Jan Kaczmarski
Informatyka II rok, grupa 5, 15.10.2024

Algorytmy geometryczne, laboratorium 2 - sprawozdanie

1. Opis ćwiczenia

Ćwiczenie polegało na implementacji oraz testach algorytmów Grahama oraz Jarvisa, wyznaczających otoczkę wypukłą zbioru punktów.

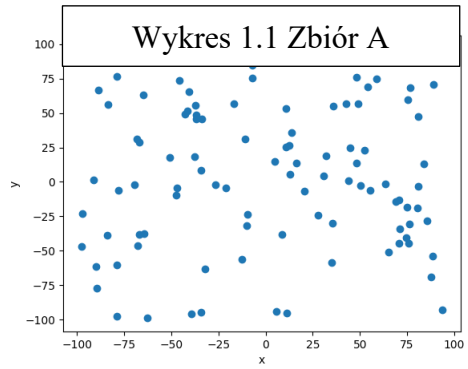
2. Środowisko, biblioteki oraz użyte narzędzia

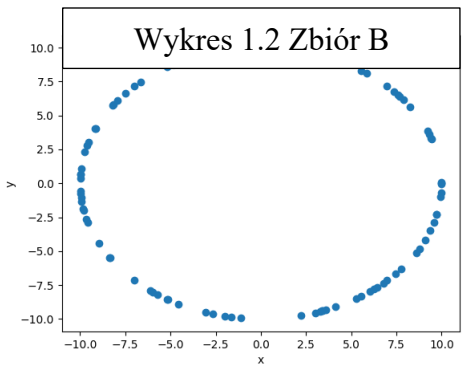
Ćwiczenie zostało wykonane w Jupyter Notebook i napisane w języku Python z wykorzystaniem narzędzia Visualizer autorstwa koła naukowego BIT do wizualizacji obliczeń i rysowania wykresów. Do wygenerowania losowych punktów użyłem biblioteki numpy. Wszystko było wykonywane na systemie operacyjnym macOS Sonoma14.6.1 I procesorze Apple M2 Pro arm64

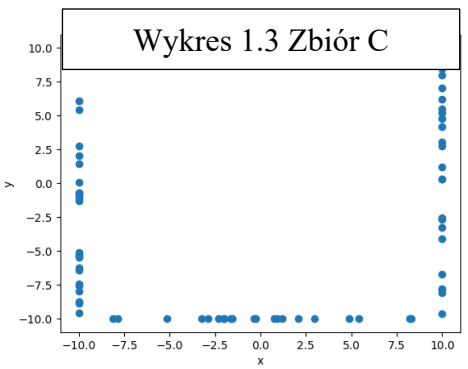
3. Generacja punktów

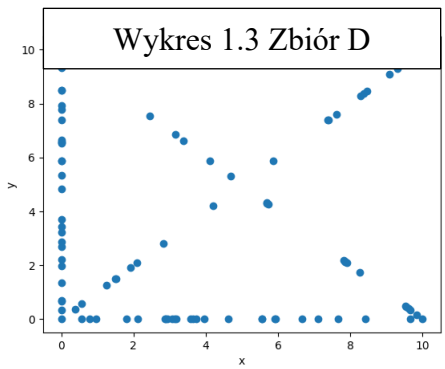
W celu wykonania ćwiczenia wygenerowane zostały 4 zbiory punktów (2D, typu double), które zostały umieszczone w osobnych tablicach:

- Zbiór A - 100 losowych punktów o współrzędnych z przedziału [-100, 100]
- Zbiór B - 100 losowych punktów leżących na okręgu o środku (0,0) i promieniu R=10
- Zbiór C - 100 losowych punktów leżących na bokach prostokąta o wierzchołkach (-10, 10), (-10,-10), (10,-10), (10,10)
- Zbiór D - zawierający wierzchołki kwadratu (0, 0), (10, 0), (10, 10), (0, 10) oraz punkty wygenerowane losowo w sposób następujący: po 25 punktów na dwóch bokach kwadratu leżących na osiach i po 20 punktów na przekątnych kwadratu.









