Algorytmy geometryczne
laboratorium 2 - sprawozdanie

Radosław Rolka
Informatyka WI, II rok
Gr 6, tydz. A czwartek 13:00

Spis treści

1 Wstęp

1.1 Cel ćwiczenia

Ćwiczenie wprowadzające w zagadnienia otoczki wypukłej – implementacja algorytmów Grahama oraz Jarvisa wyznaczające otoczkę wypukłą dla zadanych zbiorów, przeprowadzenie testów, wizualizacja i opracowanie wyników.

1.2 Program ćwiczenia

- Przygotuj program generujący następujące zbiory punktów na płaszczyźnie (współrzędne rzeczywiste typu double):
  a) Losowo wygenerowane punkty o współrzędnych z zadanego przedziału
  b) Losowo wygenerowane punkty leżące na okręgu o zadanym środku i promieniu
  c) Losowo wygenerowane punkty leżące na bokach prostokąta o zadanych wierzchołkach
  d) Zawierający wierzchołki zadanego kwadratu oraz punkty wygenerowane losowo na dwóch bokach kwadratu leżących na osiach i na przekątnych kwadratu
- Zaimplementuj algorytmy Grahama oraz Jarvisa wyznaczające otoczkę wypukłą dla zadanego zbioru punktów.
- Uruchom aplikację graficzną tak, aby można było zilustrować graficznie poszczególne kroki realizacji algorytmu.

2 Wykorzystane narzędzia

2.1 Środowisko

Ćwiczenie zostało wykonane w Jupyter Notebook wykorzystując język programowania Python oraz dodatkowe biblioteki, które zostały zawarte w projekcie dostarczonym na zajęciach.

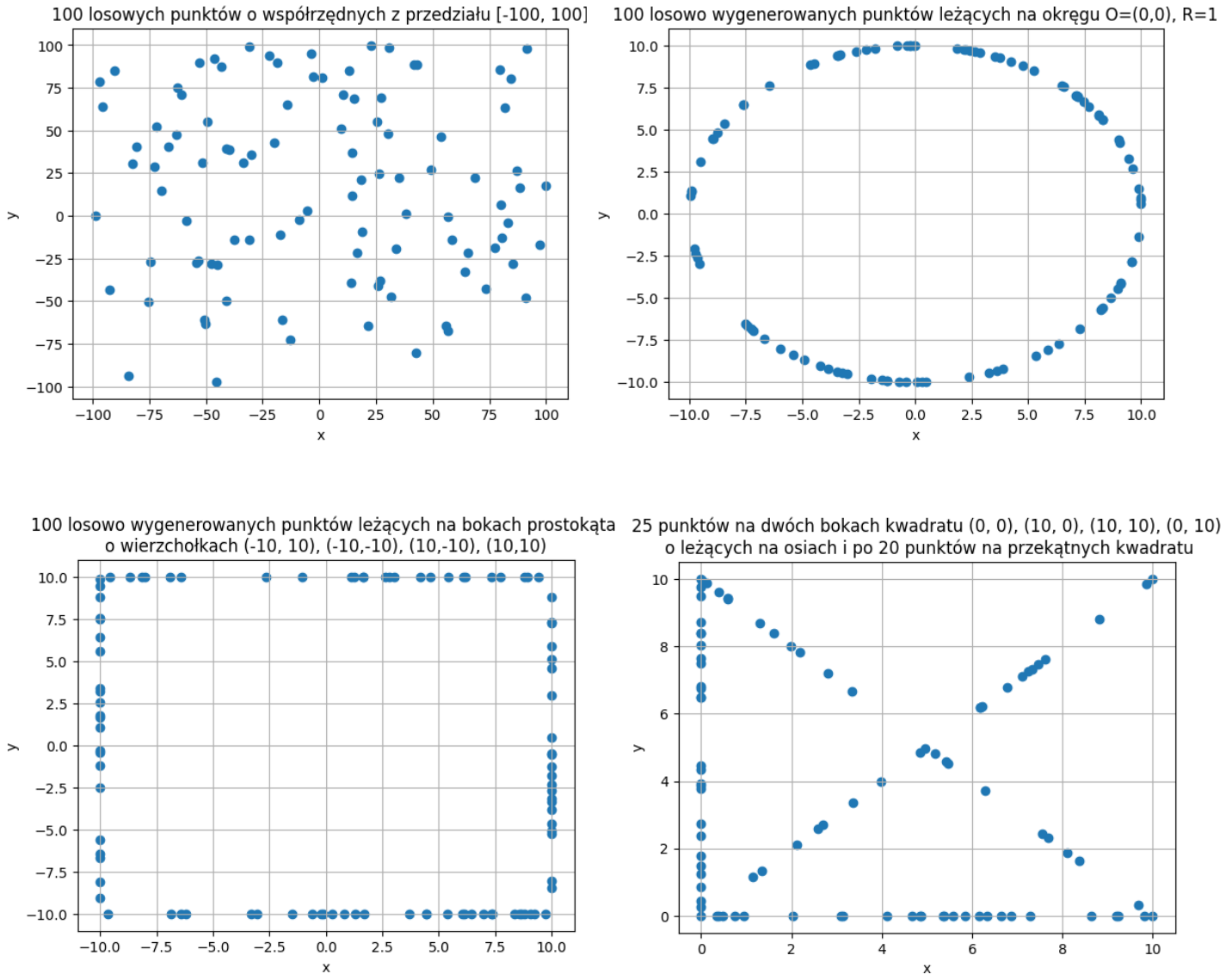
2.2 Sprzęt

Do wykonania został wykorzystany procesor Intel(R) Core(TM) I5-10300H 2.50GHz oraz system operacyjny Microsoft Windows 10 64bit ver 22H2.

3 Przebieg ćwiczeń

3.1 Generowanie punktów

Do generowania punktów wykorzystałem funkcję *numpy.random.uniform*, aby uzyskać równomierny rozkład punktów. Wszystkie funkcje są zaimplementowane w sposób umożliwiający modyfikacje zadanych parametrów.



Rysunek 1: Wykresy rozkładów wygenerowanych punktów

3.2 Przygotowanie algorytmu Grahama

Przy implementacji algorytmu Grahama wykorzystałem funkcję *cmp_to_key* z biblioteki *functools*, aby wykorzystać funkcję własnej implementacji jako porównywarka do ustalania kolejności punktów.

Złożoność czasowa: $O(n * log(n))$

Kolejne kroki algorytmu Grahama dla zbioru Q:

1. Znajdź punkt p_0 w zbiorze Q, który ma najmniejszą współrzędną y oraz najmniejszą współrzędną x w przypadku, gdy wiele punktów ma tą samą współrzędną y
2. Posortuj pozostałe punkty w Q zgodnie z przeciwnym ruchem wskazówek wokół punktu p_0
3. Jeśli kilka punktów tworzy ten sam kąt z p_0, to usuń te punkty (z wyjątkiem najbardziej odległego od p_0)
4. Utwórz stos zawierający p_0 oraz pierwsze dwa elementy posortowanej list Q
5. Dla każdego punktu Q, startując od trzeciego elementu:
    5.1. Dopóki kąt utworzony przez dwa ostatnie punkty stosu oraz aktualnie przetwarzany punkt tworzy lewostronny skręt to:
        5.1.1. Usuń punkt ze stosu
    5.2. Dodaj aktualnie przetwarzany punkt na sts
6. Zwróć otrzymany stos

3.3 Przygotowanie algorytmu Jarvisa

. Algorytm korzystający z idei $gift - wrapping$. Istnieje możliwość wyznaczania fragmentów otoczki (np. górna otoczka). Jeśli liczba punktów otoczki jest ograniczona przez k, wtedy:

