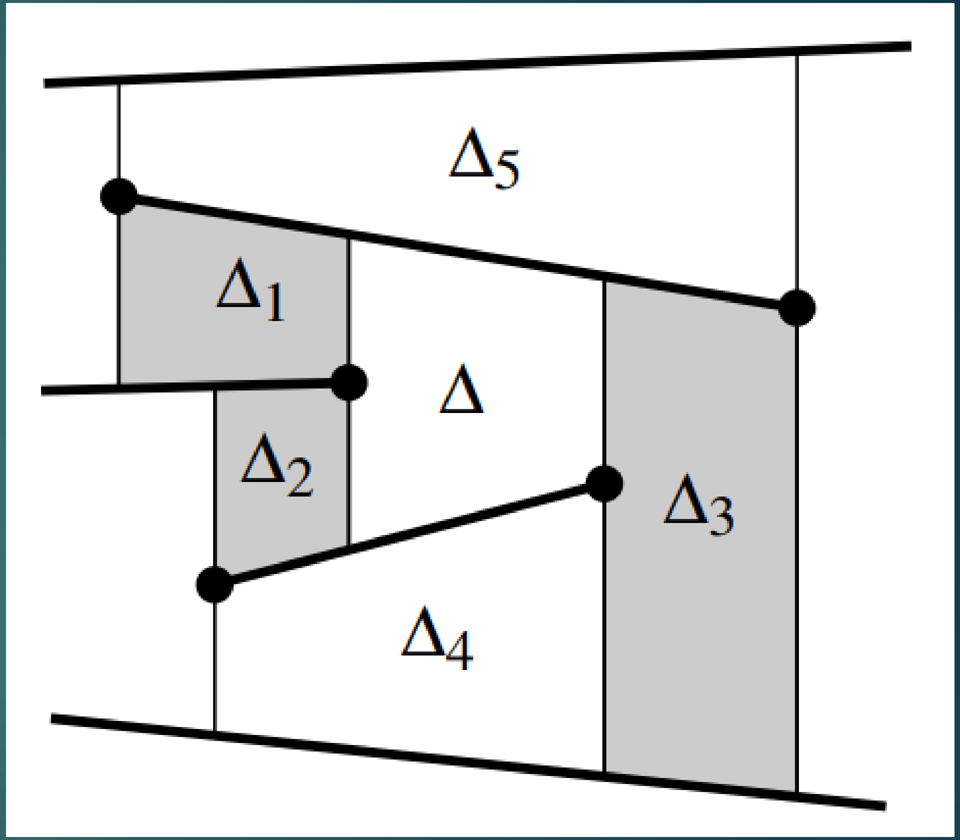


Mapa trapezowa wygenerowana dla zbioru odcinków z poprzedniego slajdu

Struktura mapy trapezowej

Mapa trapezowa **T(S)** reprezentowana jest przez graf trapezów. Mówimy, że dwa trapezy sąsiadują ze sobą jeżeli ich pionowe boki mają część wspólną. Dzięki założeniu, że odcinki reprezentujące podział płaszczyzny są w położeniu ogólnym, wiemy, że każdy trapez może mieć co najwyżej czterech sąsiadów, do których to posiada wskaźnik.

Każdy trapez ma również wskaźnik do odpowiadającego mu wierzchołka w strukturze przeszukiwań opisanej w dalszej części prezentacji.



