Algorytmy geometryczne

dr Anna Beata Kwiatkowska

Geometria obliczeniowa

Dział informatyki zajmujący się algorytmami geometrycznymi nazywamy geometrią obliczeniową.

Najczęściej rozpatrywane są problemy:

- **Na płaszczyźnie** podstawowe problemy:
  - Kiedy dwa odcinki przecinają się?
  - Kiedy punkt przynależy do wielokąta?
  - Jak znaleźć wypukłą otoczkę zbioru punktów na płaszczyźnie?
  - Jak stwierdzić, czy istnieją przecinające się pary odcinków?

  mają zastosowanie także do problemów w przestrzeni.
- **W przestrzeni** problemy ważne ze względu na zastosowanie w animacji komputerowej.

Zakładamy, że mamy do dyspozycji tylko cztery działania **+, -, *, /.**
Nie korzystamy z funkcji trygonometrycznych.
Mamy bardzo dokładne obliczenia – nieskończona precyzja.

Geometria obliczeniowa

- Rozważania ograniczamy do geometrii na płaszczyźnie.

- Podstawowe obiekty geometryczne:
  - **punkt** p reprezentujemy parą współrzędnych p=(x, y) w ustalonym wcześniej układzie współrzędnych kartezjańskich
  - **odcinek** wyznaczony przez parę punktów p, q oznaczamy przez p−q
  - **wektor** o początku w punkcie p i końcu w punkcie q oznaczamy przez p→q
  - **prosta** będzie reprezentowana przez zawarty w niej odcinek lub wektor.

Podstawowe fakty z geometrii obliczeniowej na płaszczyźnie

Jeżeli x jest liczba rzeczywistą, to **znak** liczby x oznaczamy przez sgn(x) i definiujemy:

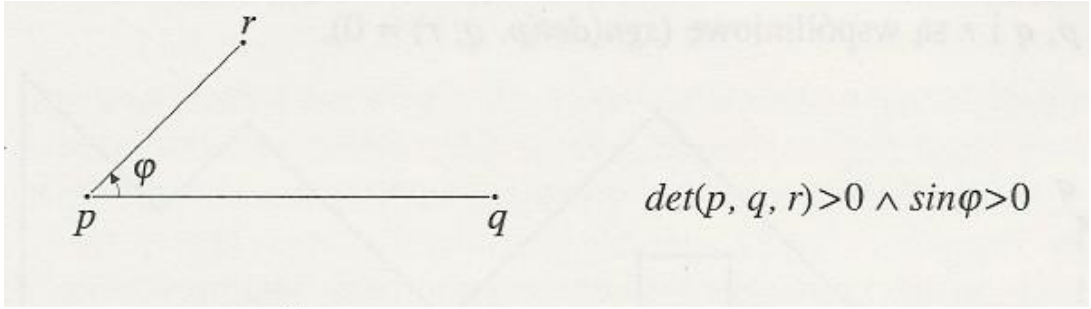
$$\text{sgn}(x) = \begin{cases} +1 & \text{dla } x>0 \\ 0 & \text{dla } x=0 \\ -1 & \text{dla } x<0 \end{cases}$$

Niech będą dane punkty p, q i r o współrzędnych $p=(x_p,y_p)$, $q=(x_q, y_q)$, $r=(x_r, y_r)$, a det(p,q,r) będzie wyznacznikiem macierzy

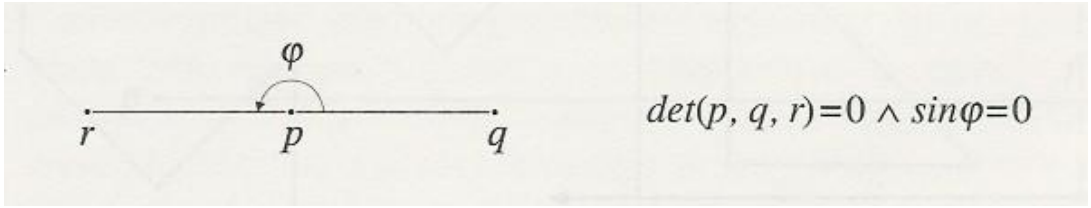
$$\det(p,q,r) = \begin{bmatrix} x_p & y_p & 1 \\ x_q & y_q & 1 \\ x_r & y_r & 1 \end{bmatrix}$$

Znak wyznacznika det(p, q, r) jest równy znakowi sinusa kąta nachylenia wektora $p \rightarrow r$ do wektora $p \rightarrow q$.

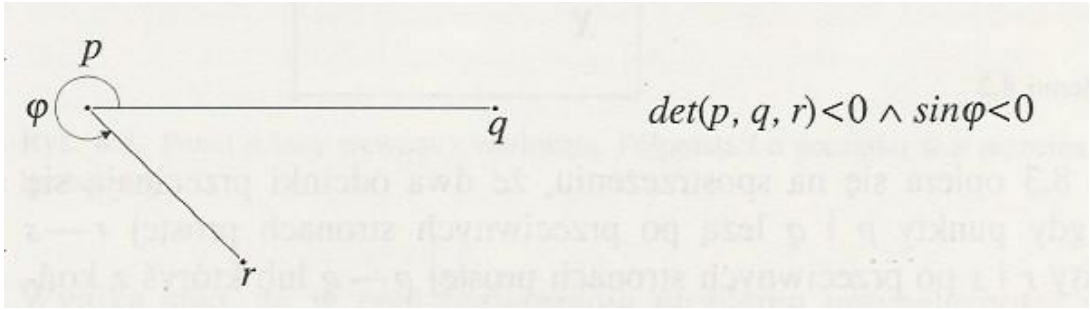
Podstawowe fakty z geometrii obliczeniowej na płaszczyźnie



Punkt r leży po **lewej stronie** wektora p→q.



Punkty p, q, r są **współliniowe**.



Punkt r leży po **prawej stronie** wektora p→q.

Elementarne algorytmy

Zadanie

Sprawdzić, czy punkty r i s leżą po tej samej stronie prostej p−q.

Odpowiedź

Wystarczy sprawdzić czy sgn(det(p, q, r))= sgn(det(p, q, s)).

Zadanie

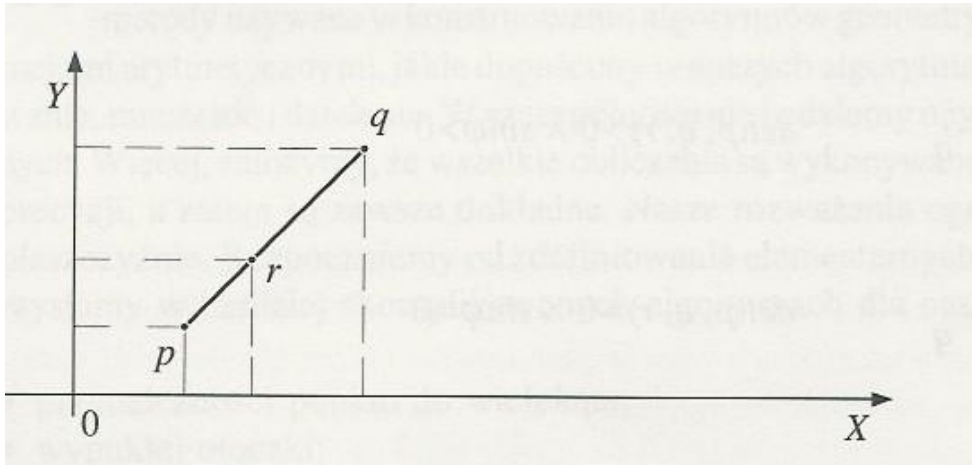
Zbadać, czy punkt r należy do odcinka p−q.

Odpowiedz

Trzeba zbadać, czy punkty p, q, r są współliniowe oraz czy

$\min(x_p,x_q) \le x_r \le \max(x_p, x_q)$

i $\min(y_p,y_q) \le y_r \le \max(y_p, y_q)$



Elementarne algorytmy

Zadanie

Kiedy dwa odcinki p−q i r−s przecinają się?

Odpowiedź

Gdy p i q leżą po przeciwnych stronach prostej r−s, a punkty r i s po przeciwnych stronach prostej p−q.

Zadanie

Kiedy punkt p należy do trójkąta?

Odpowiedź

W czasie stałym można sprawdzić, czy punkt p należy do brzegu trójkąta. Jeśli jest inaczej, P musi leżeć po tej samej stronie wszystkich boków trójkąta.

Wszystkie powyższe problemy są rozwiązywane w czasie stałym i tylko z użyciem operacji arytmetycznych.

Problem przynależności do wielokąta wypukłego

Dane:

n – liczba naturalna, n≥0

punkty $w_0, \ldots, w_{n-1}$ reprezentujące kolejne na obwodzie wierzchołki wielokąta wypukłego W,

punkt p.

Wynik: Odpowiedź na pytanie: czy p∈W?

Przypomnienie:

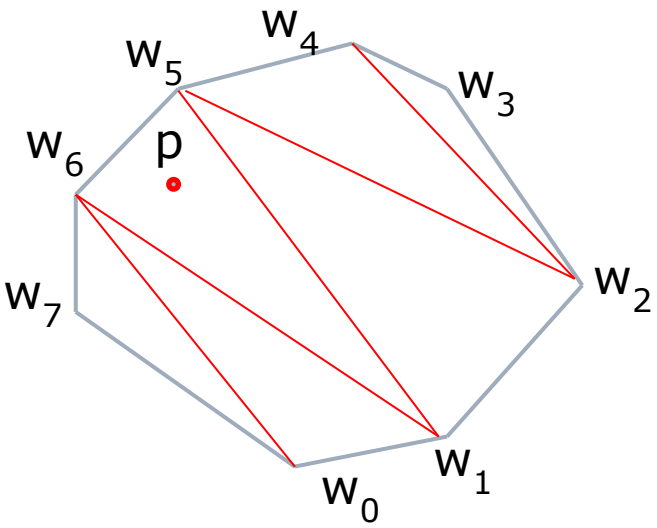
- Wielokąt jest wypukły wtedy i tylko wtedy, gdy każdy odcinek o końcach należących do wielokąta jest w nim całkowicie zawarty.
- Trinagulacja wielokąta – podział wielokąta na trójkąty nieprzecinającymi się przekątnymi.
- Dla wielokąta wypukłego o n wierzchołkach liczba trójkątów w triangulacji jest równa n-2, liczba sposobów podziału na trójkąty jest równa liczbie Catalana:

$$C_n = \frac{1}{n+1}\binom{2n}{n} = \frac{(2n)!}{(n+1)!n!}$$

Problem przynależności do wielokąta wypukłego

Algorytm I

Wykonaj dowolną trinagulację wielokąta W.



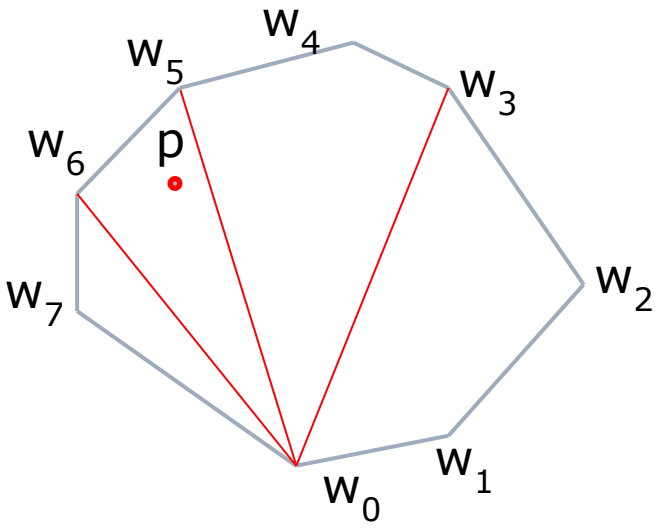
Dla każdego trójkąta sprawdź, czy p należy do niego.

W czasie stałym można sprawdzić przynależność punktu do trójkąta. Jest n-1 trójkątów, dlatego złożoność obliczeniowa tego algorytmu to O(n).

Problem przynależności do wielokąta wypukłego

Algorytm II

Posługujemy się metodą wyszukiwania binarnego.



Niech wielokąt ma wierzchołki numerowane od k do s. Są trzy możliwości:

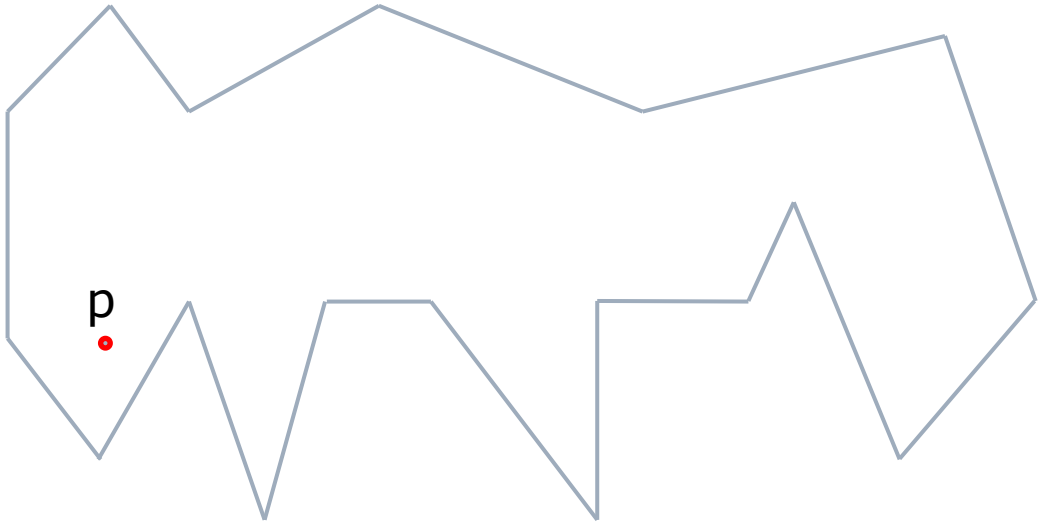
- p leży na odcinku $w_k - w_{(s+k)/2}$, wtedy badamy przynależność do odcinka
- p leży po lewej stronie, wtedy rekurencyjnie badamy wielokąt po lewej.
- p leży po prawej stronie, wtedy rekurencyjnie badamy wielokąt po prawej.

W czasie stałym można sprawdzić przynależność punktu do trójkąta.
Złożoność tego algorytmu to O(log n).

Problem przynależności punktu do dowolnego wielokąta

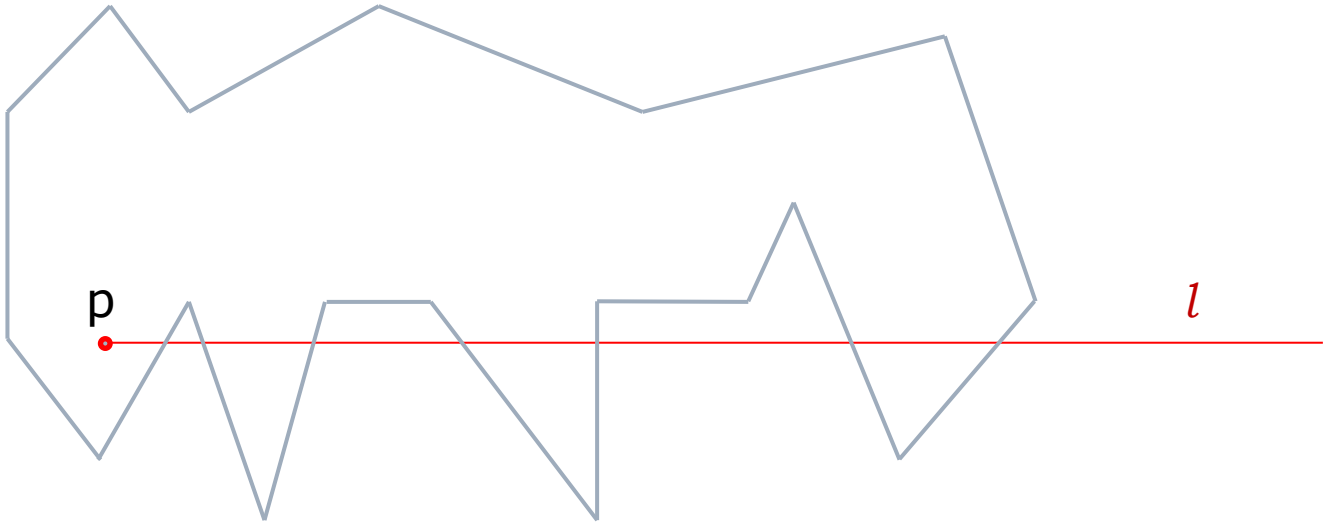
Dane:
n – liczba naturalna
n punktów w_0, …, w_{n-1} reprezentujących kolejne wierzchołki wielokąta W
punkt p.

Wynik: Odpowiedź na pytanie: czy $p \in W$?



Problem przynależności - algorytm

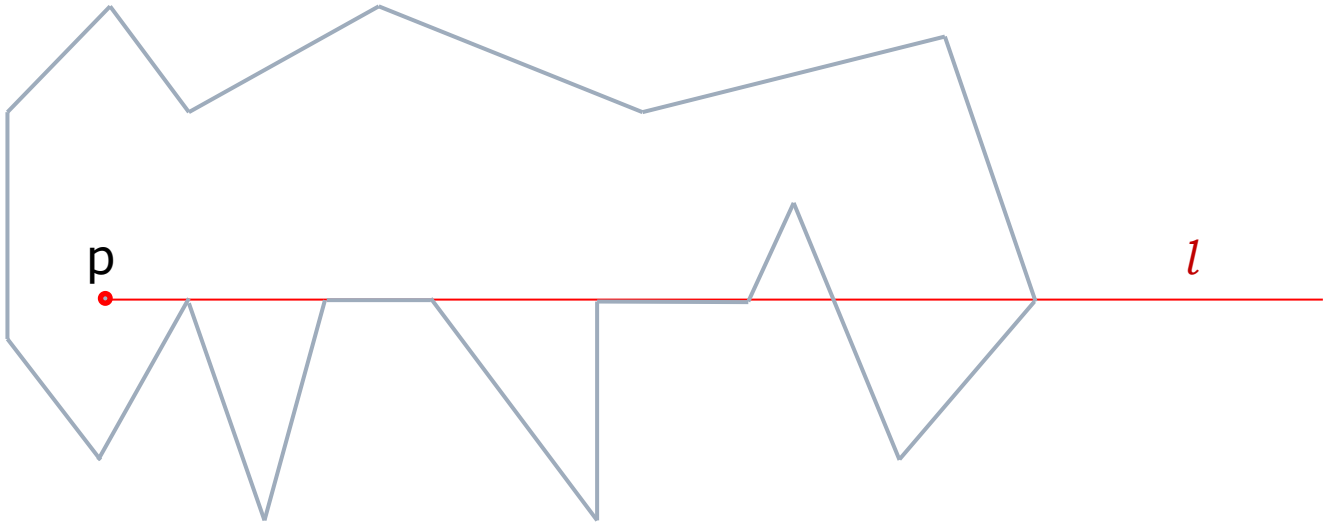
W czasie liniowym umiemy sprawdzić, czy p należy do któregoś z boków wielokąta. Jeśli jest inaczej, wyznaczamy półprostą l o początku w p.



