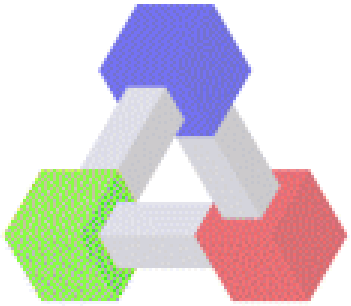


Katedra za informatiku, Fakultet tehničkih nauka Novi Sad

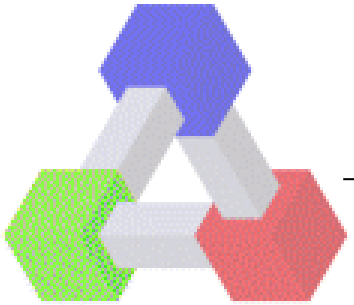
Projekat iz predmeta: Soft Computing

Korišćenje probabilističkih metoda za poređenje i ispravku pogrešno unetih naziva ulica „Probabilistic Street Matching"

Profesor:
doc. dr Đorđe Obradović
Asistent:
Marko Jocić

2014/15.

Projektni tim:
Rade Radišić, RA 69/2011



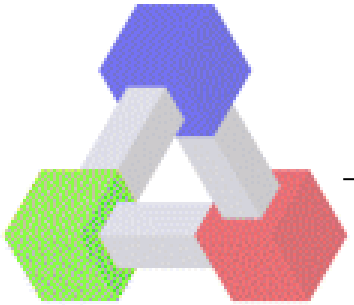
Zadatak

- Implementirati sistem za poređenje i pronalaženje istih imena ulica između zvaničnog registra ulica grada Novog Sada i ručno upisivanih adresa prilikom popunjavanja formi

- Prepoznavanje istih entiteta je veoma popularna tema, samim tim postoje i alati koji to olakšavaju (npr. OYSTER)

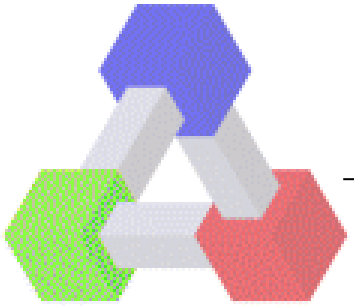
- Algoritam bi se sastojao od probabilističkog pojedinačno svake torke iz ciljane baze sa svakom torkom iz rečnika, gde će poređenje koje je u najvećem procentu tačno rezultirati ubacivanjem torke iz rečnika na mesto stare torke

- Verifikacija bi predstavljala mogućnost da registar ulica predstavlja referencijalni integritet nad „popravljenom" bazom



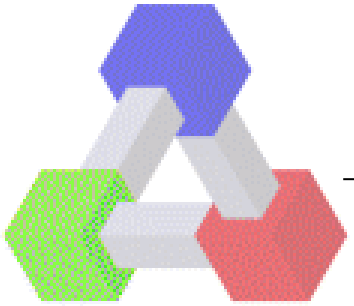
Uvod

- Istorija
- Metode
- Matematički model probabilstičkog poređenja



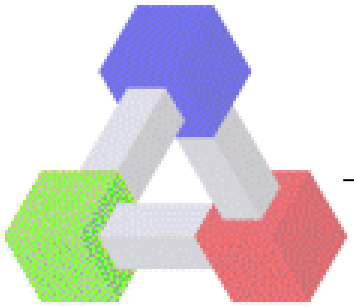
Uvod

- Istorija
  - 1946. Halbert L. Dunn – „Record Linkage“
  - 1959. H. B. Newcombe, J.M. Kennedy, S.J. Axford, A. P. James „Automatic Linkage of Vital Records“
  - Fellegi, Ivan, Sunter, Alan (December 1969). „A Theory for Record Linkage“
    - Predstavlja matematičku osnovu i formalizam za probabilističko poređenje; sve ostale teorije se oslanjaju na ovu
  - ...



