

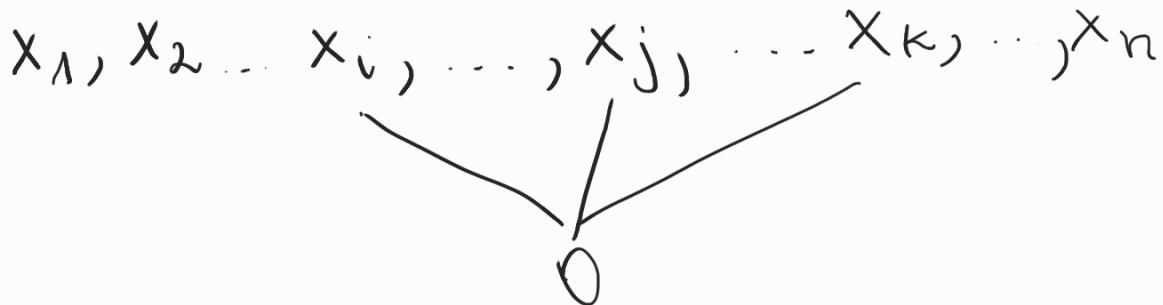
2. (2pkt) Rozważmy następujący problem:

PROBLEM:

Dane: Liczby rzeczywiste x_1, \dots, x_n .

Wynik: 'TAK' - jeśli $\exists_{1 \leq i < j < k \leq n} x_i + x_j + x_k = 0$,
'NIE' - w przeciwnym przypadku.

Udowodnij, że $\Omega(n \log n)$ jest dolną granicą na rozwiązanie tego problemu w modelu liniowych drzew decyzyjnych.



Ponieważ jeden krok naszego algorytmu to przejście o jeden poziom w dół w drzewie to chcemy pokazać, że wysokość drzewa musi wynosić $n \log n$.

Wiemy, że gdy drzewo ma k wież chotkowych to jego wysokość wynosi $\log k$.

$$n \log n = \log n^h$$

$n! \approx \left(\frac{n}{e}\right)^n$
 $\log \left(\frac{n}{e}\right)^n =$
 $= n (\log n - \log e)$

Polubimy rozważali problem sortowania to wygodnie byłoby pokazać, że liści musi być $n!$.

CZEMU?

w przypadku ciągu o długosci n
mamy $n!$ możliwości

1

$\frac{n}{n-1} \frac{(n-1)}{n-2} \dots$ jedw. wysokość działa to k to ilosci 2^k .

TUTAJ MAMY JEDYNIE DWIE MOŻLIWOŚCI
ODPOWIEDZI \rightarrow TAK lub NIE

Fajnie byłoby więc pokazać, że
ilość możliwych do tego samego
zbioru, czyli ilość z tego samego
odpowiedzia muszą znajdować się
w różnych ilościach, czyli że
droga do osiągnięcia tej odpowie-
dzi była inna.

Dodatekowo należy pokazać, że tych
ilości będą dostateczne dla to.

Również w zdeniu tym
 najbardziej zależy nam na
 porównaniu przy użyciu operacji $=$,
 ale wtedy potrzebujemy dokładność
 porównania przy użyciu operacji $\leq >$
 to do rozwiązywania tego prob-
 lemów wykorzystamy dziesię-
 trójkowe.

Chcemy ustosować dla $x_1 \dots x_n$ z
 punktem w $(x_1, \dots, x_n) \in \mathbb{R}^n$,
 tablicę taką, aby prowadziła punkt w przestrzeni n
 wynikowej.

Mozemy zdefiniować przedstawienie
 afiernowe.

$$c + \sum_{i=1}^n a_i x_i \geq 0$$

Jesieli tak to punkty należą

do podpłaty za afiliację

$$n+1, n+3, n+5, \dots, 2, \underbrace{\dots}_{\left(\frac{n}{2}\right)!} \text{ NIE} \underbrace{\dots}_{\frac{n}{2}}$$
$$2^{n-2}, 2^{n-4}, 2^{n-6} \dots$$

w każdym miejscu wykonuję - myj jakieś operacje (kombinacyjne itp). Wynik tej operacji może być $=, >, <$ od D. (sumę jakichś 3 współczynników)

WIELOSIĘĆ WYPUKŁY (WKŁĘŚTY) to wielosioń, który leży (nie leży) całkowicie po jednej stronie każdej z dwóch

Królestwo ze Sudem.

