FbHash: shema za izračun podobnosti datotek v digitalni forenziki

Timotej Knez[1], Sebastian Mežnar[2], Jasmina Pegan[1]

[1]Fakulteta za računalništvo in informatiko
Univerza v Ljubljana

[2]Fakulteta za matematiko in fiziko
Univerza v Ljubljana

Kazalo

- Uvod
- Sorodna dela
- Algoritem
- Eksperimenti v članku
- Naši eksperimenti
- Zaključek

Uvod

- Avtomatizacija preiskave datotek
- Algoritmi za iskanje približnega ujemanja
  - ssdeep, sdhash, FbHash
- Prispevki članka
  - odporna shema, dve različici, analiza varnosti
- Implementacija algoritma
  - različne funkcije za uteževanje
- Testiranje na istih množicah kot v članku

Sorodna dela

- ssdeep
  - Temelji na algoritmu spamsum
  - Funkcija z drsečim oknom
- sdhash
  - Statistično najmanj verjetni deli datoteke
  - Bloomovi filtri
- MRSH-v2
  - Multi-resolution similarity hashing
  - Funkcija z drsečim oknom, Bloomovi filtri
- mvHash-B
  - Krajšanje zapisa: glasovanje, kompakten zapis
  - Bloomovi filtri

Algoritem

- FbHash-B za nestisnjene in FbHash-S za stisnjene datoteke
- Glavna komponenta so deli, sestavljeni iz k zlogov

$$
\begin{aligned}
\mathrm{ch}_0^D &= \mathrm{B}_0^D, \mathrm{B}_1^D, \mathrm{B}_2^D, \ldots\ldots, \mathrm{B}_{k-2}^D, \mathrm{B}_{k-1}^D \\
\mathrm{ch}_1^D &= \mathrm{B}_1^D, \mathrm{B}_2^D, \mathrm{B}_3^D, \ldots\ldots, \mathrm{B}_{k-1}^D, \mathrm{B}_k^D \\
\mathrm{ch}_2^D &= \mathrm{B}_2^D, \mathrm{B}_3^D, \mathrm{B}_4^D, \ldots\ldots, \mathrm{B}_k^D, \mathrm{B}_{k+1}^D \\
&\vdots \\
\mathrm{ch}_i^D &= \mathrm{B}_i^D, \mathrm{B}_{i+1}^D, \mathrm{B}_{i+2}^D, \ldots\ldots, \mathrm{B}_{i+k-2}^D, \mathrm{B}_{i+k-1}^D \\
&\vdots
\end{aligned}
$$

Figure: Oblika delov datoteke

FbHash-B

Algoritem FbHash-B:

1. Za vsak del datoteke izračunaj zgoščeno vrednost in preštej pojavitve.
2. Uravnoteži dele datoteke glede število pojavitev
3. Preštej in uravnoteži pojavitve delov v bazi podatkov
4. Izračunaj končno oceno za datoteko iz uteži

Funkcije za izračun uteži

- Funkcije za uteži delov znotraj datoteke:
  - $W^D_{ch_i}(chf^D_{ch}) = 1 + \log_{10}\left(\frac{chf^D_{ch}}{n}\right)$
  - $W^D_{ch_i}(chf^D_{ch}) = \log_2\left(1 + \frac{chf^D_{ch_i}}{n}\right)$
  - $W^D_{ch_i}(chf^D_{ch}) = \frac{chf^D_{ch_i}}{n}$
- Funkcije za uteži delov v podatkovni bazi
  - $docw^D_{ch_i}(df_{ch}) = \log_{10}\left(\frac{m}{df_{ch}}\right)$
  - $docw^D_{ch_i}(df_{ch}) = 1 - \log_{10}\left(\frac{df_{ch}}{m}\right)$

Algoritem

- Primerjava datotek s kosinusno razdaljo

$$Similarity(D_1, D_2) = \frac{\sum_{i=0}^{n-1} W_{ch_i}^{D_1} \cdot W_{ch_i}^{D_2}}{\sqrt{\sum_{i=0}^{n-1} (W_{ch_i}^{D_1})^2} \cdot \sqrt{\sum_{i=0}^{n-1} (W_{ch_i}^{D_2})^2}} \cdot 100$$