4. Implementacja algorytmu Grahama

Algorytm działa w następujący sposób: Na początku znajduje punkt o najmniejszej współrzędnej y, jeśli takich punktów jest kilka, wybiera ten o najmniejszej współrzędnej x. Jest to punkt startowy. Następnie dokonuje sortowania punktów. Punkty sortowane są wg. kąta jaki tworzy wektor punktu startowego z rozważanym względem dodatniego kierunku osi x. W przypadku gdy rozpatrywane punkty tworzą taki sam kąt, pierwszeństwo mają punkty bardziej oddalone od punktu startowego. Następnie standardowo sprawdzamy orientację punktu względem ostatniej krawędzi i dokładamy punkty do otoczki bądź cofamy się. Po wykonaniu głównej pętli sprawdzamy czy ostatni punkt nie jest zbędny.

5. Implementacja algorytmu Jarvisa

Podobnie jak w przypadku poprzedniego algorytmu na początku znaleziono punkt startowy. W wewnętrznej pętli algorytmu poszukiwano punktu, dla którego wszystkie pozostałe leżą po lewej stronie odcinka wyznaczonego przez poprzedni punkt otoczki i tenże punkt. W przypadku znalezienia kilku współliniowych punktów spełniających to założenie wybierano ten najbardziej oddalony od ostatniego punktu otoczki (tak aby nowo powstała krawędź była jak najdłuższa). Znaleziony punkt dodawano do otoczki i kontynuowano do momentu dotarcia z powrotem do punktu startowego.

6. Otrzymane wyniki oraz ich analiza

W poniższej tabeli przedstawiam liczbę wierzchołków otoczki wypukłej obliczonej przez każdy z dwóch algorytmów dla każdego ze zbiorów.

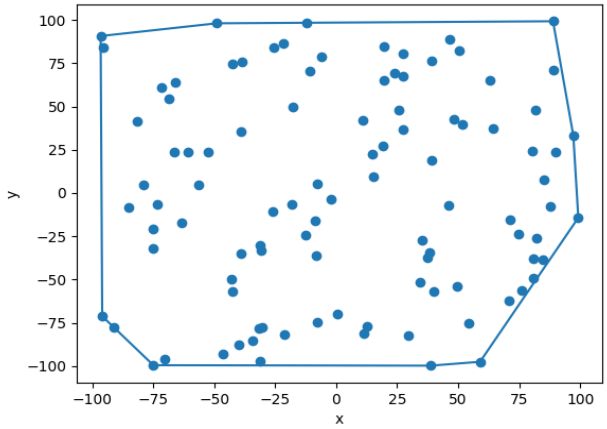
Zbiór Punktów	Liczba wierzchołków otoczki – algorytm Grahama	Liczba wierzchołków otoczki – algorytm Jarvisa
Zbiór A	9	9
Zbiór B	100	100
Zbiór C	8	8
Zbiór D	4	4

Tabela 6.1 Liczba wierzchołków otoczki w obu algorytmach dla danych zbiorów punktów

Jak widać w tabeli 6.1, oba algorytmy wskazują taką samą liczbę wierzchołków otoczki dla każdego z przygotowanych zbiorów punktów. Okazuje się, że istotnie są to te same otoczki, dlatego tutaj umieszczę po jednym rysunku dla każdego zbioru, natomiast np. na potrzeby porównania, wszystkie wyznaczone otoczki dla obu algorytmów znajdują się w pliku z implementacją.

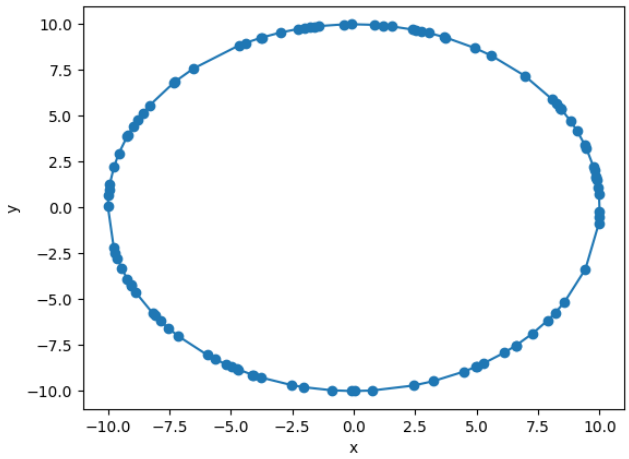
Poniżej oraz na następnych stronach znajdują się graficzne przedstawienia otoczki wypukłej.

