Fakultet elektrotehnike i računarstva

Zavod za primjenjeno računarstvo

**Napredni algoritmi i strukture podataka**

2. laboratorijska vježba

Tyrannizer, 10563

Zagreb, 10.12.2016.

# Zadatak

Zadatci za 11 bodova

Napisati program (poželjno C, C++, C# ili Java) s jednostavnim grafičkim korisničkim sučeljem koji će pretražiti zadani tekst nekim od algoritama koji se obrađuju u predmetu NASP (ne „naivni“). Može se primijeniti i algoritam izvan gradiva NASP, a bolji od naivnog, samo tada treba opisati i teorijsku pozadinu te dokazati ispravnost i složenost.

−sučelje može biti i samo jedan obrazac (forma) s dva polja za tekst (text-box); jedno za unos teksta koji se pretražuje, a drugo za unos traženog niza znakova

−polje s tekstom koji se pretražuje može biti i mjesto za prikaz rezultata; dovoljno je pronađene uzorke nekako označiti (recimo drugom bojom ili podebljati itd.). Uglavnom, moraju biti lako uočljivi.

# Rješenje zadatka

## Teorijski uvod

Pretraživanje teksta formalno se definira na sljedeći način. Tekst je polje *T* duljine *n* i uzorak je polje *P* duljine *m*. Pretpostavlja se da elementi od *P* i *T* znakovi iz konačne abecede, npr. ili [1]. U ovom rješenju implementirani su algoritmi Rabin-Karp i Knuth-Morris-Pratt.

## Implementacija

Implementacija je napisana u jeziku C# koristeći razvojno okruženje Visual Studio 15, za implementaciju nije korištena nijedna vanjska biblioteka već isključivo standardna biblioteka jezika C# (odnosno .NET framework). Implementiran je grafički program koji sadrži jednu formu na kojoj se nalaze kontrole za odabir algoritma, unos teksta, unos uzorka i prikaz rezultata.

### Pretraživanje Rabin-Karp algoritmom

Rabin-Karp algoritam ima složenost od za preprocesiranje i složenost pretraživanja.

Pseudokod funkcije za pretraživanje dan je u nastavku.

Rabin-Karp-Searcher(T, P, d, q)

n = T.length

m = P.length

h = dm-1 % q

p = 0

t0 = 0

for i = 1 to m

p = (dp + P[i]) % q

t0 = (dt0 + T[i] % q

for s = 0 to n – m

if p == ts

if P[1..m] == T[s + 1..s + m]

print s

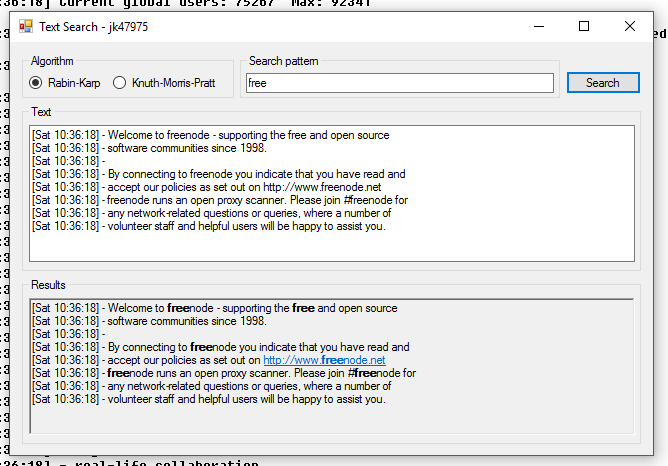
if s < n – m

ts + 1 = (d(ts – T[s+1]/h) + T[s + m + 1]) % q

Ako je *P = am* i T = an pretraživanje traje budući da se na svakom pomaku nalazi uzorak koji treba validirati. U mnogim primjenama očekivano je da se nalazi tek nekoliko pojava uzorka, npr. neka konstanta *c*. U tom slučaju vrijeme pretraživanja je Budući da je očekivano vrijeme je [1].

