Nejc Ribič, Klemen Jesenovec in Miha Stele

Poročilo

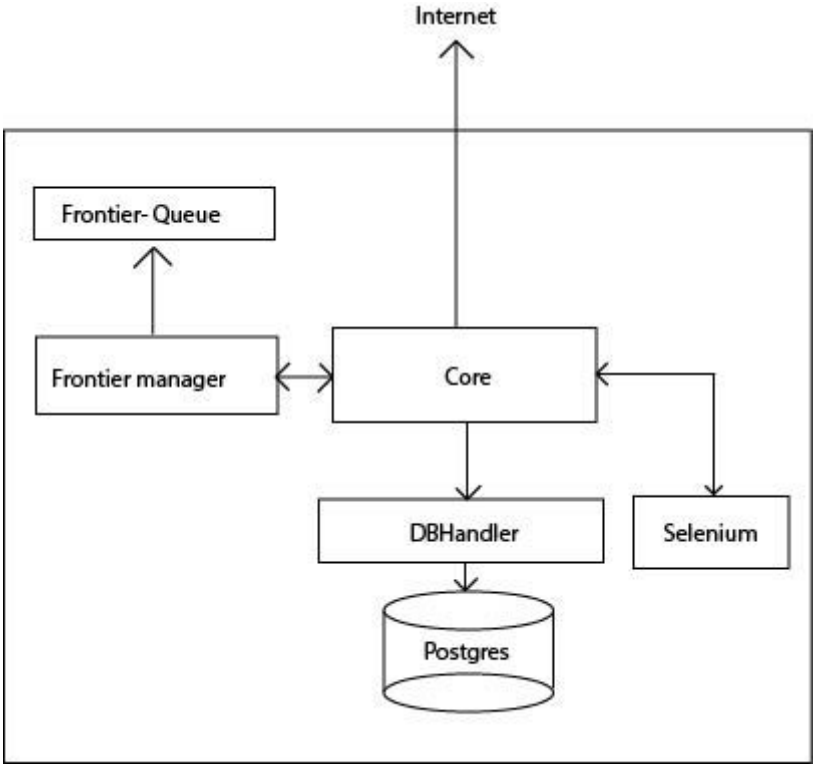
Seminar 1 - IEPS

1. Uvod

Seminarsko nalogo smo se lotili s kreacijo same arhitekture, kot smo to omenjali na predavanjih. Zajeli smo pribl. 23 000 strani v roku 2 dneh. Podatkovna zbirka je zasnovana tako, kot je predpisano v navodilih brez dodatnih entitet. Uporabili smo programski jezik Python z uporabo knjižnice Selenium in ostalih knjižnic, ki omogočajo lažji zajem.

2. Struktura pajka

Arhitektura



Sistem je zasnovan po specifikacijah s seminarske naloge, katerega smo prilagodili smiselno za uporabljen programski jezik. Frontier manager je komponenta, ki upravlja naš frontier, zasnovan v podatkovni strukturi vrsta. Hkrati poskrbi tudi za vnašanje in deljenje URLjev večih niti hkrati. Jedro poskrbi za smiselno paralelizacijo in pravilno upravljanje povezovanje zbrike, frontierja in integracijo s knjižnico Selenium. Logika preverja, če strani vsebujejo Robots Exclusion Standard, jih razčleni in upošteva. Hkrati pa notri nastopa logika za odpiranje strani, obvladovanje robnih pogojev, kot so npr. upravljanje z SSL certifikatom

in podobno. Ko se stran naloži, se nato preverijo povezave na druge strani, katere gredo potem v frontier, ločimo pa še povezave slik (tudi slike z določene strani) in dokumentov. Upravljalec podatkovne zbirke zagotovi primerne poizvedbe z zbirke, tj. vstavljanje strani, posodabljanje strani in tako naprej. Selenium je brezglavni poganjalec brskalnika, katerega smo uporabljali za dostopanje spletnih strani.

Nastavitve

Parametre podava v datoteki `settings.py`, primeri ključnih parametrov pa so:

- `NUMBER_OF_SPIDERS` - število instanc pajkov, ki delujejo vzporedno,
- `TIME_BETWEEN_REQUESTS` - čas zakasnitve naslednjega zahtevka,
- `HEADLESS_BROWSER` - možnost, da se ne prikaže okno brskalnika ob začetku luščenja.

Knjižnice in različica programskega jezika

Uporabili smo Python, različice 3.7.2, z naslednjimi knjižnicami:

- `selenium`
- `chromedriver`
- `pypiwin32`
- `robots-txt-parser`
- `psycopg2`
- `postgres`
- `beautifulsoup4`
- `requests`
- `w3lib`

Implementacija frontierja

Za frontier smo uporabili vrsto. Ker pa je potrebna logika za vzporednost ter za vzporedno upravljanje smo si zastavili še *`Frontier manager`*.