Uvod

- Metode
  - Data preprocessing
  - Identity Resolution
  - Data Matching
  - Deterministic record linkage
  - Probabilistic record linkage



Uvod

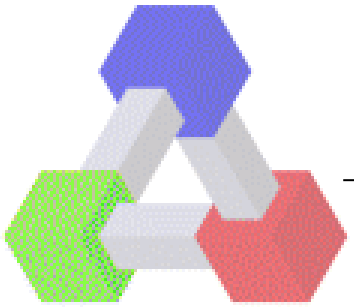
- Matematički model
  - Identični i različiti entiteti smeštaju se u skupove respektivno

$$M = \{(a,b); a = b; a \in A; b \in B\}$$

$$U = \{(a,b); a \neq b; a \in A, b \in B\}$$

  - Definisan je skup koji sadrži nizove sličnosti i razlika u svakoj od osobina K

$$\gamma\left[\alpha(a), \beta(b)\right] = \{\gamma^1\left[\alpha(a), \beta(b)\right], ..., \gamma^K\left[\alpha(a), \beta(b)\right]\}$$

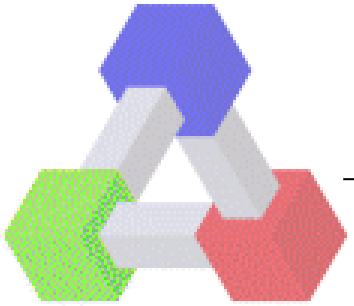


Uvod

- Matematički model
  - Na osnovu prethodnog navedenog, može se izračunati uslovna verovatnoća da su karakteristike slične, odnosno različite

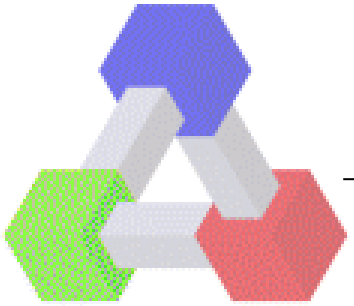
$$m(\gamma) = P\{\gamma[\alpha(a), \beta(b)] \,|\, (a,b) \in M\} = \sum_{(a,b)\in M} P\{\gamma[\alpha(a), \beta(b)]\} \cdot P[(a,b)|M]$$

$$u(\gamma) = P\{\gamma[\alpha(a), \beta(b)] \,|\, (a,b) \in U\} = \sum_{(a,b)\in U} P\{\gamma[\alpha(a), \beta(b)]\} \cdot P[(a,b)|U]$$



Uvod

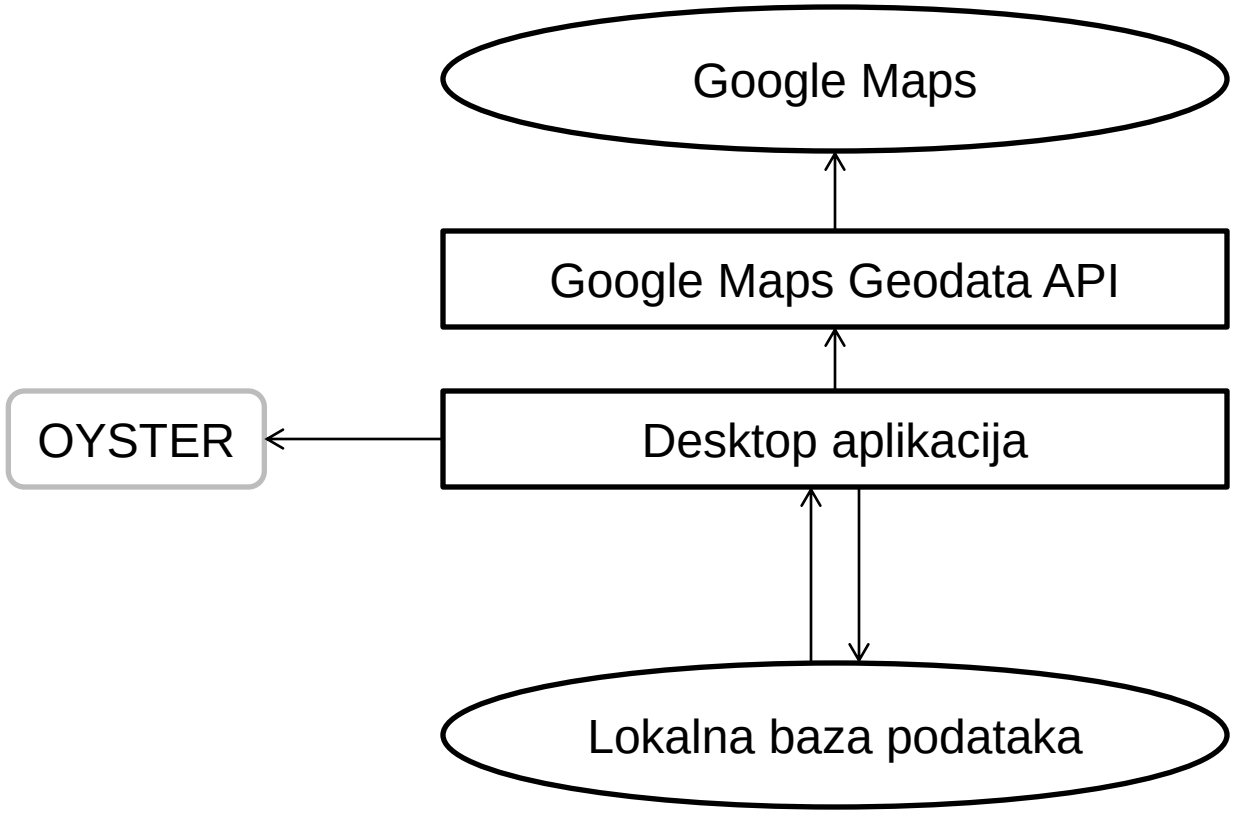
- Pregled postojećeg stanja u domenu problema „Entity Resolution“
  - „Stanford Entity Resolution Framework“ - http://infolab.stanford.edu/serf/
  - OYSTER Entity Resolution - http://sourceforge.net/projects/oysterer/