Chcemy pokazać, że wys. drzewa musi wynosić $\lceil \log n \rceil$, bo każdy $\lceil \log n \rceil$ wierzchołek w dół to $\lceil \log n \rceil$ krok naszego algorytmu. Najlepiej pokazać, że musi być $n!$ liści.

W przypadku sortowania jest to łatwe, bo przy ciągu n -elementowym mamy $n!$ możliwych wyników, więc $n!$ liści, bo $\lceil \log n \rceil$ liść $= 1$ wynik.

W przypadku tego problemu to nie jest łatwe, bo mamy tylko 2 możliwe wyniki - TAK lub NIE .

Idea tego rozwiązania:

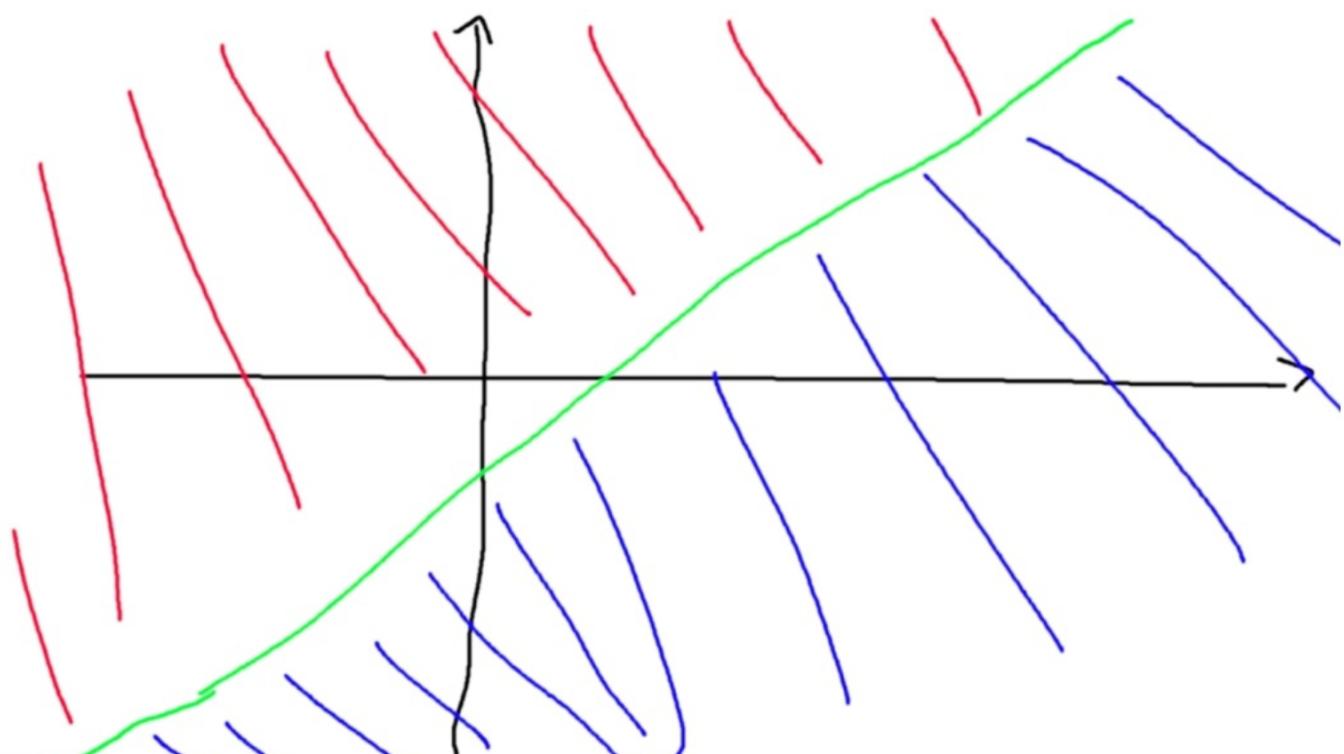
Stworzyć taki zbiór danych, które mimo tego, że mają taką samą odpowiedź, to muszą znajdować się w różnych liściach, to znaczy, że droga algorytmu do osiągnięcia tej odpowiedzi musiała być różna (wykonywać trochę różne kroki). Jeśli pokażemy, że tych liści jest dostatecznie dużo, to by oznaczało, że w przynajmniej jakimś przypadku algorytm musiał wykonać $O(n \log n)$ kroków.