W przypadku, gdy żaden z wierzchołków wielokąta nie leży na tej półprostej, punkt p leży wewnątrz wielokąta W wtedy i tylko wtedy, gdy przecina brzeg W nieparzystą liczbę razy.

Problem przynależności - obserwacja

Może zdarzyć się, że l przecina brzeg wielokąta w wierzchołkach lub zawiera krawędź wielokąta.

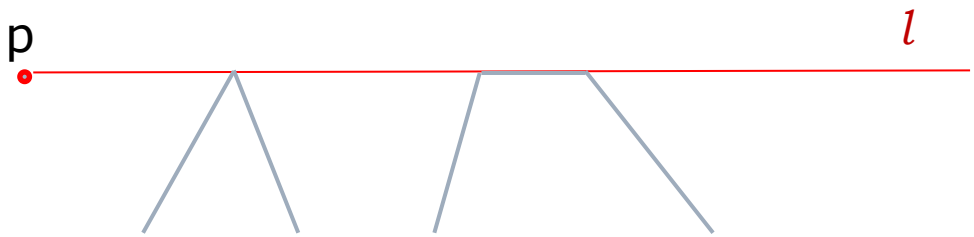


Obserwacja:

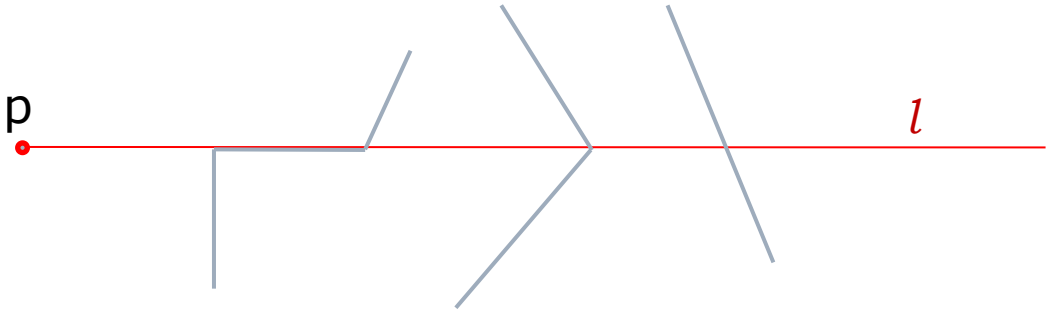
Każdą półprostą o początku w punkcie p można tak obrócić dookoła p o niewielki kąt, żeby otrzymać półprostą nie przecinającą W w wierzchołkach.

Problem przynależności - przypadki

Rozważamy dwa przypadki (i wszystkie analogiczne do nich):



Liczba punktów przecięcia nie zmienia się.



Liczba punktów przecięcia zwiększa się o jeden.

Złożoność O(n) – koszt obliczeń związany z każdym wierzchołkiem i bokiem jest stały, a mamy n wierzchołków.

Wypukła otoczka

Dane:

n – liczba punktów na płaszczyźnie

Zbiór n punktów S = $\{(x_i, y_i)$ dla $0 \le i, j \le n-1\}$, punkty zadane są przez współrzędne kartezjańskie.

Wynik:

Kolejne punkty wypukłej otoczki zbioru S (najmniejszy wielokąt wypukły zawierający S).

Wypukła otoczka

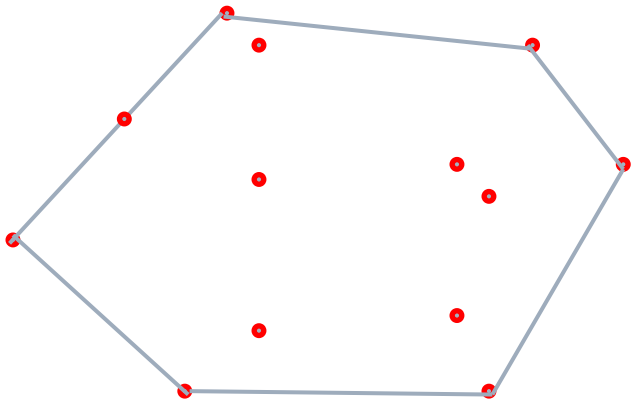
Dane:

n – liczba punktów na płaszczyźnie

Zbiór n punktów S = $\{x_i, y_i$ dla $0 \le i, j \le n-1\}$, punkty zadane są przez współrzędne kartezjańskie.

Wynik:

Kolejne punkty wypukłej otoczki zbioru S (najmniejszy wielokąt wypukły zawierający S).



Wypukła otoczka

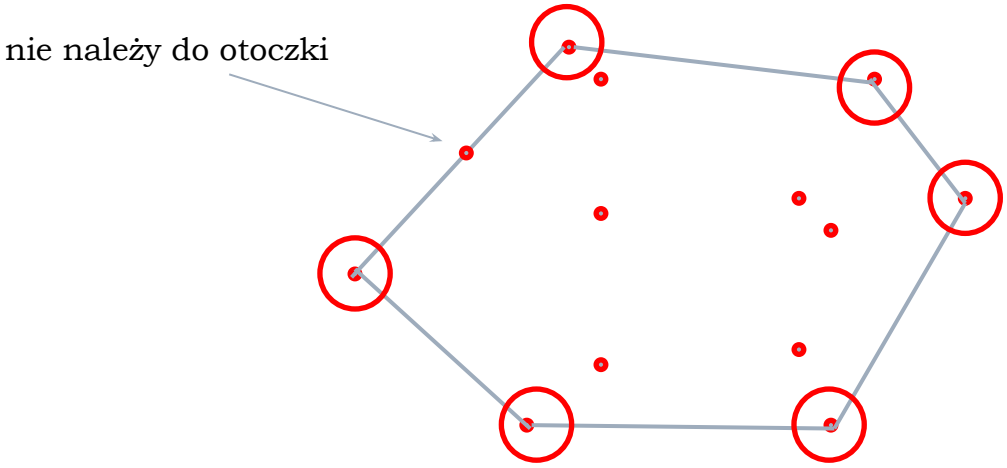
Dane:

n – liczba punktów na płaszczyźnie

Zbiór punktów S = { $p_i = (x_i, y_i)$ dla $0 \le i, j \le n-1$}, punkty zadane są przez współrzędne kartezjańskie.

Wynik:

Kolejne punkty wypukłej otoczki zbioru S (najmniejszy wielokąt wypukły zawierający S).



Wypukła otoczka – algorytm naiwny

Krok 1. Znaleźć wszystkie wierzchołki wypukłej otoczki zbioru S.

Krok 2. Uporządkować w kolejności występowania na obwodzie wypukłej otoczki.

Fakt

Punkt p nie jest wierzchołkiem wypukłej otoczki wtedy i tylko wtedy, gdy leży wewnątrz pewnego trójkąta o wierzchołkach z S, różnych od p, lub należy do odcinka łączącego dwa punkty z S różne od p.

Ponieważ dla n punktów mamy co najwyżej $\binom{n}{3}$ różnych trójkątów. Dlatego realizacja kroku1. zabiera czas $O(n^4)$.

Czy musimy sprawdzać wszystkie $\binom{n}{3}$ trójkąty?

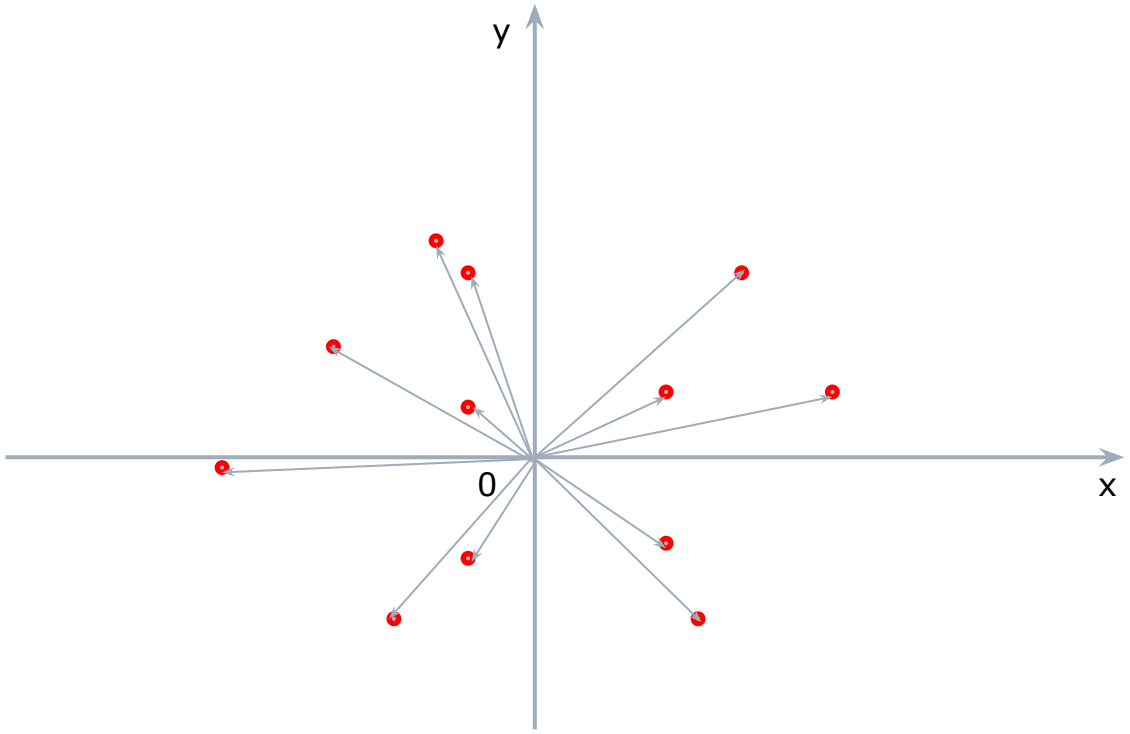
Można wybrać punkt, np. c - centroid, i uznać go za początek układu współrzędnych. Rozpatrujemy wtedy trójkąty o wierzchołkach c, p_i, p_{i+1}.

Algorytm ma wtedy złożoność $O(n^2)$.

Wypukła otoczka – sortowanie biegunowe

Układ współrzędnych polarnych – układ współrzędnych na płaszczyźnie wyznaczony przez pewien punkt O zwany biegunem oraz półprostą o początku w punkcie O zwaną osią biegunową.

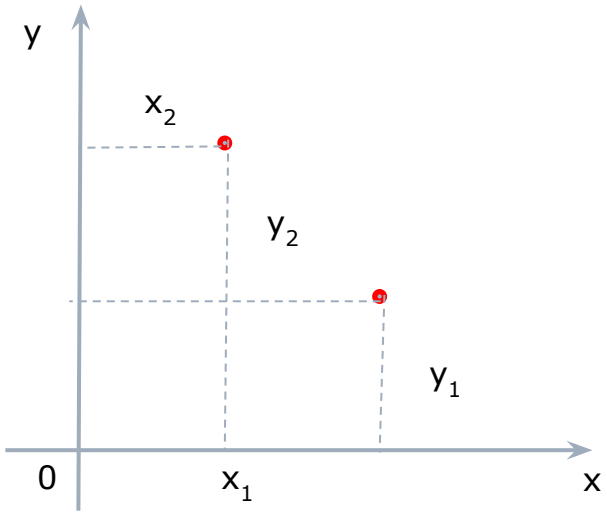
Sortowanie biegunowe – sortowanie względem współrzędnych biegunowych wektorów (promień i kąt nachylenia do osi OX).



Wypukła otoczka – sortowanie biegunowe

Posortujemy wierzchołki bez obliczania kątów. Niech alfa będzie funkcją określoną dla punktów płaszczyzny różnych od punktu 0, o wzorze:

$$alfa(p) = \begin{cases} \frac{y_p}{d_p}, & gdy\ \ x_p \geq 0\ i\ y_p \geq 0 \\ 2 - \frac{y_p}{d_p}, & gdy\ x_p \leq 0\ i\ y_p \geq 0 \\ 2 + \frac{|y_p|}{d_p}, & gdy\ x_p \leq 0\ i\ y_p \leq 0 \\ 4 - \frac{|y_p|}{d_p}, & gdy\ x_p \geq 0\ i\ y_p < 0 \end{cases}$$



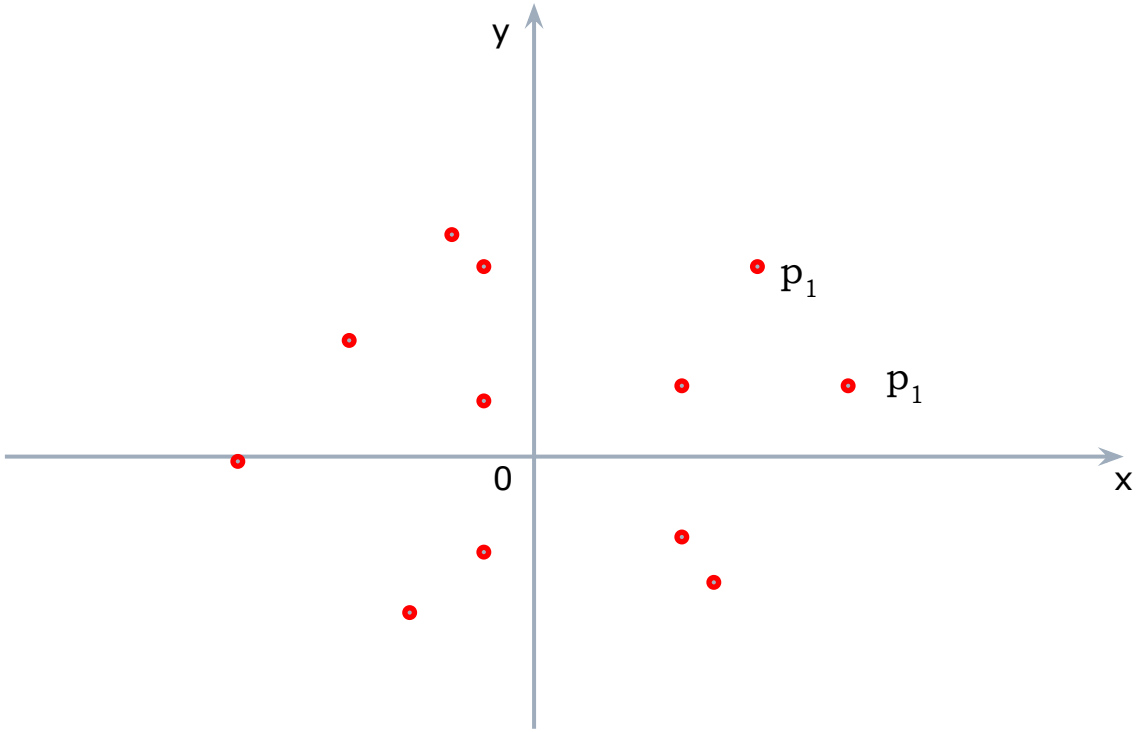
gdzie $d_p = |x_p| + |y_p|$.

Funkcja alfa pozwala wyznaczyć kolejności wierzchołków na obwodzie wielokąta wypukłego w czasie O(n logn).

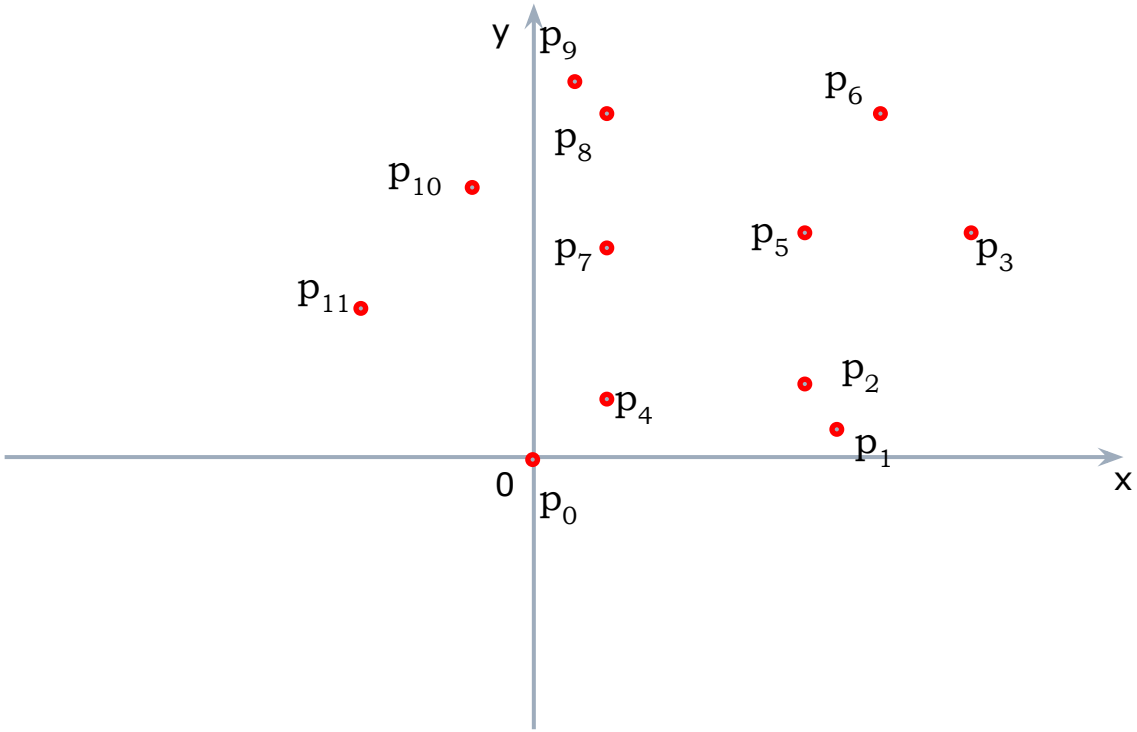
Algorytm Grahama

- W algorytmie Grahama używamy stosu, który zawiera kandydatów na wierzchołki otoczki.
- Każdy punkt z wejściowego zbioru jest raz wkładany na stos, natomiast punkty nie będące wierzchołkami otoczki są ze stosu zdejmowane.
- W momencie zakończenia działania algorytmu stos zawiera punkty występujące na otoczce w kolejności odwrotnej do ruchu wskazówek zegara.