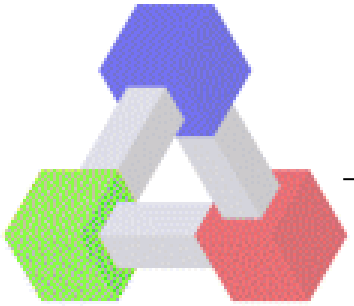


Implementacija

- Preprocesiranje podataka
  - Sva imena ulica moraju biti napisana latiničnim pismom

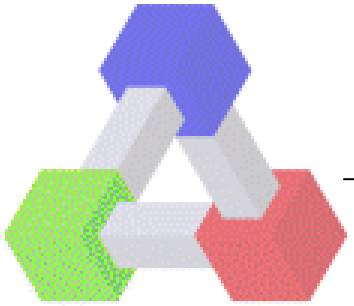
Implementacija





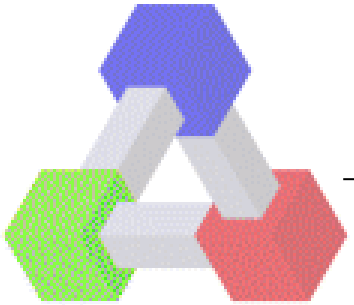
Implementacija

- Inicijalizacija:
  - Iz Google Maps baze podataka slanjem HTTP/GET zahteva sa koordinatama na šta će API vratiti odgovarajuće podatke
    - Napomena: koordinate se ograničavaju na površinu pravougaonika koja pokriva teritoriju grada Novog Sada
  - Iz dobijenih podataka će se uzeti polje sa adresom, iz kog će se parsirati ime ulice
  - Poslaće se upit lokalnoj bazi sa prethodno isparsiranim imenom ulice, te ukoliko ono ne postoji, dodaće se kao nova torka



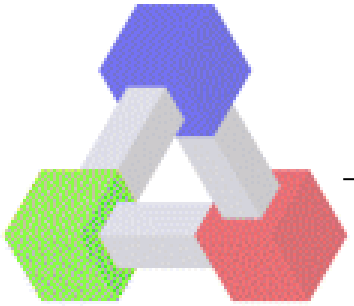
Implementacija

- Uklanjanje duplikata
  - Za svaki entitet vršiće se prolaz kroz sve ostale i korišćenjem OYSTER softverskog paketa, odnosno funkcionalnosti probabilistic i direct linking koje poseduje, vršiće se uklanjanje duplikata
  - Duplikati će biti zamenjeni jednom vrednošću koja sa svim ostalim ima najveći procenat poklapanja, odnosno, sve ostale će biti uklonjene iz lokalne baze podataka



