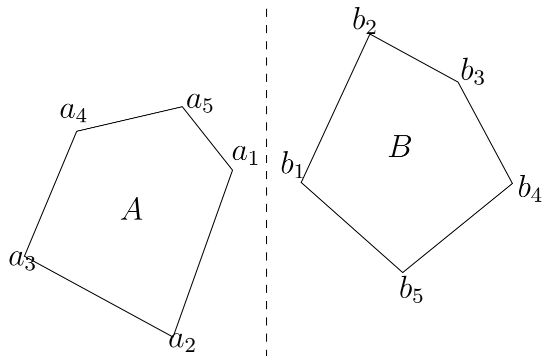
Zadanie 3. *Otoczką wypukłą zbioru P, punktów na płaszczyźnie, nazywamy najmniejszy wielokąt wypukły zawierający (w swoim wnętrzu lub na brzegu) wszystkie punkty z P. Naturalny, oparty na zasadzie dziel i zwyciężaj, algorytm znajdowania otoczki wypukłej dla zbioru P, dzieli P na dwa (prawie) równoliczne podzbiory (np. pionową prostą), znajduje rekurencyjnie otoczki wypukłe dla tych podzbiorów, a następnie scala te otoczki. Podaj algorytm wykonujący tę ostatnią fazę algorytmu, tj. algorytm scalania dwóch otoczek wypukłych.*

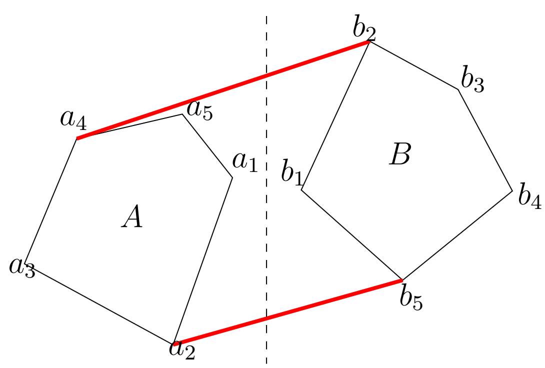
Konstrukcja otoczki

Załóżmy, że nasz algorytm znajdujący otoczkę wypukłą dzieli punkty na dwa podzbiory pionową prostą. Niech wielokąty zwrócone przez algorytm dla podproblemów to będą $A = (a_1, a_2, ..., a_{n_1})$ i $B = (b_1, b_2, ..., b_{n_2})$, takie, że kolejne punkty stanowią wielokąt w kolejności zgodnej z ruchem wskazówek zegara. Przykładowo, a_1 jest połączone z a_2, a_2 z a_3, ..., a_n z a_1. Niech A będzie na lewo od dzielącej prostej, a B na prawo. Przykład takich wielokątów został przedstawiony na rysunku 1.



Rysunek 1: Przykładowe A i B.

Chcemy znaleźć taką prostą wyznaczaną przez punkty $(a, b) \in A \times B$, że żaden z punktów a_i i b_i nie leży powyżej tej prostej (dalej nazywanej górną styczną) oraz analogicznie (a', b') takie, że żaden punkt nie leży poniżej (dolna styczna). W ten sposób po dodaniu odcinków (a, b) i (a', b') i usunięciu punktów nie stanowiących brzegu wielokąta uzyskamy wielokąt wypukły zawierający wszystkie punkty z S. Dla naszego przykładowego rysunku wybraliśmy $(a, b) = (a_4, b_2)$ i $(a', b') = (a_2, b_5)$.



Rysunek 2: Dodane odcinki (a_4, b_2) i (a_2, b_5).

Intuicyjnie widzimy, że wystarczają dwa takie odcinki, aby uzyskać jeden wielokąt wypukły zawierający A i B. Spróbujmy (ze względu na mój brak doświadczenia geometrycznego mało formalnie) uzasadnić, że tak jest.

Twierdzenie 1. *Odcinki przechodzące przez styczne pewnych rozłącznych, oddzielonych prostą wielokątów wypukłych wraz z tymi wielokątami wyznaczają wielokąt wypukły.*

Dowód. Niech S będzie dowolnym zbiorem punktów, a A i B niech będą wielokątami spełniającymi wszystkie nasze założenia utworzonymi z pewnego podziału zbioru S. Niech (a, b) będzie taką parą punktów, że wszystkie punkty z $A \cup B$ są nie wyżej niż prosta przechodząca przez (a, b). Podobnie niech (a', b') wyznacza prostą, poniżej której nie ma żadnych punktów z $A \cup B$. Oczywiście dla każdego skończonego zbioru punktów takie cztery punkty istnieją.

Wszystkie punkty S znajdują się w obszarach ograniczonych przez A lub B, a więc znajdują się w wielokącie utworzonym przez dodanie odcinków (a, b) i (a', b') i usunięcie punktów wewnętrznych.