Trapez Δ sąsiaduje z trapezami Δ_1, Δ_2, Δ_3, ale nie sąsiaduje z Δ_4 oraz Δ_5

Struktura przeszukiwań

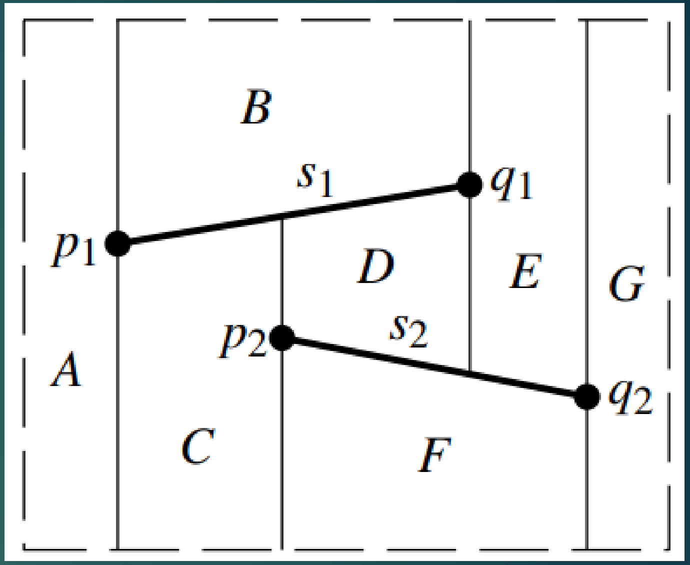
Struktura przeszukiwań to acykliczny graf skierowany (DAG), budową przypominający drzewo binarne, w którym mamy trzy typy węzłów:

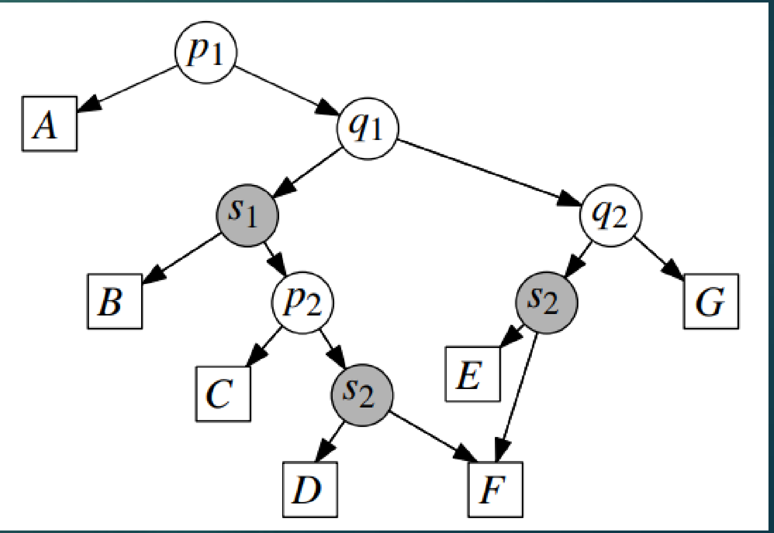
- **Węzły** ze wskaźnikiem do **trapezu**, będące równocześnie "liścmi" grafu
- **Węzły wierzchołków** ze wskaźnikiem do jednego końca odcinka z **S** oraz wskaźniki do dwóch dzieci (lewego oraz prawego) w DAGu
- **Węzły odcinków** ze wskaźnikiem do danego odcinka z **S** oraz, podobnie jak poprzednio, dwa wskaźniki do dzieci w strukturze przeszukiwań

Działanie struktury przeszukiwań

Zapytanie o położenie danego punktu zaczyna działać w korzeniu grafu i następnie:

- W przypadku, jeżeli jesteśmy w węźle trapezu to znaleźliśmy szukany obszar
- Jeżeli znaleźliśmy się w węźle wierzchołka, to sprawdzamy po której jego stronie znajduje się szukany punkt i idziemy do odpowiedniego dziecka (jeżeli po prawej to do prawego, po lewej – do lewego). Jeżeli punkt jest równy wierzchołkowi, to algorytm kończy działanie i zwraca o tym informację
- Jeżeli doszliśmy do węzła odcinka, to sprawdzamy czy szukany punkt znajduje się nad, czy pod odcinkiem i wędrujemy odpowiednio do lewego lub prawego dziecka, albo zwracamy odpowiednią informację, gdy punkt leży na odcinku