Algorytm Grahama

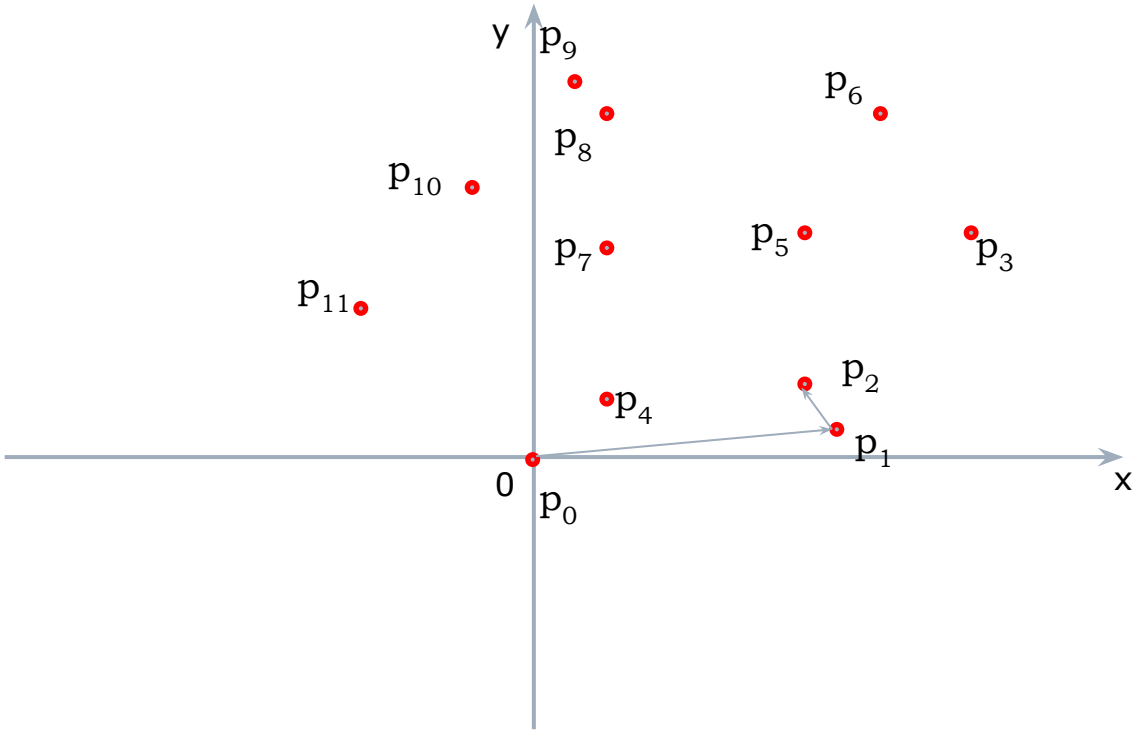


Algorytm Grahama



Algorytm Grahama

STOS



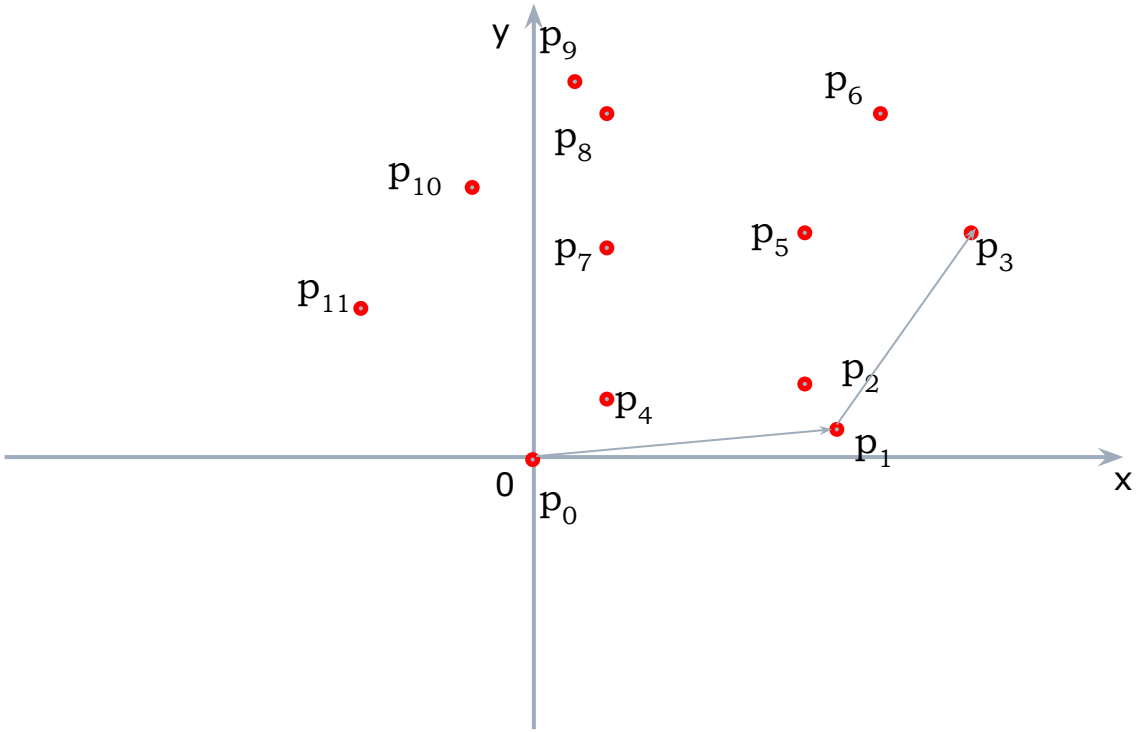
p_2

p_1

p_0

Algorytm Grahama

STOS

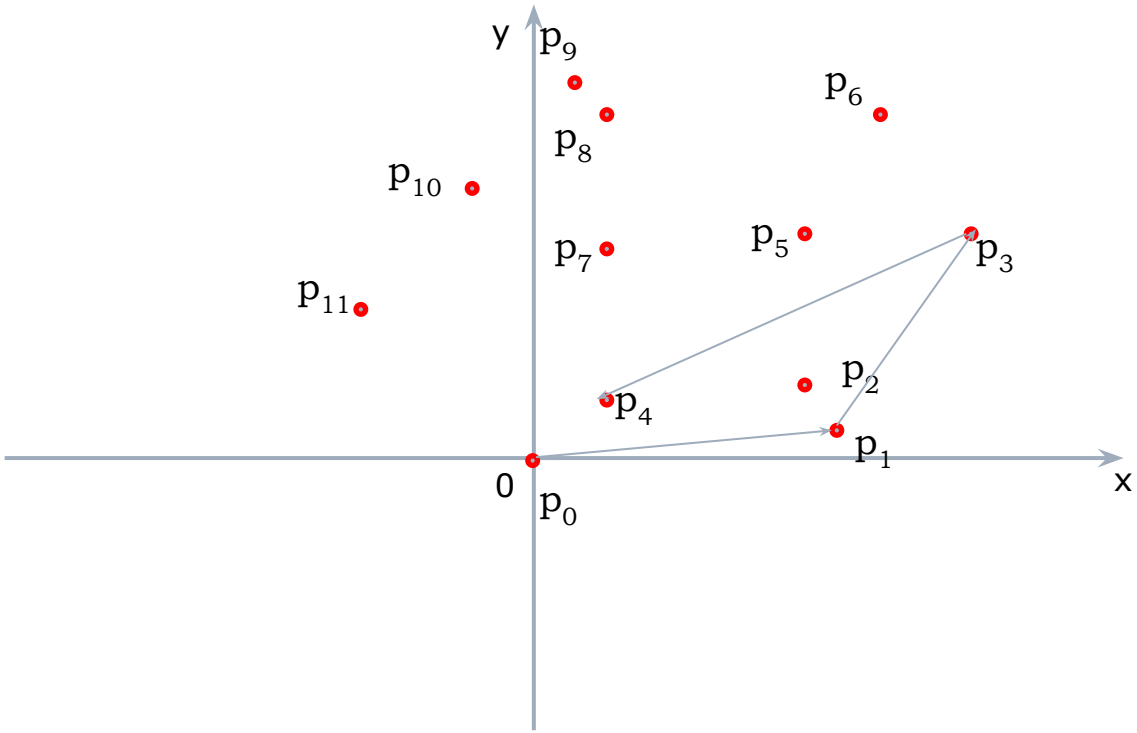


p_3

p_1

p_0

Algorytm Grahama



STOS

p_4

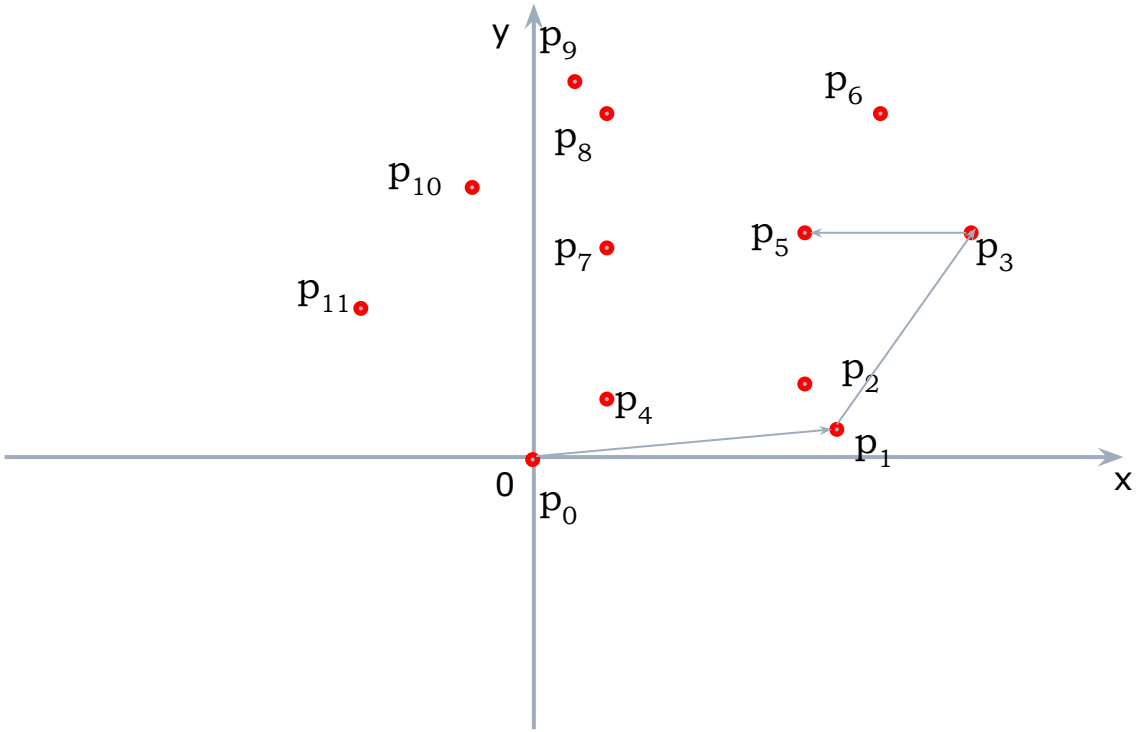
p_3

p_1

p_0

Algorytm Grahama

STOS



p_5

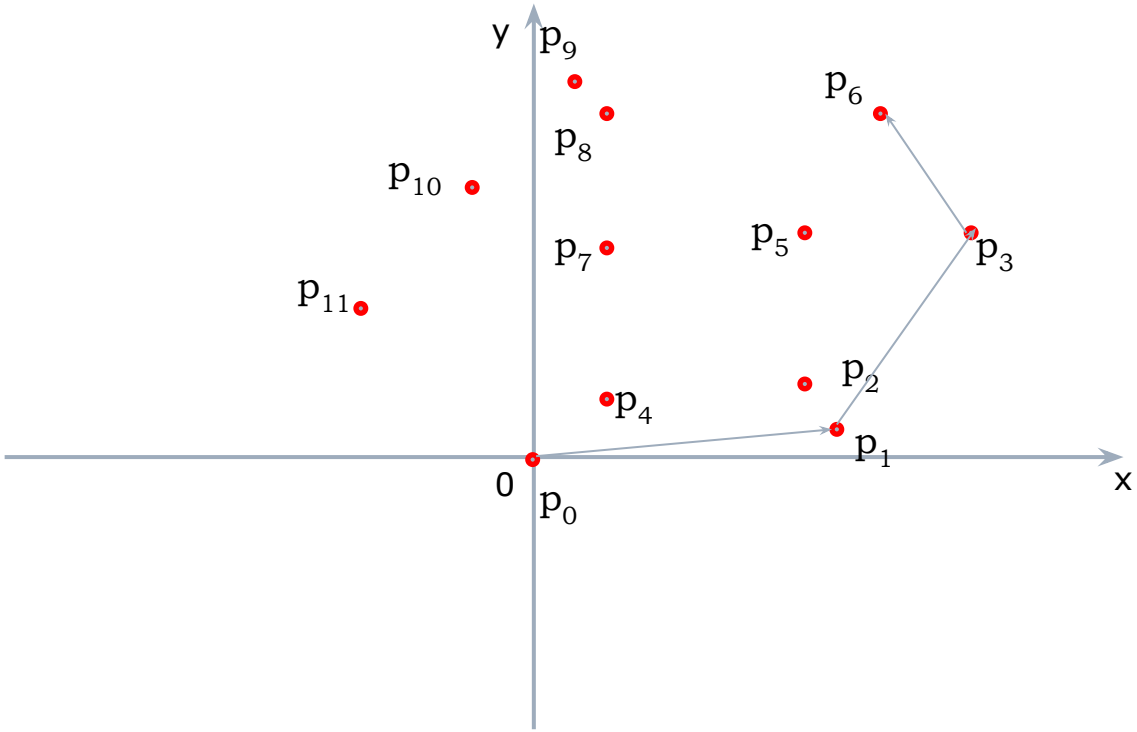
p_3

p_1

p_0

Algorytm Grahama

STOS



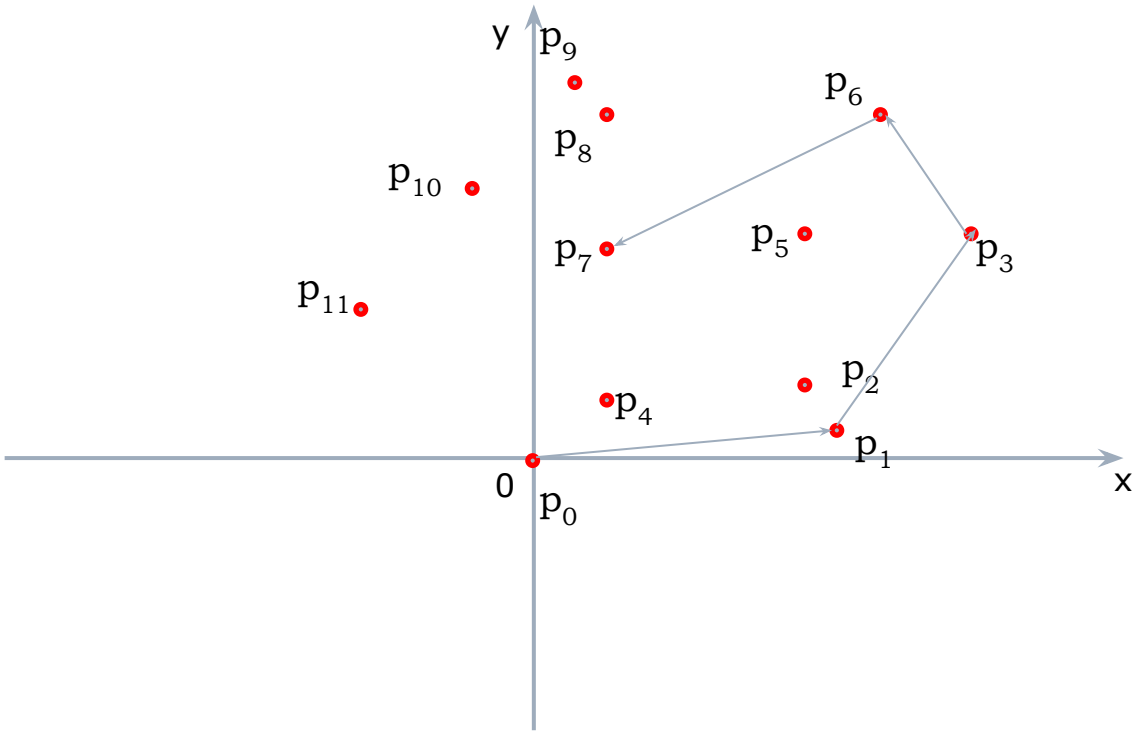
p_6

p_3

p_1

p_0

Algorytm Grahama



STOS

p_7

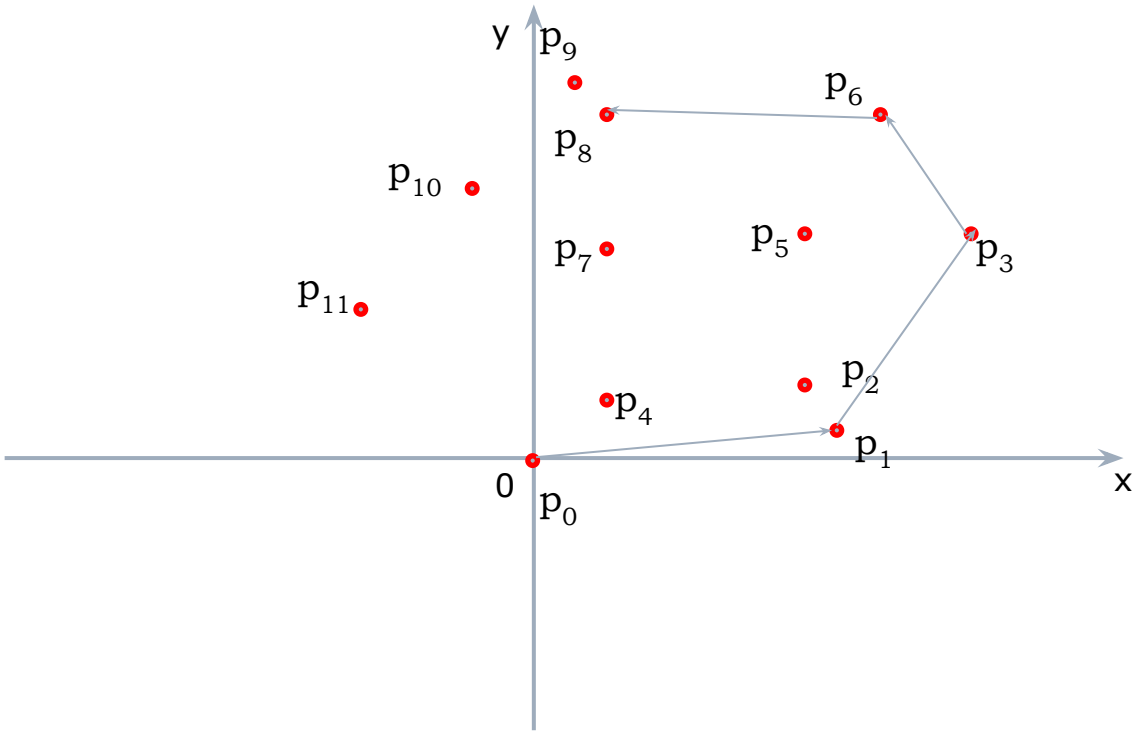
p_6

p_3

p_1

p_0

Algorytm Grahama



STOS

p_8

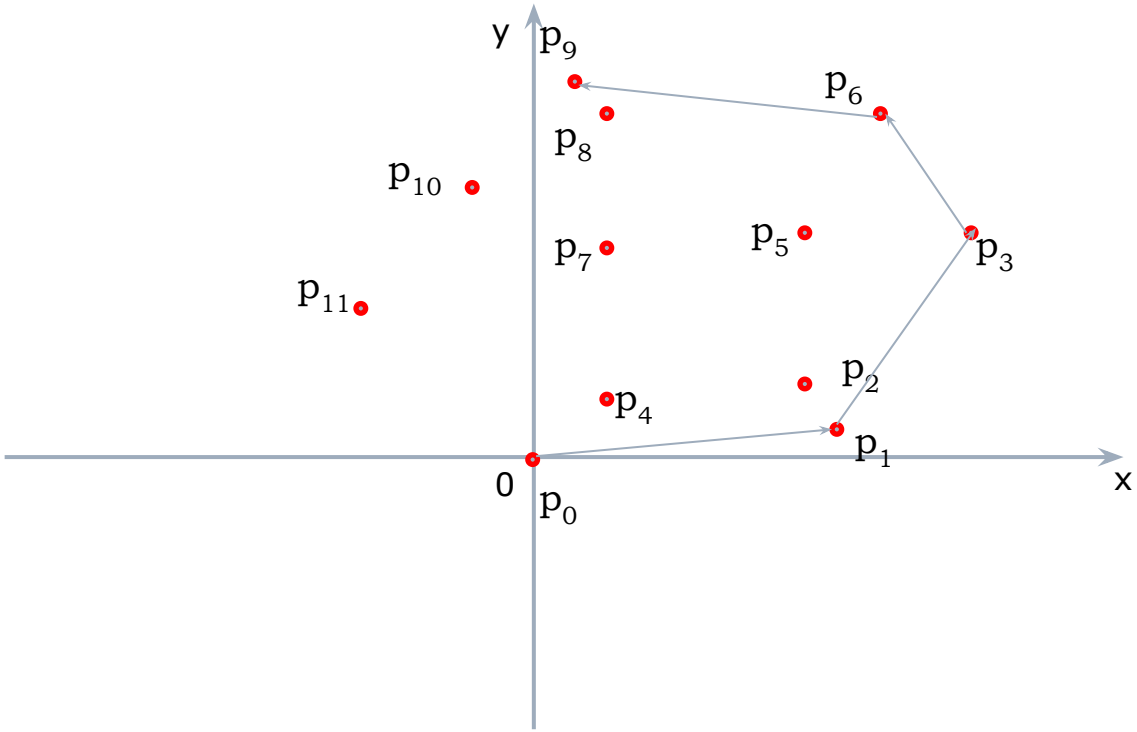
p_6

p_3

p_1

p_0

Algorytm Grahama



STOS

p_9

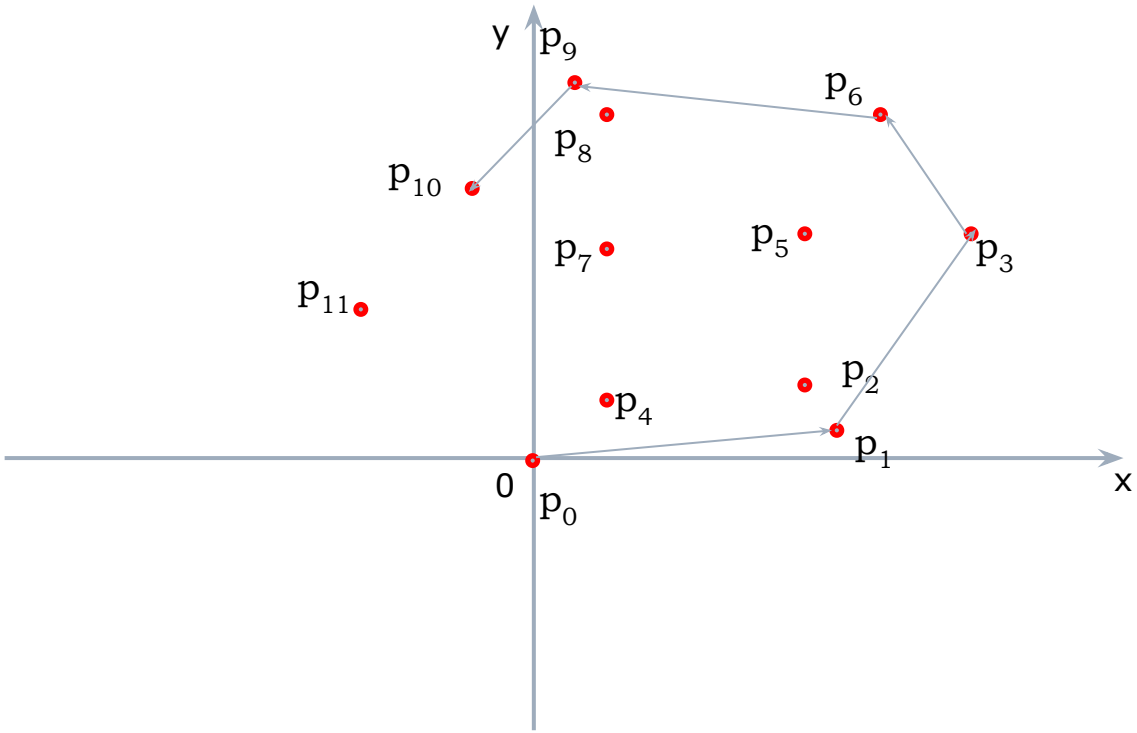
p_6

p_3

p_1

p_0

Algorytm Grahama



STOS

p_{10}

p_9

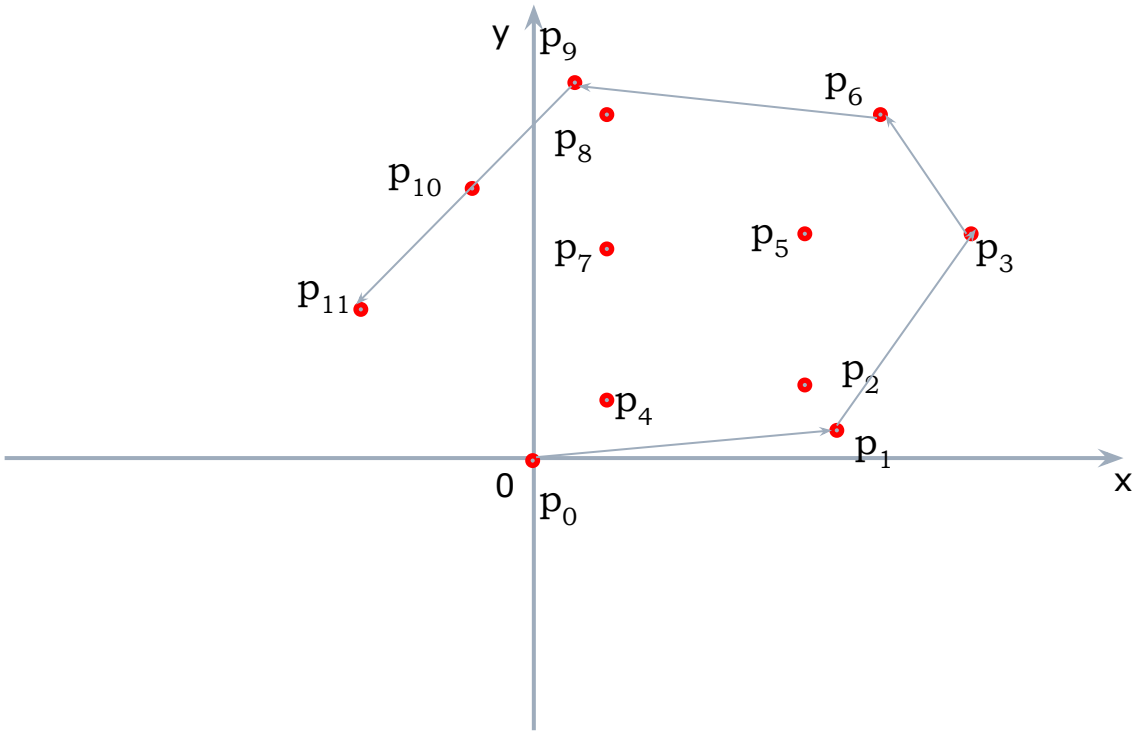
p_6

p_3

p_1

p_0

Algorytm Grahama



STOS

p_{11}

p_9

p_6

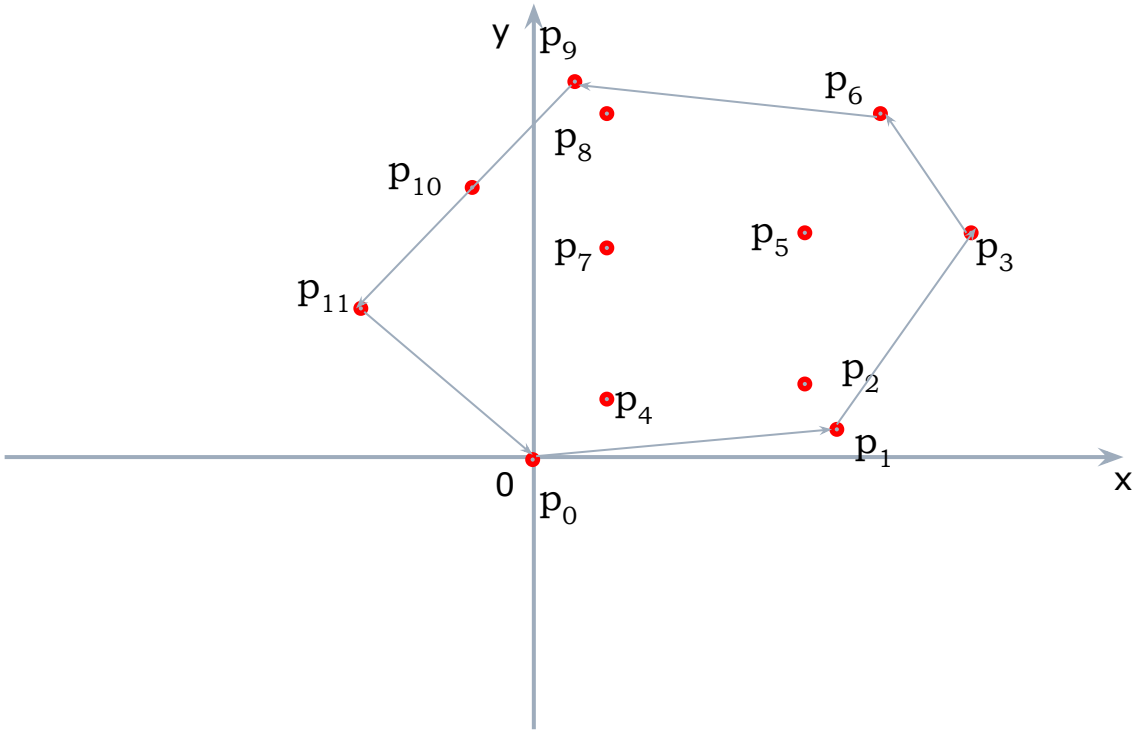
p_3

p_1

p_0

Algorytm Grahama

