Web crawling

decembrie 2017

Crawler = componenta unui motor de căutare care colectează pagini de pe web pentru a le indexa.

- web crawling: procesul de colectare eficientă a unui volum mare de pagini web, împreună cu structura de link-uri dintre pagini.
- spider = denumire alternativă a unui web crawler.

Probleme discutate în acest curs

1. Caracteristicile unui crawler (deziderate)
2. Arhitectura unui crawler; detalii de implementare

- Robustețe: să recunoască și să evite capcanele de crawlere (engl. **spider traps**):
  - un spider strap este un generator de pagini web care determină un crawler să se blocheze colectând un nr. infinit de pagini
- Politețe: Serverele web au reguli implicite și explicite care precizează cât de frecvent pot fi vizitate. Aceste principii trebuie să fie respectate de către crawlere.

Caracteristicile unui crawler

Caracteristici pe care ni le dorim

- Să poată fi executat în mod distribuit, pe mai multe mașini
- Să aibe o arhitectură scalabilă, care să tolereze adăugarea de mașini noi
- Să utilizeze eficient resursele sistemului: procesor, memorie, capacitate de rețea
- Să dea prioritate paginilor web care oferă răspunsuri de calitate la întrebările utilizatorilor
- Să opereze îm mod continuu, solicitând copii noi ale paginilor colectate anterior. Frecvența de colectare trebuie să fie aproximativ egală cu frecvența de modificare a paginii.
- Să fie extensibil, adică să poată opera cu formate noi, protocoale noi, etc. $\Rightarrow$ arhitectura crawlerului trebuie să fie modulară.

Cum funcționează un crawler?

1. Pornește de la o mulțime inițială S de URL-uri, numită seed set
2. Se setează $F = S$; F se numește frontiera URL a crawler-ului
3. Se alege un URL $u \in F$ și colectează pagina web P corespunzătoare
   - P se analizează lexical și se extrage textul T și link-urile L din P.
   - Se indexează textul T al URL-ului u
   - $F := (F \cup L) \setminus \{u\}$
   - Se revine la pasul 2.

$\Rightarrow$ traversare recursivă a grafului web care conține nodurile mulțimii inițiale de URL-uri S.

Design multi-threaded: capabil să colecteze $\approx 10^9$ pagini/lună $\Leftrightarrow$ sute de pagini/secundă