- FbHash-S uporabi FbHash-B na razširjenih datotekah

Eksperimenti v članku

- Algoritem varen proti napadu z aktivnim napadalcem
- Detekcija fragmentov
- Korelacija skupnega dela datoteke

Detekcija fragmentov

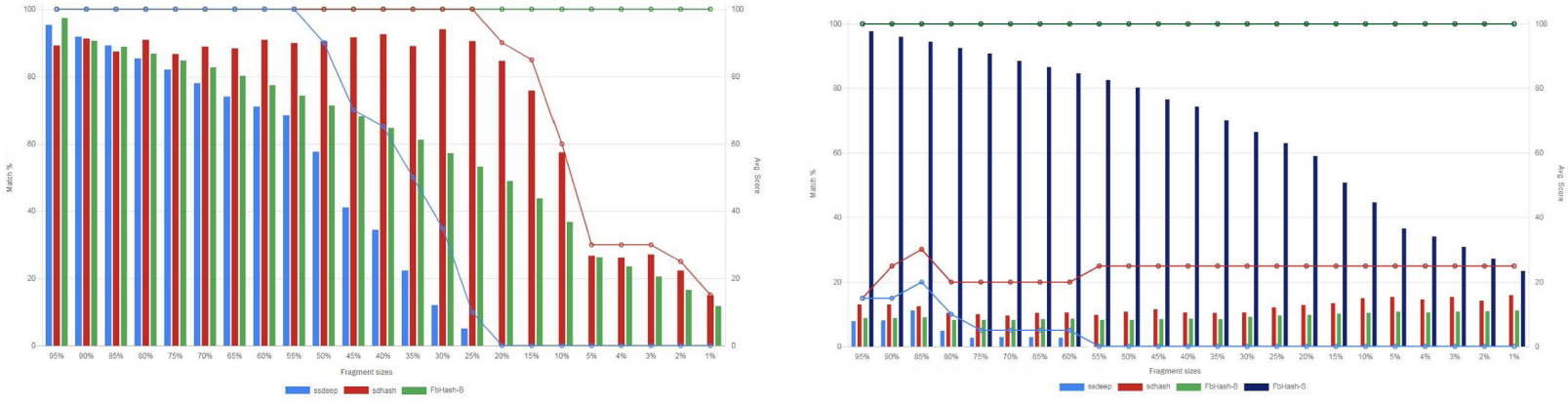


Figure: Rezultati detekcije fragmentov iz članka

Korelacija skupnega dela datoteke

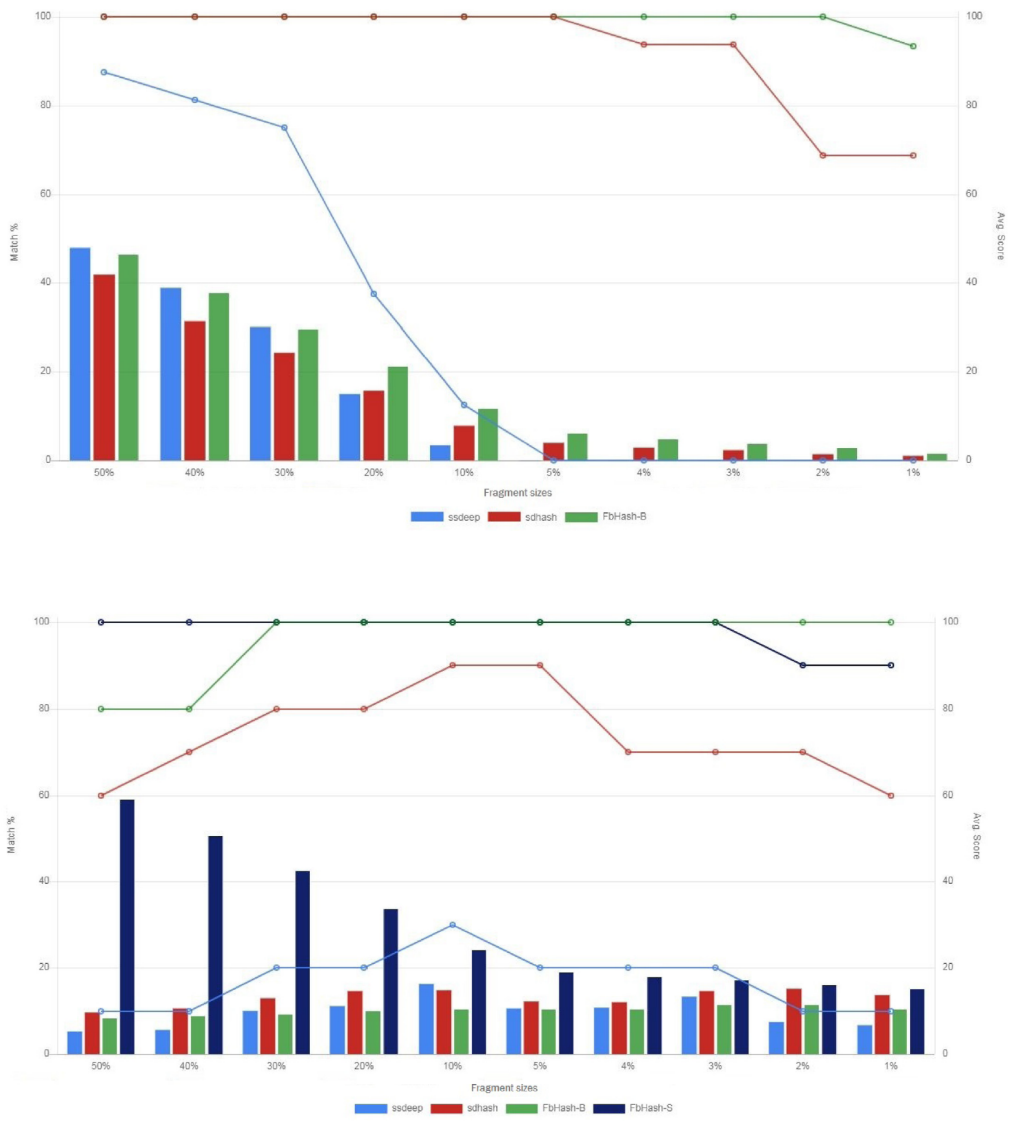