O złożoności decyduje sortowanie, który można wykonać w O(n logn). Pozostałe kroki są wykonywane w czasie liniowym.



STOS

p_{11}

p_9

p_6

p_3

p_1

p_0

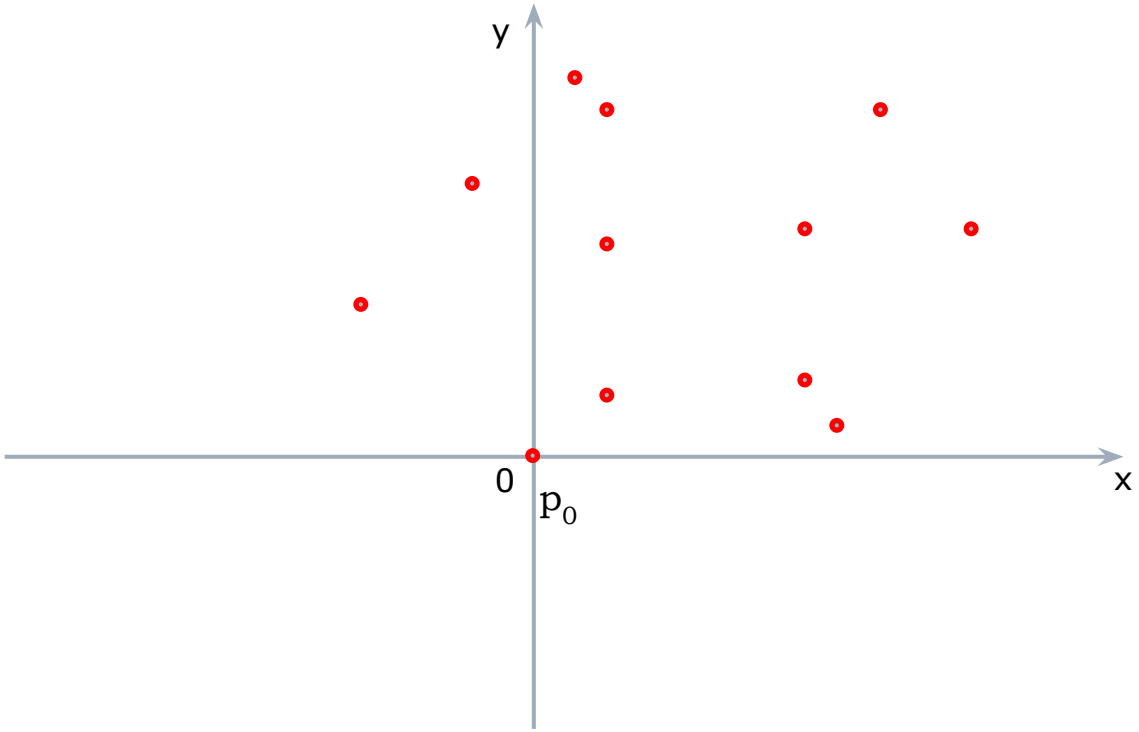
Algorytm Grahama

1. Niech p_0 będzie punktem o najmniejszej współrzędnej y, jeśli takich jest kilka, bierzemy ten o najmniejszej współrzędnej x. Punkt ten uznajemy za początek układu współrzędnych.
2. Sortujemy pozostałe punkty niemalejąco względem współrzędnych polarnych.
3. Włóż na stos S punkty p0, p1, p2.
4. Dla punktów od 3 do n-1:
   a) tak długo, jak kolejny punkt p_i jest na prawo od wektora utworzonego z przedostatniego i ostatniego wierzchołka na stosie zamień na stosie ostatni element na pi
   b) włóż pi na stos.
5. Wynikiem są punkty na stosie.

O złożoności decyduje punkt 2, który można wykonać w O(n logn). Kroki 1. i 4. (zasada magazynu) są wykonywane w czasie liniowym.

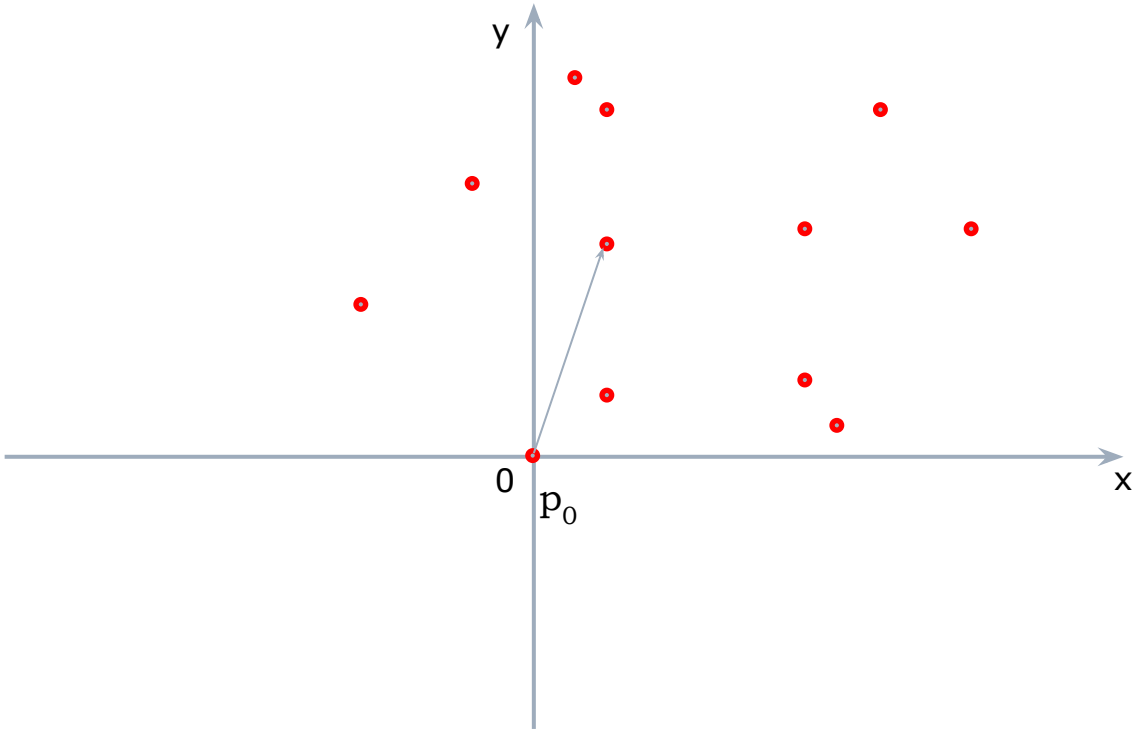
Algorytm Jarvisa

Niech p_0 będzie punktem o najmniejszej współrzędnej y, jeśli takich jest kilka, bierzemy ten o najmniejszej współrzędnej x.