Algorytm konstruujący mapę trapezową

Do stworzenia mapy trapezowej **T(S)** wykorzystywany jest randomizowany algorytm przyrostowy. Działanie algorytmu możemy opisać w następujący sposób:

1. Tworzymy prostokąt zawierający w sobie wszystkie odcinki należące do zbioru **S** (inicjalizuje on naszą mapę trapezową)
2. Generujemy losową permutację odcinków ze zbioru S
3. Rozpatrujemy kolejno odcinki w kolejności takiej jak w wylosowanej permutacji:
   - Znajdujemy wszystkie trapezy z aktualnej mapy, które przecinają się z aktualnie rozpatrywanym odcinkiem (nie liczymy przecięć z wierzchołkami). Robimy to w następujący sposób: wykonujemy zapytanie w aktualnej strukturze przeszukiwań dla lewego wierzchołka naszego odcinka. Następnie do momentu napotkania prawego wierzchołka przechodzimy do kolejnych sąsiadów, zaczynając od znalezionego trapezu i dodajemy ich do listy przeciętych trapezów jeżeli zawierają w sobie rozważany odcinek

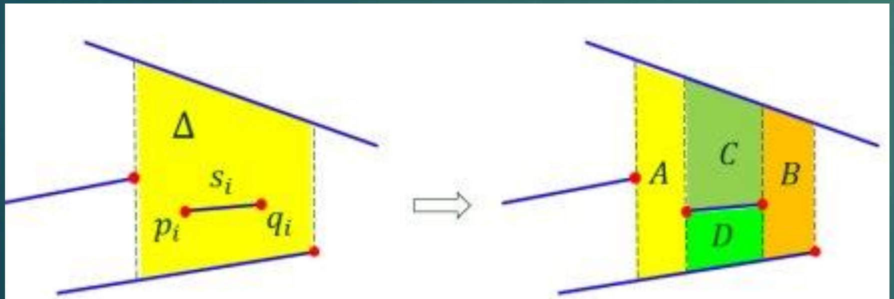
Algorytm konstruujący mapę trapezową c.d.

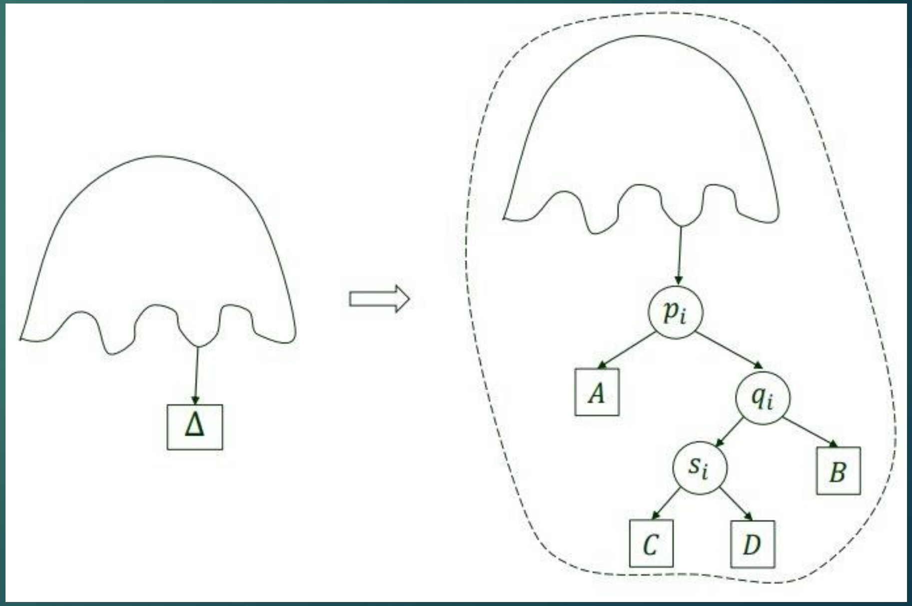
3. ...
   - Usuwamy znalezione trapezy z mapy i zastępujemy je trapezami powstałymi po dodaniu aktualnego odcinka oraz aktualizujemy informacje o tych trapezach (wskaźniki do ich sąsiadów oraz pola definiujące ich położenie w przestrzeni)
   - Usuwamy ze struktury przeszukiwań węzły odpowiadające usuniętym trapezom i dodajemy nowopowstałe trapezy odpowiednio łącząc je brakującymi węzłami wierzchołków oraz węzłami odcinków

