SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

PROJEKT IZ BIOINFORMATIKE

Ukkonenov Algoritam

Mislav Marković  
Filip Matijević  
Marko Raguž

Voditelj: Mislav Marković

Zagreb, siječanj, 2019.

**Sadržaj**

[1. Uvod 4](#_Toc535239059)

[2. Opis algoritma 5](#_Toc535239060)

[3. Rezultati testiranja 11](#_Toc535239061)

[4. Zaključak 13](#_Toc535239062)

[5. Literatura 14](#_Toc535239063)

# Uvod

Da bismo objasnili kako radi Ukkonenov algoritam potrebno je prvo objasniti što je to sufiksno stablo. Sufiksno stablo je je komprimirani trie koji sadrži sve sufikse zadatnog teksta gdje su ključevi sufiksi a vrijednosti su pozicije u originalnom tekstu.

Naivna implementacija sufiksnog stabla zahtjeva O(n^2) ili čak O(n^3) vremensku složenost.

Ukkonenov Algoritam je online (slovo po slovo) algoritam koji gradi sufiksno stablo u linearnom vremenu, predložen od Eska Ukkonena prema kojem je dobio i ime.

# Opis algoritma

Struktura podataka, tj. Sufiksno stablo, koje na kraju trebamo stvoriti iz danog teksta je poput search trie-a ali umjesto jednog znaka po bridu kao oznaka brida koristit će se par brojeva [od, do]. To su pokazivači na originalni text te tako svaki brid predstavlja String proizvoljne duljine, ali zauzima samo O(1) memorije jer se spremaju samo 2 pokazivača.

Objasnit ćemo algoritam ukratko na jednostavnom primjeru. Za početak potrebne su dodatne varijable.

1. Aktivna točka - trojka( aktivni\_čvor, aktivni\_brid, aktivna\_dužina). Govori nam gdje moramo početi sa umetanjem novog sufiksa.
2. Ostatak – Govori nam koliko suffiksa moramo umetnuti izravno. Npr ako je tekst abcdefg, a ostatak je 3 onda moramo procesirati zadnja 3 sufiksa “efg”, “fg”, te “g”.

Uzmimo da nam je tekst “abcabd”. Prvo dodajemo specijalni znak $ na kraj teksta tako da je krajnji tekst “abcabd$”.

Korak 1: obrađeno : “”

aktivni\_čvor: 0

aktivni\_brid: none

aktivna\_dužina: 0

ostatak: 0



Korak 2: obrađeno: “a”

aktivni\_čvor: 0

aktivni\_brid: none

aktivna\_dužina: 0

ostatak: 0



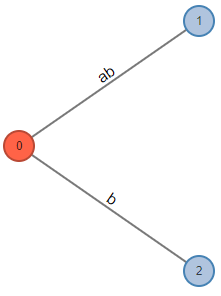
Korak 3: obrađeno: “ab”

aktivni\_čvor: 0

aktivni\_brid: none

aktivna\_dužina: 0

ostatak: 0



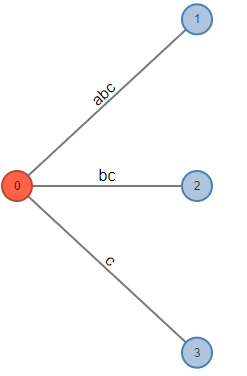
Korak 4: obrađeno: “abc”

aktivni\_čvor: 0

aktivni\_brid: none

aktivna\_dužina: 0

ostatak: 0



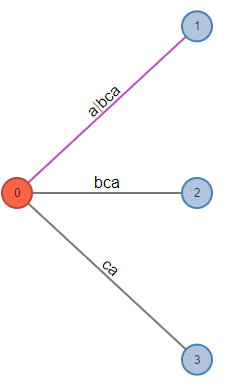
Korak 5: obrađeno: “abca”

aktivni\_čvor: 0

aktivni\_brid: a

aktivna\_dužina: 1

ostatak: 1



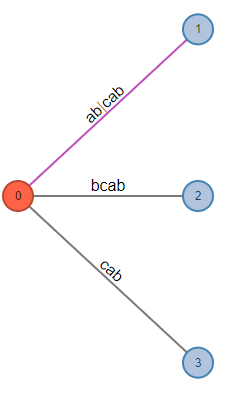
Korak 6: obrađeno: “abcab”

aktivni\_čvor: 0

aktivni\_brid: a

aktivna\_dužina: 2

ostatak: 2



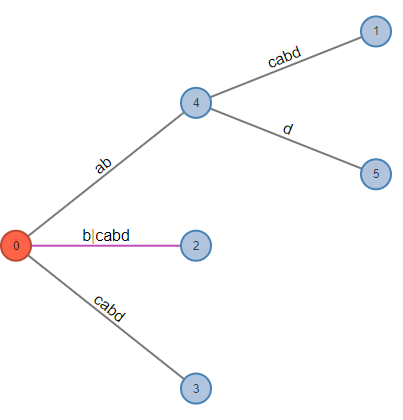
Korak 7: obrađeno: “abcabd”

