Rozpoznawanie wzorców w tekście.

24 listopada 2016

Algorytm Knutha-Morrisa-Pratta

Intuicja Załóżmy że w tekście AABAAAAABAAA.... chcemy wyszukać wzorzec AABAAB. Zaczynamy porównywać poszczególne litery
AABAA**A**AABAAA
AABAA**B**
Różnica w tekście występuje na 5 pozycji (rozpoczynając numerację od 0). W algorytmie naiwnym następna interacja polegałaby na rozpoczęciu sprawdzania 1 znaku tekstu z 0 znakiem wzorca. Przyglądając się przykładowi, można zauważyć, że istnieje inne wyjście z sytuacji. Ponieważ zgodny ciąg wzorca i tekstu AABAA posiada prefikso-sufiks AA (to znaczy ciąg ten rozpoczyna i kończy się literami AA), algorytm może od razu porównać 2 znak wzorca z 5 znakiem tekstu (dwa poprzednie znaki są zgodne, ponieważ wiemy że jest to prefikso-sufiks).
AABAA**A**AABAAA
AA**B**AAB
Wykorzystanie prefikso-sufiksów poszukiwanego wzorca znacząco redukuje liczbę operacji potrzebną do jego znalezienia.
tekst: ABCEABCDABCDAG
wzór: ABCDABG
Przebieg algorytmu:

1. krok, porównywanie znaków tekstu i wzorca, rozpoczynając od początko

   ABC**E**ABCDABCDABG

   ABC**D**ABG

   Różnica występuje przy 3 znaku. Ponieważ w sprawdzonym ciągu - ABC nie ma powtarzających się znaków na początku i końcu (brak prefikso-sufiksa), należy porównać 3 znak tekstu z 0 znakiem wzorca. Znaki się nie zgadzają.

2. krok, ponieważ znaki się nie zgadzają, rozpoczynamy porównywanie od porównania 4 znak tesktu z 0 znakiem wzorca

   ABCEABCDAB**C**DABG

   ABCDAB**G**

   Różnica występuje przy 9 znaku tekstu. W sprawdzonym ciągu ABCDAB, powtarzającymi się znakami na początku i końcu są znaki AB (AB jest prefikso-sufiksem). Możemy przejść do sprawdzenia 9 znaku testu z 2 znakiem wzorca (zgodność znaku 7 tekstu z 0 znakiem wzorca i 8 znaku tekstu z 1 wzorca mamy zagwarantowany).

3. krok

   ABCEABCDABCDABG

   ABCDABG

   wzorzec został znaleziony

W celu zaimplementowaniu algorytmu najpierw należy stworzyć tablicę prefikso-sufiksów poszukiwanego wzorca.
Abstrakcyjny zapis postępowania prowadzącego do stworzenia tablicy prefikso-sufiksów:

1. Deklaracja zmiennych
   - pomocnicze i,j
   - tablica prefikso-sufiksów t o długości wzorca w
   - i,j=0 oraz t[0]=0

2. Dla i z zakresu (1, długość(w)):

   jeżeli w[i] == w[j] to t[i] = j+1, j=j+1

   jeżeli w[i]!=w[j] i j =0 to t[i]=0 i wykonaj kolejną iterację dla i

   jeżeli w[i]!=w[j] i j>0 to j=t[j-1] i sprawdź poprzednie punkty

Przykładowy przebieg algorytmu dla wzorca:
wzorzec: w = ABCDABCA
0 1 23 4 5 67
tablica prefikso-sufiksów: t = [0,0,0,0,0,0,0,0]
zmienne pomocnicze i,j

1. Początkowo ustawiamy: t[0]=0
2. ustawiamy j=0 i=1, sprawdzamy czy w[i]==w[j]] (w[1]==w[0] tzn. czy A==B), ponieważ tak nie jest wstawiamy t[1] = 0
3. ustawiamy i=2, sprawdzamy czy w[i]==w[j] (w[2]==w[0] tzn. czy A==C), ponieważ tak nie jest wstawiamy t[2] = 0
4. ustawiamy i=3, sprawdzamy czy w[i]==w[j] (w[3]==w[0] tzn. czy A==D), ponieważ tak nie jest wstawiamy t[3] = 0
5. ustawiamy i=4, sprawdzamy czy w[i]==w[j] (w[4]==w[0] tzn. czy A==A), ponieważ tak jest wstawiamy t[4] = j+1, czyli t[4] = 1, ustawiamy j=1
6. ustawiamy i=5, sprawdzamy czy w[i]==w[j] (w[5]==w[1] tzn. czy B==B), ponieważ tak jest wstawiamy t[5] = j+1, czyli t[5] = 2, ustawiamy j=2
7. ustawiamy i=6, sprawdzamy czy w[i]==w[j] (w[6]==w[2] tzn. czy C==C), ponieważ tak jest wstawiamy t[6] = j+1, czyli t[6] = 3, ustawiamy j=3
8. ustawiamy i=7, sprawdzamy czy w[i]==w[j] (w[7]==w[3] tzn. czy A==D), ponieważ tak nie jest, ustawiamy j=t[j], czyli j=t[3]=0, sprawdzamy czy w[i]==w[j] (w[7]==w[0], czyli czy A==A), ponieważ tak jest wstawiamy t[7]=j+1=1

Abstrakcyjny zapis algorytmu

1. Wygenerowanie tablicy prefikso-sufiksów TPS
2. inicjalizacja zmiennych i,j=0
3. rozpoczęcie porównywania wzorca i tekstu

   dopóki i+t<długość tekstu

   jeżeli wzorzec[i]==tekst[i+j] to

   jeżeli i = długość wzorca -1 to wzorzec został znaleziony

   w innym wypadku i =i+1
4. jeżeli wzorzec[i]!=tekst[i+j] to

   jeżeli i>0 to j = j +i - TPS[i-1], i =TPS[i-1]

   w innym wypadku i=0, j=j+1

Punktacja

1. zaimplementowanie algorytmu obliczającego tablicę prefikso-sufiksów- 4 punkty
2. zaimplementowanie algorytmu przeszukującego tekst - 6 punktów