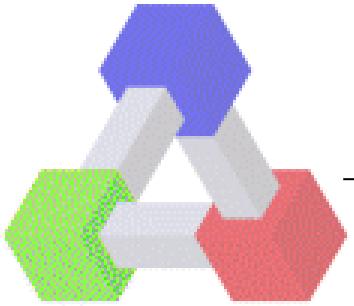
Implementacija

- Prilikom implementacije rešenja, koristiće se sledeće tehnologije i protokoli
  - Prilikom dobavljanja podataka iz baze Google Maps
    - HTTP Protocol
    - JSON (XML)
  - Desktop aplikacija, parsiranje i poređenje
    - Ruby
    - ~~OYSTER~~ (ipak izbačen, iz razloga navedenih na sl. slajdu)
  - Lokalna baza podataka
    - MySQL



Zašto ne OYSTER?

- Daje mnogo više nego što je potrebno za ovaj projekat
- Nedovoljno je dokumentovan
- XML skripte, komplikovane za korišćenje, alat koji ih generiše je još u fazi razvoja
- Njegova korisnost primetna je tek u radu sa bazama podataka sa većim brojem atributa



Rešenje

- Upotreba Jaro-Winkler rastojanja
- Jedna od Record Linkage metoda
- Sličnost stringova skalira se na vrednost između 0 i 1
  - 0 za potpuno različite stringove, 1 za deterministički iste stringove

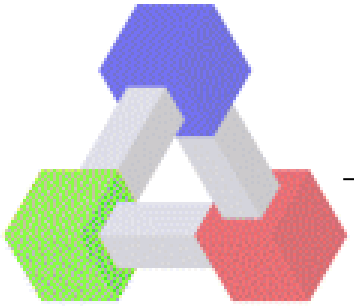
Jaro rastojanje

$$d_j = \begin{cases} 0 & \text{if } m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) & \text{otherwise} \end{cases}$$

- m – broj karaktera koji se poklapaju
- t – polovina broja transpozicija karaktera

- Karakteri se poklapaju ako i samo ako je njihovo rastojanje manje od

$$\left\lfloor \frac{\max(|s_1|, |s_2|)}{2} \right\rfloor - 1.$$



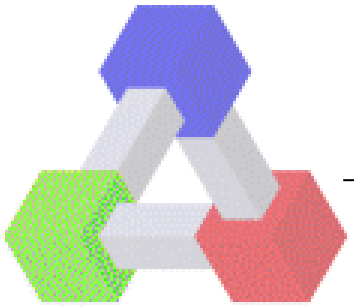
Primer Jaro rastojanja

- Za stringove 'Pera' i 'Pear', Jaro rastojanje dobija se na sledeći način:

  $m = 4, s_1 = 4, s_2 = 4$

  $t = 2/2 = 1$ (r-a, a-r)

  $d_j = 1/3 * (4/4 + 4/4 + (4-1)/4) \approx 0{,}917$

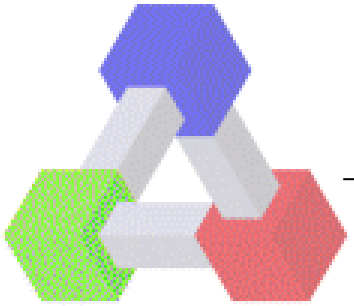


Jaro – Winkler rastojanje

- Koristi Jaro rastojanje, ali dodaje još i ponderišući faktor, da približi jedinici sličnije stringove

$$d_w = d_j + (\ell p(1 - d_j))$$

  - d_j – Jaro rastojanje
  - l – poklapanje prefiksa kod stringova (koliko god da je inače, ne ide više od 4)
  - p – faktor skaliranja (uglavnom 0.1, ali ne sme preći vrednost 0.25, jer bi to rezultiralo poklapanjima između stringova koje je veće od maksimalne vrednosti 1)