Aktualizacja struktury przeszukiwań

Usuwając liść reprezentujący usunięty z mapy trapez musimy w odpowiedni sposób zaktualizować graf przeszukiwań:

1. Gdy trapez zawierał w sobie cały odcinek, tworzymy cztery nowe trapezy: dwa leżące po prawej (B) i po lewej (A) stronie odcinka, oraz dwa leżące nad (C) i pod (D) nim, które są nim rozdzielone





Aktualizacja struktury przeszukiwań c.d.

W pozostałych przypadkach postępujemy w podobny sposób, tak, aby zachować właściwości struktury przeszukiwań:

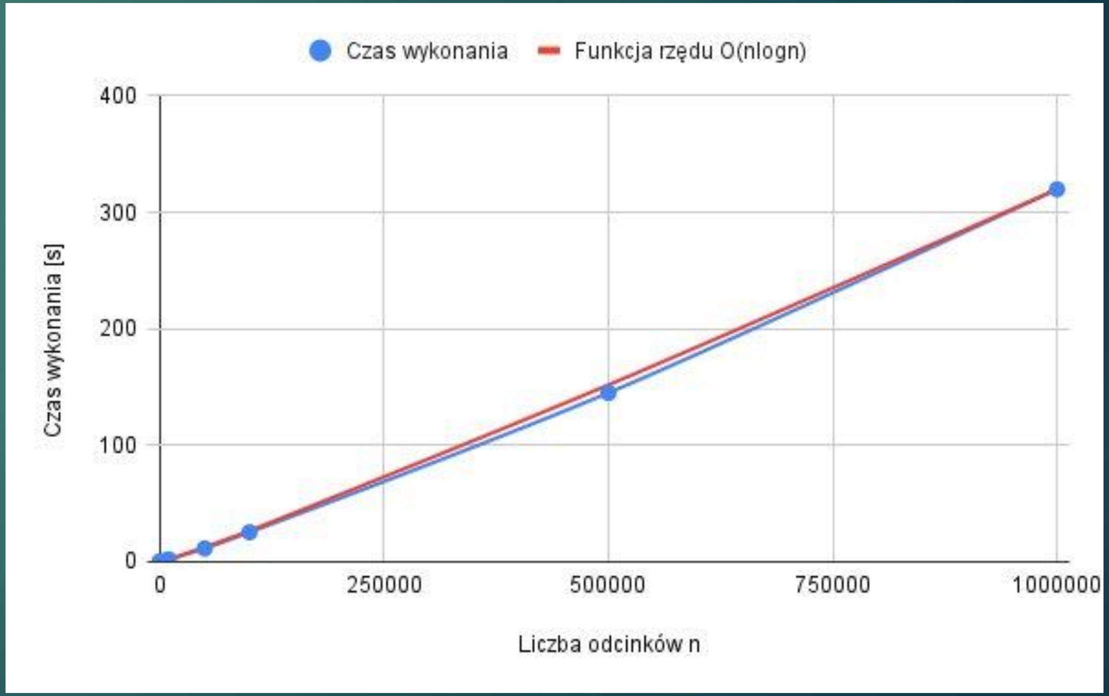
- Gdy trapez był przecięty odcinkiem, ale nie zawierał w sobie żadnego z jego wierzchołków, tworzymy dwa nowe trapezy leżące nad i pod odcinkiem i których suma jest równa usuniętemu trapezowi. Wstawiamy wtedy do struktury przeszukiwań węzeł odcinka którego lewym dzieckiem jest nowy trapez leżący wyżej, a prawym – ten leżący poniżej odcinka
- Gdy usunięty trapez zawierał dokładnie jeden koniec odcinka musimy wstawić dokładnie 3 trapezy Δ_1, Δ_2, Δ_3, takie, że Δ_1 znajduje się po jednej stronie końca odcinka, a Δ_2 i Δ_3 po jego drugiej stronie, Δ_2 jest górnym sąsiadem Δ_1, a Δ_2 jego dolnym sąsiadem. Wstawiamy wtedy węzeł wierzchołka, którego jednym dzieckiem jest węzeł trapezu Δ_1, a drugim węzeł odcinka, którego dziećmi są węzły trapezów Δ_2, Δ_3