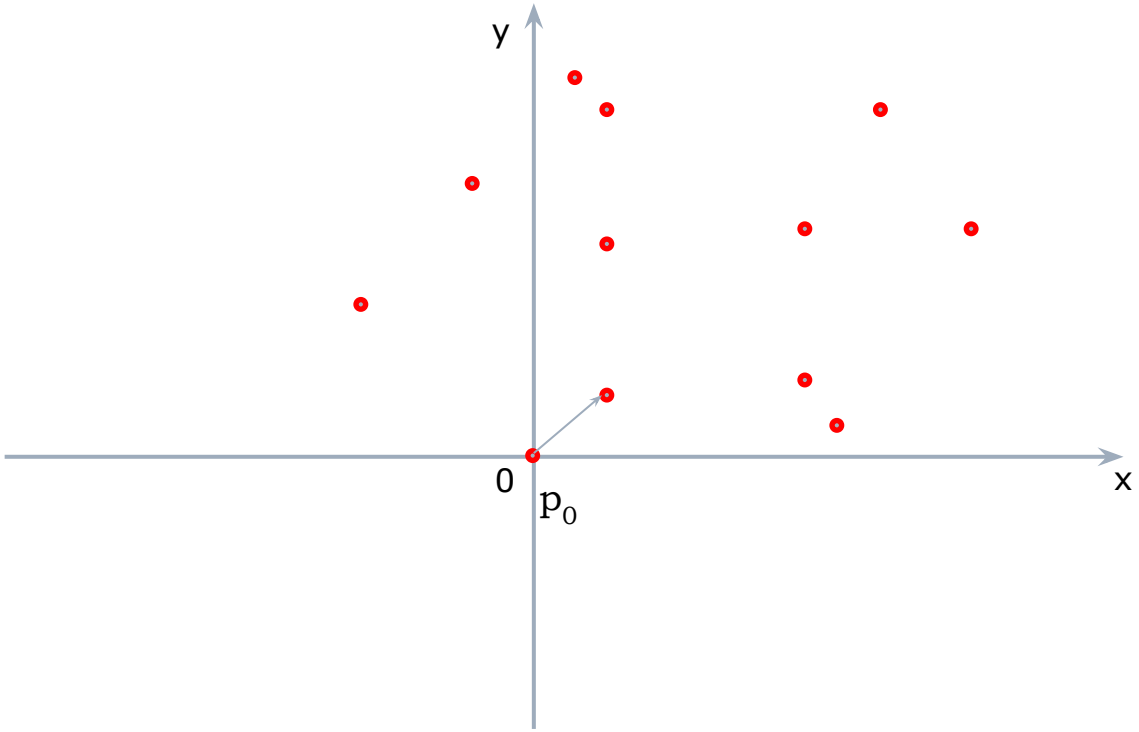
Algorytm Jarvisa

Weź dowolny punkt różny od p_0



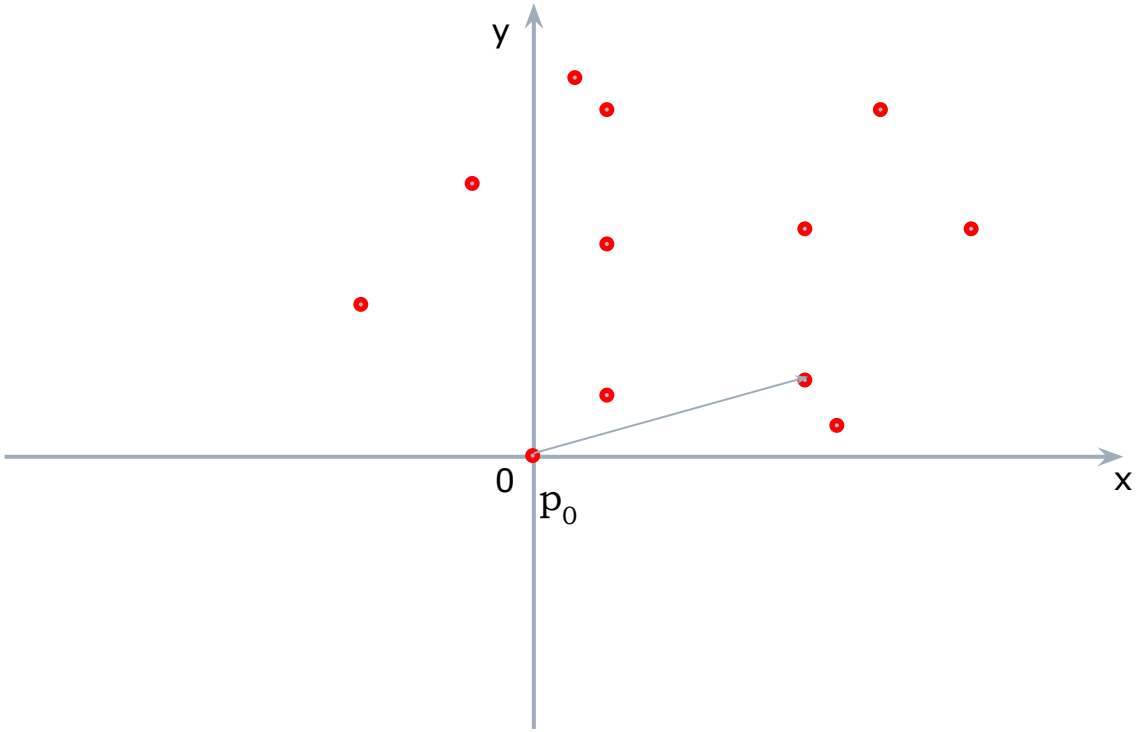
Algorytm Jarvisa

Powtarzaj dla punktów, które jeszcze nie są w otoczce: Dla punktów pi, jeśli pi leży na prawo od wektora, weź go jako koniec wektora.



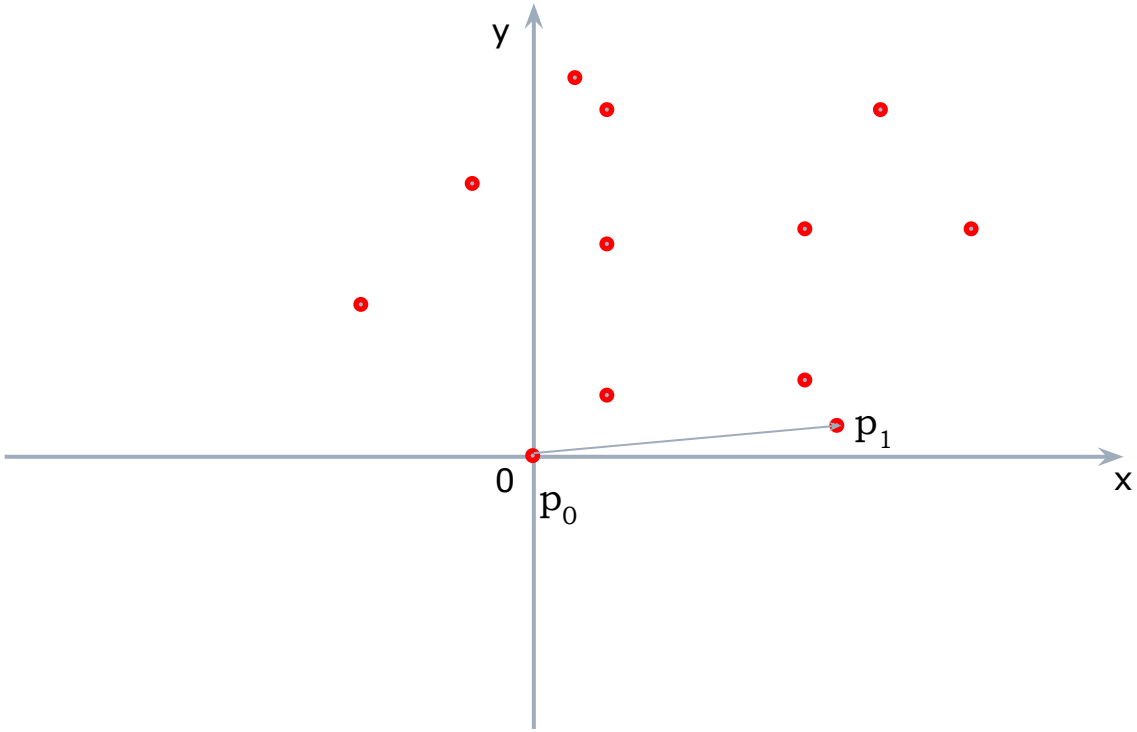
Algorytm Jarvisa

Powtarzaj dla punktów, które jeszcze nie są w otoczce: Dla punktów pi, jeśli pi leży na prawo od wektora, weź go jako koniec wektora.



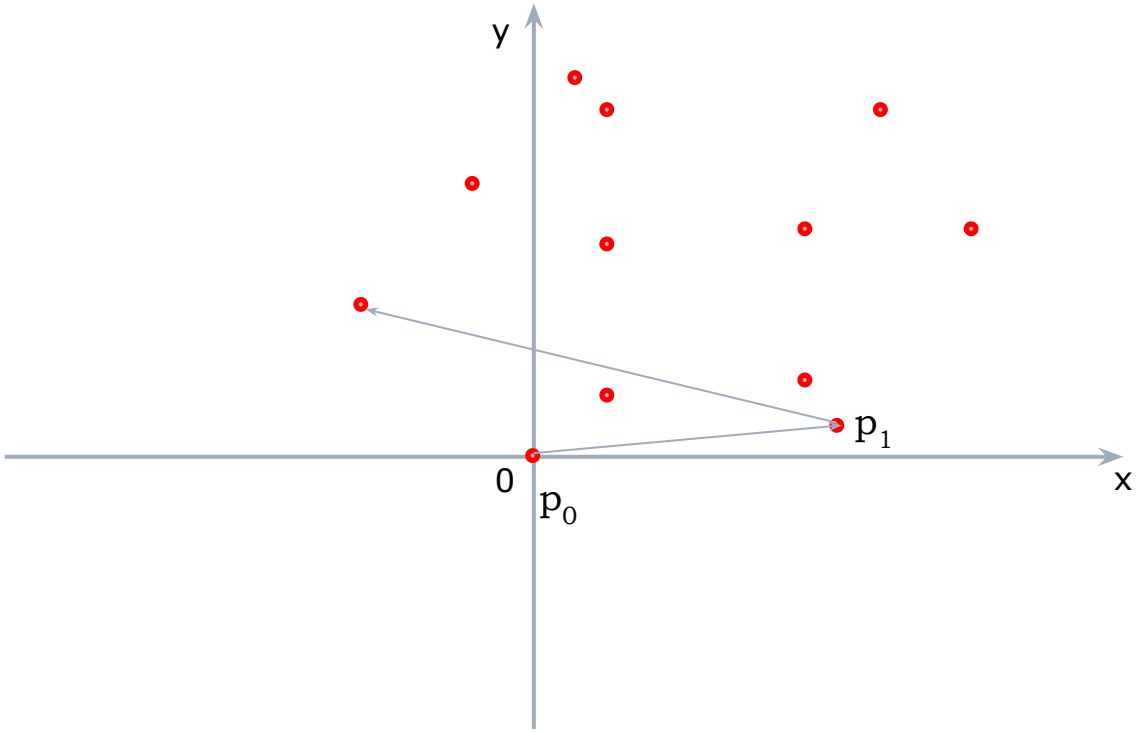
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



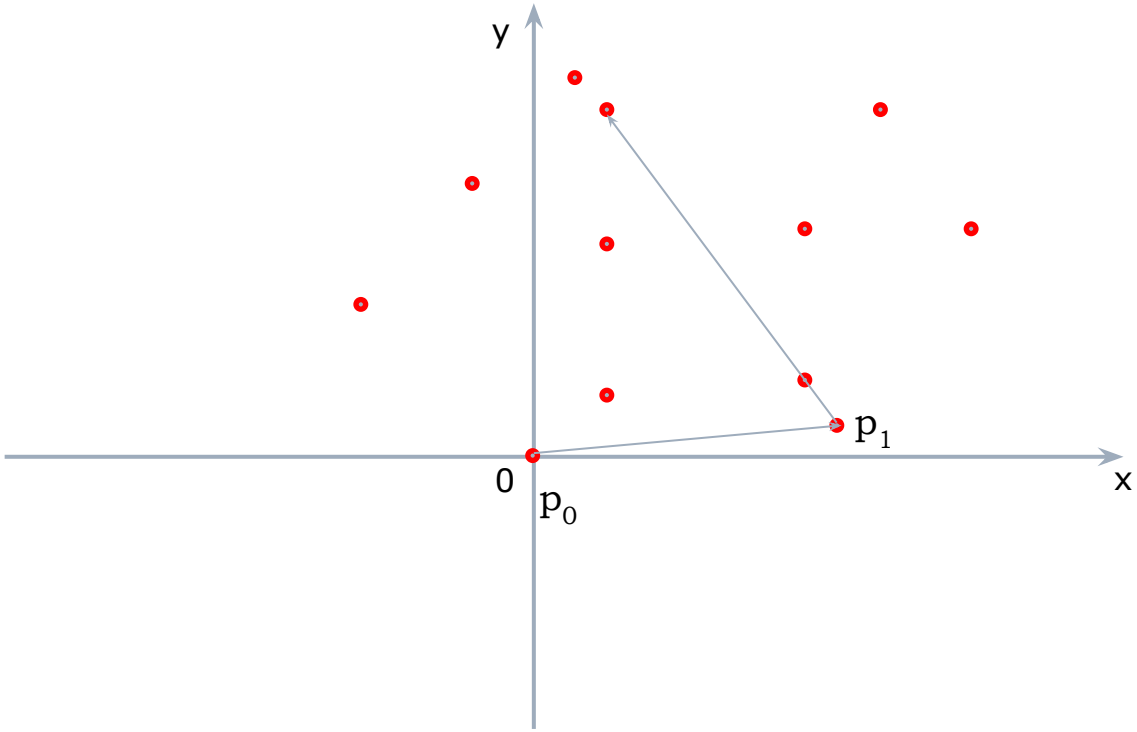
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



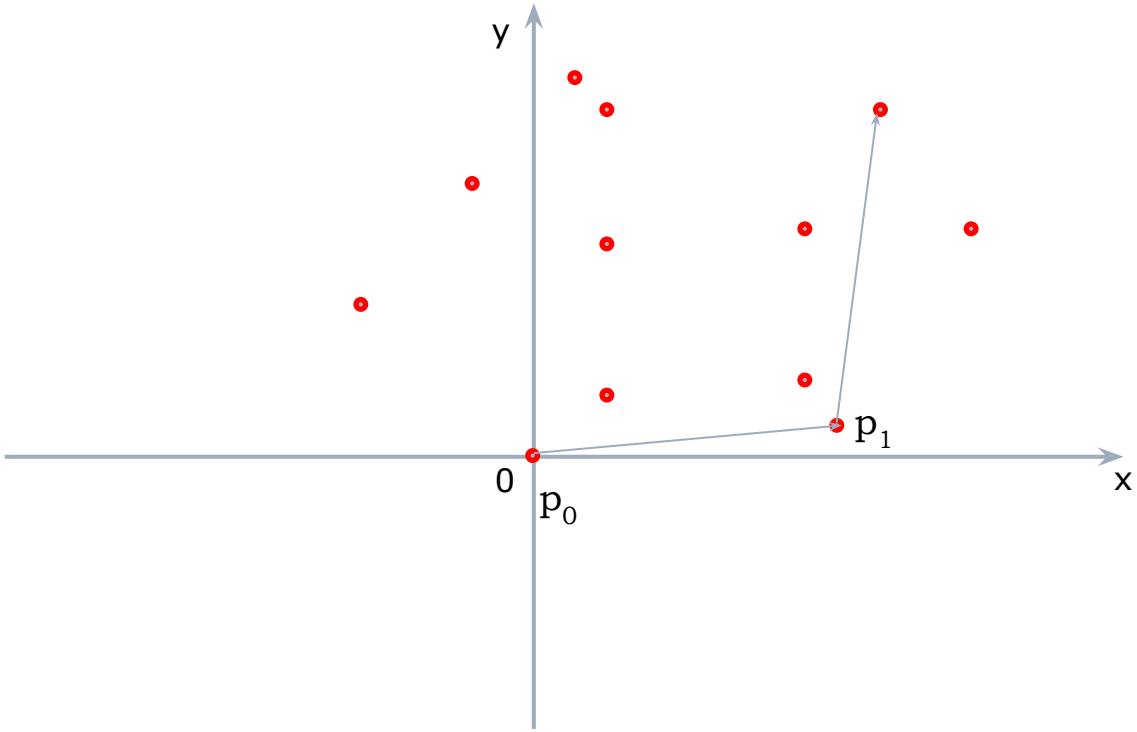
AlgorytmJarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



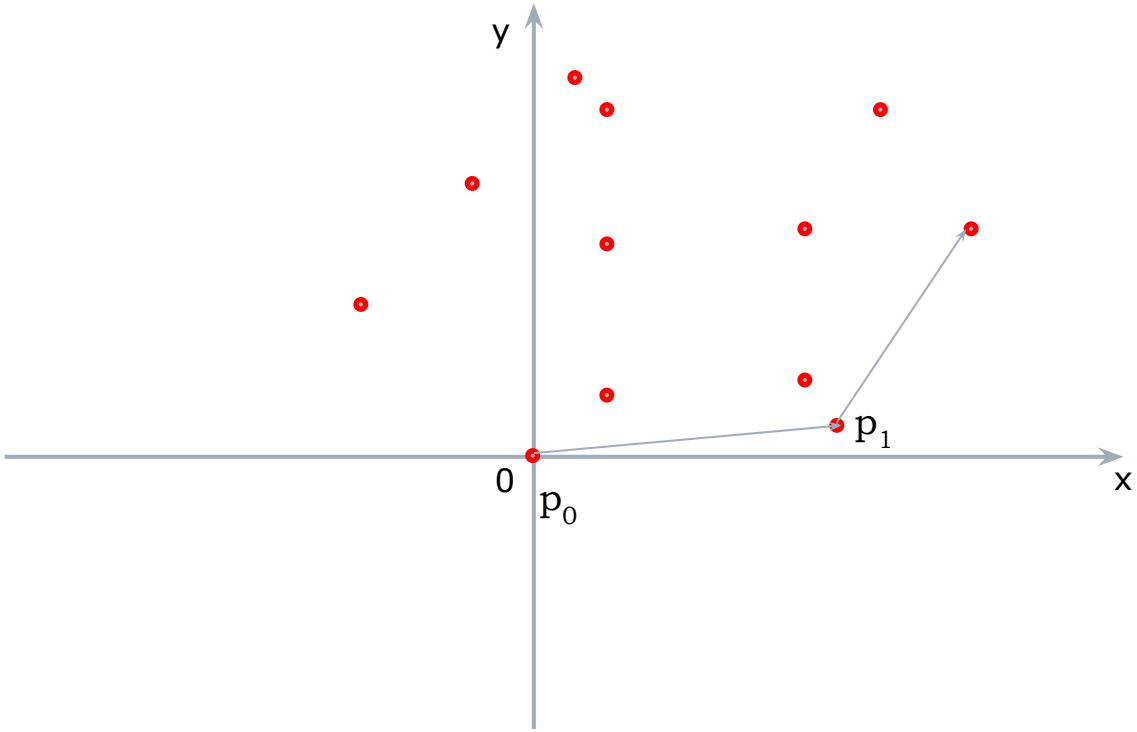
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