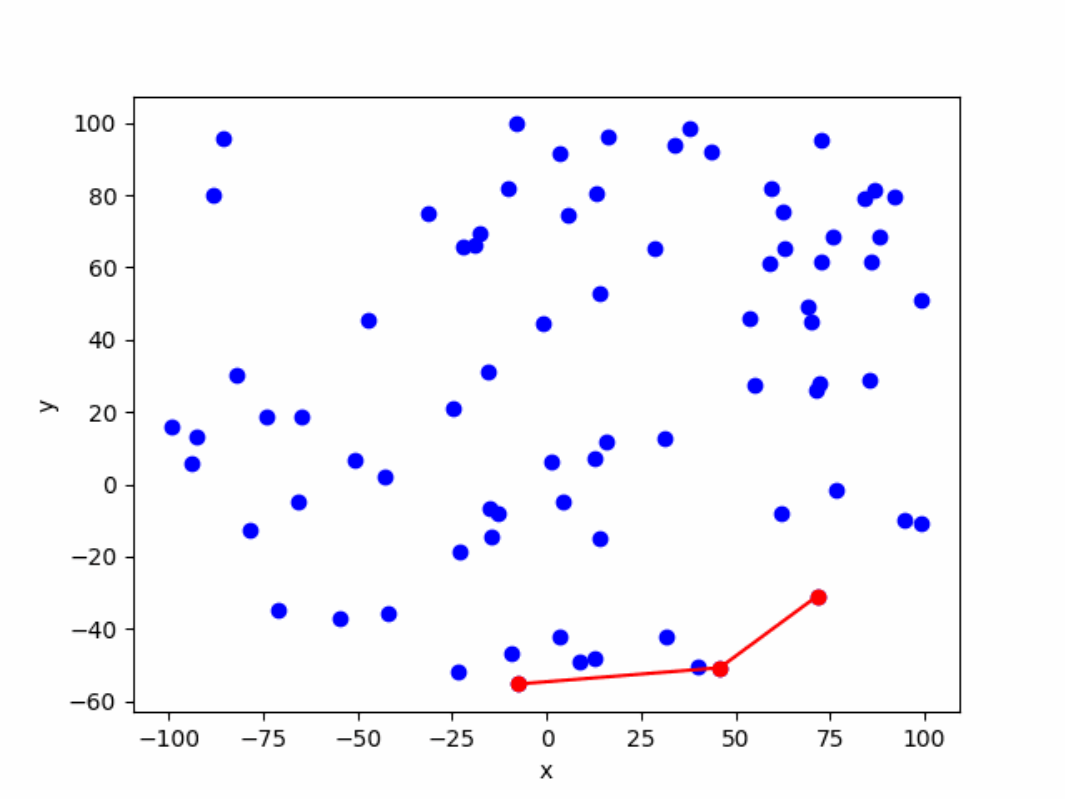
Złożoność czasowa: $O(n * k) -> O(n^2)$

Kolejne kroki algorytmu Grahama dla zbioru Q:

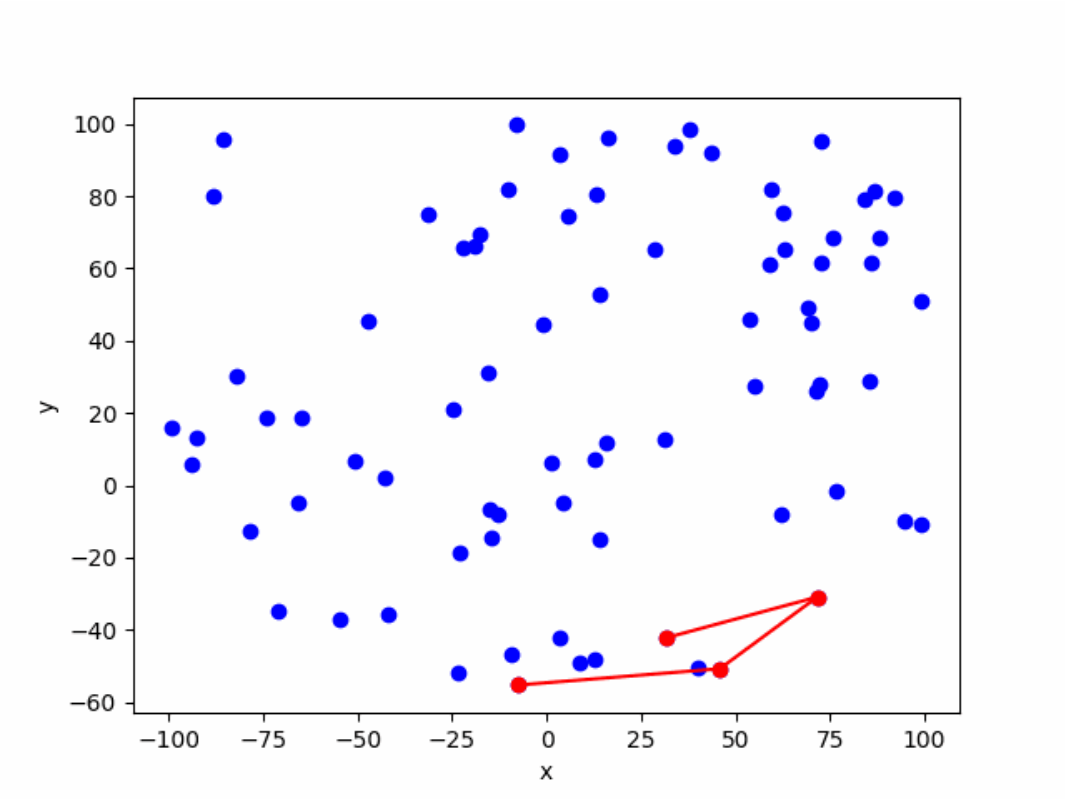
1. Znajdź punkt p_0 w zbiorze Q, który ma najmniejszą współrzędną y oraz najmniejszą współrzędną x w przypadku, gdy wiele punktów ma tą samą współrzędną y
2. Utwórz pusty stos
3. Wybierz punkt z następnym indeksem od p_0 jako potencjalnie przetwarzany wierzchołek otoczki *curr*
4. Dopóki *curr* nie jest p_0:
    4.1. Dodaj *curr* do stosu
    4.2. Dla każdego punktu *next* w Q:
        4.2.1 Jeśli kąt utworzony przez ostatni punkt stosu, *curr* oraz *next* tworzą lewostronny skręt to ustaw wartość *curr* na wartość *next*
5. Zwróć otrzymany stos