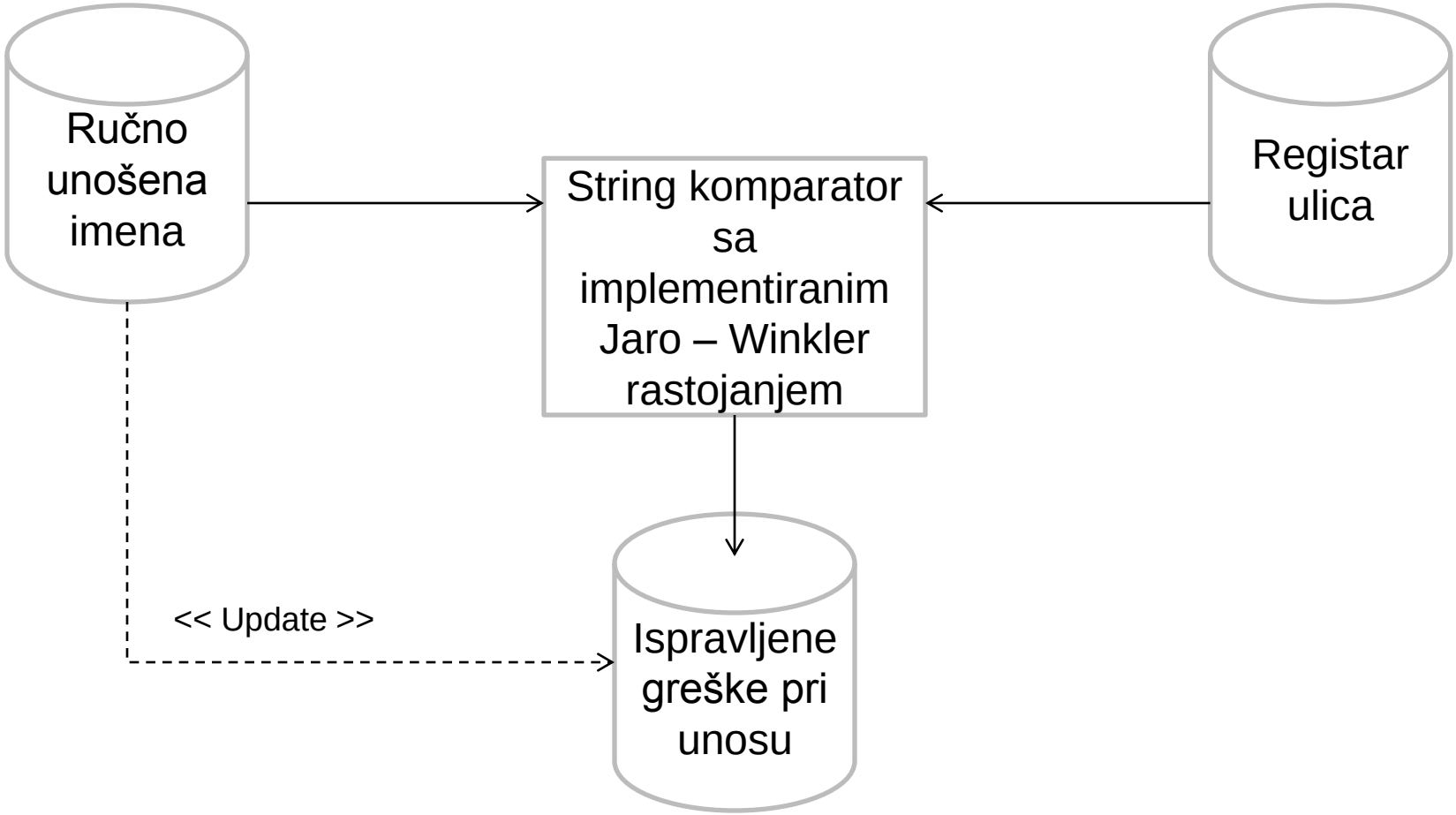
Primer Jaro – Winkler rastojanja

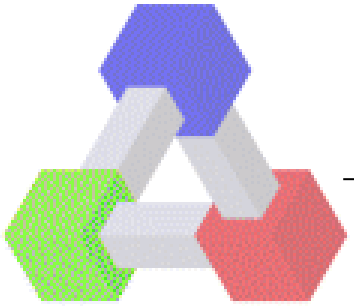
- Na osnovu Jaro rastojanja za stringove iz prethodnog primera 'Pera' i 'Pear' i težinskog faktora p = 0.1, rezultat Jaro – Winkler rastojanja biće:

