2 Otoczka wypukła w 2D

2.1 Sformułowanie problemu

–

Zadanie
Dla danych punktów na płaszczyźnie: $p_1, \ldots, p_n$, znaleźć otoczkę wypukłą zbioru tych punktów.

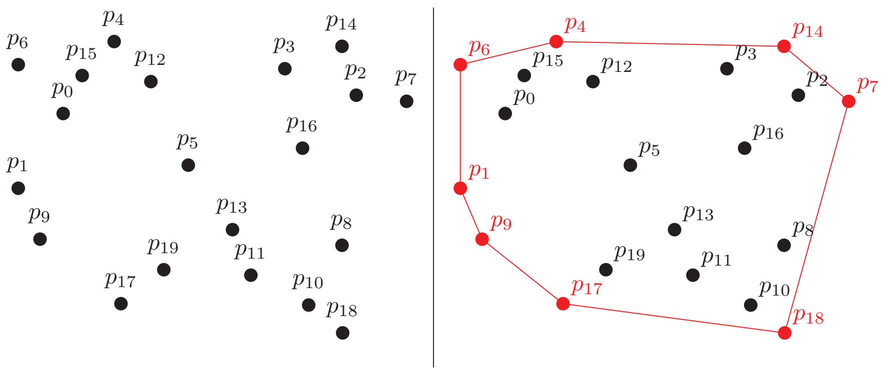
Definicje

- Zbiór P nazywamy wypukłym, jeżeli wraz z każdą parą punktów $p, q \in P$, do zbioru należy odcinek $\overline{pq}$
- Otoczką wypukłą (ang. convex hull) zbioru P nazywamy najmniejszy zbiór wypukły zawierający zbiór P. Oznaczenie $\mathsf{CH}(P)$.

Charakteryzacja
Otoczka wypukła zboru P to część wspólna wszystkich zbiorów wypukłych zawierających P.

2.2 Analiza problemu

- Tak sformułowana definicja otoczki wypukłej jest nieprzydatna:
  - zawiera nieskończoną liczbę obiektów do rozważenia
  - nie pozwala rozwiązać problemu w skończonej liczbie kroków
  - nie mówi czym jest otoczka wypukła
- Aby móc rozwiązać problem otoczki wypukłej algorytmicznie musimy znaleźć charakteryzację która jest pozbawiona tych wad.

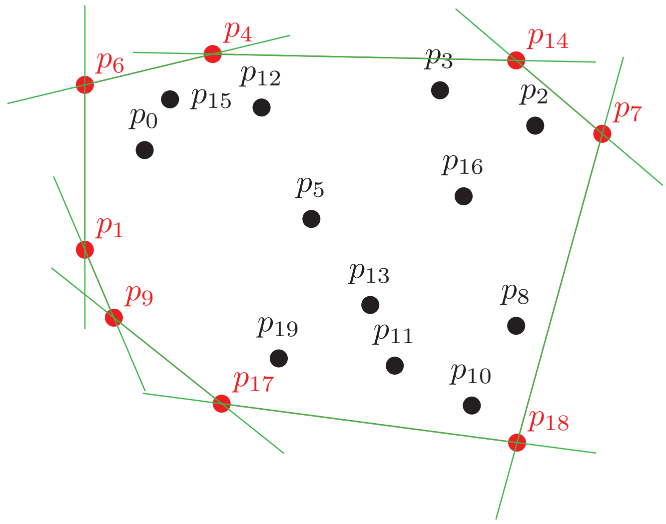


Rysunek 1: Otoczkę wypukłą możemy sobie wyobrazić jako gumkę recepturkę rozpiętą na zbiorze.

Charakteryzacja:
Otoczka wypukła (skończonego) zbioru P to wielokąt wypukły, którego wierzchołki są punktami ze zbioru P, zawierający wszystkie punkty P. Jest on wyznaczony jednoznacznie.