Złożoność algorytmu

Ze względu na to, że algorytm jest randomizowany, należy wykonać analizę probabilistyczną złożoności obliczeniowej oraz pamięciowej. Otrzymujemy wtedy następujące złożoności oczekiwane w zależności od liczby odcinków $\mathbf{n}$:

Czas tworzenia mapy – $O(\mathrm{nlogn})$

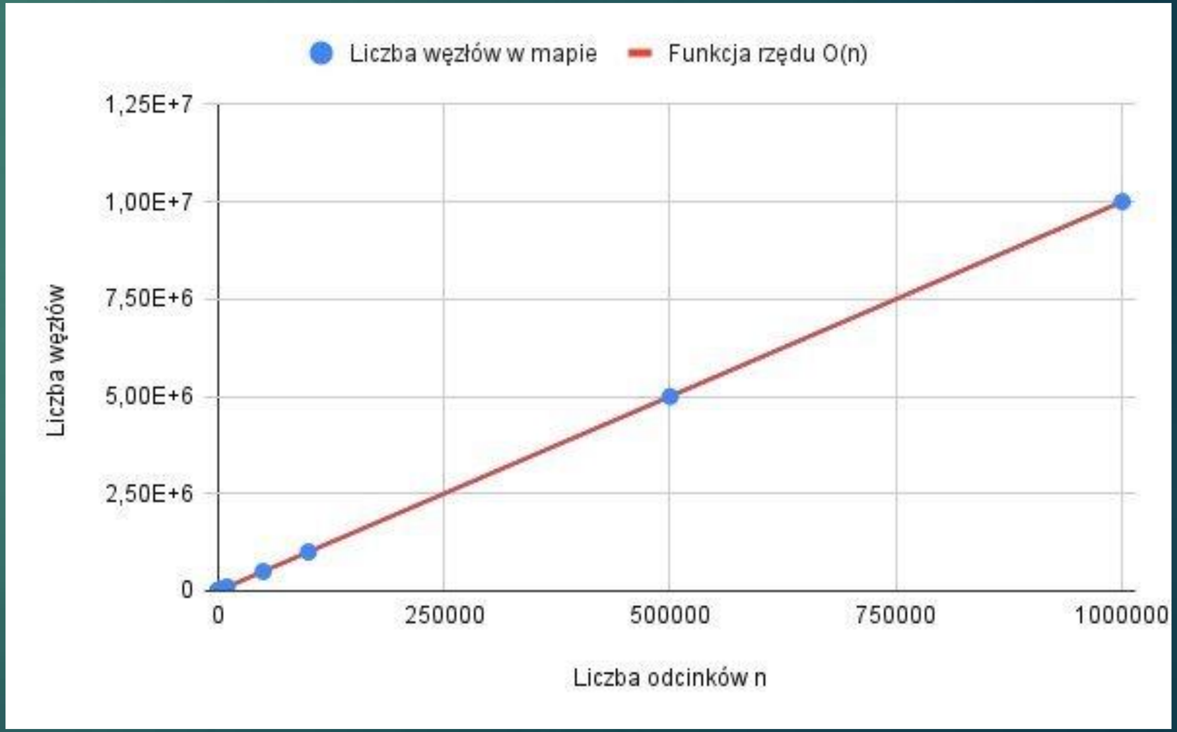
Na wykresie znajduje się porównanie czasu działania zaimplementowanego algorytmu tworzącego mapę trapezową z funkcją rzędu $O(\mathrm{nlogn})$



Złożoność algorytmu c.d.