  $p = 0.1, l = 2$

  $d_w = 0.917 + 2 * 0.1 * (1 – 0.917) \approx 0.933$

Prikaz rešenja



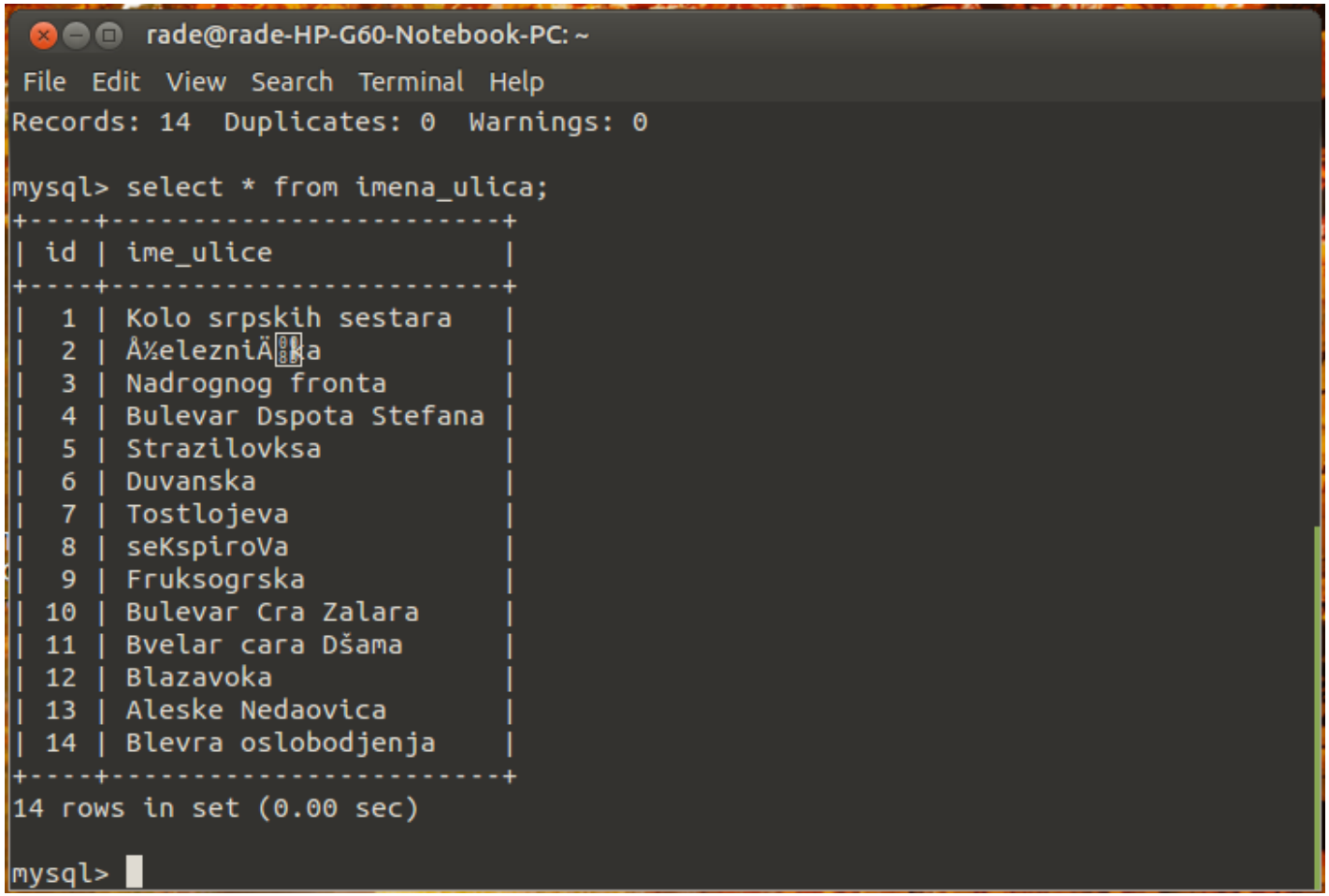


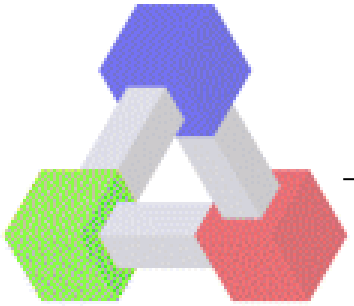
String komparator

- Za jednu torku iz ručno unošene baze traži se odgovarajući u rečniku (registar ulica), odnosno, sa maksimalnim nivoom poklapanja
- Torka sa maksimalnim nivoom poklapanja iz registra postavlja se umesto tekuće vrednosti u bazi koja se ispravlja

