

Zad 9

Zakładaćmy, że punkty $p_1 = (x_1, y_1), p_2 = (x_2, y_2), \dots, p_n = (x_n, y_n)$ podane są w kolejności, w jakiej rysować będziemy ten wielokąt.

Aby analizować rozwiązywanie tego problemu, będziemy go dzielić na podproblemy.

Wybranym punkt p_k , $k \in \{2, 3, \dots, n-1\}$: podzielimy wielokąt na trzy części: wielokąt p_1, \dots, p_k , wielokąt p_k, \dots, p_n i trójkąt p_1, p_k, p_n .

Rozwiążemy rekurencyjne podproblemy dla nowo powstających wielokątów.

Danaczymy wielokąty p_i, \dots, p_j jako $P(i, j)$. Przy podziiale w punkcie p_k najdłuższa przekątna będzie miała długość $\max\{P(1, k), P(k, n), l_{p_1, p_k}, l_{p_k, p_n}\}$.

Do rozwiązania całego problemu musimy wykonać ten proces dla każdego możliwego $k \in \{2, 3, \dots, n-1\}$ wyliczając

$$\min_k (\max\{P(1, k), P(k, n), l_{p_1, p_k}, l_{p_k, p_n}\}).$$

Zbiór rekurencyjny:

$$P(i, j) = 0, \quad j - i < 3$$

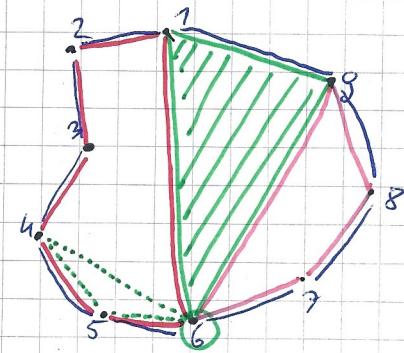
$$P(i, j) = \min_{i \leq k \leq j} \max\{P(i, k), P(k, j), l_{p_i, p_k}, l_{p_k, p_j}\}$$

Będziemy uzupełniać tablicę dwuwymiarową $P[i][j] = P(i, j)$.

Dla każdego pola tablicy musimy $O(n)$ razy policzyć maksimum (ze względu na wybór k). Zatem złożoność algorytmu wynosi $O(n^3)$.

Aby wyznaczyć zbiór tych wierzchołków wystarczy stworzyć dnołu, w którym każde uzupełnianie rekurencyjne będzie dopisywać w węźle wartość k (punkt podziału), tzn. uzupełnianie rekurencyjne $P(i, j)$ dopisze w węźle k, j , $P(i, k)$ w lewym diecku - k, i , a $P(k, j)$ w prawym diecku. W tym celu najpierw wyznaczamy k dla $P(i, j)$, po czym ponownym podproblemy $P(i, k), P(k, j)$.

Nie zmieni to złożoności obliczeniowej algorytmu.



Aby wyznaczyć liczbę takich podzbiiorów, wykorzystamy algorytm dynamiczne rozwiązujejący problem Longest Increasing Subsequence.

function LIS($T[1..n]$):

$D[1, \dots, n] \leftarrow$ tablica wypełniona wartościami 1 // Dl. najd. LIS do i
 $K[1, \dots, n] \leftarrow K[1] = 1, \text{ reszta } 0$
for i from 1 to n :

for j from 1 to $i-1$:

if $T[j] < T[i]$

if $D[i] = D[j] + 1$

$K[i] += 1$

if $D[i] < D[j] + 1$

$D[i] = D[j] + 1$

$K[i] = 1$

$ind = 0, M = 0$

for i from 1 to n :

if $M < D[i]$

$M = D[i]$

$ind = i$

return $K[i]$

Zad 8

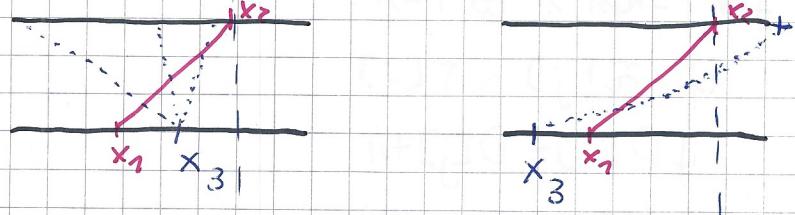
Obserwacja:

Wzajemne dwa dowolne odcinki p_1 i p_2 o końcach odpowiednio w:

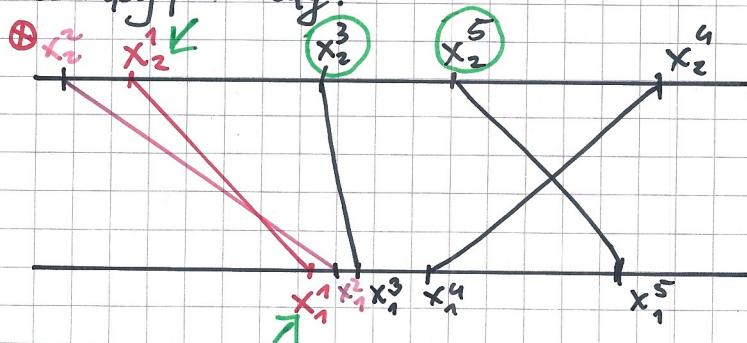
$$p_1: (x_1, y_1), (x_2, y_2)$$

$$p_2: (x_3, y_3), (x_4, y_4)$$

Odcinki przecinają się, gdy $(x_1 < x_3 \wedge x_2 > x_4) \vee (x_1 > x_3 \wedge x_2 < x_4)$



Zauważmy teraz, że gdy posortujemy odcinki po pierwszej współrzędnej x tworzącą listę, w której każdy odcinek zaczyna się nie wcześniej niż jego poprzednik. Wtedy, jeśli koniec tego odcinka jest dalej niż koniec jego poprzednika, to te odcinki się nie przecinają. Wystarczy więc znaleźć najdłuższy rosnący podciąg.



Mozemy wyznaczyć LCS 2 ciągiem powtarzanych współrzędnych x w czasie $O(n \log n + n^2) = O(n^2)$

Znalezione współrzędne są końcami odcinków uchodzących w skład rozwiązania.