Implementacija iskanja strani, razčlenjevanja in podobno

Takoj, ko se zahtevek izvede, narediva sledečo logiko:

1. branje spletne strani
2. shranjevanje strani
3. zajem vseh URL-jev
4. smiselne URL-je shrani v frontier

5. zajem in shranjevanje slik
6. shranjevanje binarnih dokumentov
7. nadaljna logika, ki ni zanimiva za ta podpoglavje

Začetne povezave v frontierju:

Vzeli smo naslednje povezave:

- `http://evem.gov.si/`
- `https://e-uprava.gov.si/`
- `https://podatki.gov.si/`
- `http://www.e-prostor.gov.si/`

Problemi

Razčlenjevanje robots.txt - robots.txt nekatere strani vsebujejo, druge ne. Včasih se zgodi, da se strani ne naloži ali pa datoteka vsebuje nepravilno obliko zapisa. Primerno smo obvladovali vse primere in ustrezno zapisali v bazo.

Vzporednost - problem je bilo smiselno uvesti paralelizacijo zaradi upravljanja s frontierjem, podatkovno zbirko. Vsaka nit globalno nastavi vrednost, ki označuje, če je končala z delom. Šele ko vse niti zaključijo in je frontier prazen se program lahko konča.

Čiščenje zbirke - Med implementacijo smo poskušali delovanje našega pajka kar v podatkovni zbirki. Zaradi napak so se pojavljale razne napake tudi v zbirki, zato smo morali zbirko počistiti po novih poskusih oz. končni ekstrakciji. Problem se je pojavljal pri Postgres administracijskem uporabniškem vmesniku pgadmin, saj čiščenje zbirke ob veliki količini ni delovalo pravilno.

Premalo pomnilnika (Out of Memory) - pri izvajanju nekaterih sql skript za namen vizualizacije so bile težave pri poizvedbenih stavkih, saj so zahtevali večjo količino pomnilnika in posledično nismo dobili rezultatov. Rešili smo problem tako, da nismo uporabili ime domene v vizualizaciji – le njeno identiteto, kar je olajšalo poizvedbeni stavek.

Odvisnost logike - Ko smo razčlenili spletne strani, smo skušali le v eni iteraciji ločiti smiselne povezave (tj. katere gredo v frontier, katere so slike ipd.). Razčlenili smo jih po atributih href (tudi z javascript logike), potem smo po smislu pregledali še ostale značke (npr. img za slike).

SSL - SSL težave nam je primarno povzročal antivirusni program, hkrati pa so se pojavljale tudi zaradi gonilnika, ki ga ponuja selenium. Antivirusni sistem je omogočal enostavno izklopitev SSL funkcinalnosti, medtem ko smo s pomočjo metod s knjižnice Selenium in try-catch stavki skoraj popolnoma odpravili težave.

upravljanje dolgotrajnih zahtevkov - Omejili smo zahtevke na nekaj sekund, v primeru da zahteve traja dalj časa, se zahtevek prekine in se pajek loti upravljati drugih strani.

upravljanje z zakasnitvijo med zahtevki - Zakasnitev med zahtevki nastavljamo z globalno nastavitvijo. Predstavlja čas čakanja v sekundah med zahtevki, ki so poslani na isto domeno. Implementirana je s pomočjo slovarja, ki hrani čas zadnjega dostopa no domeno. Preden pajek pošlje zahtevek, počaka ustrezno število sekund.

