SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

PROJEKTNA DOKUMENTACIJA

**LCSk++: mjera sličnosti za dugačke nizove znakova**

Dario Bošnjak

Renato Bošnjak

Dorian Ljubenko

Matija Marić

Matea Torbarina

Zagreb, siječanj, 2019.

**Sadržaj**

[1. Uvod 1](#_Toc534586175)

[2. Opis algoritma 2](#_Toc534586176)

[3. Rezultati testiranja 3](#_Toc534586177)

[4. Zaključak 4](#_Toc534586178)

[5. Sažetak 5](#_Toc534586179)

[6. Literatura 6](#_Toc534586180)

# Uvod

Usporedba sličnosti znakovnih nizova bitan je dio mnogih algoritama. Neke od primjena uključuju otkrivanje plagijata, sustave za verzioniranje koda, programe za usporedbu datoteka (alat diff na Unix sustavima) te bioinformatiku.

Često korištene metrike za ovaj problem su najduži zajednički podniz (engl. *longest common subsequence*, LCS) ili dužina uređivanja (engl. *edit distance*).

Zbog vrlo dugačkih nizova koji se pojavljuju kod primjena u bioinformatici (npr. genom), navedene metrike nisu primjenjive.

Metriku LCS za dva znakovna niza moguće je aproksimirati algoritmima (Baker, i dr., 2002) koji imaju vremensku složenost , odnosno , gdje *T* označava broj podudarajućih dijelova u nizovima, a *n, m* duljinu dvaju nizova.

Novi radovi iznijeli su poboljšanja, tako su (Benson, i dr., 2013) definirali mjeru LCS*k* koja pronalazi najveći broj nepreklapajućih podnizova duljine *k*. Predloženi algoritam imao je vremensku i prostornu složenost.

Korištenjem strukture podataka crveno-crnog stabla, poboljšanje predloženog algoritma (Deorowicz, i dr., 2014) donijelo je vremensku složenost koja iznosi i prostornu složenost. Oznake *m* i *n* jednake su kao i u prethodnom algoritmu, oznaka *r* definira ukupni broj podudarajućih podnizova duljine *k* u dva znakovna niza, a oznaka *l* označava duljinu optimalnog rješenja.

Iako su vremenska i prostorna složenost umanjene, algoritam LCS*k* i dalje nije savršen jer pretpostavlja da su podudarajući podnizovi točno duljine *k*.  
Problem se jasno uočava na sljedećem primjeru: *X* = ABCDE, *Y* = ABCDE, *Z* = ABCFG, *k* = 3. Označimo nepreklapajuća podudaranja podnizova duljine 3 podebljanim slovima:

1. **ABC**DE, **ABC**DE
2. **ABC**DE, **ABC**FG

Vidimo da LCS3 mjera, za sve parove dvaju ulaznih nizova, iznosi 1 jer se podudarajući podnizovi duljine 3 ne smiju preklapati.

Ovaj problem rješava mjera LCS*k*++ (Pavetić, i dr., 2014) koja relaksira uvjet dužine podudarajućih podnizova tako što zahtijeva da oni budu barem duljine *k*.

# Opis algoritma

U ovom poglavlju bit će opisane potrebne strukture podataka i logika algoritma.

## Strukture podataka

Jedna od prednosti mjere LCS*k*++ je jednostavnost implementacije. Algoritam za izračun mjere, u jednostavnoj, ali neefikasnoj varijanti, ne mora koristiti posebne strukture podataka, osim običnog polja.

### Fenwickovo stablo (binarno indeksirano stablo)

Međutim, zbog vrlo dugačkih nizova, kakvi se u području bioinformatike često pojavljuju, jednostavna implementacija algoritma postaje beskorisna. Zato se umjesto polja, u jednom dijelu algoritma koristi Fenwickovo stablo, poznato i pod nazivom binarno indeksirano stablo.

#### Problem

Imamo polje A veličine *n* elemenata, želimo ostvariti sljedeće dvije operacije:

1. Povećaj element na *i*-tom mjestu u polju – operacija *update*
2. Izračunaj sumu prvih *i* brojeva u polju A – operacija *query*

Ako koristimo strukturu polja, prvu operaciju možemo izvesti s vremenskom složenosti , dok nam za drugu vremenska složenost u najgorem slučaju iznosi .

#### Rješenje

Korištenje Fenwickovog stabla omogućuje izvođenje obje operacije s vremenskom složenosti , dok se memorijska složenost ne mijenja u odnosu na strukturu polja. Umjesto operacije dohvaćanja sume prvih *i* brojeva, algoritam LCS*k*++ koristi modificirano Fenwickovo stablo za sljedeće operacije, bez promjene vremenske složenosti:

1. Postavi element na *i*-tom mjestu u polju na vrijednost *v*, ako vrijedi *v*>A[*i*]
2. Dohvati najveći od prvih *i* elementa polja

## Način rada algoritma

# Rezultati testiranja

# Zaključak

# Sažetak

# Literatura

**Baker, Brenda i Giancarlo, Raffaele. 2002.** Sparse dynamic programming for longest common subsequence from fragments. *Journal of Algorithms.* 2002, Svez. 42, 2, str. 231--254.

**Benson, Gary, Levy, Avivit i Shalom, Riva. 2013.** *International Conference on Similarity Search and Applications.* s.l. : Springer, 2013. str. 257--265.

**Deorowicz, Sebastian i Grabowski, Szymon . 2014.** *Information Processing Letters.* s.l. : Elsevier, 2014. str. 634--638. Svez. 114.

**Pavetić, Filip, Žužić, Goran i Šikić, Mile . 2014.** LCSk++: Practical similarity metric for long strings. *arXiv preprint.* 2014.