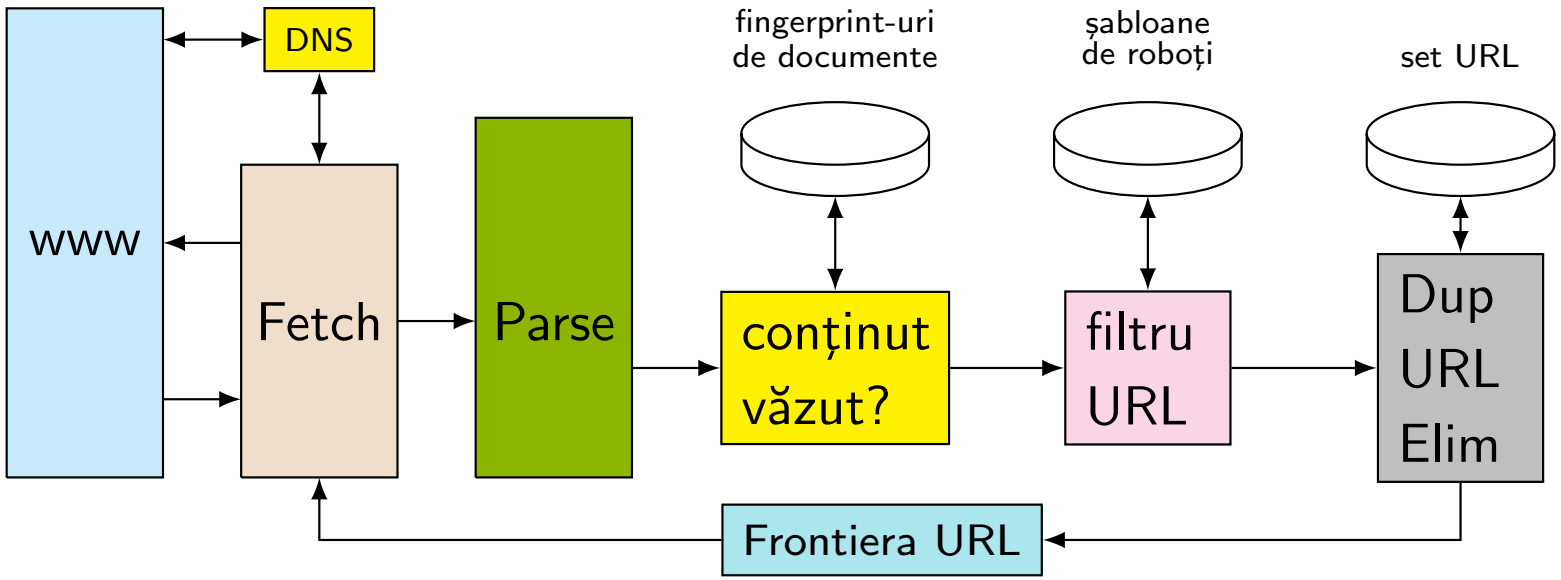


Figure: Bucla logică de operații efectuate de un thread al crawler-ului

Procesul de crawling este efectuat de sute de thread-uri: fiecare thread execută bucla logică din diagrama ilustrată.

Componentele funcționale ale unui crawler

Structură modulară (vezi arhitectura de bază):

1. **Frontiera URL**: conține URL-urile paginilor cere urmează a fi colectate de către crawler
2. **DNS**: modul de rezoluție DNS: determină serverul web de unde să descarce pagina web specificată de un URL
3. **Fetch**: modul ce folosește `http` pentru a descărca o pagină web de la un URL
4. **Parse**: modul de extragere a textului și link-urilor din paginile web colectate
5. **DupURLElim**: modul de eliminare a duplicatelor: determină dacă un link extras este deja în frontiera F, sau este nou

Ce face thread-urile unui crawler?

1. Selectează un url $u \in F$ și colectează pagina web P corespunzătoare
2. Modulul Parse extrage din P textul T și link-urile L și informații despre link-uri (de ex., textul ancoră)
   - textul ancoră se folosește pentru calculul scorului de importanță bazat pe analiza link-urilor dintre pagini.
3. Se verifică dacă s-a mai analizat vreo pagină web cu același conținut T
   - Se calculează un fingerprint (de exemplu, un checksum) al paginii web, care se reține într-o colecție de FP-uri deja văzute
     - dacă FP-ul calculat $\in$ colecția de FP-uri întâlnite $\Rightarrow$ renunță la indexarea paginii P
4. Se aplică un filtru URL care analizează fiecare URL din L, dacă trebuie adăugat la frontiera URL:
   - de exemplu, se pot exclude URL-urile `.com`
   - Unele servere web interzic crawling-ul paginilor lor, pe baza protocolului standatdizat Robots Exclusion Protocol

Protocolul de excludere a roboților
(Robot Exclusion Protocol)

- Mai întâi, robotul (=crawlerul) descarcă fișierul `robots.txt` din rădăcina ierarhiei URL a site-ului accesat.
- Fișierul `robots.txt` indică ce roboți au dreptul să acceseze pagini web de pe site-ul respectiv.

Exemplu

```
User-agent: *
Disallow: /yoursite/temp/
User-agent: searchengine
Disallow:
```

Nici un robot nu are dreptul să viziteze URL-uri din subdirectorul `/yoursite/temp/`, cu excepția robotului `searchengine`

De obicei, o pagină web conține link-uri la alte pagini web de pe același site $\Rightarrow$ pentru a evita descărcarea repetată a lui `robots.txt` de pe același site, crawler-ul reține fișierele `robots.txt` descărcate în o memorie cache.