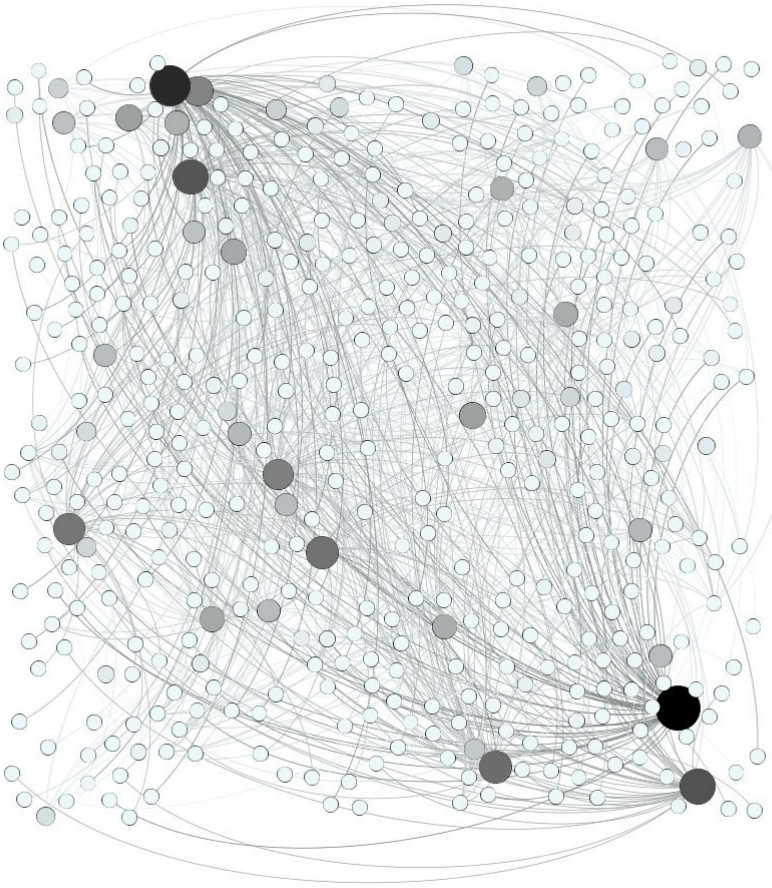
3. Statistika

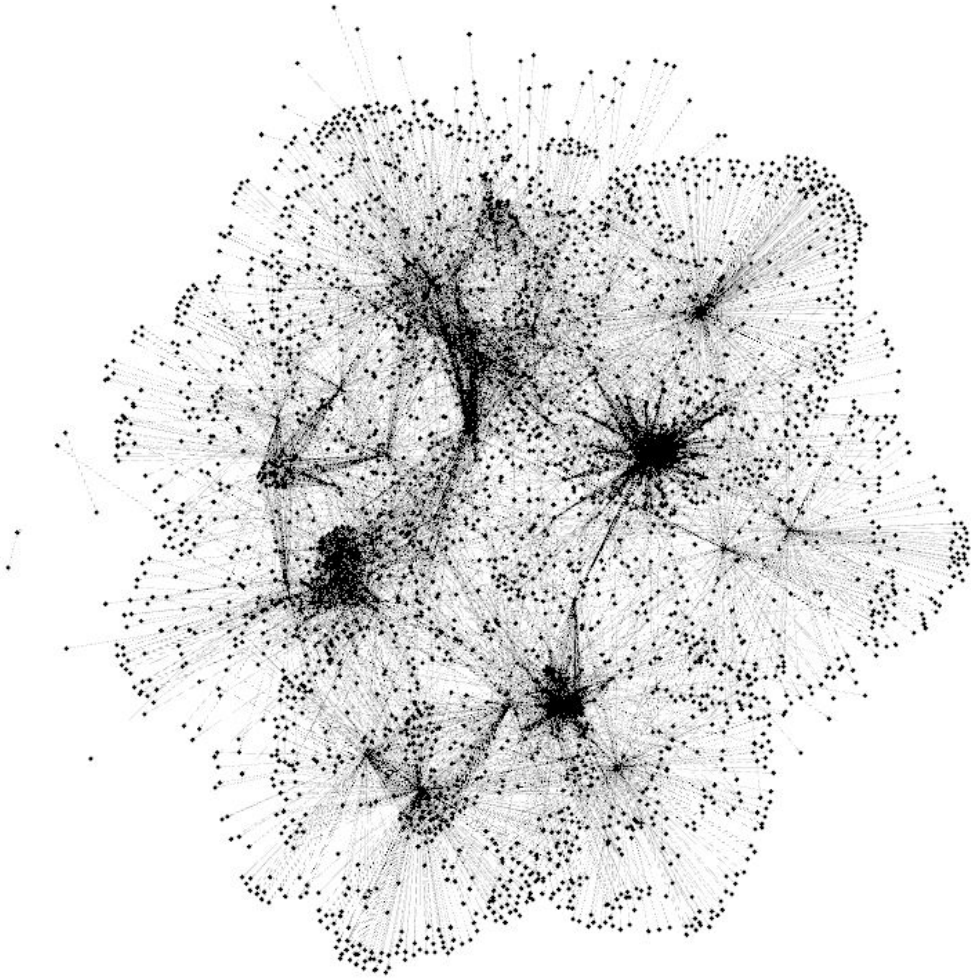
Pajek je iskal 20 ur. V tem času je našel 22 973 strani. To je 1148 strani na uro. Kar je smiselno, saj za vsako stran na isti domeni čaka okoli 10 sekund.

- število domen(tabela site): 139
- št. spletnih strani: 22 973
- št. duplikatov: 1583 (statistiko smo zbirali v času zagona)
- povprečno število slik na stran: 9 (angl. per page)
- povprečno število binarnih dokumentov na stran:

4. Vizualizacija

Za vizualizacijo podatkov smo se odločili za uporabo knjižnice gephi. Za prikaz mreže smo izvozili tabelo links v formatu csv. Gephi zahteva drugačno poimenovanje, da pravilno zazna povezave, zato smo morali imena stolpcev izvožene datoteke spremeniti.





5. Zaključek

Seminarska naloga je bila zelo zanimiva, saj nam je omogočila uporabo tehnologij po svoji želji, in nas na ta način ni ovirala glede inovativnosti in arhitekturnih pristopov. Med samo implementacijo smo se naučili, da je zelo pomembna korektnost in konsistentnost programiranja, saj smo si kot ekipa velikokrat povzročali nemalo preglavic z git konflikti. Prav tako pa smo se ogromno naučili o samih pajkih, zajemu, etiki zajema, podatkovnih zbirkah in podobnih stvareh.