Algorytmy i Struktury Danych, 14. ćwiczenia

2019-01-23

Spis treści

1 Obliczanie tablicy LCP

Tablica LCP zawiera długość najdłuższego wspólnego prefiksu pomiędzy kolejnymi sufiksami z tablicy sufiksowej.

Function LCP(T, SA)

$n = |T|$
$l = 0$
for $i = 1, \ldots, n$ **do**
  $k = SA^{-1}[i]$
  $j = SA[k-1]$
  while $T[i+l] = T[j+l]$ **do**
    $l = l + 1$
  $LCP[k] = l$
  if $l > 0$ **then** $l = l - 1$
return LCP

2 Drzewa sufiksowe definicja

`https://en.wikipedia.org/wiki/Suffix_tree`

3 Zastosowanie drzew sufiksowych

- wyszukiwanie wielu wzorców
- liczba różnych podsłów
- najdłuższe wspólne podsłowo

4 Egzamin 2017/2018, zadanie 1

W kostce $[0, n]^3$, $n > 10$, danych jest n różnych odcinków równoległych do osi układu współrzędnych i o końcach w punktach o współrzędnych całkowitych, każdy o długości 10. Zaproponuj efektywny algorytm obliczający liczbę punktów o współrzędnych całkowitych należących do co najmniej dwóch różnych odcinków.

Rozwiązanie:

Ponieważ każdy odcinek ma długość 10, to zawiera co najwyżej 11 punktów o współrzędnych całkowitych. Dla każdego odcinka P_i kodujemy jego punkty (o współrzędnych całkowitych) jako czwórki (x, y, z, i). Wszystkie w ten sposób czwórki sortujemy leksykograficznie. W ten sposób znalezienie różnych punktów, które należą do co najmniej 2 odcinków wymaga jedynie jednokrotnego przejścia po uporządkowanej liście.

5 Egzamin 2017/2018, zadanie 3

Niech x będzie słowem binarnym o długości co najmniej 2 i zawierającym co najmniej jedno 0 (zero) oraz co najmniej jedną 1 (jedynkę). Zaprojektuj efektywny algorytm, który w słowie binarnym x znajduje dwa podsłowa o maksymalnej długości, które różnią na każdej pozycji.

Rozwiązanie:

Budujemy drzewo sufiksowe dla słowa $S = x\$neg(x)\#$, gdzie $neg(x)$ oznacza negację słowa x. Wszystkie sufiksy słowa S anotujemy etykietą:

- 1 jeśli rozpoczynają się we fragmencie x słowa S
- 2 jeśli rozpoczynają się we fragmencie $neg(x)$ słowa S
- 0 wpp

. Poszukiwane podsłowa są zakodowane w drzewie sufiksowym przez węzły, które w swoim poddrzewie mają zarówno liście typu 1 jak i 2. Nas interesuje najdłuższe takie podsłowo, więc musimy znaleźć węzeł o największej głębokości (uwaga! krawędzie w drzewie sufiksowym mają wagi).

6 Egzamin 2017/2018, zadanie 4

Zaprojektuj strukturę danych, która umożliwia wydajne wykonywanie następujących operacji na dynamicznie zmieniającym się ciągu liczbowym $\alpha =< a_1, a_2, \ldots, a_n >$:

- Ini(α):: α := <pusty ciąg>; //operacja wykonywana tylko raz na początku
- Insert(α,a,i):: wstaw element a na pozycję i w ciągu α, $1 \leq i \leq |\alpha|+1$
- Delete(α,i):: usuń i-ty element z ciągu α, $1 \leq i \leq |\alpha|$
- Add(α,e,i,j):: do każdego z elementów podciągu α[i..j] dodaj e, $1 \leq i \leq j \leq |\alpha|$
- MaxInc(α):: podaj długość najdłuższego spójnego podciągu rosnącego w ciągu α

Rozwiązanie:

Budujemy drzewo czerwono-czarne w którego liściach są zapisane elementy ciągu α, każdy węzeł wewnętrzny utrzymuje następujące informacje:

- liczba liści,
- wartość skrajnie lewego liścia,
- wartość skrajnie prawego liścia,
- najdłuższy spójny podciąg rosnący w poddrzewie,
- najdłuższy spójny podciąg rosnący w poddrzewie który jest prefiksem podciągu z poddrzewa,
- najdłuższy spójny podciąg rosnący w poddrzewie który jest sufiksem podciągu z poddrzewa,

Możemy pokazać, że każdy z tych atrybutów jesteśmy w stanie aktualizować w czasie $O(1)$ na podstawie wartości z poddrzew.

7 Egzamin 2016/2017, zadanie 3

Zaprojektuj efektywny algorytm, który dla danych słów x, y nad alfabetem $\{d, i, k, s\}$ obliczy ile różnych słów będących cyklicznymi przesunięciami słowa x jest podsłowami słowa y.

Rozwiązanie:

Budujemy drzewo sufiksowe dla słowa $S = y\$xx\#$. Następnie dla każdego węzła na głębokości $|x|$ sprawdzamy czy w jego poddrzewie istnieją sufiksy zaczynające się zarówno we fragmencie y jak i we fragmencie xx.