aktivni\_čvor: 0

aktivni\_brid: b

aktivna\_dužina: 1

ostatak: 1



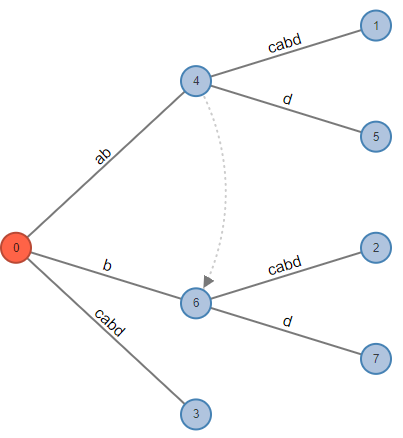
Korak 8: obrađeno: “abcabd”

aktivni\_čvor: 0

aktivni\_brid: none

aktivna\_dužina: 0

ostatak: 0



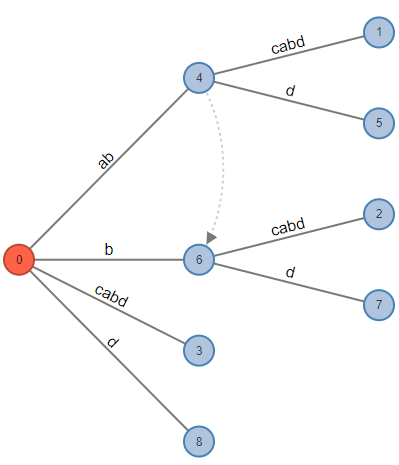
Korak 9: obrađeno: “abcabd”

aktivni\_čvor: 0

aktivni\_brid: none

aktivna\_dužina: 0

ostatak: 0



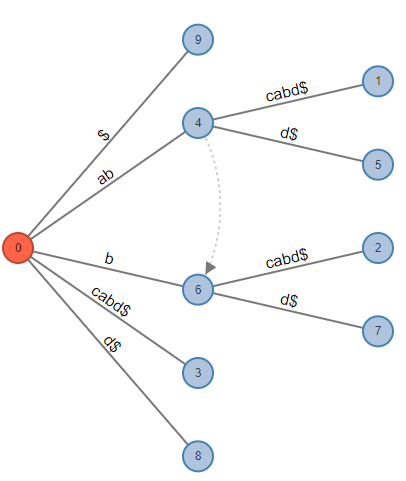
Korak 10: obrađeno: “abcabd$”

aktivni\_čvor: 0

aktivni\_brid: none

aktivna\_dužina: 0

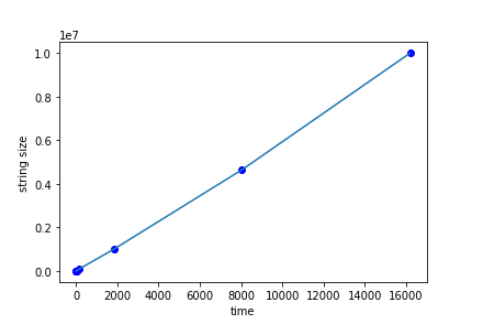
ostatak: 0



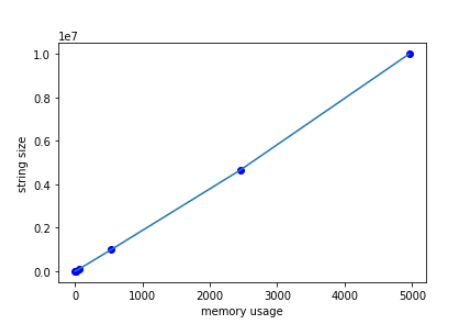
Time je sufiksno stablo dovršeno. Svi sufiksi se eksplicitno nalaze u stablu. Ovdje je algoritam objašnjen na jednostavnom primjeru, za detaljnije definicije i objašnjenje pogledati u poveznice koje su stavljene pod literaturu.

# Rezultati testiranja

Algoritam smo testirali na tekstovima različite duljine, tj. 100, 10k, 100k, 1mil, te 10mil znakova. Vremenska i memorijska složenost je prikazana na grafovima.   
Algoritam je testiran na procesoru Intel(R) Core(TM) i7- 4710HQ CPU @ 2.50GHz 2.5GHz s radnom memorijom od 12GB.



Vremenska složenost



Memorijska složenost

# Zaključak

Testiranjem smo uočili da je i vremenska i memorijska složenost linearna, što je puno bolje od naivne implementacije sufiksnog stabla( O(n^2)/ O(n^3) vremenska složenost). Ukkonenov algoritam je također on-line algoritam, u svakom trenutku ima sufiksno stablo spremno od trenutno pročitanog teksta, što može biti korisno ukoliko nemamo potpuni tekst u početku.

# Literatura

1.) <https://www.cs.helsinki.fi/u/ukkonen/SuffixT1.pdf>

2.) <https://en.wikipedia.org/wiki/Ukkonen%27s_algorithm>

3.) <https://stackoverflow.com/questions/9452701/ukkonens-suffix-tree-algorithm-in-plain-english>

4.) <https://en.wikipedia.org/wiki/Suffix_tree>

5.) <https://en.wikipedia.org/wiki/Trie>