Figure: Rezultati korelacije skupnega dela datoteke iz članka

Naši eksperimenti

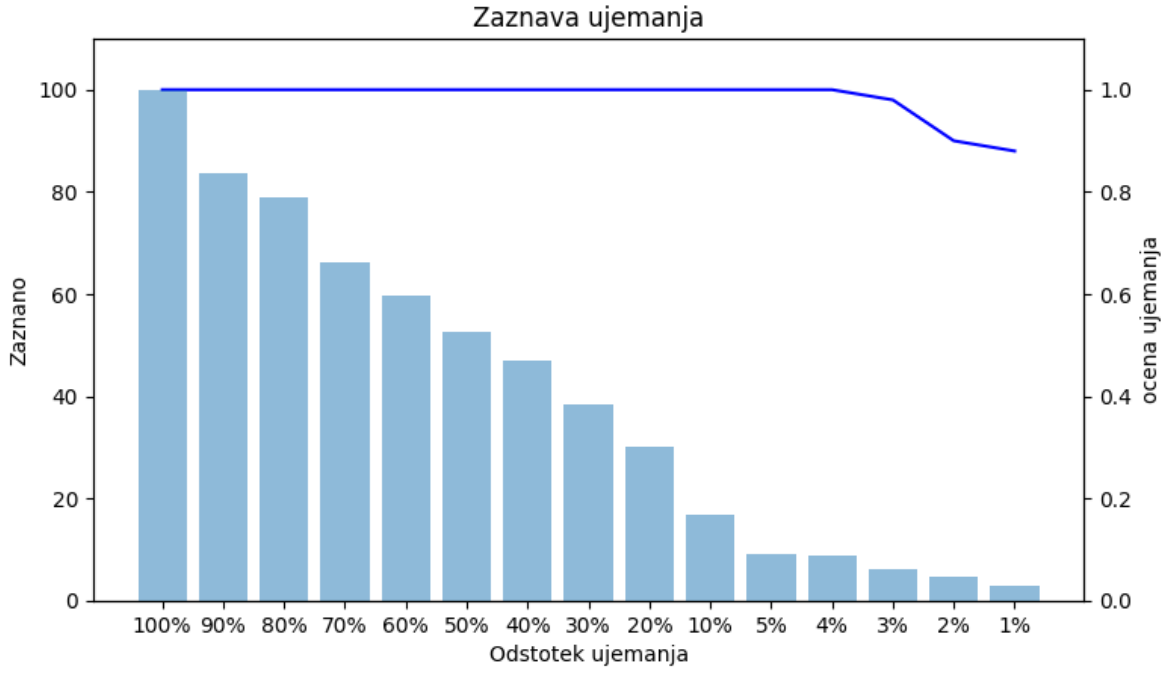
- Implementacija algoritma v jeziku python
- Implementacija FbHash-S in FbHash-B
- Testiranje na prosti zbirki dokumentov t5 (http://roussev.net/t5/t5.html)
- Testiranje delovanja algoritma pri različnih funkcijah za izračun uteži

