Naloga 1 (Iskanje in ekstrakcija podatkov iz spleta)

Jaka Kordež, Anže Gregorc

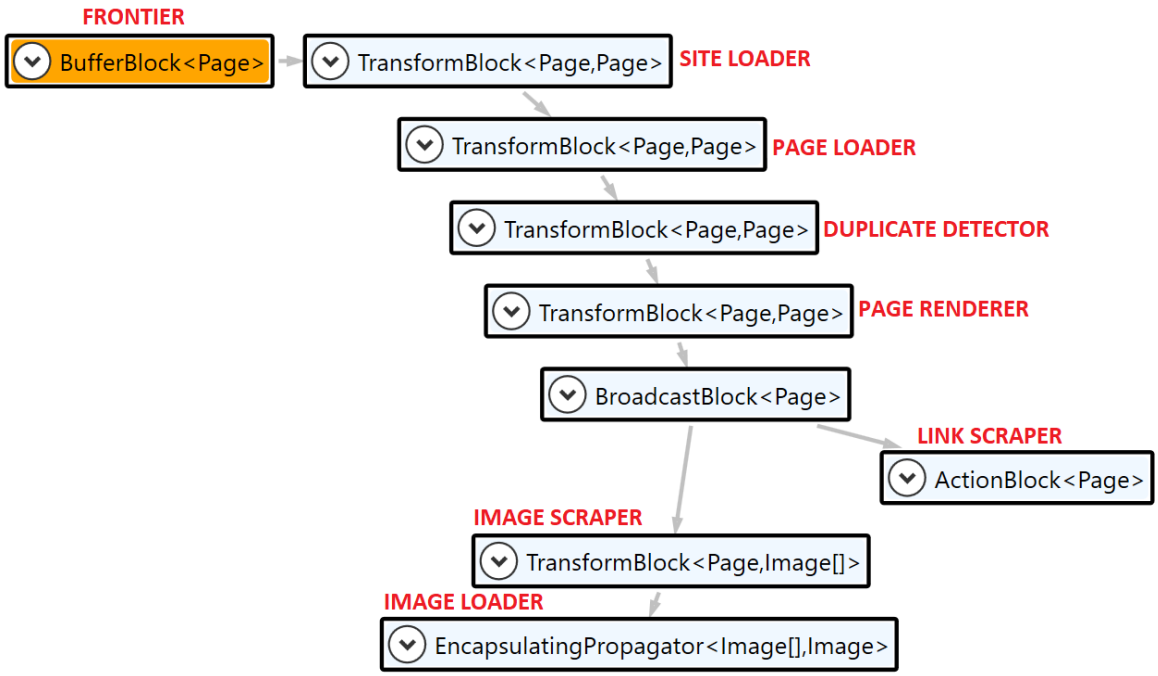
Uvod

Cilj te naloge je bil izdelati spletnega pajka. Naš program bo obiskal le povezave s končnico .gov.si. Razdeljen je na več delov:

1. HTTP downloader in renderer, ki poskrbita za prenos in upodobitev spletne strani.
2. Data extractor: iz spletne strani najde povezave in slike.
3. Duplicate detector: odkrije strani, ki so enake.
4. URL frontier: Vrsta URL-jev, ki še čakajo, da na prenos.
5. Datastore: Podakovna baza, kjer shranimo vse podatke.

Struktura spletnega pajka

Naš spletni pajek ima strukturo podobno cevovodu. Vsak del programa se dogaja v določenem koraku cevovoda. Razdeljen je na 9 stopenj. To so: frontier, site loader, page loader, page duplicate detector, page renderer, link scraper, image scraper in image loader.