W końcach tych odcinków utworzyliśmy cztery nowe kąty; pozostałe kąty wielokąta pochodzą z oryginalnych A i B, więc są większe niż 180°. Pozostaje więc rozważyć kąty w punktach a, a', b, b'. Rozważymy tylko punkt a; pozostałe przypadki są bardzo podobne.

a jest połączony z punktem b oraz pewnym innym punktem x. Wiemy, że x znajduje się nie wyżej niż prosta przechodząca przez (a, b). Skoro każde z x, b znajduje się po jednej stronie (lub na) tej prostej przechodzącej przez a, to kąt nie może być większy niż 180°.

Skoro wszystkie kąty są nie większe niż 180°, to wielokąt jest wypukły. □

Twierdzenie 2. *Tak uzyskany wielokąt jest minimalny.*

Dowód. Na pewno nie możemy zmniejszyć obszarów ograniczonych przez A i B — inaczej tracilibyśmy punkty lub łamali warunek wypukłości, bo A i B są minimalne. Z tego wynika też, że punkty nie stanowiące brzegu tych obszarów nie są interesujące dla tworzenia otoczki. Zatem nowa otoczka musi powstać z pewnego podzbioru punktów w A i B. Gdyby wziąć dowolny wielokąt (bez dziur), który jest mniejszy, oszczędzając coś na obszarze „łączącym” A i B, to któryś z odcinków (a, b) lub (a', b') nie byłby zawarty w całości w wielokącie, więc taki wielokąt nie byłby wypukły. □

Wyszukiwanie stycznych

Naiwne wyszukiwanie odpowiednich par punktów byłoby kosztowne, a chcielibyśmy ograniczyć czas spędzony podczas scalania do $O(n)$, tym samym ograniczając złożoność rekurencji do $O(n \log n)$.

Załóżmy teraz dla wygody, że wszystkie współrzędne x są różne[1]. Skróci to trochę zapis i uzasadnienia. Dodatkowo przyjmiemy, że indeksy punktów w odpowiednich wielokątach są cykliczne, czyli np. $a_i = a_{(i \mod n_1)+1}$.

Niech a_l będzie punktem z A o najmniejszej współrzędnej x, a a_r o największej współrzędnej x. Analogicznie definiujemy b_l, b_r.

Chcemy dla wielokątów A i B poprowadzić styczną górną. Wiemy, że A i B są wypukłe, więc intuicja podpowiada, że punkty $(a, b) \in A \times B$ wyznaczające taką styczną powinny znajdować się

[1] Możemy pozbyć się tego założenia uważniej dzieląc punkty ze względu na współrzędne, aby wielokąty A i B zawsze były rozłączne.

w górnych „częściach” swoich wielokątów. W zdegenerowanych przypadkach możemy mieć kilka dobrych punktów, które są współliniowe na stycznej. W szczególnych sytuacjach na stycznej mogą leżeć punkty nie należące do górnej części. Pokażemy, że zawsze istnieją tam punkty z części górnej.

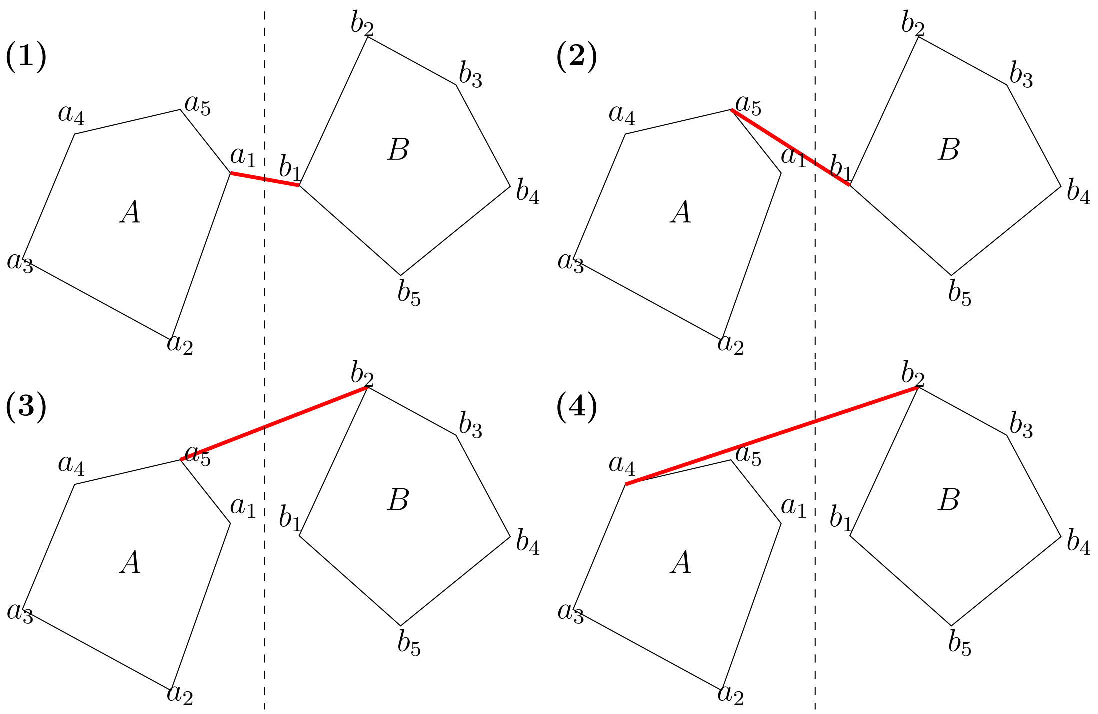
Twierdzenie 3. *Zawsze możemy wybrać* (a, b) *tak, że są w górnych częściach swoich wielokątów, czyli* $a \in \{a_l, a_{l+1}, ..., a_r\}$, $b \in \{b_l, b_{l+1}, ..., b_r\}$.