Zbiór A



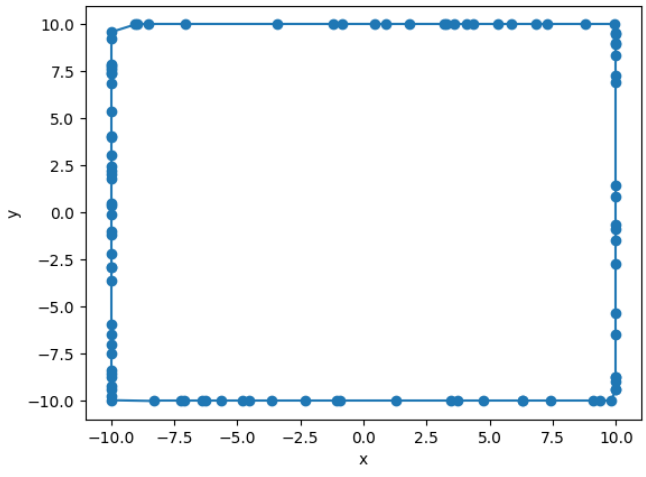
Rys. 6.2 Otoczka wypukła zbioru A wyznaczona przez algorytmy Grahama i Jarvisa

Zbiór B



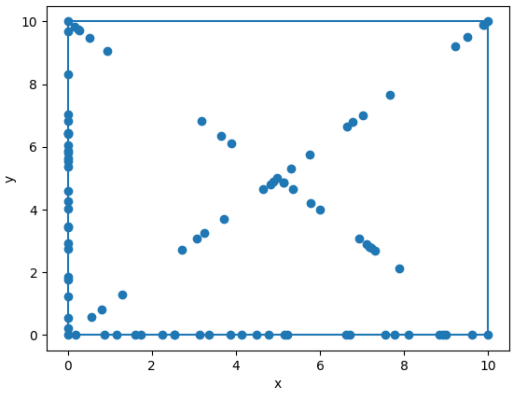
Rys. 6.3 Otoczka wypukła zbioru B obliczona przez algorytmy Grahama i Jarvisa

Zbiór C



Rys. 6.4 Otoczka wypukła zbioru C obliczona algorytmami Grahama i Jarvisa

Zbiór D



Rys. 6.5 Otoczka wypukła zbioru D wyznaczona algorytmami Grahama i Jarvisa

7. Porównanie złożoności czasowej algorytmów

Teoretyczna złożoność czasowa algorytmu Grahama to $O(nlogn)$, natomiast algorytmu Jarvisa $O(kn)$, gdzie n oznacza liczbę wierzchołków zbioru, dla którego wyznaczamy otoczkę, a k to liczba wierzchołków otoczki. Pesymistycznie zatem algorytm Jarvisa ma złożoność $\Theta(n^2)$. Aby porównać czasy obliczania otoczki, przygotuję zmodyfikowane zbiory punktów:

- Nowy zbiór A: n losowych punktów o współrzędnych z przedziału [-1000, 1000]
- Nowy zbiór B: m losowych punktów leżących na okręgu o środku w punkcie (100,100) i promieniu R=500
- Nowy zbiór C: n losowych punktów leżących na bokach prostokąta o wierzchołkach (-200,50), (-200,-150), (100,-150), (100,50)
- Nowy zbiór D: zawierający wierzchołki kwadratu (0,0), (100,0), (100,100), (0,100) oraz po $n1$ punktów leżących na osiach i po n_2 punktów na przekątnych (łącznie n),

gdzie $n \in \{100,1000,5000,10000,50000\}$, $m \in \{10,100,500,1000,5000\}$ oraz n_1 i n_2 są równe odpowiednio $0{,}3 \cdot n - 1$ oraz $0{,}2 \cdot n - 1$. Dla zbioru B przyjęto inną ilość punktów z powodu złożoności algorytmu Jarvisa $O(kn)$, dla zbiour B w którym wszystkie punkty należą do otoczki algorytm ma złożoność czasową $O(n^2)$. Dla liczby punktów 10000 i 50000 zakończenie algorytmu zajmuje zdecydowanie za dużo czasu, żeby go zmierzyć i brać pod uwagę w naszym rozważaniu. Dlatego dla zbioru B liczba punktów należy do zbioru m.