4 Wizualizacja graficzna

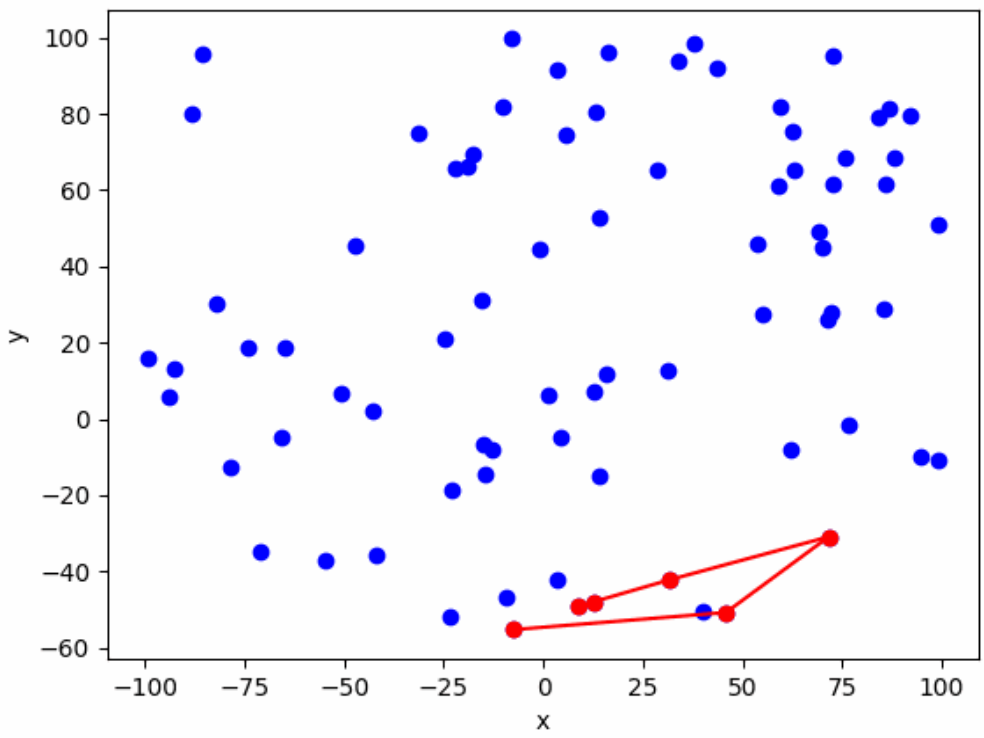
4.1 Algorytm Grahama



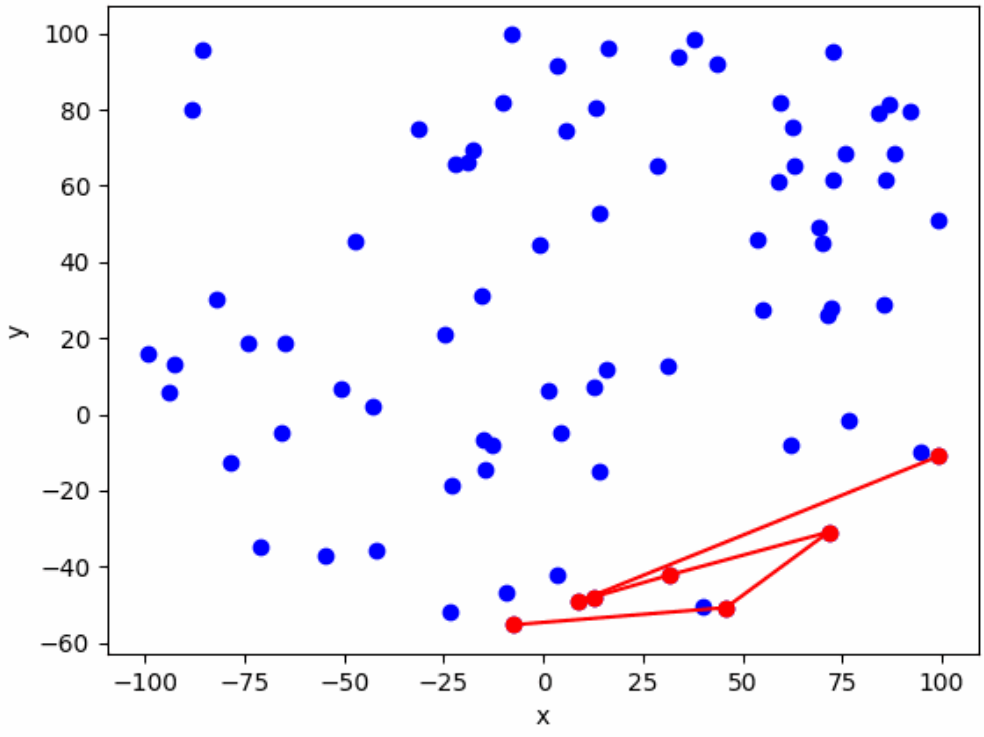
Rysunek 2: Jesteśmy w momencie po dołączeniu drugiego odcinka budowy otoczki



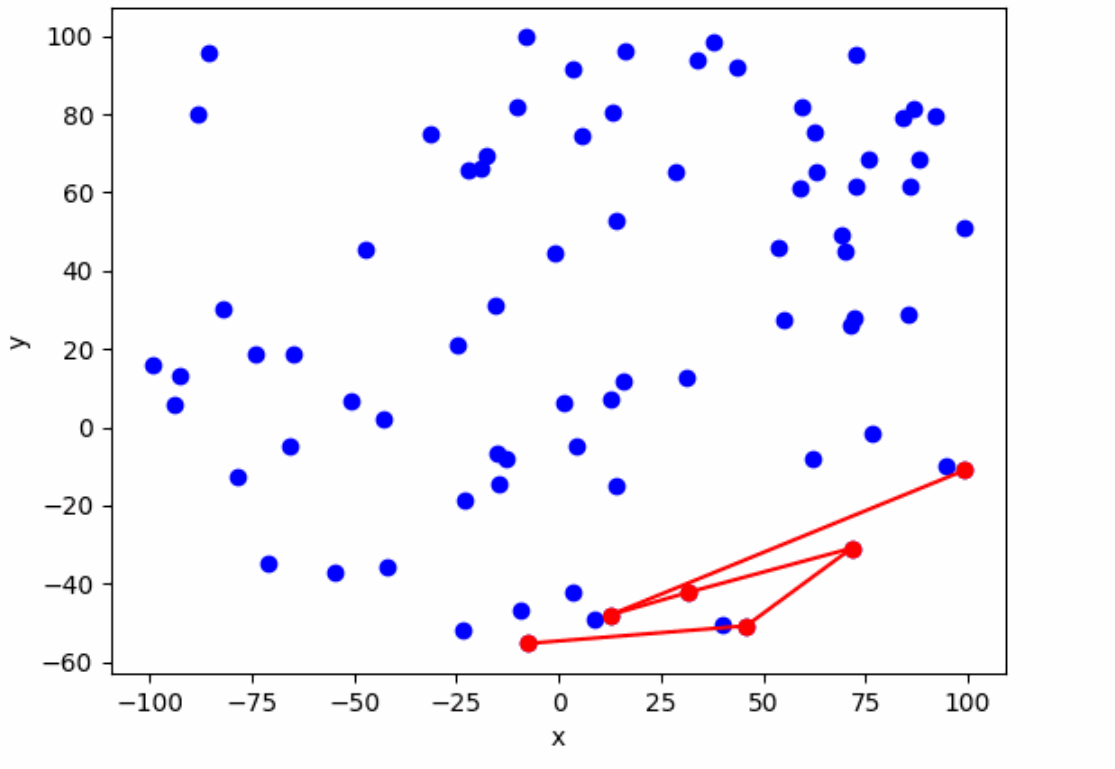
Rysunek 3: Dodaję kolejny punkt do tworzonej otoczki