Wnioski:

- odcinek tworzący bok wielokąta zawiera dokładnie dwa punkty zbioru P jako swoje końce
- wielokąt wypukły jest częścią wspólną półpłaszczyzn wyznaczonych przez proste zawierające jego boki



Rozważmy dwa kolejne wierzchołki $p, q \in P$ wielokąta $\mathsf{CH}(P)$ oraz krawędź (skierowaną) $\overline{pq}$.

Obserwacja 1
Wszystkie punkty zbioru P leżą po tej samej stronie prostej $\overline{pq}$

Obserwacja 2
Jeżeli wszystkie punkty zbioru $P \setminus \{p, q\}$ leżą po tej samej stronie prostej $\overline{pq}$, to $\overline{pq}$ jest krawędzią $\mathsf{CH}(P)$.

Przeformułowanie problemu
Dla danej listy punktów P na płaszczyźnie wyznaczyć listę tych elementów które są wierzchołkami wielokąta $\mathsf{CH}(P)$.

Reprezentacja wielokąta
Wielokąt opisujemy jako listę wierzchołków, wypisanych w kolejności zgodnej z ruchem wskazówek zegara. Lista może się zaczynać od dowolnego wierzchołka.

2.3 Algorytm naiwny

Algorytm 1 NaiveConvexHull(P)

Input: P - zbiór punktów na płaszczyźnie
Output: lista wierzchołków wielokąta $\mathsf{CH}(P)$

```
E = ∅
for all p, q ∈ P do
  valid = true
  for all r ∈ P, r ≠ p, q do
    if r leży po lewej stronie prostej pq then
      valid = false
  if valid then
    dodaj pq do E
ze zbioru E, utwórz listę wierzchołków CH(P)
```

Duża złożoność obliczeniowa
Algorytm naiwny ma złożoność obliczeniową $O(n^3)$, gdzie n - liczba wierzchołków. Zbyt dużą dla zastosowań, poza małymi zbiorami.

Nie zawsze działa poprawnie
W przypadku zdegenerowanym - punkty współliniowe - algorytm działa niepoprawnie. Jeżeli punkty p, q, r są współliniowe, to nie dodaje krawędzi.

Nie jest stabilny numerycznie
Jeżeli punkty są prawie, prawie, ale to prawie współliniowe, to w błędy zaokrągleń mogą spowodować nieprzewidywalne zachowanie

Epic fail! Epic fail! Epic fail! Epic fail! Epic fail! Epic fail!
Dodatkowy problem:

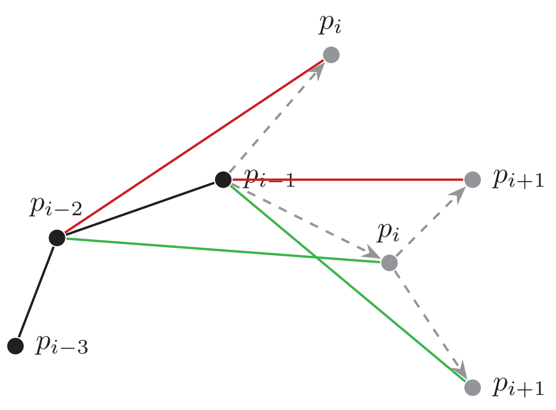
Punkty współliniowe
Punkty p, q, r są współliniowe i leżą na brzegu $CH(P)$ w tej kolejności. Czy otoczka wypukła zawiera krawędzie $\overline{pq}$ i $\overline{qr}$ czy tylko $\overline{pr}$?

2.4 Algorytm GiftWrapping

- Zastosujemy podejście przyrostowe - wyznaczymy $\mathsf{CH}(P)$ dodając po jednym punkcie, za każdym razem aktualizując rozwiązanie.
- Na początek sortujemy punkty względem pierwszej współrzędnej.
- Nastepnie wyznaczymy górną krawędź zaczynając od „najbardziej lewego” punktu kończąc na „najbardziej prawym”
- Potem wyznaczymy dolną krawędź, przechodząc w przeciwnym kierunku.