Zbióry Punktów	Liczba punktów				
	100	1000	5000	10000	50000
Nowy zbiór A	0.0009 s	0.0069 s	0.0104 s	0.0163 s	0.0856 s
Nowy zbiór C	0.0003 s	0.0018 s	0.0080 s	0.0159 s	0.0860 s
Nowy zbiór D	0.0001 s	0.0012 s	0.0062 s	0.0120 s	0.0627 s

Tabela 7.1 Czasy wykonania algorytmu Grahama dla różnych zbiorów

Zbióry Punktów	Liczba punktów				
	10	100	500	1000	5000
Nowy Zbiór B	0.0000 s	0.0002 s	0.0008 s	0.0017 s	0.0095 s

Tabela 7.2 Czasy wykonania algorytmu Grahama dla zbiorów podobnych do B

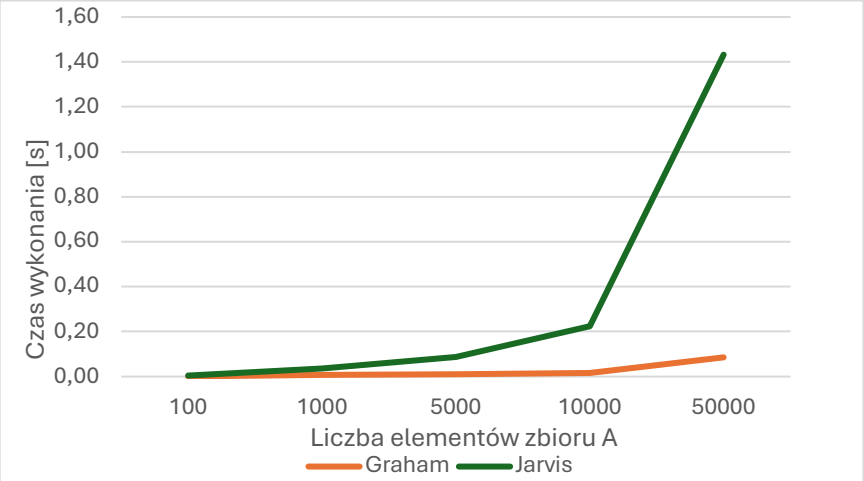
Zbióry Punktów	Liczba punktów				
	100	1000	5000	10000	50000
Nowy zbiór A	0.0046 s	0.0355 s	0.0886 s	0.2228 s	1.4325 s
Nowy zbiór C	0.0007 s	0.0071 s	0.0343 s	0.0688 s	0.3482 s
Nowy zbiór D	0.0004 s	0.0034 s	0.0170 s	0.0336 s	0.1691 s

Tabela 7.3 Czasy wykonania algorytmu Jarvisa dla różnych zbiorów

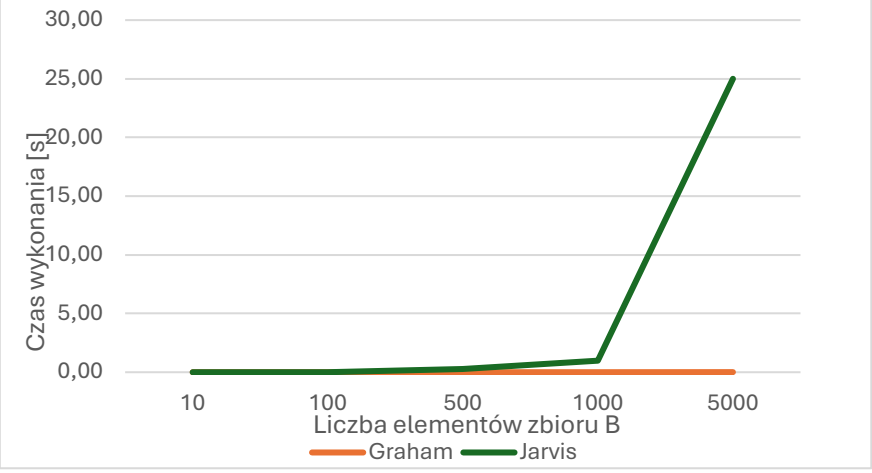
Zbióry Punktów	Liczba punktów				
	10	100	500	1000	5000
Nowy Zbiór B	0.0001 s	0.0098 s	0.2487 s	0.9876 s	25.0012 s