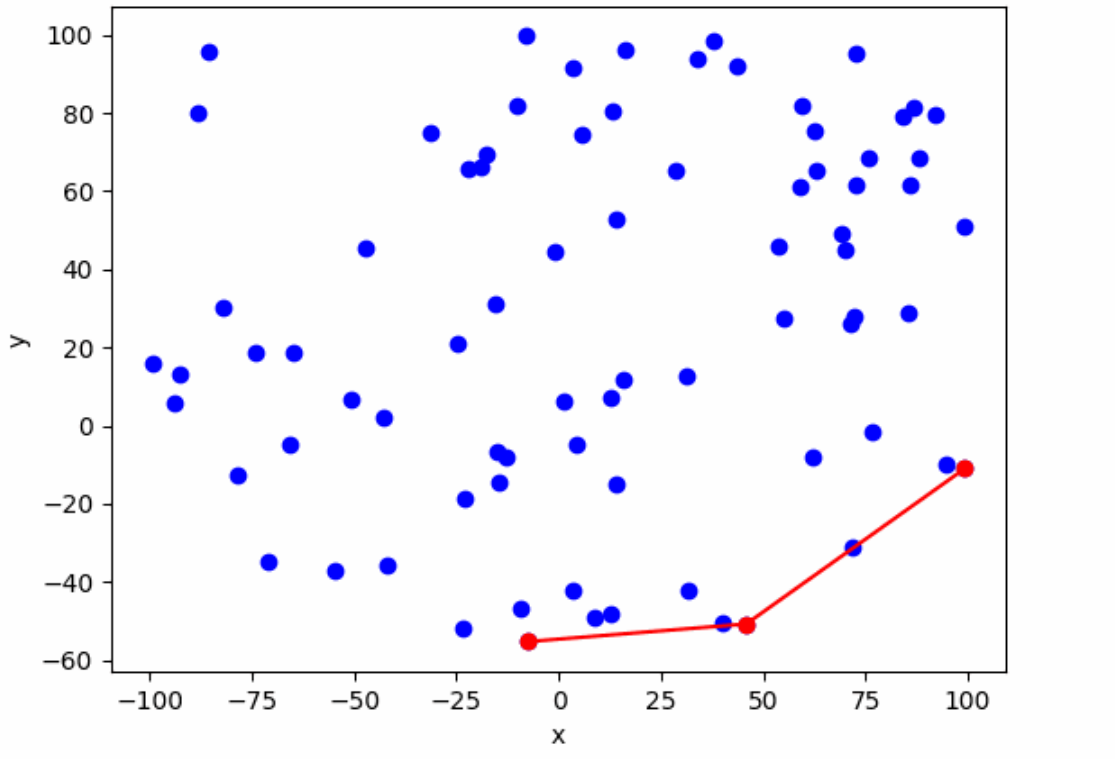
Rysunek 4: Do tego momentu wszytkie kolejne punkty utworzyły prawostronny skręt, więc więc są na stosie jako potencjalne punkty otoczki



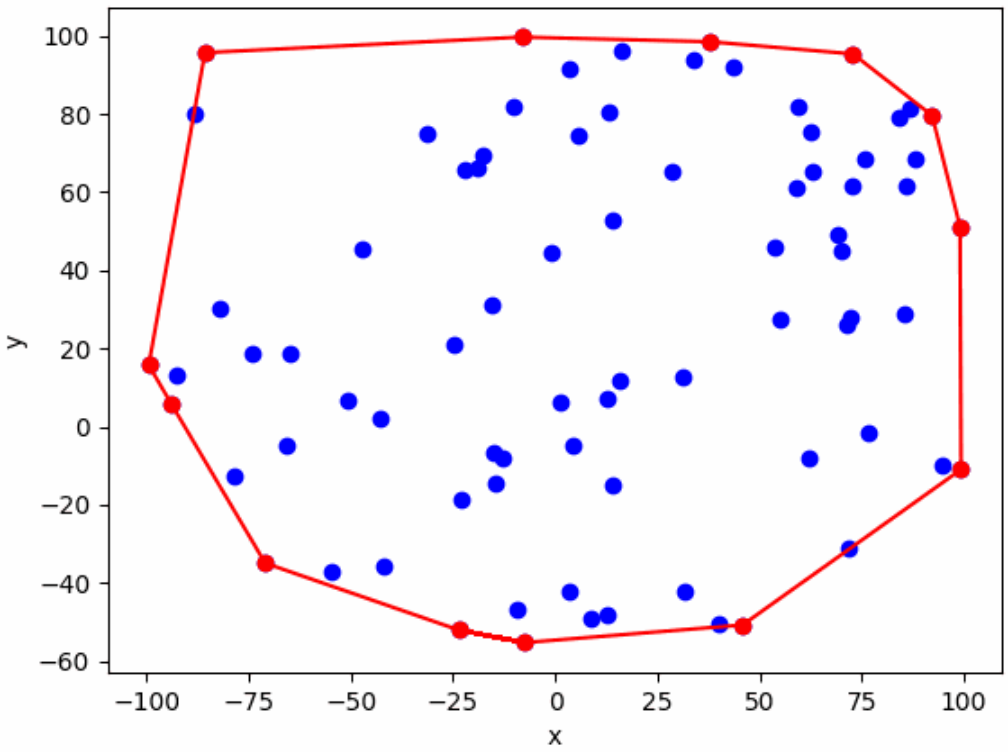
Rysunek 5: Jednak w tym momencie tworzy się lewostronny skręt i dopóki taki będzie, trzeba usuwać kolejne punkty ze stosu



Rysunek 6: Po usunięciu jednego punktu dalej musimy usuwac punkty ze stosu

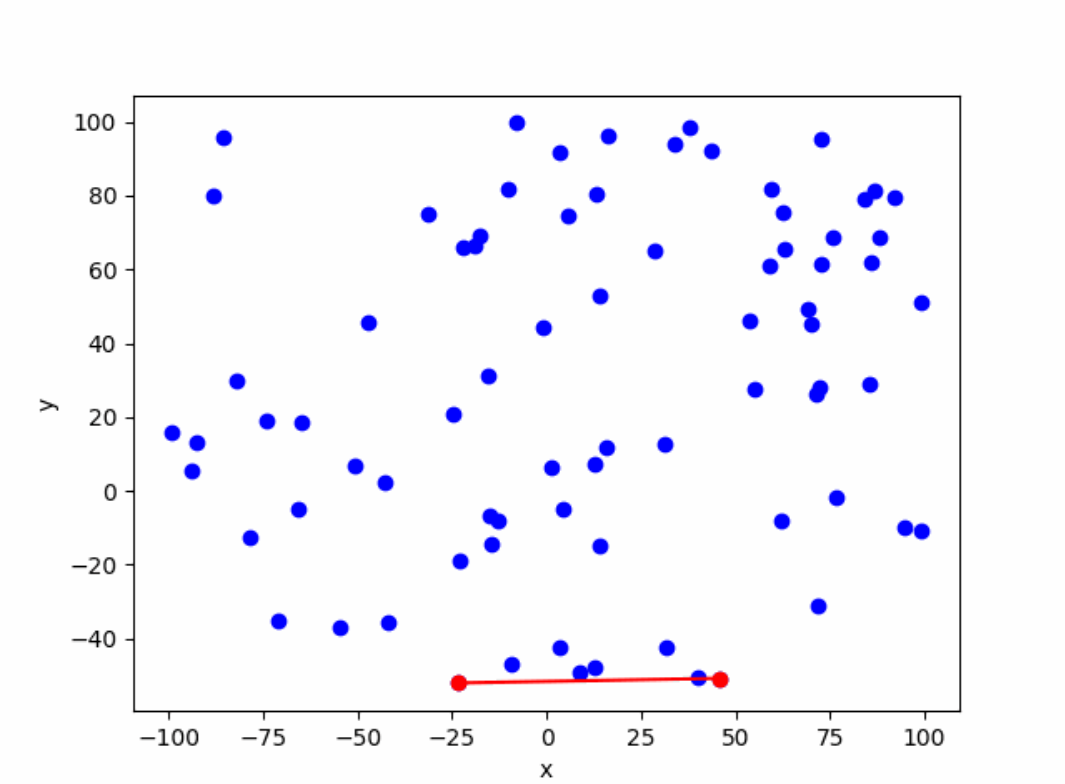


Rysunek 7: Dopiero po usunięciu czterech punktów ze stosu będziemy przetwarzać kolejny punkt z posortowanego zbioru