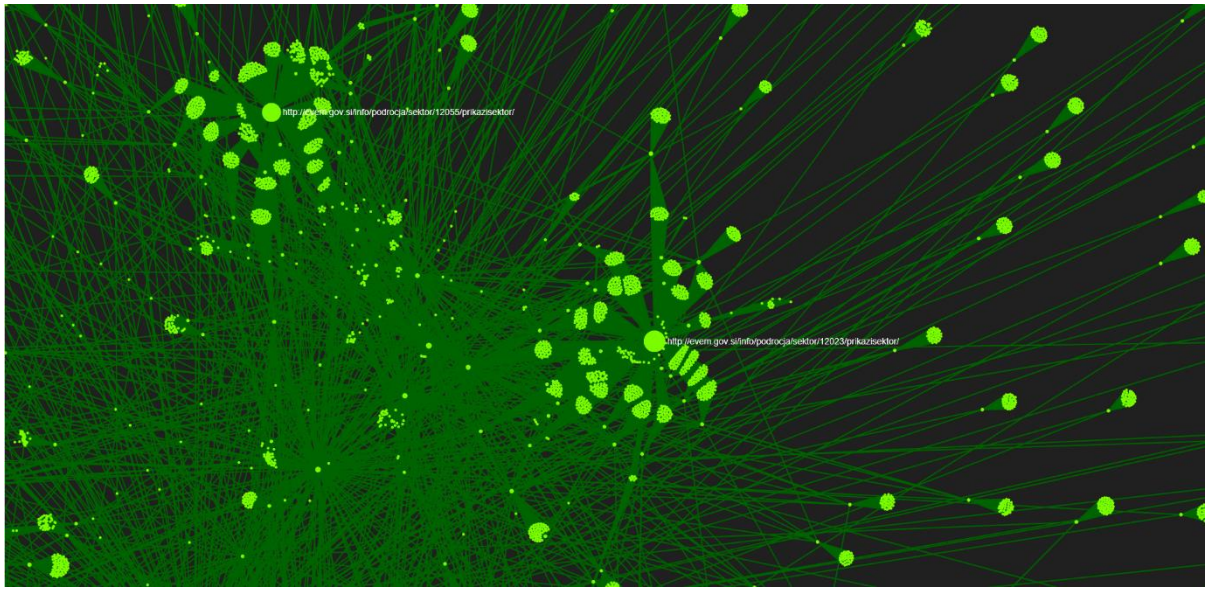
Slika prikazuje strukturo našega spletnega pajka. Potrebno je bilo implementirati pajka, ki bi deloval asinhrono. To smo storili preprosto s parametrom `MaxDegreeOfParallelism`, ki nam ga ponuja objekt `TransformBlock`, s katerim smo implementirali korake. V nadaljevanju je zapisan kratek opis posameznega koraka, morebitne parametre ter druge značilnosti.

1. Frontier: predstavlja le blok URL-jev, ki čakajo na vrsto, da se jih pošlje v naslednji korak. Implementirana je po metodi FIFO (prvi url, ki gre v frontier, gre tudi prvi naprej). Tako smo zadostili strategiji iskanja v širino (breadth-first).
2. Site loader: najde domeno in poskusi najti datoteko robots.txt. Če jo najde, se njegova vsebina shrani ter upošteva vse pomembne informacije (tj. User-agent, Allow, Disallow, Crawl-delay in Sitemap).
3. Page loader: pridobi spletno stran. Kot ogrodje smo vzeli Selenium. Tukaj tudi oblikujemo URL v kanonično obliko. V našem primeru to pomeni: vzamemo le del za ? in odstranimo del `www.` ter `.html.`
4. Duplicate detector: zazna duplikate. Zaznamo le identične spletne strani in jih v podatkovni bazi tudi primerno označimo.
5. Page renderer: upodobimo spletno stran. V našem primeru jo le ovijemo v razred `HtmlParser`, ki nam stori vse potrebno.
6. Link scraper: Iz spletne strani najdemo vse povezave (tj. location.href ali dobument.location). V primeru, da je povezava relativna, jo primerno preoblikujemo.
7. Image scraper: Iz spletne strani najde vse slike.
8. Image loader: Prenese slike ter jih shrani v podatkovno bazo.

Reševanje problemov

V času izdelave spletnega pajka smo imeli kar nekaj težav. Kot prvi problem izpostavimo težavno razhroščevanje. Ko je cevovod prenehal delovati, smo imeli težavo najti korak pri katerem se je zataknilo. Klub temu smo po nekem času vse napake odpravili. Težave smo imeli tudi pri shranjevanju podatkov v podatkovno bazo. Ker je pajek asinhron, smo morali vsako interakcijo z bazo zakleniti. S tem smo se rešili največjih težav pri zapisovanju v podatkovno bazo.

Vizualizacija



Slika prikazuje vizualizacijo domene evem.gov.si. Prikazan je le del vseh strani in vozlišč, saj je celotna slika prevelika za prikaz.

Osnovna statistika

- Število vseh domen: 39
- Število vseh spletnih strani: 20099
- Število duplikatov: 2119
- Število binarnih datotek: 298
    - Število datotek s končnico .pdf: 234
    - Število datotek s končnico .doc: 37
    - Število datotek s končnico .docx: 26
    - Število datotek s končnico .ppt: 1
    - Število datotek s končnico .pptx: 0
- Število vseh slik: 680
- Povprečno število slik na spletno stran: 0,33

Zaključek

Z našim delom smo zadovoljni. Žal smo imeli nekaj stiske s časom in tako nismo uspeli nalogo še bolj izpopolniti.