Złożoność pamięciowa grafu przeszukiwań - O(n)

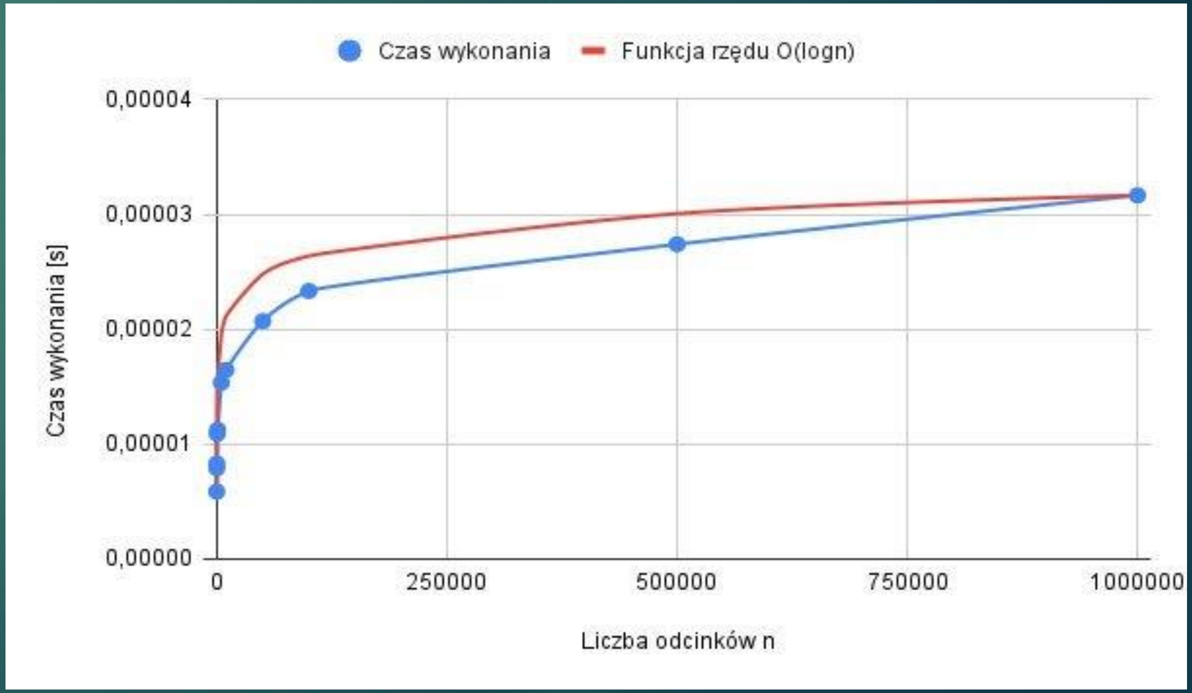
Na wykresie znajduje się porównanie wielkości grafu przeszukiwań dla różnych ilości odcinków z funkcją rzędu $O(n\log n)$



Złożoność algorytmu c.d.

Czas lokalizacji punktu w używając struktury przeszukiwań – $O(\log n)$

Na wykresie znajduje się porównanie czasu lokalizacji punktu dla różnych ilości odcinków, z funkcją rzędu $O(n\log n)$



Bibliografia

- M. van de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. (1997) *Computational Geometry*, Springer
- Lecture Notes for Introduction to Computational Geometry (Com S 418/518) Yan-Bin Jia, Iowa State University

https://faculty.sites.iastate.edu/jia/files/inline-files/13.%20trapezoidal%20maps.pdf

Dziękujemy za uwagę