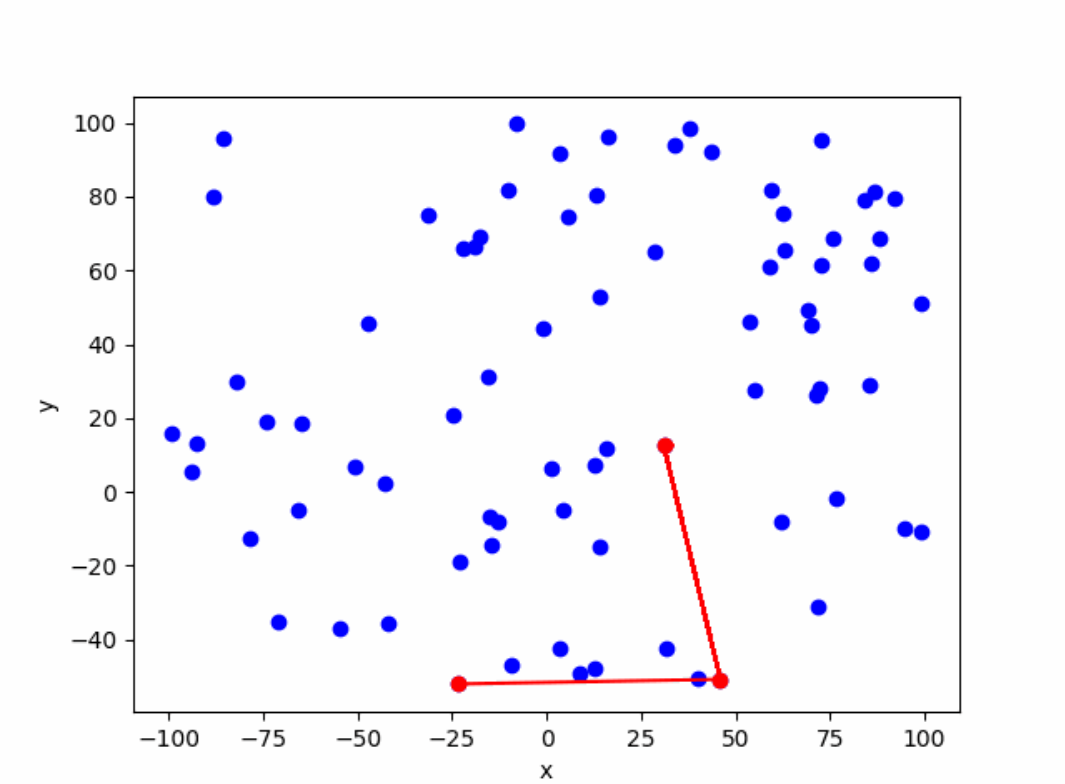


Rysunek 8: Powstała otoczka po zakończeniu pracy algorytmu

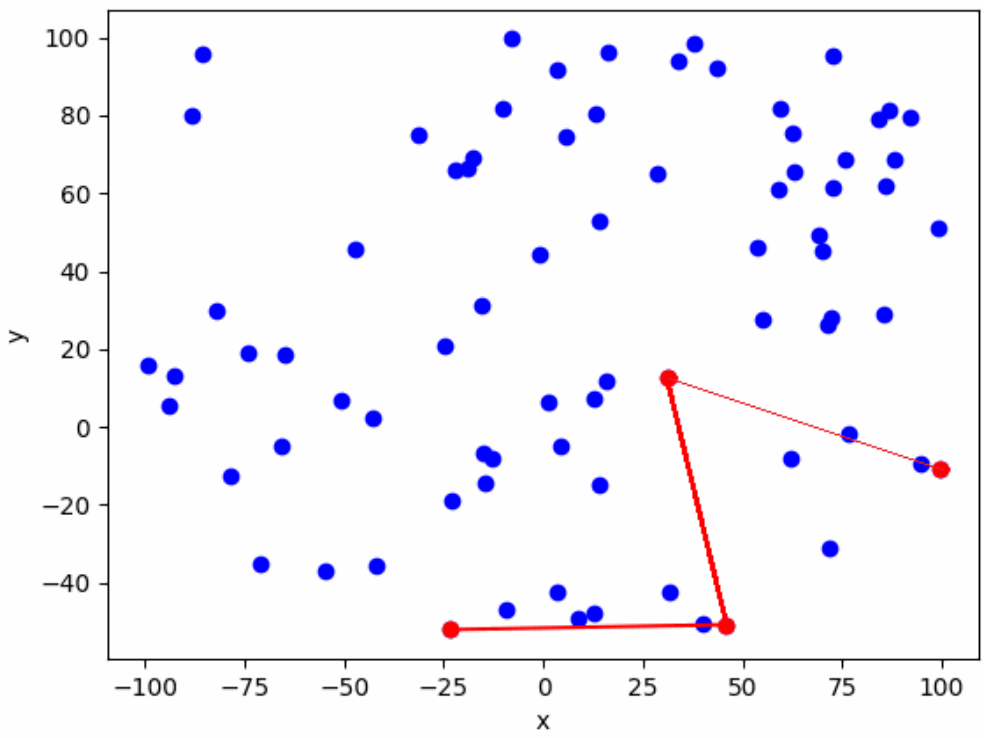
4.2 Algorym Jarvisa



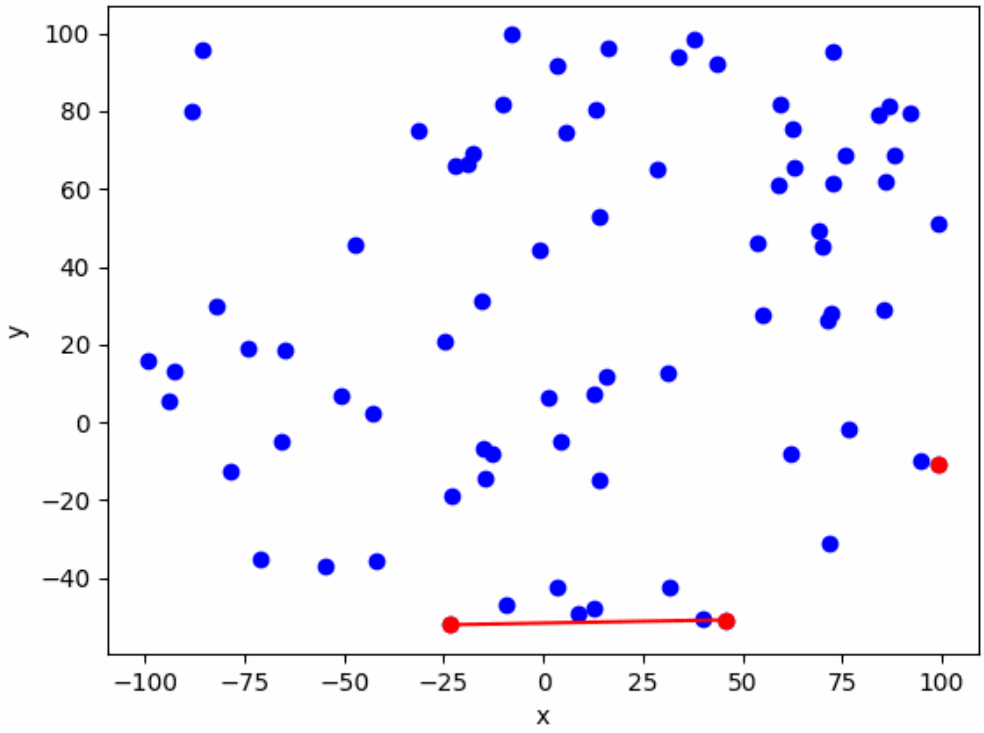
Rysunek 9: Jesteśmy w momencie po dołączeniu pierwszego odcinka otoczki