Normalizarea URL-urilor și eliminarea duplicatelor

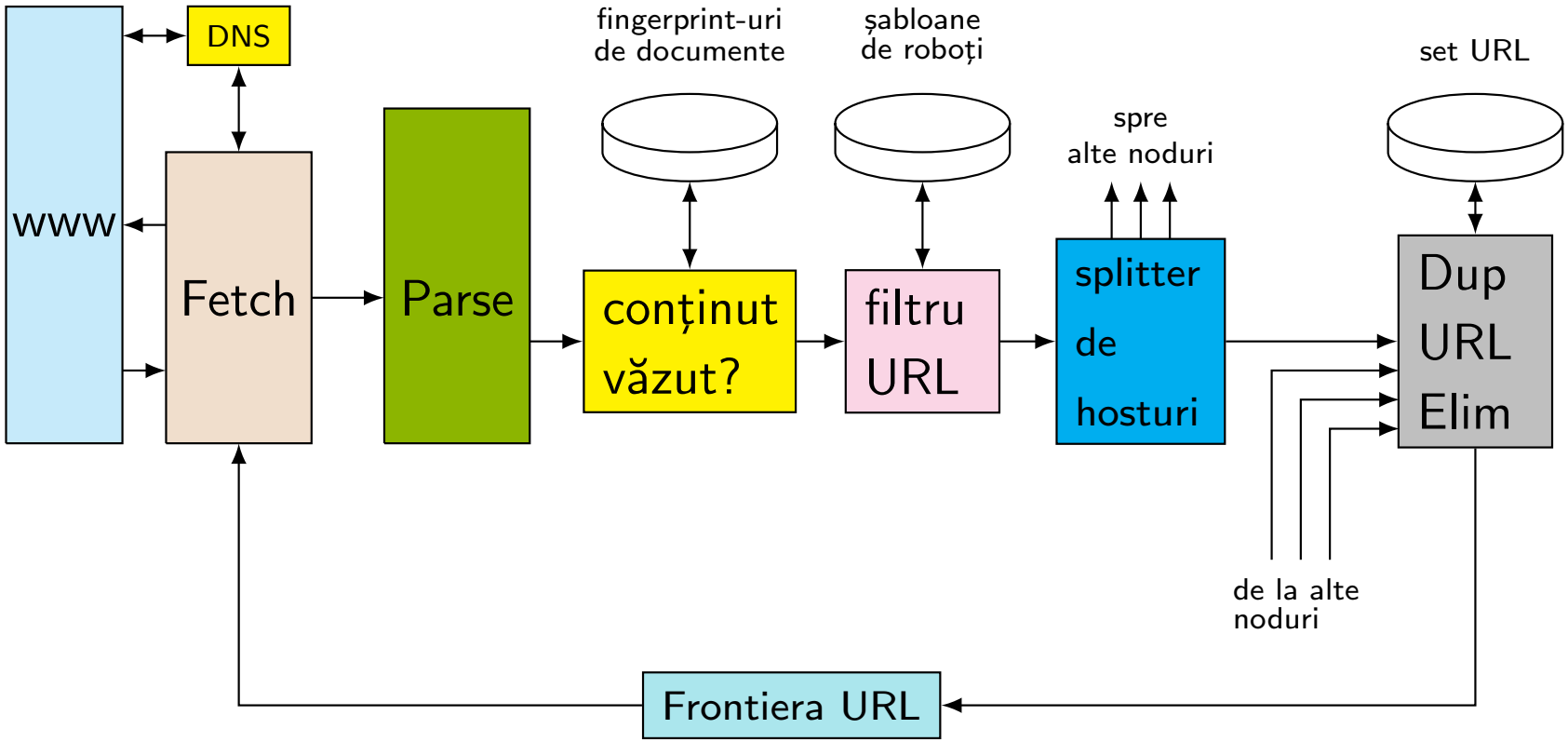
- Un crawler **normalizează** URL-urile din frontierea URL: link-urile relative din sursele `html` ale paginilor web se transformă în URL-uri absolute
- Crawlerul verifică dacă URL-ul normalizat este în frontiera URL, sau dacă a fost deja descărcat
  - dacă da, URL-ul normalizat nu se adaugă la frontiera URL
- Frontiera URL este adesea implementată ca o **coadă cu priorități**: fiecare URL poate fi asociat cu o prioritate de control al momentului când să fie colectată pagina web de la URL-ul respectiv

Crawlere distribuite

Pentru a putea opera pe o rețea de mărimea WWW, thread-urile unui crawler se execută în procese diferite, pe noduri(=mașini) diferite ale unui sistem de crawling distribuit:

- crawlerul are o funcție hash $h : \mathcal{N} \to \mathcal{P}(URL)$ unde
  - $\mathcal{N}$ este mulțimea de noduri a crawlerului
  - $h(n)$ este mulțimea de site-uri accesate de procesul de crawling de pe nodul $n \in \mathcal{N}$
- Procesele de crawling care rulează pe noduri diferite comunică și operează în comun pe o mulțime de URL-uri:
  - Bucla logică din Figura 1 se execută simultan pe fiecare nod $n \in \mathcal{N}$ al crawler-ului, **cu 2 modificări simple**: (1) după filtrul URL, se adaugă un splitter de hosturi care redirecționează fiecare URL filtrat u la nodul n pentru care $u \in h(n)$; (2) output-ul fiecărui splitter de hosturi este transmis la toate modulele DupURLElim ale crawlerului distribuit

Crawlere distribuite
Arhitectura de bază



Bibliografie

Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze: Capitolul 20: *Web crawling and indexes* din An Introduction to Information Retrieval.
Ediție online (c) 2009 Cambridge UP.