Za pretraživanje ovim algoritmom potrebno je u gornjem lijevom kutu odabrati algoritam, upisati tekst, upisati pojam za pretraživanje i kliknuti na gumb „Search“.

Nakon toga, u donjem polju prikazuju se rezultati pretraživanja, pojave uzorka u tekstu prikazane su podebljano.



### Pretraživanje Knuth-Morris-Pratt algoritmom

Knuth-Morris-Pratt algoritam ima složenost od za preprocesiranje i složenost pretraživanja.

Pseudokod funkcije za pretraživanje dan je u nastavku.

Knuth-Morris-Pratt-Searcher(T, P)

n = T.length

m = P.length

lps = Compute-Prefix-Function(P)

q = 0

for i = 1 to n

while q > 0 and P[q + 1] != T[i]

q = lps[q]

if P[q + 1] == T[i]

q++

if q == m

print i – m

q = lps[q]

Funkcija koristi dodatnu pomoćnu funkciju koja računa tablicu prefiksa za uzorak pretraživanja koja govori kako uzorak odgovara pomacima od sebe samog. Ova tablica se koristi kako bi se izbjegla testiranja nepotrebnih pomaka [1].

Compute-Prefix-Function(P)

m = P.length

lps = int[m]

lps[0] = 0

k = 0

for q = 2 to m

while k > 0 and P[k + 1] != P[q]

k = lps[k]

if P[k + 1] == P[q]

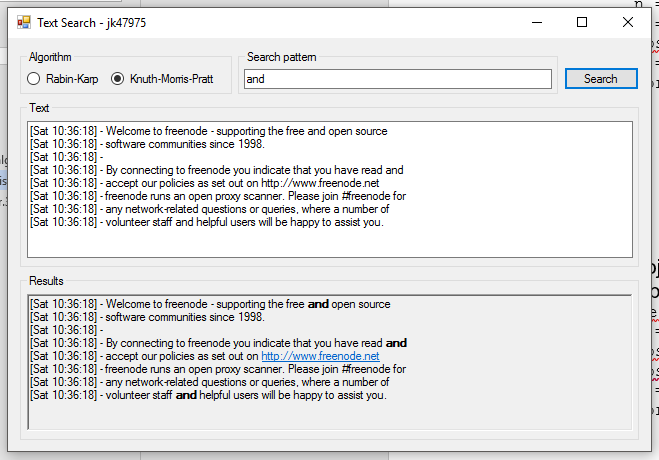
k++

lps[q] = k

return lps

Za pretraživanje ovim algoritmom potrebno je u gornjem lijevom kutu odabrati algoritam, upisati tekst, upisati pojam za pretraživanje i kliknuti na gumb „Search“.

Nakon toga, u donjem polju prikazuju se rezultati pretraživanja, pojave uzorka u tekstu prikazane su podebljano.



# Zaključak

Pretraživanje teksta je zanimljivo područje s mnogo primjena, na primjer provjera plagijata ili pretraživanje podataka u bazi. Moguća poboljšanja rješenja ovog rješenja su implementacija nekih drugih algoritama poput Boyer-Moore ili Aho-Corasick i Commentz-Walter algoritama koji su proširenja ovdje implementiranih algoritama s mogućnošću pretraživanja više uzoraka odjednom.

# Literatura

Tablica 1. Formati citiranja literature.[[1]](#endnote-1)

|  |  |
| --- | --- |
| **Vrsta** | **Format** |
| Poglavlje knjige | Cormen, et. al.; String-Matching, Introduction to Algorithms, MIT Press, 2009, 985 - 1013 |
|  |  |

1. Derivat IEEE stila citiranja kojeg također možete koristiti. Više detalja o IEEE stilu na: <http://www.ieee.org/documents/ieeecitationref.pdf> [↑](#endnote-ref-1)