Fakulteta za računalništvo in informatiko, Univerza v Ljubljano

Mihael Švigelj, Lovro Vražič in Luka Zakšek

Crawler

Domača naloga pri predmetu Iskanje in ekstrakcija podatkov s spleta

MENTOR: prof. dr. Marko Bajec

V nalogi smo po danih navodilih v Pythonu razvili spletnega pajka, ki preiskuje le spletne strani domene *.gov.si*.

1. Uvod

Nalogo smo implementirali v jeziku Python. Za podatkovno bazo smo uporabili spletno dostopno PostgreSQL bazo. Pri implementaciji smo uporabili knjižnice: selenium, psycopg2, BeautifulSoup, tldextract in requests. Vizualizacijo pa smo naredili z orodjem gephi.

2. Struktura programa

Program začne s tem, da v podatkovno bazo in v frontier (več o frontieru v naslednjem poglavju) doda osnovne seed naslove, v našem primeru: “gov.si”, “evem.gov.si”, “e-uprava.gov.si” in “e-prostor.gov.si”. Zatem se ustvari toliko niti, kot je podan argument ob zagonu programa. Če argument ni podan, se jih ustvari 16. Med ustvarjanjem niti vedno preteče 10 sekund. Niti v neskončni zanki opravljajo funkcijo “get_images_links” na linkih, ki jih vrača frontier, dokler frontier vsaj 30 sekund ni prazen.
Funkcija “get_images_links” z uporabo knjižnice Selenium inicializira headless chrome driver, ki poskuša pridobiti vsebino spletne strani na danem url-ju. Nato se z uporabo knjižnice tldextract iz url-ja razbere domena spletne strani, s pomočjo katere pridobimo tudi primerno vrstico v tabeli crawldb.site. S knjižnico BeautifulSoup razberemo html_content, nato s klicem funkcije get_content_type preverimo za kakšen tip spletne strani gre - html, binary ali duplicate. V primeru, da je content_type html sledi ekstrakcija slik in linkov iz vsebine spletne strani.
V nadaljevanju se sprehodimo skozi vse edinstvene najdene linke ter, v primeru da vsebujejo *gov.si*, naredimo sledeče: če najden url ne vsebuje html vsebine, ampak eno od upoštevanih tipov vsebin (doc, docx, pdf, ppt, pptx), ustvarimo page_data objekt, ki ga kasneje vstavimo v bazo v istoimensko tabelo s klicem metode insert_page_data_to_db. Če najden url še ni v bazi v tabeli page, preverimo če nam robots.txt dovoljuje crawlanje ter pridobimo id site-a, kateremu url pripada, nato ustvarimo vnos strani v tabelo page. Prav tako dodamo stran z najdenim url-jem v frontier. Če je najden url na novi domeni ustvarimo še nov vnos v crawldb.site in frontierju. Potem ustvarimo še objekt link, ki vsebuje id trenutno

obdelovane strani ter id najdene strani - ta objekt nato vstavimo v bazo v tabelo link s klicem funkcije put_link_in_db. Sledi ustvarjanje objektov image za vse najdene slike na obdelovani spletni strani - te nato vstavimo v bazo s klicem metode insert_images_to_db. Na koncu obdelano stran tudi posodobimo v tabeli page.