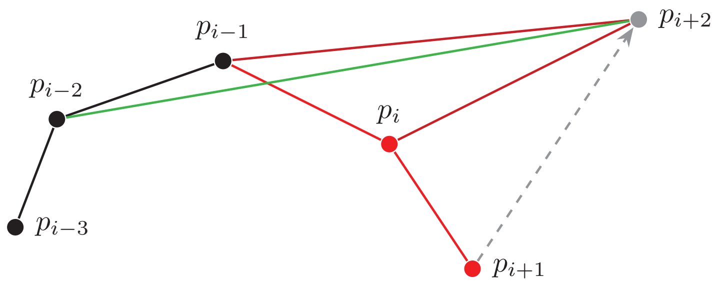
Podstawowy krok:

- Załóżmy, że wyznaczyliśmy górna krawędź L_{up} dla dla punktów $p_1, \ldots, p_{i-1}$.
- Dodajemy kolejny punkt p_i.



- Obserwacja 1: poruszając się po brzegu wielokąta skręcamy w lewo lub w prawo.
- Obserwacja 2: jeżeli wielokąt jest wypukły, to skręcamy tylko w prawo

- Dodajemy p_i do L_{up}. Zauważmy, że p_i należy do otoczki wypukłej punktów $p_1, \ldots, p_i$, bo jest najbardziej na prawo.
- Jeżeli trzy ostatnie punkty tworzą zakręt w prawo: OK, dodajemy następny punkt



- W przeciwnym wypadku, usuwamy środkowy,
- Powtarzamy sprawdzenie dla nowych trzech ostatnich, lub aż zostaną tylko dwa lub pojawi się zakręt prawo.

Algorytm 2 GiftWrapping(P)

Input: P - zbiór punktów na płaszczyźnie
Output: L - lista wierzchołków wielokąta $\mathsf{CH}(P)$

posortuj wierzchołki wg pierwszej współrzędnej
$L_{up} \leftarrow p_1$, $L_{up} \leftarrow p_2$,
for $i = 3$ **to** n **do**
    $L_{up} \leftarrow p_i$
    while L_{up} zawiera więcej niż 3 punkty **and** ostanie trzy nie tworzą zakrętu w prawo **do**
        usuń środkowy z trzech ostatnich
$L_{down} \leftarrow p_n$, $L_{down} \leftarrow p_{n-1}$
for i=n-2 **to** 1 **do**
    $L_{down} \leftarrow p_i$
    while L_{down} zawiera więcej niż 3 punkty **and** ostanie trzy nie tworzą zakrętu w prawo **do**
        usuń środkowy z trzech ostatnich
Usuń pierwszy i ostatni punkt z L_{down}
return połączone listy L_{up} i L_{down}

Nie działa poprawnie

- Jeżeli dwa punkty mają tą samą pierwszą współrzędną, nie mamy dobrego porządku (co to powoduje?). Problem rozwiązujemy sortując leksykograficznie.
- Punkty współliniowe nie tworzą zakrętu. Należy je traktować, jako tworzące zakręt w lewo.

Dobra złożoność

Algorytm ma złożoność obliczeniową $O(n \log n)$, gdzie n - liczba wierzchołków. Bez sortowania leksykograficznego jest $O(n)$, bo sortowanie zajmuje najwięcej czasu.

2.5 Inne algorytmy

Istnieje wiele algorytmów, także wyznaczających otoczkę w 3D:

- Naiwny $O(n^3)$
- GiftWrapping $O(n \log n)$
- Graham $O(n \log n)$
- Jarvis $O(nh)$, pesymistyczna $O(n^2)$
- QuickHull $O(n \log n)$, pesymistyczna $O(n^2)$
- Merge
- Chan (1996) $O(n \log h)$, pesymistyczna $O(n \log n)$