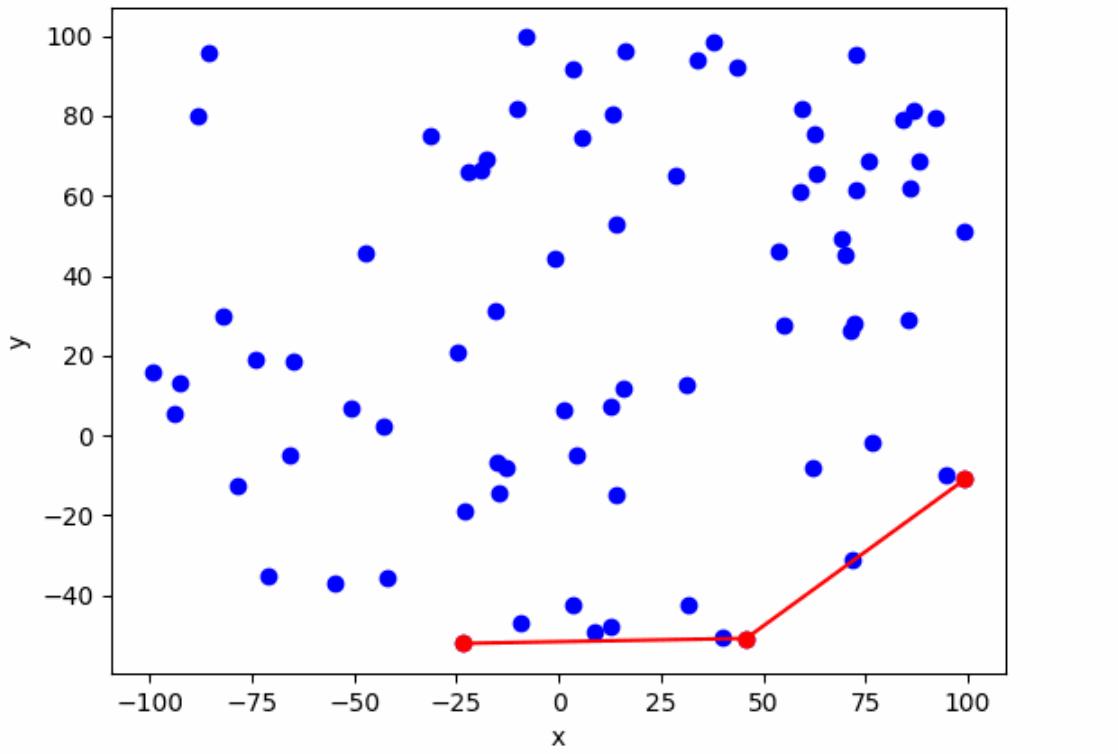
Rysunek 10: Wybieramy kolejny indeksem punkt i ustawiamy go jako *curr*



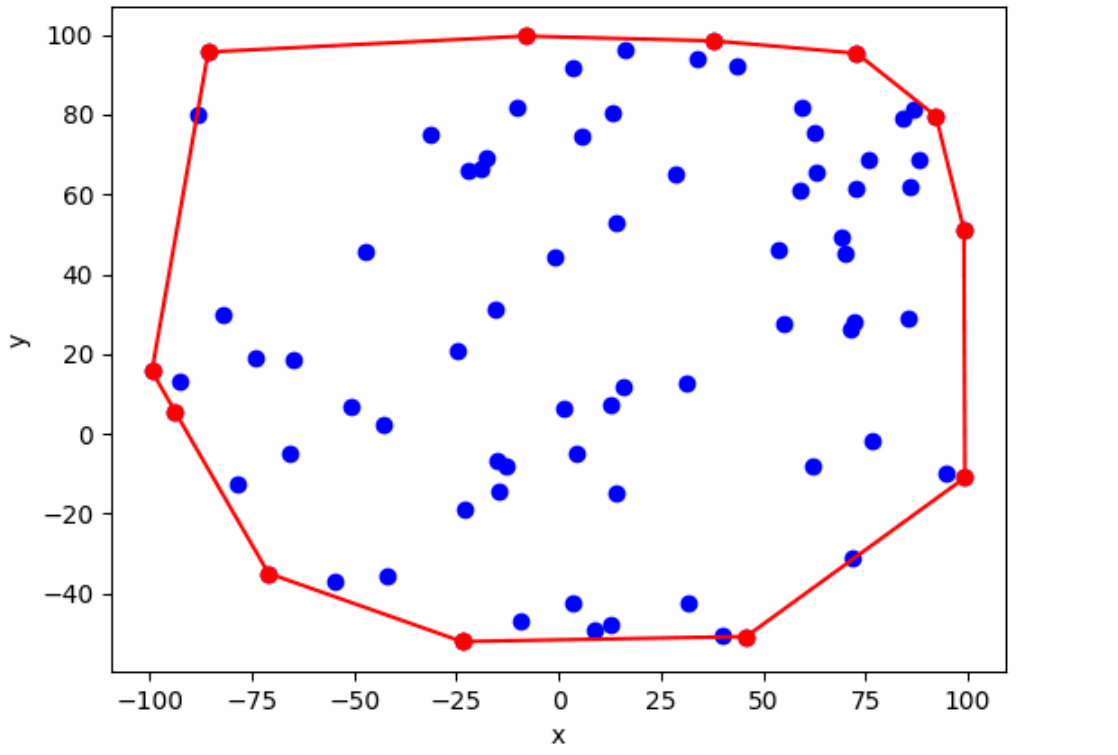
Rysunek 11: Jednak następny punkt *next* tworzy lewostronny skręt, więc to on zostanie nowym *curr*



Rysunek 12: Po przejściu przez pozostałe punkty, żaden nie utworzył lewostronnego skrętu, więc to ten zostanie dodany jako kolejny punkt otoczki



Rysunek 13: Analogicznie kroki powtórzą się dla kolejnych punktów, póki nie zostanie wyznaczona cała otoczka



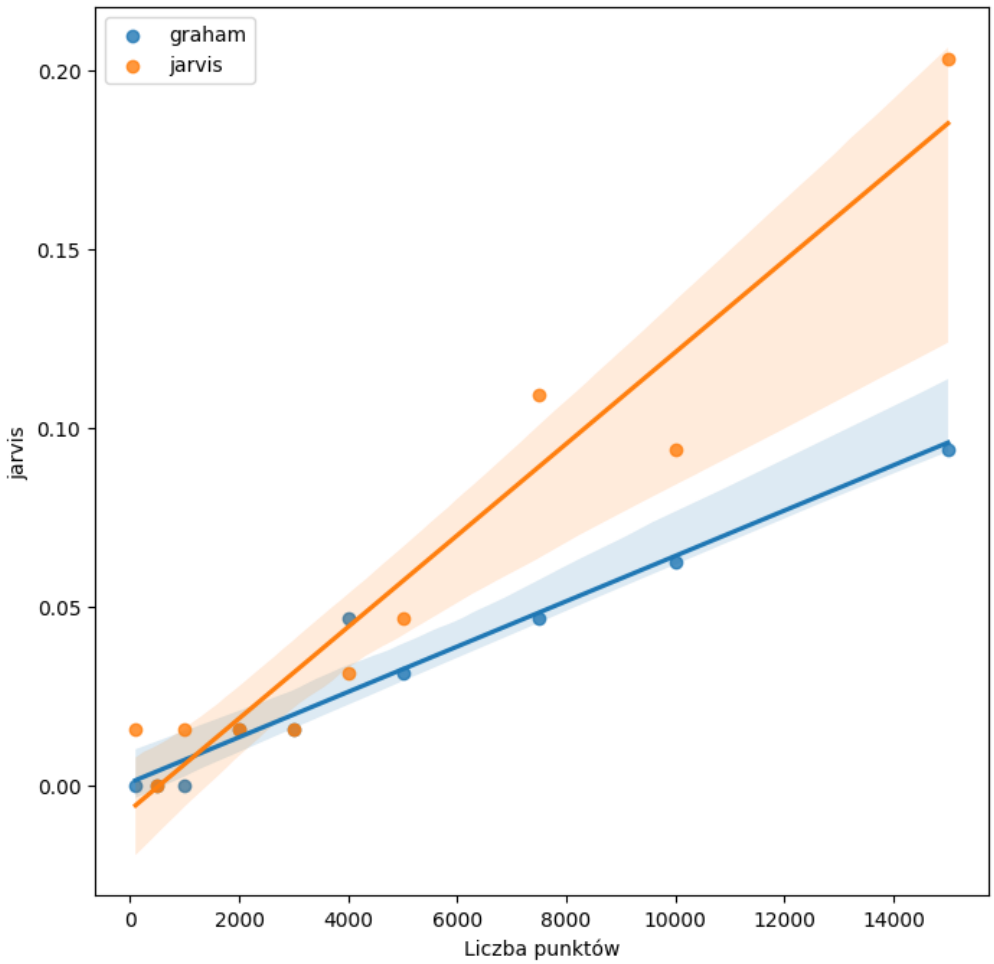
Rysunek 14: Powstała otoczka po zakończeniu pracy algorytmu