3. Breadth-First Search oziroma implementacija Frontierja

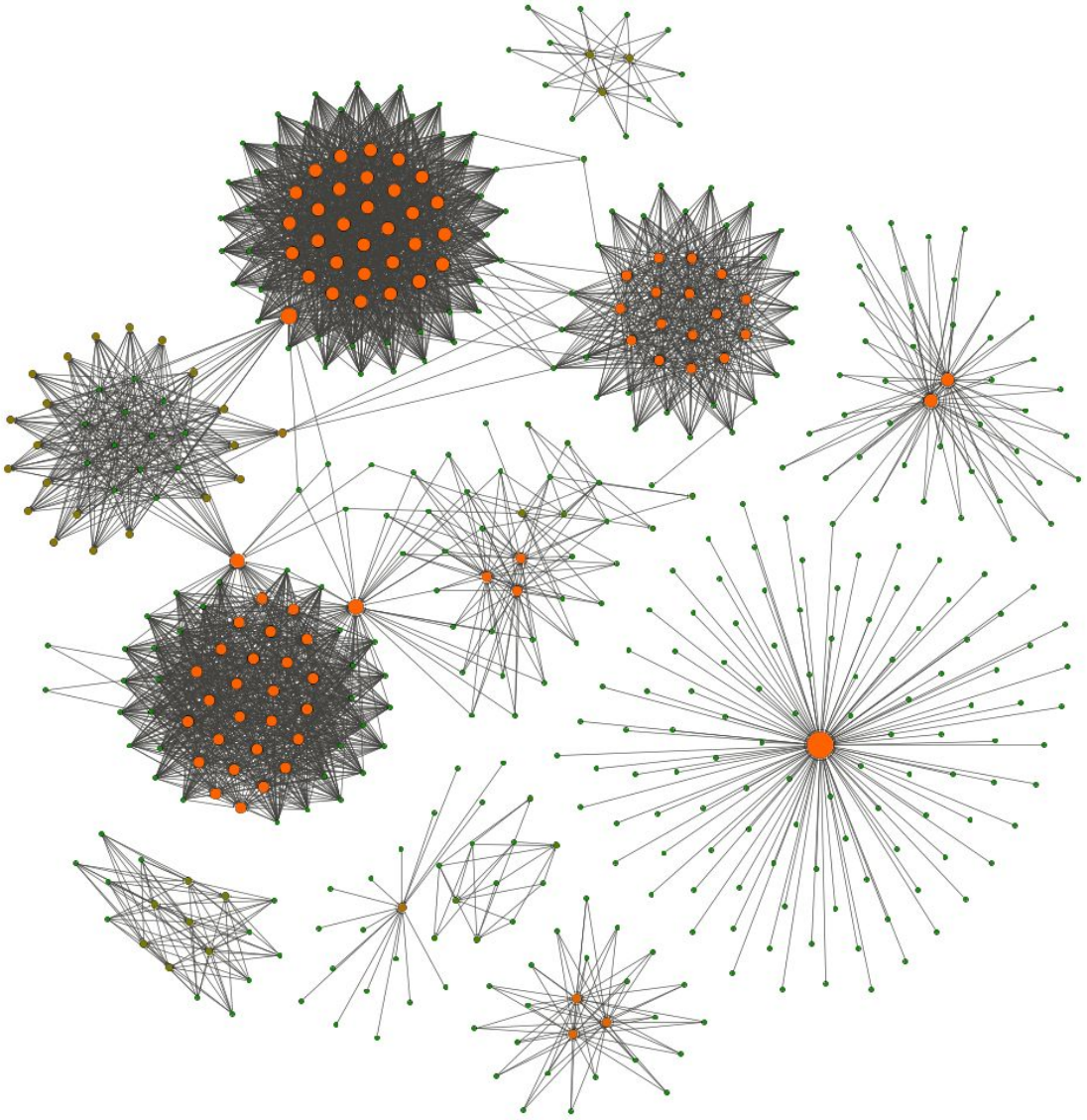
BFS frontier je implementiran lokalno z uporabo podatkovnih struktur Queue. Frontier vsebuje Queue domen, vsaka izmed domen pa vsebuje še Queue linkov. Kadar crawler zazna novo domeno, se ta, ko se doda v podatkovno bazo, doda tudi v Frontier objekt. Nato se vsi linki, ki spadajo pod to domeno, dodajo v Queue linkov za to domeno. Tudi kasneje med crawlanjem se linki, s katerimi se prvič srečamo, dodajajo v Queue domene h kateri spadajo.
Ko se od frontierja zahteva naslednji link za obdelavo, ta vrne link na začetku Queue-a linkov znotraj domene, ki je na začetku Queue-a domen. Domeno se obenem ponovno vstavi v Queue domen.
Frontier za vsako domeno tudi beleži zadnji čas dostopa in v primeru, da je od zadnjega dostopa poteklo manj kot pet sekund, nit izvajanja blokira.

4. Splošna statistika

	site	page	duplicates	binary	images	image/page	link/page
vse domene	157	37407	2454	234	2756	1,05	1,5
gov.si	/	1798	46	0	173	0,09	0,99
evem.gov.si	/	511	34	0	448	0,87	2,21
e-uprava.gov.si	/	4637	0	0	104	0,02	1,00
e-prostor.gov.si	/	384	1	0	2	0,005	0,2

5. Vizualizacija

Prikazani so podatki povezav za prvih 3000 obdelanih strani. Povprečna stopnja je 1.268. Na sliki so prikazane samo strani z več kot 30 izhodnimi povezavami. Uporabljen je t. i. Force-directed graph drawing algoritem. Velikost in barva vozlišča prikazujeta število vhodnih povezav, kjer je večje in bolj oranžno vozlišče bolj pomembno.



6. Težave

Glavne težave, s katerimi smo se soočali:

- detekcija duplikatov strani - detekcijo duplikatov smo realizirali samo s primerjavo url naslovov in točne vsebine strani s podatkovno bazo.
- določanje domene - več časa smo se ukvarjali z iskanjem najboljše metode natančnega in konsistentnega določanja domene, ker so imena domen precej različna, poleg tega pa je v urlju včasih vsebovan tudi www, včasih ne, in podobno.
- koordinacija - ker nismo najbolj izkušeni pri uporabi gita in tudi ker si na začetku nismo najbolje razdelali zasnove programa smo imeli pogoste težave s sočasnim delom.

7. Zaključek

Razvit program ni popolen. Najbolj očitna potencialna izboljšava je detekcija duplikatov, saj trenutno to dosežemo zgolj z primerjanjem URL naslovov in md5 hasha same html datoteke. Najverjetneje bi se lahko izboljšala tudi hitrost sočasnega procesiranja, saj za zagotavljanje varnosti podatkov vsako delo s podatkovno bazo delamo s ključavnico, najverjetneje marsikje tudi brez potrebe.
Kljub temu program v grobem doseže zastavljene cilje - to je, poglobi se v mrežo “.gov.si” strani in dokumentira povezave in vsebovane podatke.