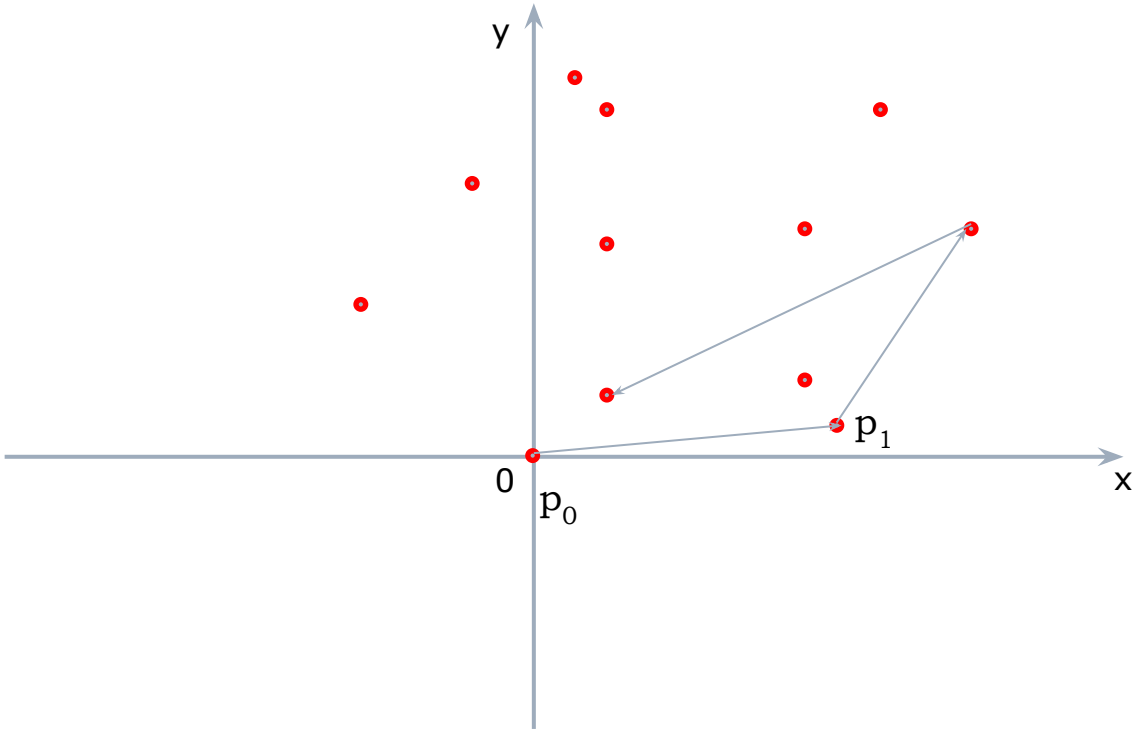
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



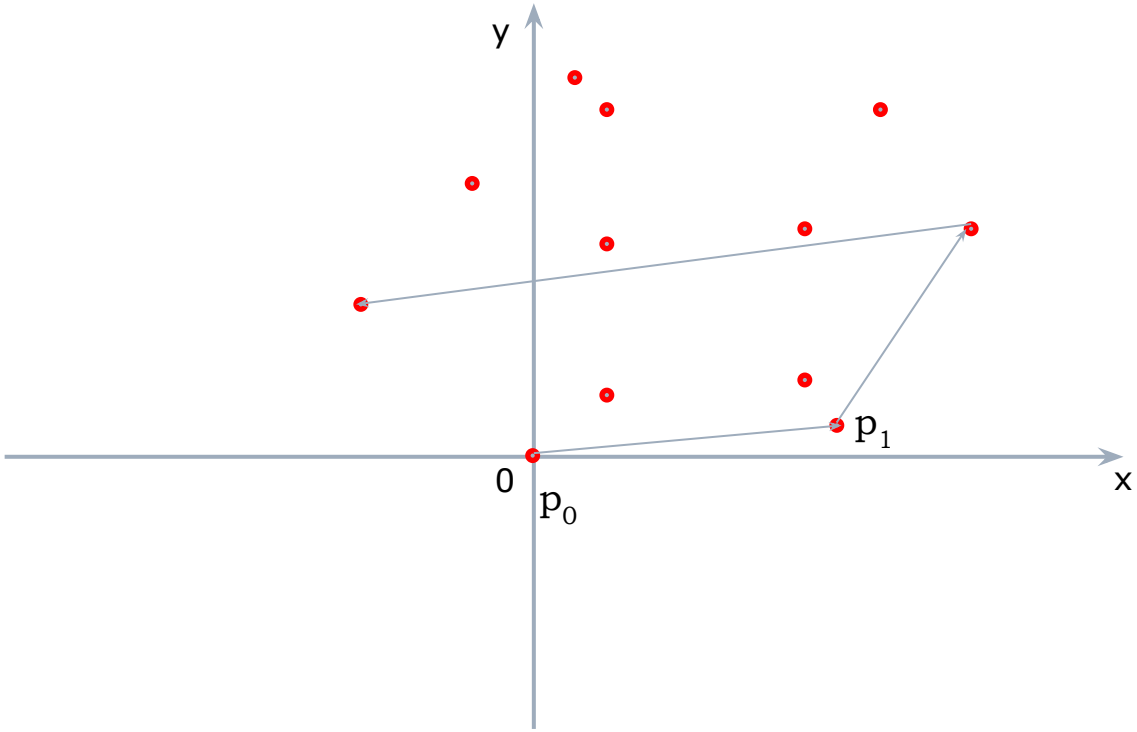
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



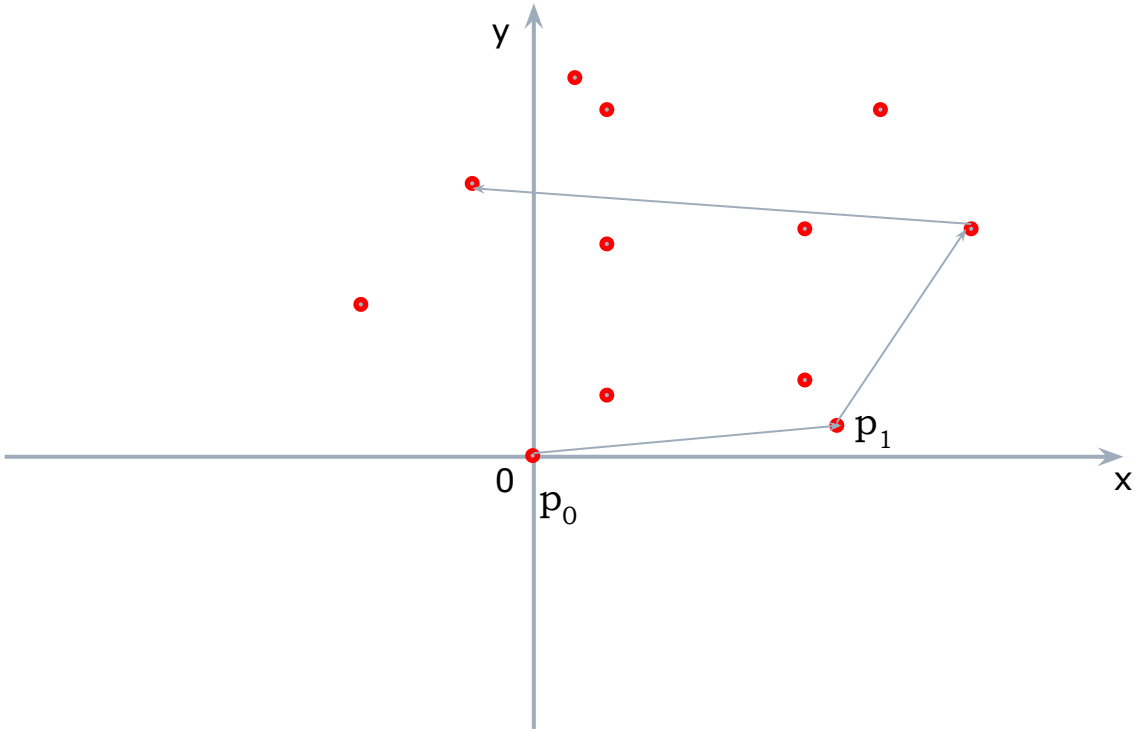
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



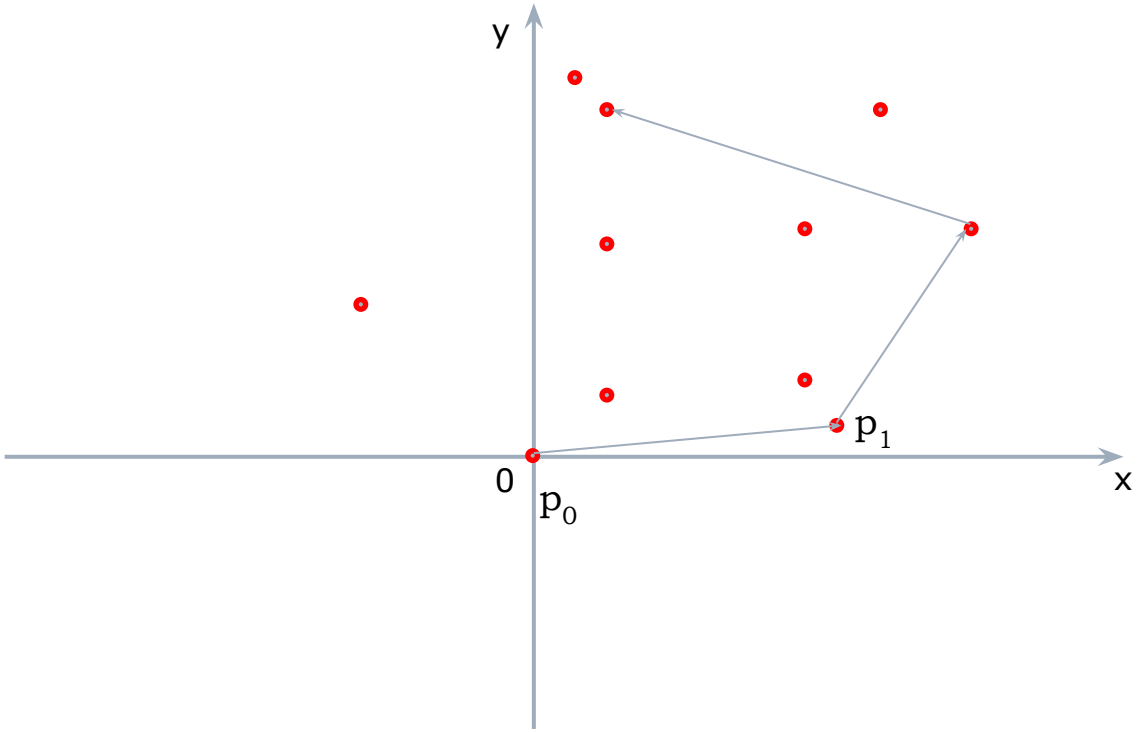
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



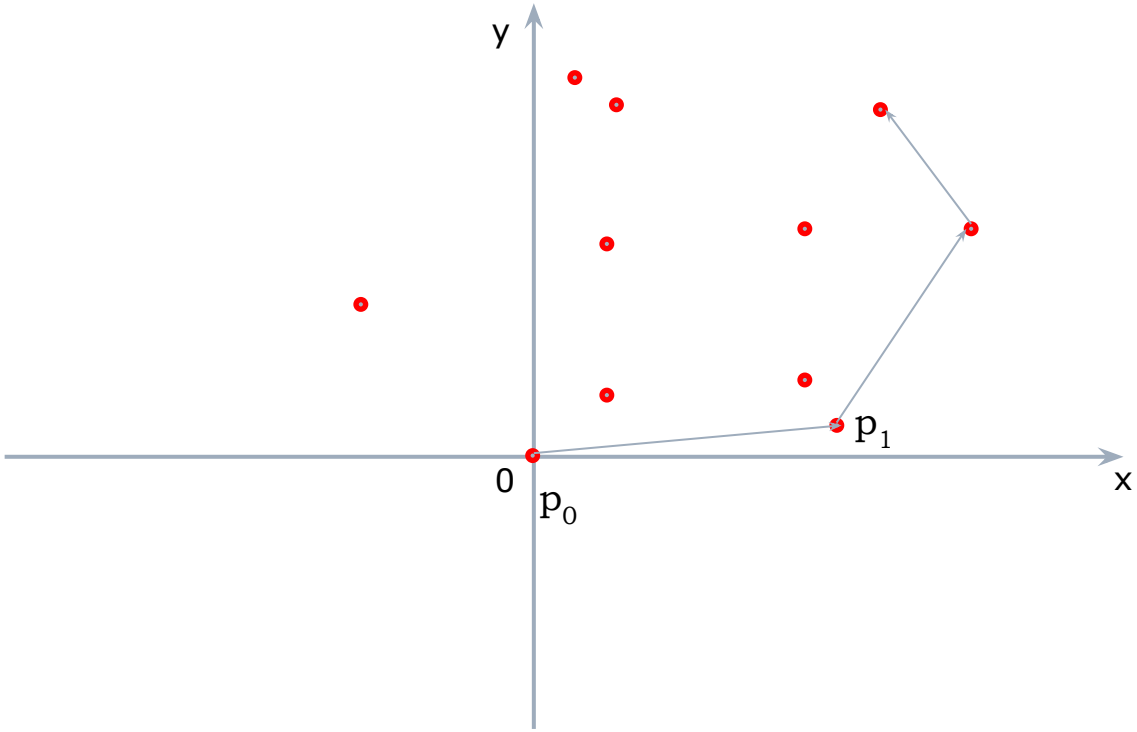
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



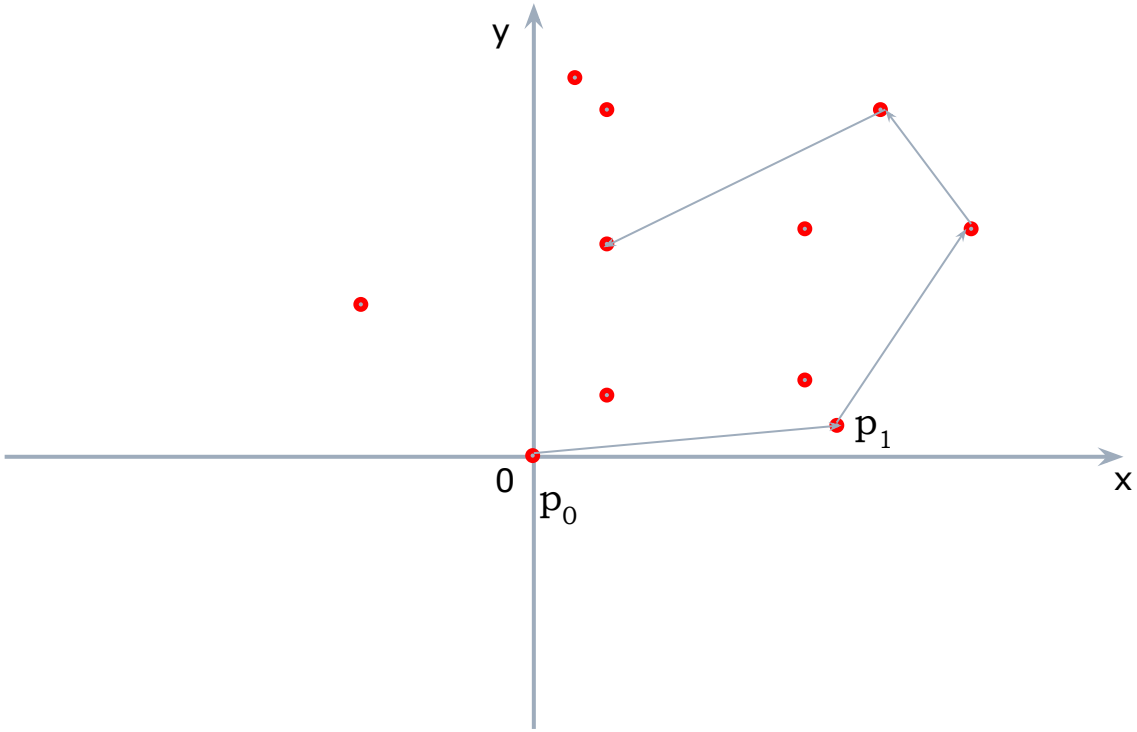
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



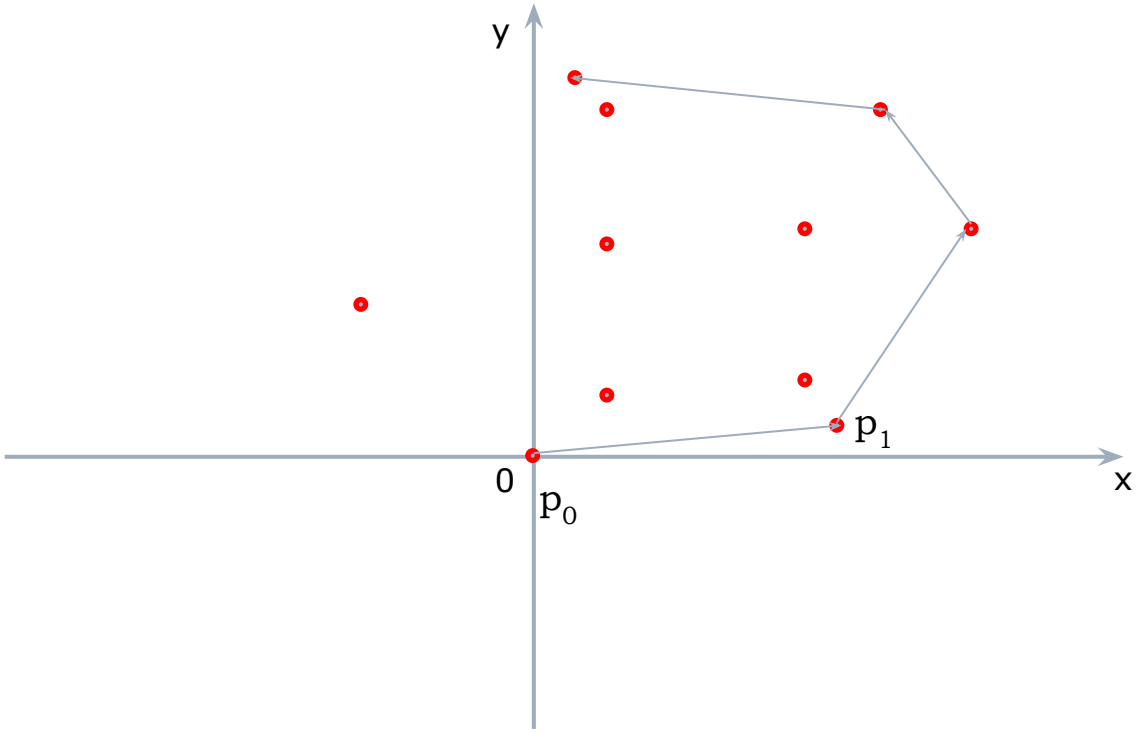
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