5 Porównanie czasu działania algorytmów

5.1 Zbiór A

	Liczba punktów	dolna granica	górna granica	graham [s]	jarvis [s]	szybszy	róznica [s]
1	100	-200	200	0.000000	0.000000	equal	0.000000
2	500	-200	200	0.000000	0.015625	graham	0.015625
3	1000	-200	200	0.000000	0.000000	equal	0.000000
4	2000	-200	200	0.000000	0.031250	graham	0.031250
5	3000	-200	200	0.015625	0.015625	equal	0.000000
6	4000	-200	200	0.015625	0.046875	graham	0.031250
7	5000	-200	200	0.015625	0.062500	graham	0.046875
8	7500	-200	200	0.062500	0.078125	graham	0.015625
9	10000	-200	200	0.062500	0.109375	graham	0.046875
10	15000	-200	200	0.078125	0.187500	graham	0.109375

Rysunek 15: Wyniki pomiarów dla chmury punktów

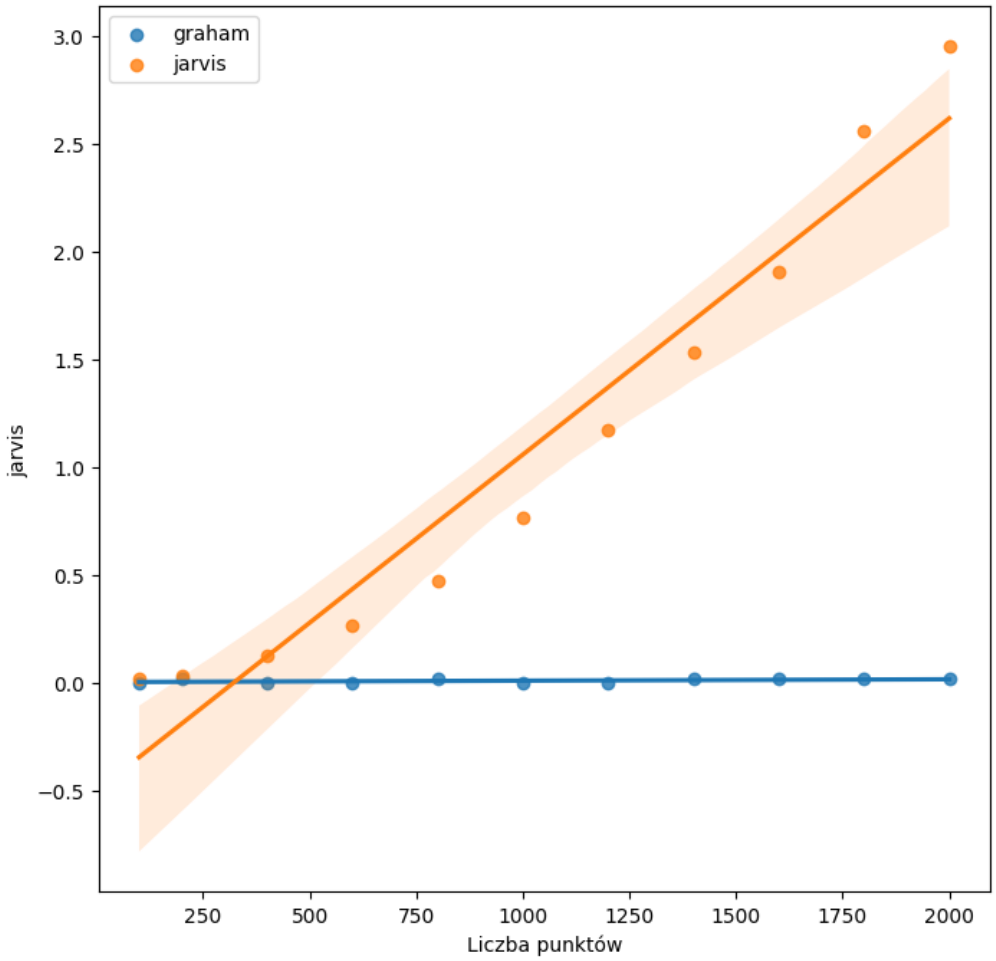


Rysunek 16: Wykres czasu pomiaru od liczby punktów

5.2 Zbiór B

	Liczba punktów	środek	promień	graham [s]	jarvis [s]	szybszy	róznica [s]
1	100	(0, 0)	10	0.000000	0.000000	equal	0.000000
2	200	(0, 0)	10	0.000000	0.031250	graham	0.031250
3	400	(0, 0)	10	0.000000	0.140625	graham	0.140625
4	600	(0, 0)	10	0.000000	0.296875	graham	0.296875
5	800	(0, 0)	10	0.015625	0.437500	graham	0.421875
6	1000	(0, 0)	10	0.000000	0.656250	graham	0.656250
7	1200	(0, 0)	10	0.000000	0.984375	graham	0.984375
8	1400	(0, 0)	10	0.000000	1.406250	graham	1.406250
9	1600	(0, 0)	10	0.000000	1.812500	graham	1.812500
10	1800	(0, 0)	10	0.015625	2.234375	graham	2.218750
11	2000	(0, 0)	10	0.015625	3.156250	graham	3.140625