Zatem jak to zrobić:

Chcemy utożsamić nasz ciąg $(x_1 \dots x_n)$ z punktem w $((x_1, x_2, \dots, x_n) \in \mathbb{R}^n)$

Budujemy drzewo trynarne, gdzie w każdy liściu wykonujemy jakąś operację na punkcie (P_i) (kombinację liniową), gdzie wynik tej kombinacji to może być $<0, =0, >0$. Nie obchodzi nas, co to jest, ale w przypadku tego zadania mogłyby to być np. suma jakichś 3 współrzędnych.

Takie podziały będą budowały podprzestrzenie o takim samym lub niższym wymiarze - hiperpłaszczyzna powstała przez ' $=0$ ' będzie dzieliła na 2 wypukłe podprzestrzenie. Np w tym przypadku, w \mathbb{R}^2 jednowymiarowa płaszczyzna dzieli \mathbb{R}^2 na 2 mniejsze:



Niech, tak jak na wykładzie, $\{s(v)\}$ dla wierzchołka v to będzie zbiór punktów, które osiągają v . Oczywiście $\{s(v)\} \subseteq R^n$ i jest to jedna z takich przestrzeni jak na rysunku.

Lemat z wykładu

$\{s(v)\}$ jest wielościanem wypukłym. To oznacza, że jak weźmiemy dowolne 2 punkty z $\{s(v)\}$, to jak narysujemy linię między nimi, to wszystkie punkty na tej linii też będą w $\{s(v)\}$ - w naszym przypadku ma to taką implikację, że wszystkie te punkty prowadzą do takiej samej odpowiedzi (TAK) lub (NIE).

Chcemy więc skonstruować taki przykład, który prowadzi do ilości liści rzędu $n!$. Weźmy takie liczby:

$$\{1, 3, 5, 7, \dots, n-1, 2n, -(2n+2), -(2n+4), -(2n+6), \dots, -(2n + n - 2)\}$$

Odpowiedź dla tego zbioru to NIE , ponieważ musimy wziąć jakąś liczbę ujemną. Jak już jakąś weźmiemy, to musimy wyeliminować $(-2n)$ a jedyny sposób na to to wzięcie 2 - wzięcie jakichś 2 liczb nieparzystych będzie zawsze za małe, bo 2 największe sumują się do zaledwie $(2n - 3)$. Zostaje nam liczba parzysta, ale nie możemy jej wyeliminować, bo zostały same nieparzyste.

Załóżmy, że n jest parzyste.

Rozważmy teraz permutacje tego podzbioru takie, że liczby parzyste zostają takie same, ale permutujemy liczby nieparzyste. Zatem mamy $\frac{n}{2}!$ różnych punktów, które oznaczymy $P_1 \dots P_{\frac{n}{2}!}$. Dla wszystkich oczywiście odpowiedź to NIE i poprawny algorytm taką powinien dawać. Chcemy pokazać, że odpowiedź na nie musi trafić zawsze do innego liścia.

Dowód:

Weźmy dowolne punkty P_i, P_j i współrzedną którą się różnią o indeksie k i założmy, że są w tym samym liściu. Wiadomo, że skoro są w tym samym liściu, to są w tej samej hiperpłaszczyźnie, więc na linii między P_i oraz P_j wszystkie punkty należą do tej płaszczyzny czyli dają taką samą odpowiedź NIE . Zatem istnieje jakiś punkt P między nimi, który na k -tej współrzednej ma liczbę parzystą. Weźmy go zatem.

Ale skoro ma on liczbę parzystą na k -tej pozycji, to algorytm powinien zwracać dla niego tak, bo $P(k) + 2n + -(2n + P(k)) = 0$. Wiemy, że $(-2n + P(k))$ należy do naszego zbioru danych, bo skoro ten punkt leży między P_i a P_j , to też $1 < k < n$. Stąd sprzeczność.

Z tego wynika, że nasze drzewo ma $\frac{n}{2}!$ liści, ze wzoru Stirlinga: $n! \approx (\frac{n}{e})^n$

A wys. drzewa o $\lfloor n! \rfloor$ liści wynosi przynajmniej tyle, zatem:

$$\sqrt{2}^n \geq n! \Rightarrow h \geq \log(n!) \approx \log(n^n) \approx n \log n$$