Postopek testiranja

- Generiranje datotek z znanim ujemanjem
  - Vzamemo dve datoteki iz zbirke
  - Naključni blok prve vstavimo v drugo
  - Primerjamo prvo datoteko z novo nastalo datoteko
- Algoritem naučimo na celotni zbirki dokumentov
- Z algoritmom primerjamo ustvarjena dokumenta
- Opazujemo
  - Delež datotek z zaznanim ujemanjem
  - Povprečno oceno ujemanja glede na resnično ujemanje
  - F-oceno

Rezultati testiranja

- Izračun uteži kot v izvornem članku
- $W^D_{ch_i}(chf^D_{ch}) = 1 + \log_{10}\left(\frac{chf^D_{ch}}{n}\right)$
- F ocena: 0.94
- Rezultati skladni s člankom



Rezultati testiranja

- Uporaba različnih uteži
  - Levo: $W^D_{ch_i}(chf^D_{ch}) = \log_2\left(1 + \frac{chf^D_{ch_i}}{n}\right)$
  - Desno: $W^D_{ch_i}(chf^D_{ch}) = \frac{chf^D_{ch_i}}{n}$

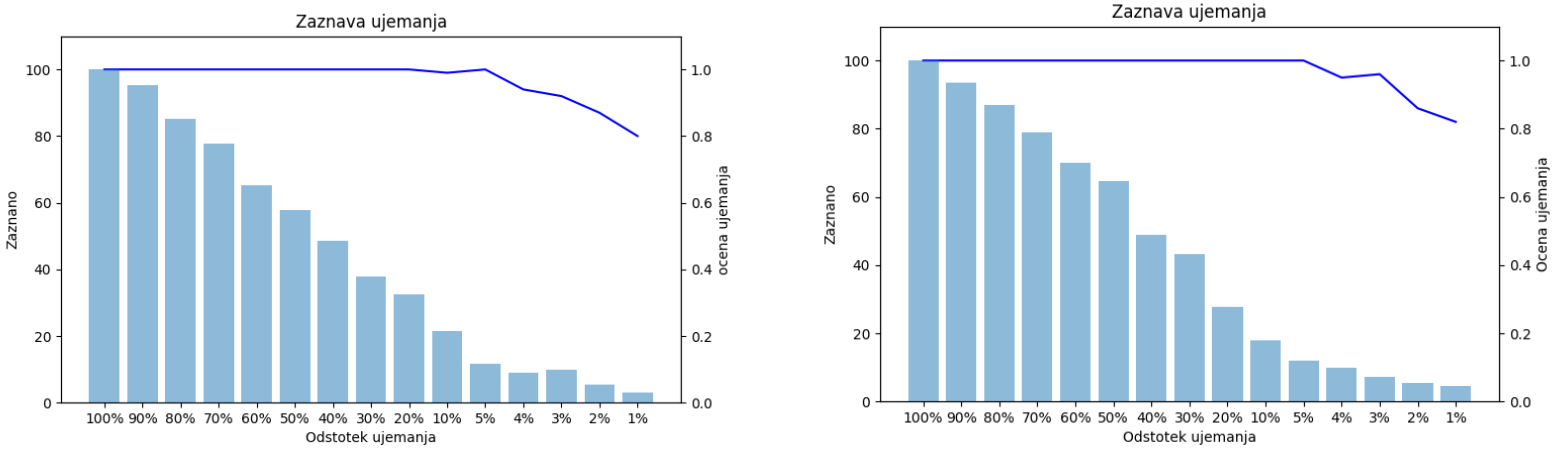
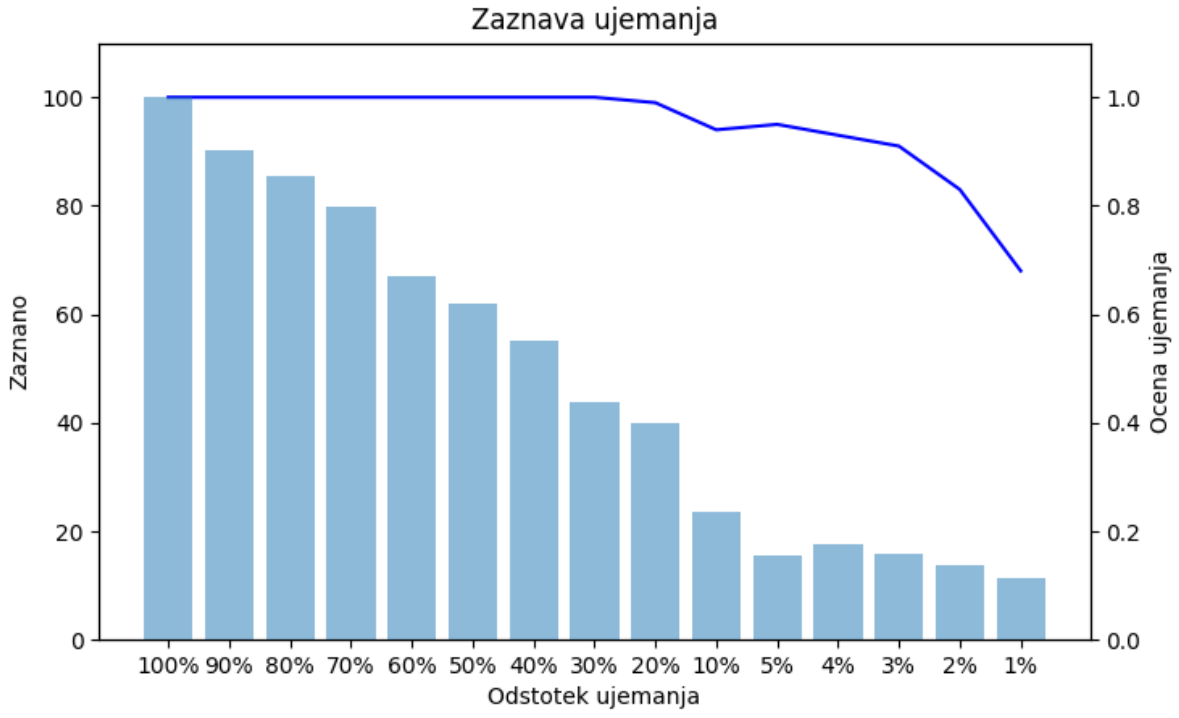


Figure: Testiranje delovanja algoritma z različnimi utežmi

Rezultati testiranja

- Razpoznavanje brez zbirke datotek
- F ocena: 0.87
- Dovolj dober rezultat, da bi lahko algotitem uporabili tudi brez zbirke datotek.



Zaključek

- The proposed approach improves the average and maximum relative errors compared to the existing square approximations.

Zaključek

- The proposed approach improves the average and maximum relative errors compared to the existing square approximations.
- Error analysis has shown that an error in the circuit is directly proportional to the trouble it can cause.