Dowód. Udowodnimy tezę dla wielokąta A; dla B dowód przebiega analogicznie.

Załóżmy nie wprost, że istnieje taki zbiór punktów S i jego podział na A, B, że na jego górnej stycznej nie leżą żadne punkty z górnej części A. Wybierzmy dowolny punkt a na tej stycznej.

a_l i a_r są nie wyżej niż styczna. Z założenia nie wprost żaden z tych punktów nie znajduje się na stycznej. To oznacza, że a znajduje się powyżej prostej między a_l i a_r, a więc wielokąt A nie jest wypukły — sprzeczność. $\square$

Odcinek pomiędzy a_r i b_l nie przecina żadnego z wielokątów. Zaczynając w tych punktach, możemy przemieszczać się po wielokącie A przeciwnie do ruchu wskazówek zegara, i po B zgodnie z ruchem wskazówek zegara. Po A przesuwamy się, dopóki zmniejsza to nachylenie. Po B, dopóki nachylenie się zwiększa. Kiedy nie ma już możliwości przesunięcia, (a, b) zostały znalezione.



Rysunek 3: Przykład działania algorytmu.

Znalezione punkty spełniają nasze wymaganie, że wszystkie pozostałe znajdują się nie wyżej, niż ich prosta: załóżmy nie wprost, że jest inaczej, czyli istnieje punkt ponad wyznaczoną prostą. Bez straty ogólności niech będzie to pewien punkt $a_i \in A$. Jeżeli $x(a_i) < x(a)$, to, z wypukłości A, wykonalibyśmy przynajmniej jedno więcej przesunięcie od punktu a, nie kończąc algorytmu w tym punkcie — sprzeczność. Jeżeli natomiast $x(a) < x(a_i)$, to algorytm nigdy by do a nie dotarł, bo zwiększałoby to nachylenie, a więc znowu sprzeczność.

Taka iteracja nigdy nie zajdzie dalej niż punkty a_l i b_r, a więc zawsze zostanie znalezione rozwiązanie. Znajdowanie dolnej stycznej odbywa się analogicznie.

Pamiętając, że operacje na indeksach odbywają się modulo odpowiednie rozmiary, pseudokod procedury łączącej dwie otoczki mógłby wyglądać następująco:

```
Function slope(a, b)
    return (y(a) - y(b)) / (x(a) - x(b))
Procedure MergeConvexHulls(A = (a_1, a_2, ..., a_{n_1}), B = (b_1, b_2, ..., b_{n_2}))
    i, i' ← indeks punktu A o maksymalnym x
    j, j' ← indeks punktu B o minimalnym x
    while True do
        if slope(a_{i-1}, b_j) < slope(a_i, b_j) then
            i ← i - 1
        else if slope(a_i, b_{j+1}) > slope(a_i, b_j) then
            j ← j + 1
        else
            break
    while True do
        if slope(a_{i'+1}, b_{j'}) > slope(a_{i'}, b_{j'}) then
            i' ← i' + 1
        else if slope(a_{i'}, b_{j'-1}) < slope(a_{i'}, b_{j'}) then
            j' ← j' - 1
        else
            break
return (a_{i'}, a_{i'+1}, ..., a_i, b_j, b_{j+1}, ..., b_{j'})
```

Zauważmy, że jesteśmy w stanie zwracać uzyskaną otoczkę wypukłą w pożądanej postaci, tj. posortowaną zgodnie z ruchem wskazówek zegara. Dlatego nie potrzebujemy wykonywać dodatkowej pracy pod warunkiem, że dla przypadków brzegowych rekurencji ta własność też jest zachowana.

Złożoność

Przedstawiona procedura scalania otoczek odbywa się w $O(n)$. Zatem rozważany algorytm szukania otoczki wypukłej może posortować punkty po współrzędnej x raz w $O(n \log n)$, i wykonać procedurę rekurencyjną o złożoności $O(n \log n)$. Stąd złożoność algorytmu to również $O(n \log n)$.