Izgled rada aplikacije

Početno stanje





Izgled rada aplikacije

Za vreme procesa

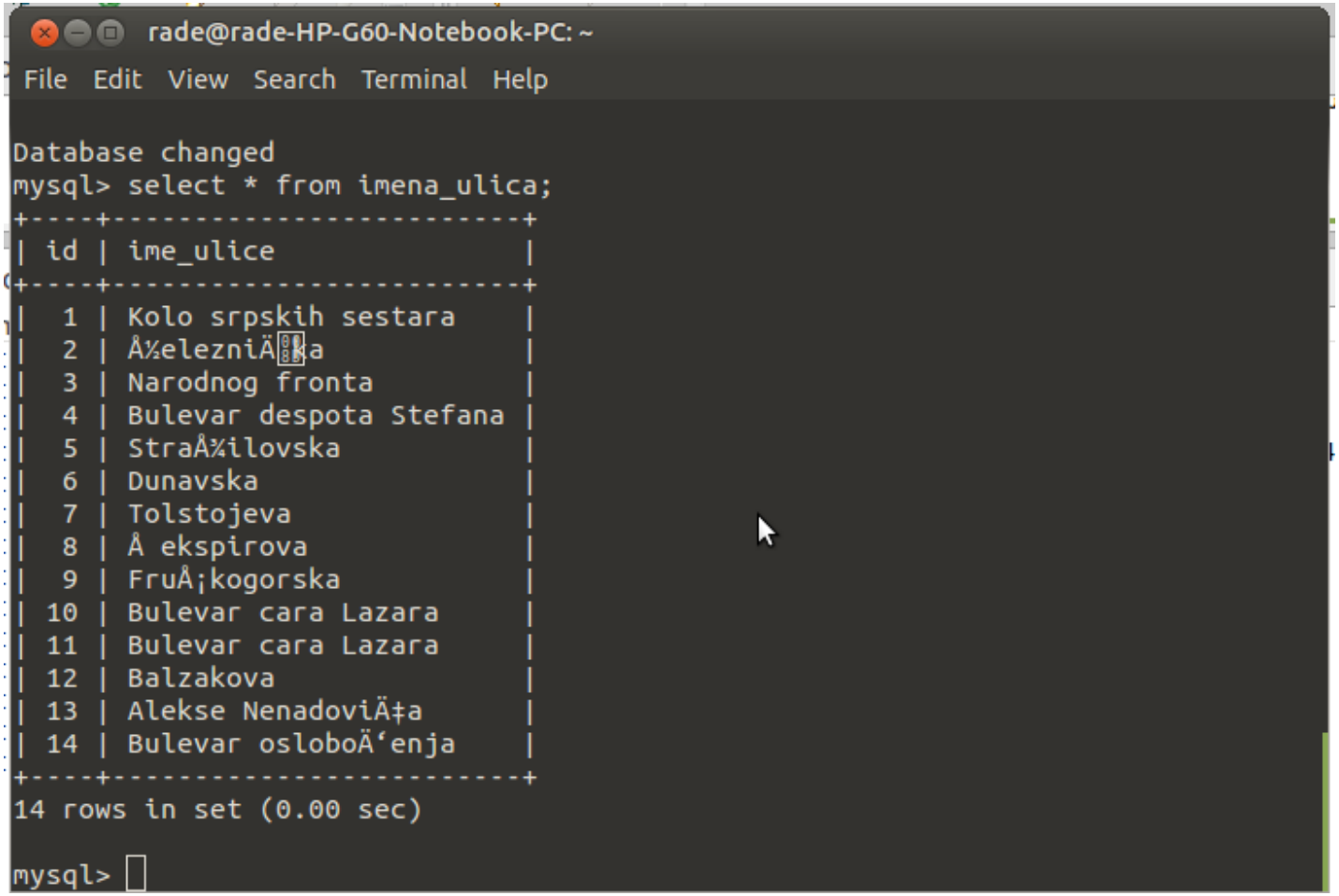
Console × Problems

<terminated> street_matcher.rb [Ruby Application] /usr/bin/ruby

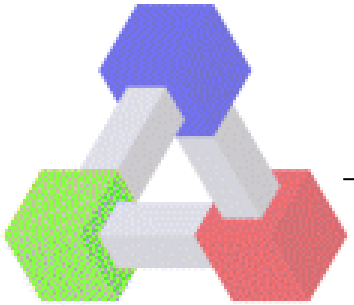
```
Best candidate for Kolo srpskih sestara is: Kolo srpskih sestara distance: 1.0
Best candidate for Železnička is: Železnička distance: 0.8222222222222223
Best candidate for Nadrognog fronta is: Narodnog fronta distance: 0.9299999999999999
Best candidate for Bulevar Dspota Stefana is: Bulevar despota Stefana distance: 0.9191040843214756
Best candidate for Strazilovksa is: Stražilovska distance: 0.9398989898989898
Best candidate for Duvanska is: Dunavska distance: 0.9666666666666667
Best candidate for Tostlojeva is: Tolstojeva distance: 0.9600000000000001
Best candidate for seKspiroVa is: Šekspirova distance: 0.7523809523809524
Best candidate for Fruksogrska is: Fruškogorska distance: 0.8932323232323233
Best candidate for Bulevar Cra Zalara is: Bulevar cara Lazara distance: 0.8653216374269005
Best candidate for Bvelar cara D?ama is: Bulevar cara Lazara distance: 0.8466828836797876
Best candidate for Blazavoka is: Balzakova distance: 0.9333333333333333
Best candidate for Aleske Nedaovica is: Alekse Nenadovića distance: 0.9268545751633988