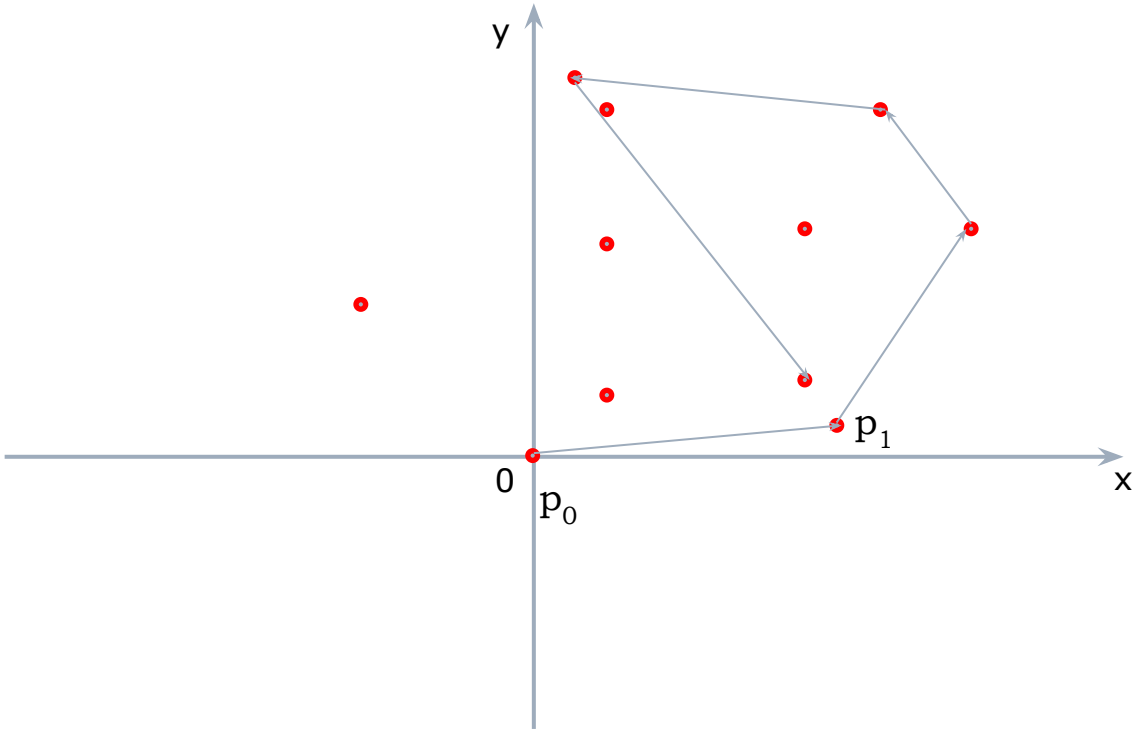
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



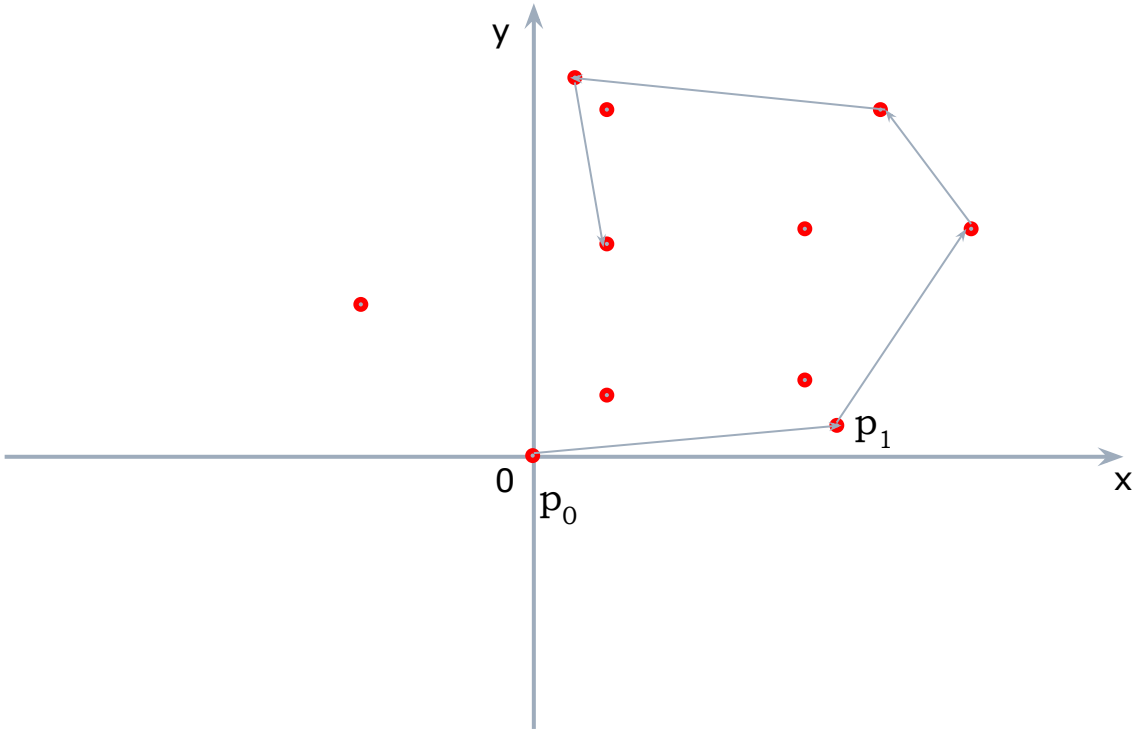
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



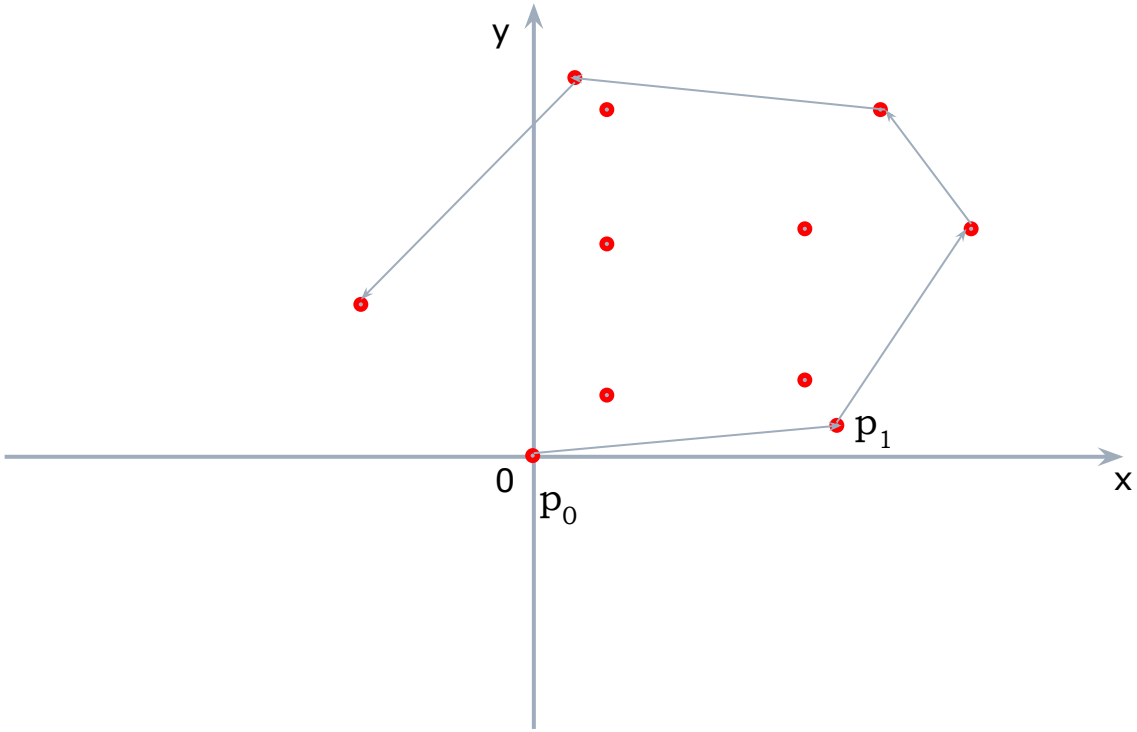
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.



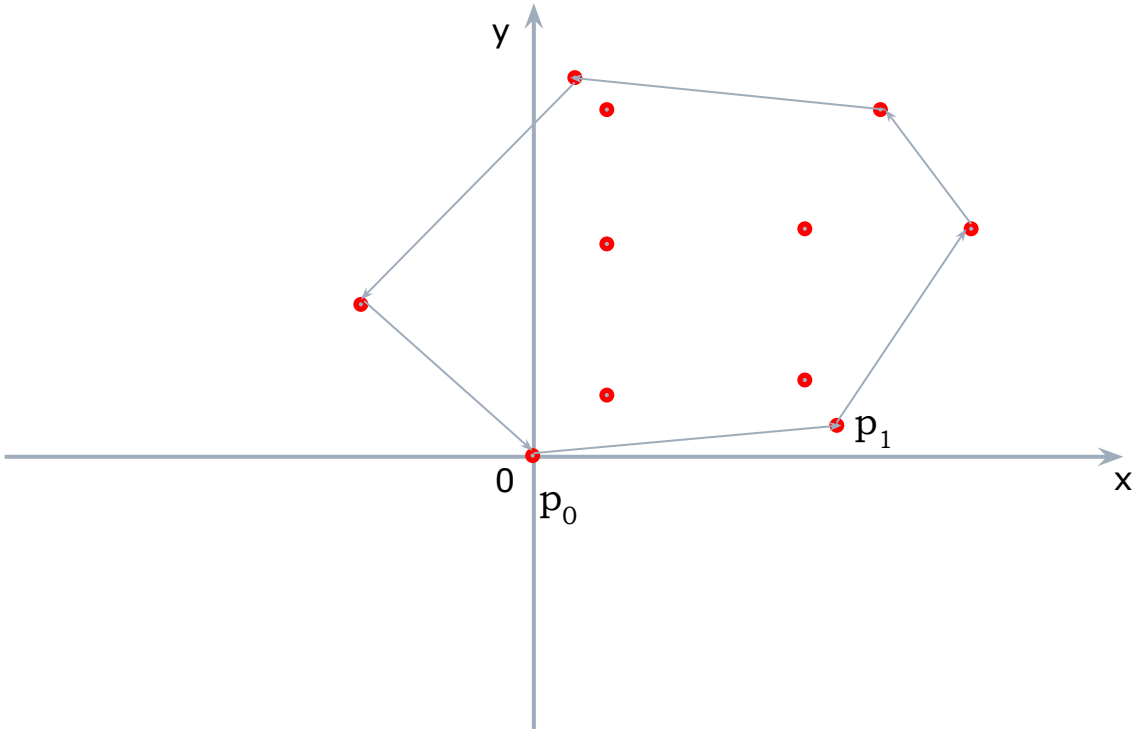
Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę.

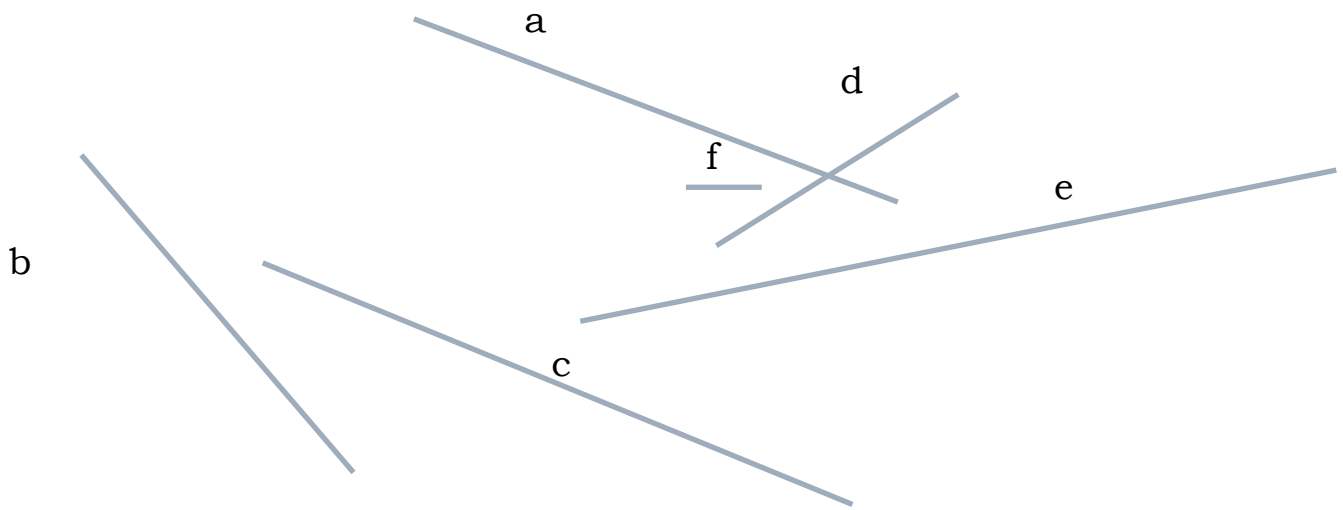


Algorytm Jarvisa

Powtarzaj, aż znajdziesz całą otoczkę. Złożoność obliczeniowa algorytmu Grahama to 0(nk), gdzie k jest liczba punktów na otoczce.



Przecinanie się par odcinków



Dane:

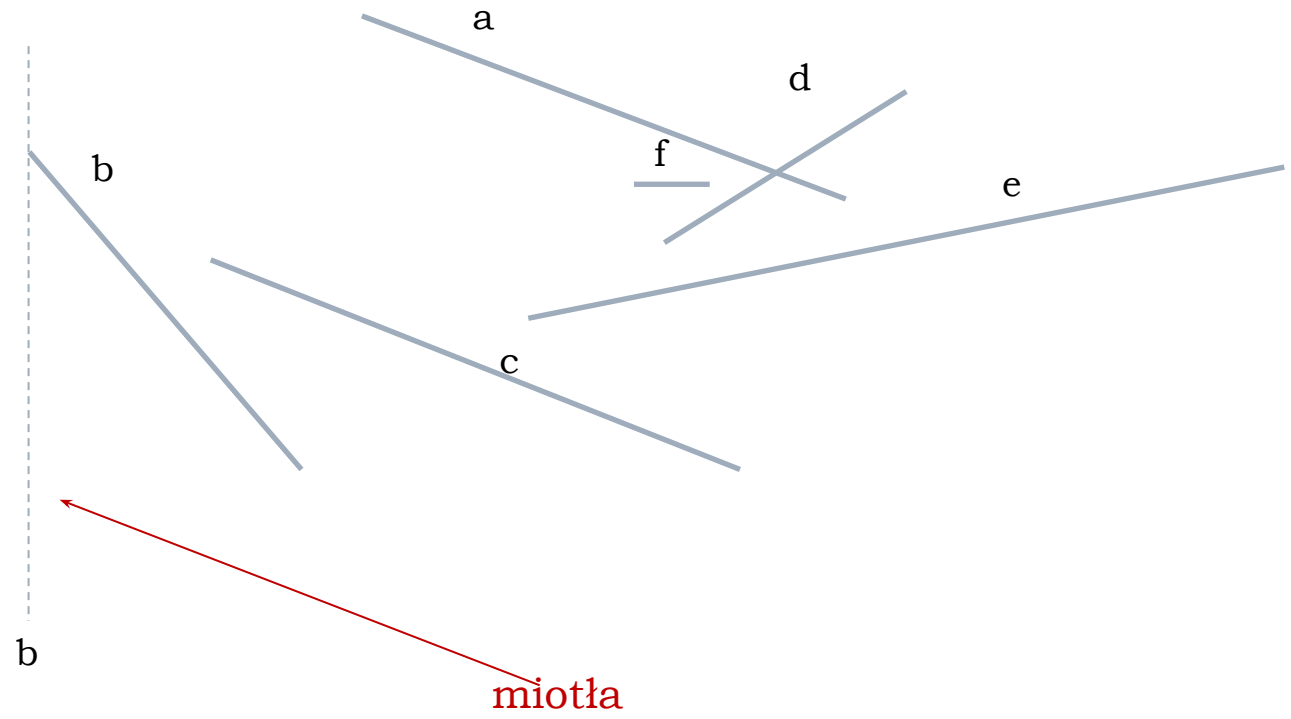
n – liczba odcinków

n par punktów tworzących odcinki, nie rozpatrujemy odcinków równoległych do osi współrzędnych, każdy odcinek przecina się z innym co najwyżej raz

Wynik:

Odpowiedź na pytanie: czy wśród odcinków są co najmniej dwa przecinające się?