Tabela 7.4 Czasy wykonania algorytmu Jarvisa dla zbiorów podobnych do B

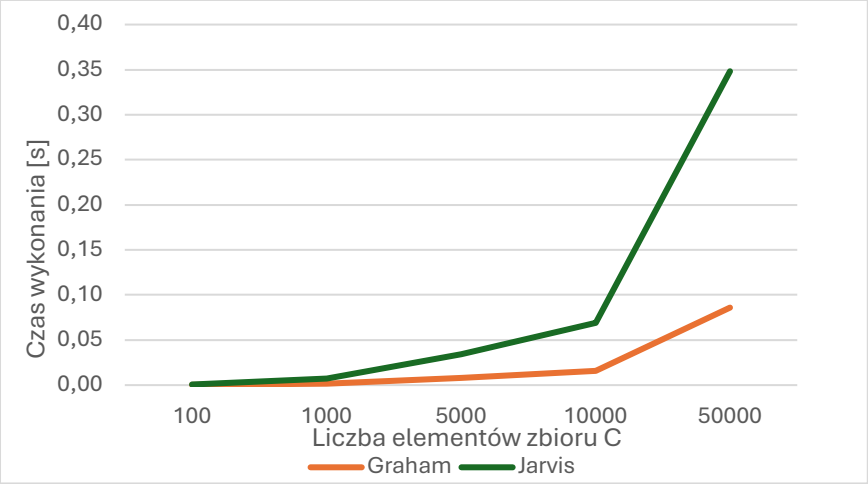
(*) Czas 0,0000 s oznacza, że czas wykonania był mniejszy niż 0,5·10-5 s, ale nie zerowy.



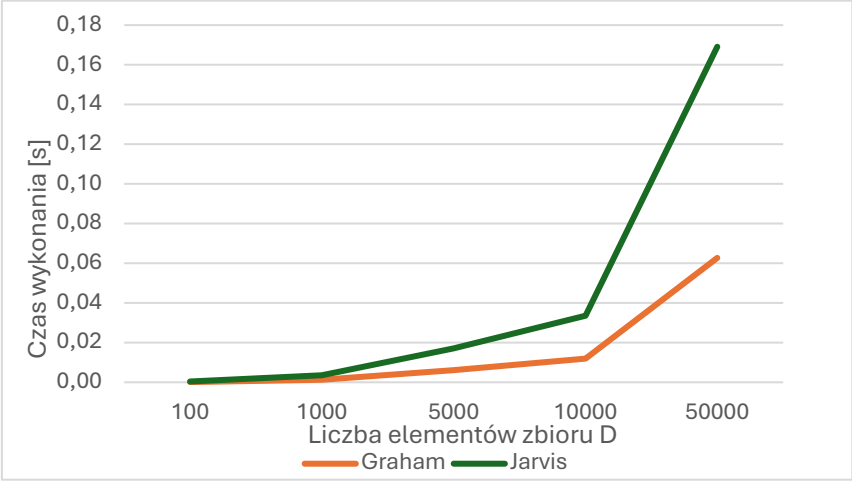
Wykres 7.5 Czas wykonania algorytmów Grahama i Jarvisa dla zestawu danych A w zależności od ilości punktów w zestawie



Wykres 7.6 Czas wykonania algorytmów Grahama i Jarvisa dla zestawu danych B w zależności od ilości punktów w zestawie



Wykres 7.7 Czas wykonania algorytmów Grahama i Jarvisa dla zestawu danych C w zależności od ilości punktów w zestawie



Wykres 7.7 Czas wykonania algorytmów Grahama i Jarvisa dla zestawu danych C w zależności od ilości punktów w zestawie

Wnioski

Na podstawie uzyskanych wyników testów można stwierdzić, że algorytmy całkiem dobrze radziły sobie z podstawowymi zbiorami danych. Pewne odstępstwa mogły wynikać z ograniczeń w precyzji przechowywania liczb rzeczywistych przez komputer – na przykład niektóre współliniowe punkty o identycznym kącie nie zostały wykluczone, ponieważ wartość wyznacznika była poza zakresem ustalonej tolerancji. Tolerancja dla zera została przeze mnie dobrana metodą prób i błędów, opierając się na założeniu, że wyniki dla testów zaproponowanych przez AGH Bit będą poprawne, zakładając rzetelne przygotowanie tych zbiorów danych.

Przypuszczam, że specyficzne zestawy punktów zostały zaproponowane właśnie ze względu na potencjalne trudności obliczeniowe. Zbiór A jest raczej "neutralny", podczas gdy zbiór B stanowi wyzwanie, gdyż wszystkie jego punkty tworzą otoczkę wypukłą. Natomiast zbiory C i D zawierają wiele współliniowych punktów, co umożliwia przetestowanie eliminacji wszystkich punktów poza tymi najbardziej oddalonymi przy wyznaczaniu otoczki wypukłej. Uważam, że mój program sprostał wymaganiom tego zadania.