Best candidate for Blevra oslobodjenja is: Bulevar oslobođenja distance: 0.8927244582043343
```

Izgled rada aplikacije

Stanje nakon završetka*

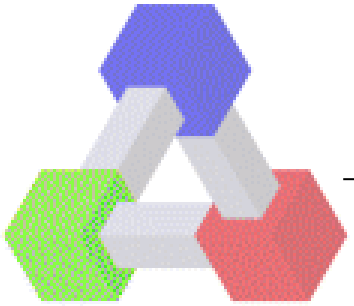


*napomena – MySQL klijent ne prikazuje dobro UTF-8 karaktere



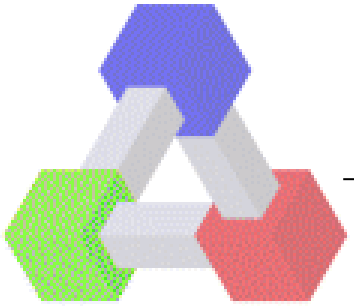
Nedostatci aplikacije

- Torke obe baze učitavaju se u program kao iterativna struktura podataka, što je za prezentaciju ovog projekta bilo sasvim korektno, ali, kao alat koji bi obrađivao veliki broj podataka ne bi, zbog memorijskog zauzeća
- Predlog rešenja: parcijalno učitavanje podataka iz baze, čuvanje rezultata na nivou lokalnog maksimuma, glavni rezutat je maksimum na nivou cele „funkcije"
  - Pogodno za paralelizaciju algoritma



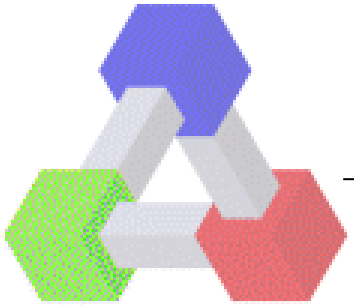
Predlog daljeg toka razvoja

- Mogućnost da korisnik bira bazu rečnik i bazu koja se popravlja, kao i odabir grada za koji će se formirati rečnik (registar) iz specijalizovane GUI aplikacije ili Web servisa, bez ikakve potrebe za intervencijom u kodu
- Otklanjanje prethodno navedenih nedostataka aplikacije i potencijalnih uskih grla u performansama



Literatura

- Ivan P. Felligi, Alan B. Sunder – „A Theory For Record Linkage“
- Winkler, W. E. - String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage



Kontakt

- Rade Radišić, RA 69/2011
  - email: radisic.rade@gmail.com