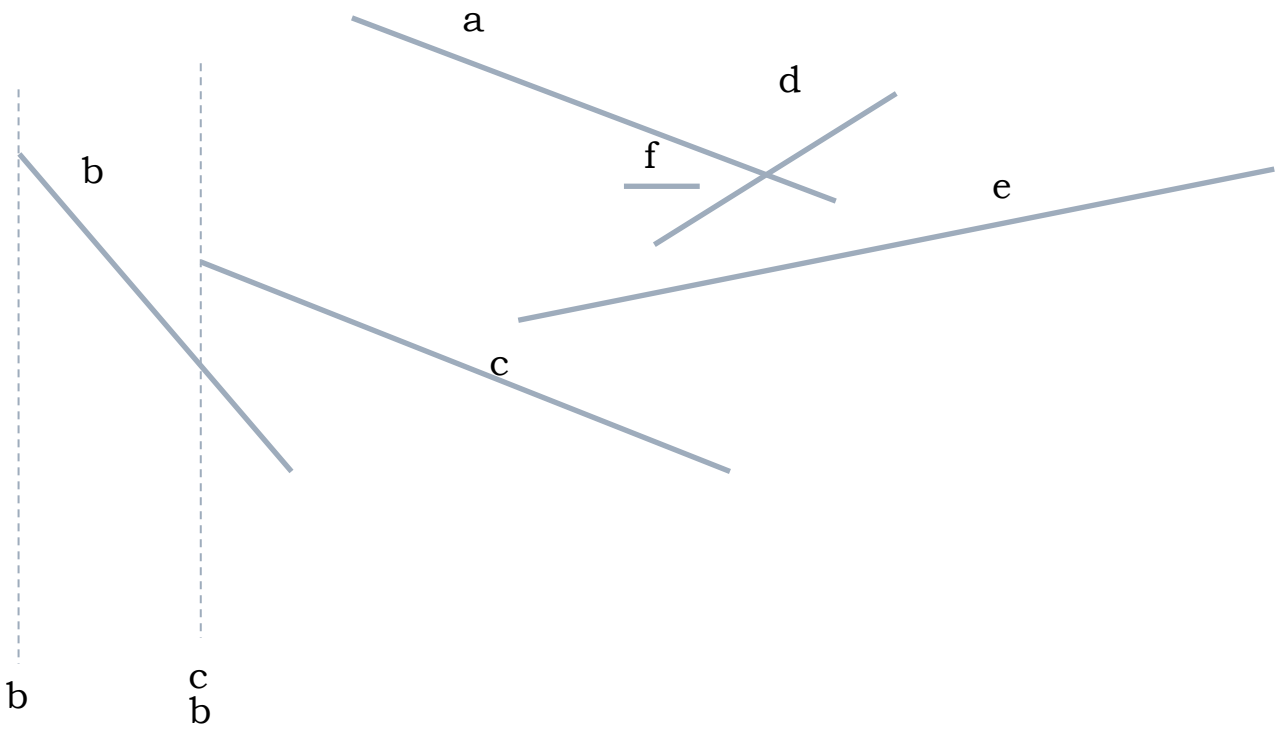
Przecinanie się par odcinków



Algorytm korzysta z metody przez zamiatanie i polega na tym, że:

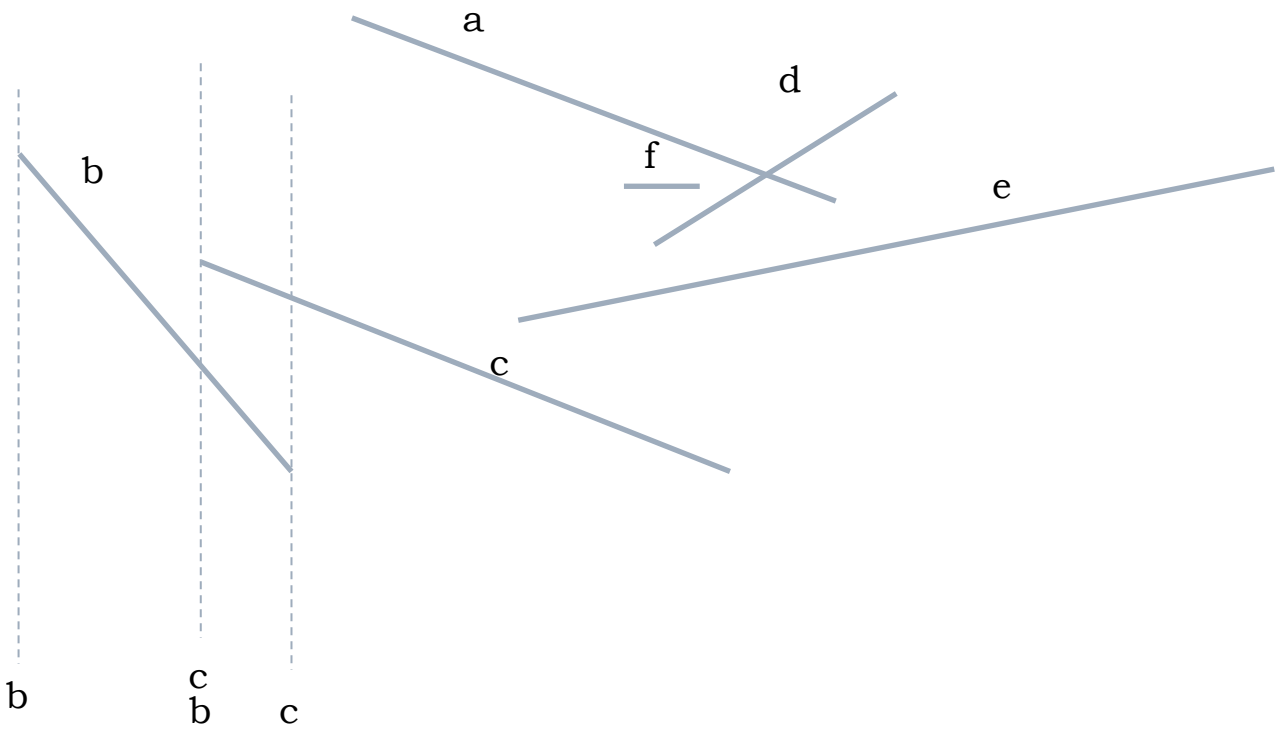
- sortujemy odcinki niemalejąco względem współrzędnych x początków odcinków.
- prowadzimy miotłę po tych współrzędnych.

Przecinanie się par odcinków



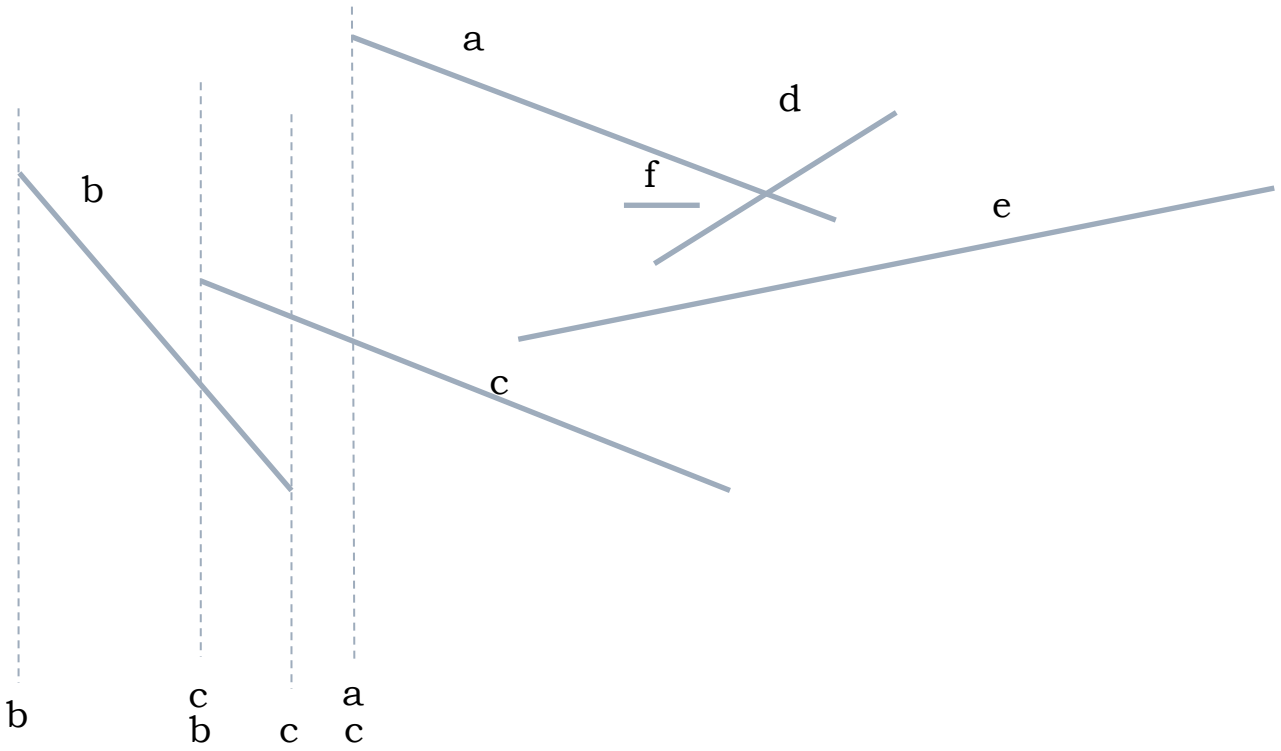
Rozpoczynający się odcinek wkładamy do miotły binarnie, zgodnie z relacją leżenia po lewej/prawej – po lewej wyżej, po prawej niżej. Sprawdzamy czy sąsiadujące odcinki nie przecinają się.

Przecinanie się par odcinków

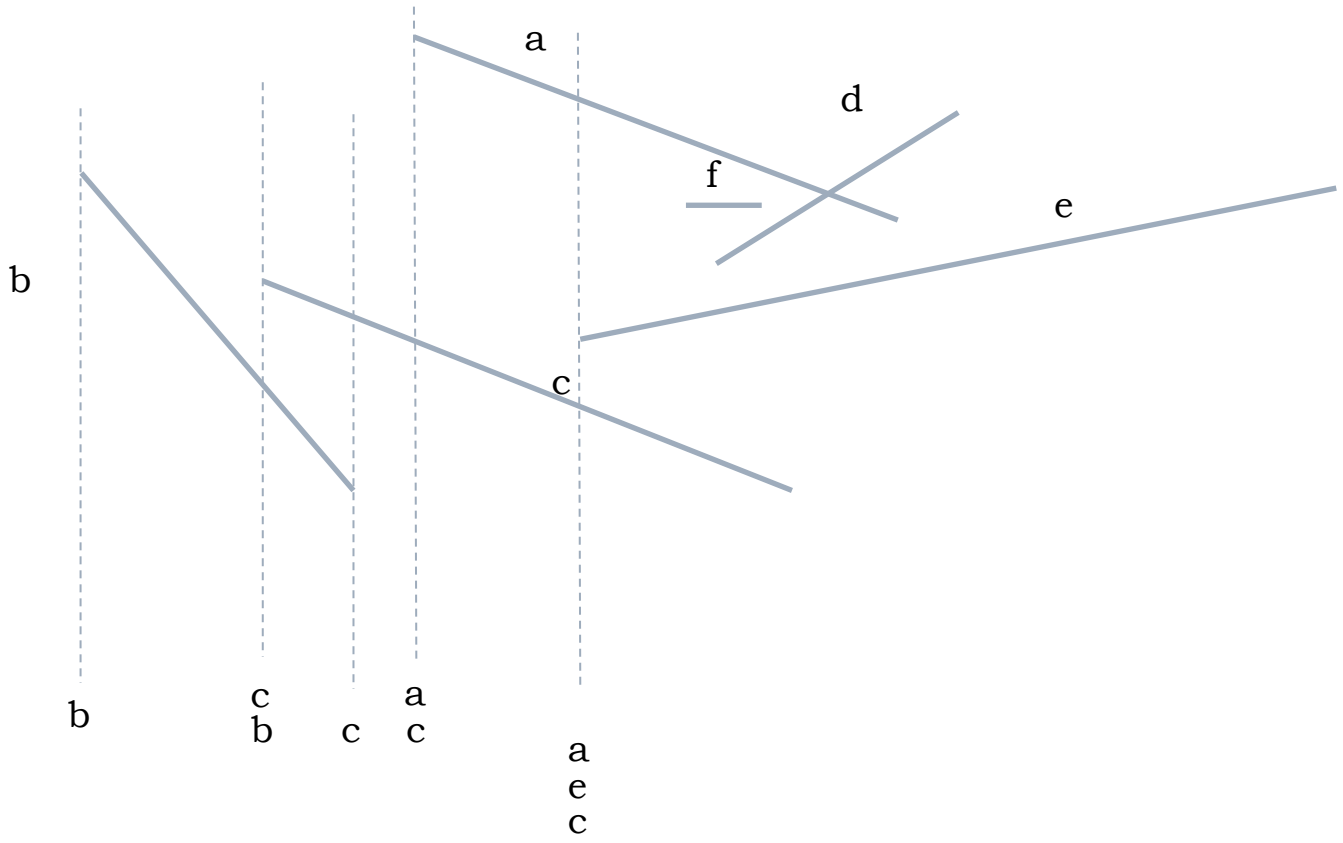


Gdy odcinek się kończy, wyrzucamy go zmiotły i sprawdzamy, czy jego dwaj sąsiedzi nie przecinają się.

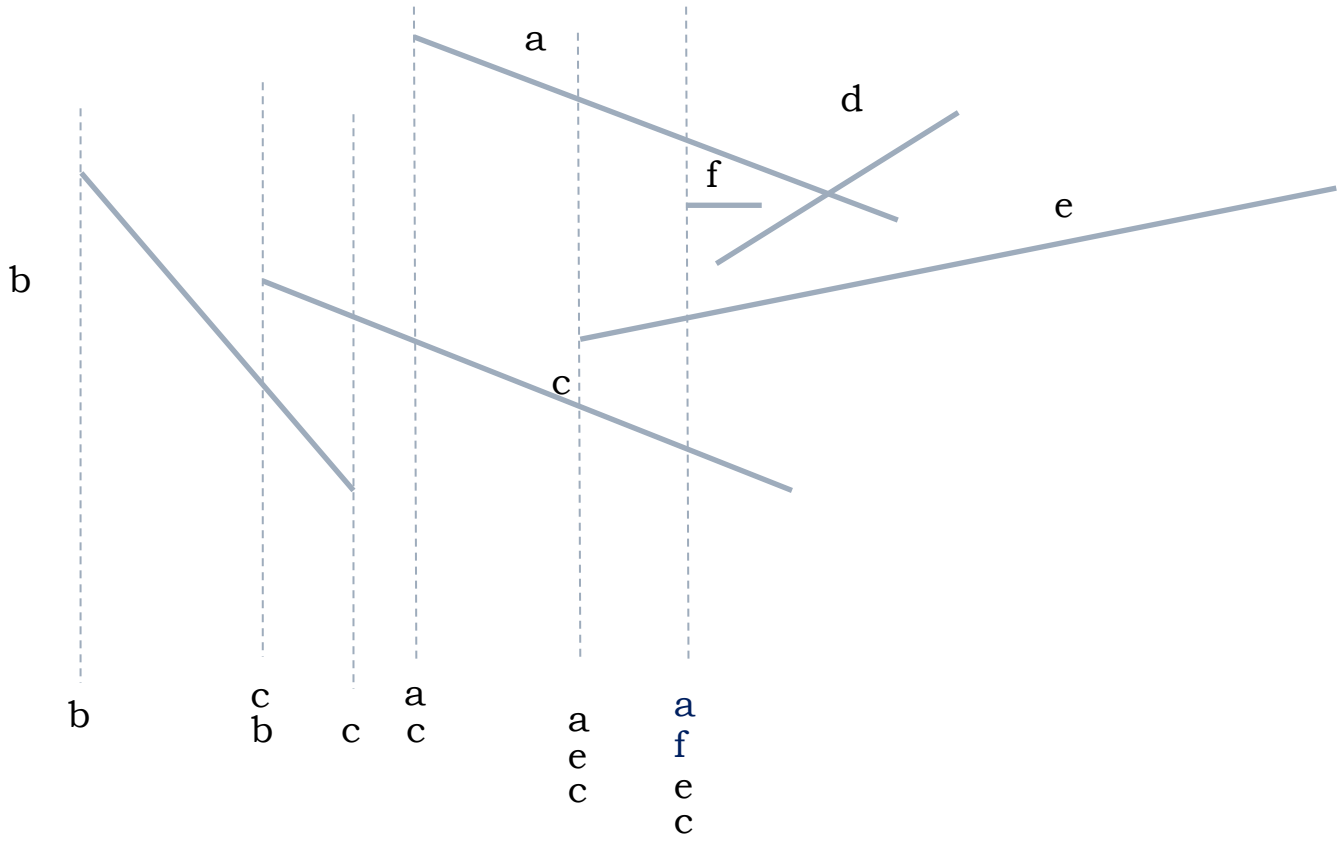
Przecinanie się par odcinków



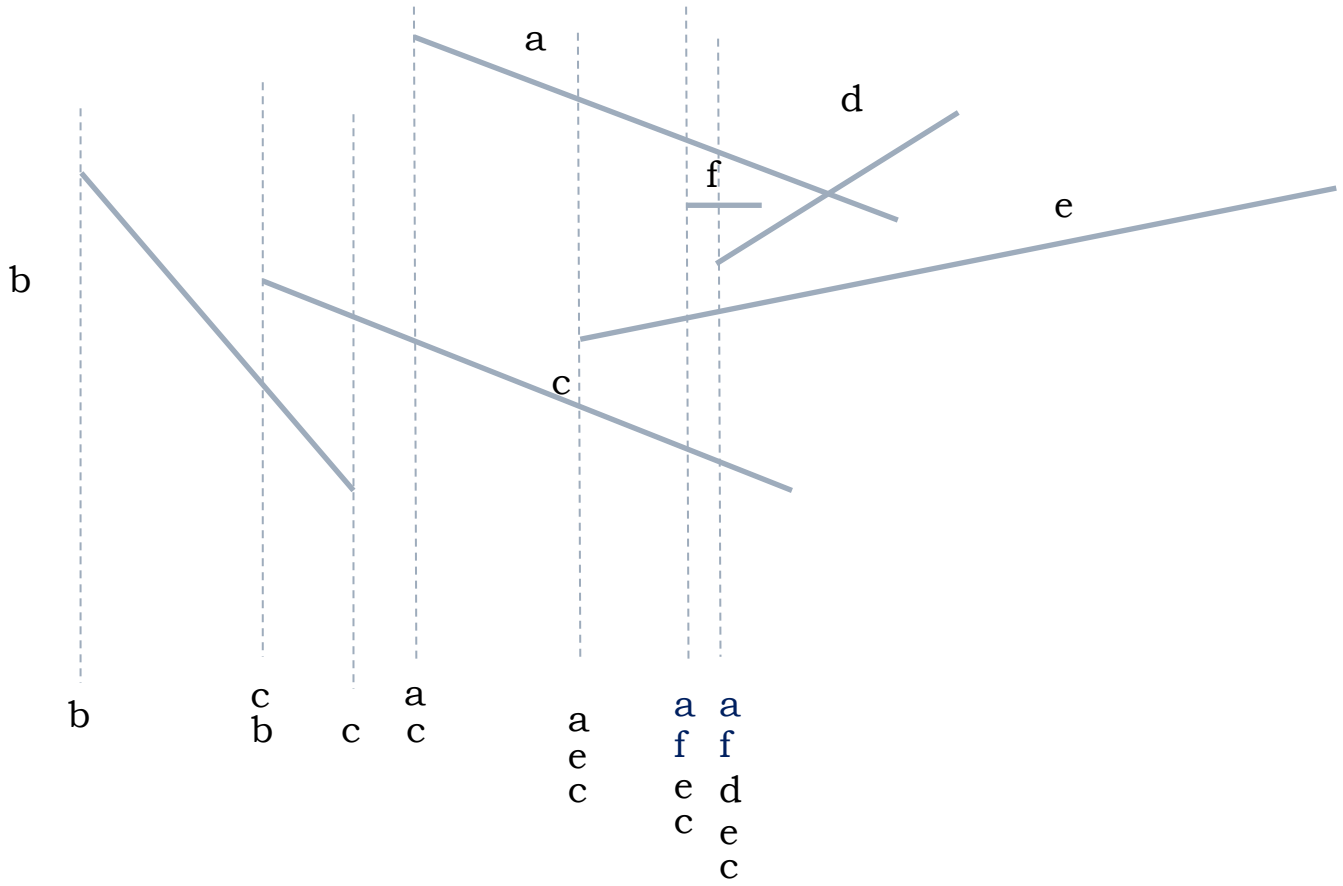
Przecinanie się par odcinków



Przecinanie się par odcinków



Przecinanie się par odcinków



Przecinanie się par odcinków