Rysunek 17: Wyniki pomiarów dla punktów na okręgu

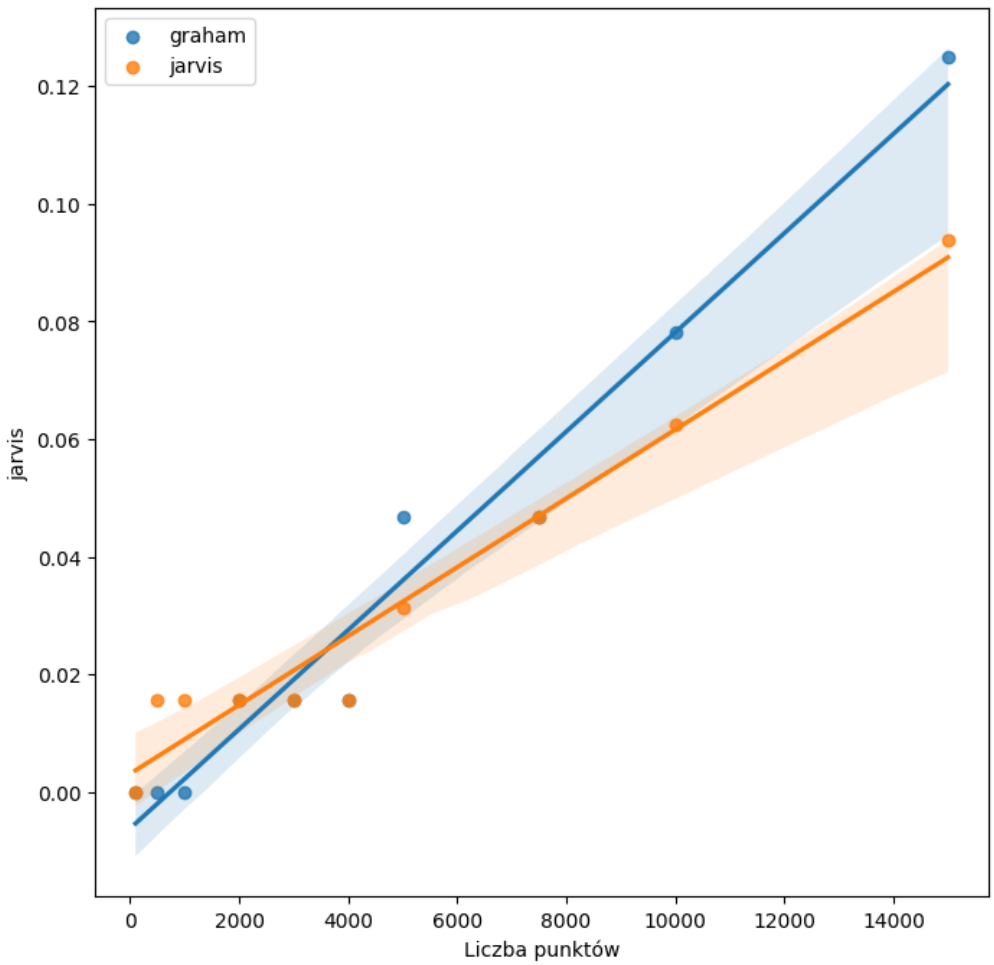


Rysunek 18: Wykres czasu pomiaru od liczby punktów

5.3 Zbiór C

	Liczba punktów	lewy-dół	prawa-góra	graham [s]	jarvis [s]	szybszy	róznica [s]
1	100	(-10, -10)	(10, 10)	0.000000	0.000000	equal	0.000000
2	500	(-10, -10)	(10, 10)	0.000000	0.000000	equal	0.000000
3	1000	(-10, -10)	(10, 10)	0.015625	0.000000	jarvis	0.015625
4	2000	(-10, -10)	(10, 10)	0.015625	0.015625	equal	0.000000
5	3000	(-10, -10)	(10, 10)	0.015625	0.015625	equal	0.000000
6	4000	(-10, -10)	(10, 10)	0.031250	0.015625	jarvis	0.015625
7	5000	(-10, -10)	(10, 10)	0.031250	0.015625	jarvis	0.015625
8	7500	(-10, -10)	(10, 10)	0.046875	0.031250	jarvis	0.015625
9	10000	(-10, -10)	(10, 10)	0.078125	0.046875	jarvis	0.031250
10	15000	(-10, -10)	(10, 10)	0.109375	0.062500	jarvis	0.046875

Rysunek 19: Wyniki pomiarów dla punktów na krawędzi prostokąta

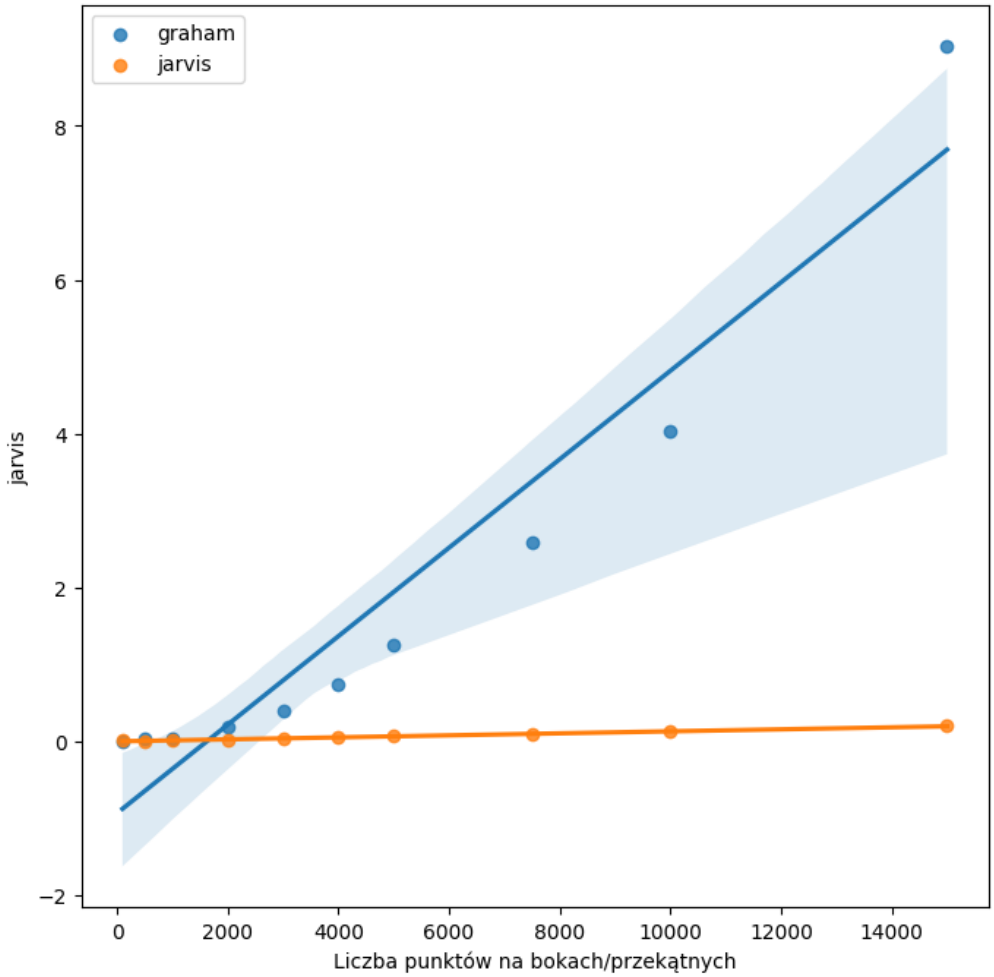


Rysunek 20: Wykres czasu pomiaru od liczby punktów

5.4 Zbiór D

	Liczba punktów na bokach/przekątnych	lewy-dół	prawa-góra	graham [s]	jarvis [s]	szybszy	róznica [s]
1	100	(0, 0)	(10, 10)	0.000000	0.000000	equal	0.000000
2	500	(0, 0)	(10, 10)	0.015625	0.015625	equal	0.000000
3	1000	(0, 0)	(10, 10)	0.046875	0.015625	jarvis	0.031250
4	2000	(0, 0)	(10, 10)	0.234375	0.031250	jarvis	0.203125
5	3000	(0, 0)	(10, 10)	0.406250	0.031250	jarvis	0.375000
6	4000	(0, 0)	(10, 10)	0.703125	0.078125	jarvis	0.625000
7	5000	(0, 0)	(10, 10)	1.265625	0.078125	jarvis	1.187500
8	7500	(0, 0)	(10, 10)	2.703125	0.109375	jarvis	2.593750
9	10000	(0, 0)	(10, 10)	4.640625	0.125000	jarvis	4.515625
10	15000	(0, 0)	(10, 10)	10.812500	0.171875	jarvis	10.640625

Rysunek 21: Wyniki pomiarów dla punktów na dwóch bokach i przekątnych kwadratu



Rysunek 22: Wykres czasu pomiaru od liczby punktów

6 Wnioski

- Oba algorytmy prawidłowo wyznaczyły otoczki wypukłe dla wszystkich zbiorów punktów. Dla zbiorów A i B algorytm Grahama był szybszy, jednak w pozostałych przypadkach to algorytm Jarvisa kończył swoje zadanie pierwszy.
- Można zauważyć, że algorytm Grahama otrzymuje podobne wyniki czasowe dla różnych zbiorów, w odróżnieniu od algorytmu Jarvisa. Spowodowane jest to tym, że złożoność algorytmu Grahama nie zależy od liczby punktów na otoczce, tylko jest stała.
- Z tego wynika, że przy nieznanym rozkładzie punktów to algorytm Grahama będzie lepszym wyborem. Jednakże dla zbiorów które mają wiele punktów współlinowych, to własnie algorytm Jarvisa będzie naturalnym wyborem.