UNIVERSITATEA TEHNICĂ „Gheorghe Asachi” din IAȘI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DOMENIUL: Calculatoare și Tehnologia Informației
SPECIALIZAREA: Tehnologia Informației

Crawler WEB

--REGĂSIREA INFORMAȚIILOR PE WEB--

Profesor îndrumător
Ș.l. dr. ing. Alexandru Archip

Student
Temătoru Daniel-grupa 1410B

Capitolul 1. Problema propusă spre rezolvare

Aplicația corespunzătoare celei de a doua componente de proiect trebuie să implementeze un Crawler WEB. Acest modul trebuie să realizeze corect cereri HTTP utilizând versiunea 1.1 a protocolului (atât cereri directe, cât și cereri prin intermediul unui proxy HTTP) și să salveze conținutul HTML al resursei indicate. Aplicația trebuie proiectată și implementată astfel încât să permită rularea încontinuu. În acest sens, modulul URL Frontier (coada de explorare) aferent va include inițial un set de două/trei URL-uri. Rularea încontinuu implică următorii pași:

1. se va prelua următorul URL din coada de explorare și se procesează astfel încât să se extragă numele domeniului explorat și URL-ul relativ al resursei dorite;

2. dacă domeniul este la prima explorare, atunci se va solicita resursa /robots.txt; dacă aceasta există, se trece la pasul 3, dacă nu se continuă cu pasul 4;

3. (dacă există robots.txt) se verifică clauzele Disallow pentru URL-ul relativ curent; dacă REP permite accesul pe resursă, atunci se trece la pasul 4, dacă nu, se trece la următorul URL din coada de explorare;

4. se preia resursa indicată de URL și se salvează local – pentru fiecare domeniu se va crea un director, apoi, în cadrul acestui director, se va urma structura de directoare din cadrul URL-ului;

5. (dacă este cazul) dacă se primește un cod 301 Moved Permanently, atunci se va reface cererea pentru noua locație și se vor actualiza datele deja salvate;

6. se analizează tag-ul HTML <meta, name="robots">; în cazul în care este permisă extragerea link-urilor incluse în document, se vor extrage aceste link-uri sub forma unui set de URL-uri absolute; din cadrul acestui set se elimină link-urile care nu respectă REP sau care se află deja în coada de explorare;

7. se reia pasul 1.

Cerințe bonus

Aplicația poate respecta următoarele cerințe bonus:

1. Cache DNS: aplicația implementează propriul mecanism de caching al înregistrărilor DNS.

2. performanțe: în mod secvențial, în cazul în care REP nu impune restricții de viteză, se dorește o rată de transfer de aproximativ 100 pagini/minut;

3. Crawler-ul WEB va fi distribuit/paralelizat pentru a spori performanțele; dacă, de exemplu, se vor utiliza două module de tip fetcher, atunci rata de transfer dorita va fi de cel put¸in 200 pagini/minut.

Capitolul 2. Prezentarea aplicației propuse ca soluție

Aplicația propusă a fost scrisă în limbajul Java, utilizând mediul de dezvoltare Eclipse IDE. De asemenea, au fost utilizată o librărie externă, și anume **JSoup** (pentru procesarea fișierelor HTML).Interfața cu utilizatorul este reprezentată de linia de comandă, meniul și interacțiunea fiind în mod text, simplu.

Exemplu de execuție a aplicației

```
----Link-ul: 95 ----
Adresa din coada: http://riweb.tibeica.com/crawl/hand-pub-alg.html
----Numele de domeniu  a fost gasit in cache----

----Link-ul: 96 ----
Adresa din coada: http://riweb.tibeica.com/crawl/dir-filter-of.html
----Numele de domeniu  a fost gasit in cache----

----Link-ul: 97 ----
Adresa din coada: http://riweb.tibeica.com/crawl/dir-filter-of.html
----Numele de domeniu  a fost gasit in cache----

----Link-ul: 98 ----
Adresa din coada: http://riweb.tibeica.com/crawl/dir-other-phm.html
----Numele de domeniu  a fost gasit in cache----

----Acest link a mai fost vizitat http://riweb.tibeica.com/crawl/dir-other-par.html----
----Link-ul: 99 ----
Adresa din coada: http://riweb.tibeica.com/crawl/dir-other-phm.html
----Numele de domeniu  a fost gasit in cache----

----Acest link a mai fost vizitat http://riweb.tibeica.com/crawl/dir-other-par.html----
----Link-ul: 100 ----
Adresa din coada: http://riweb.tibeica.com/crawl/inst-trouble.html
----Numele de domeniu  a fost gasit in cache----


Cele 100 link-uri au fost preluate cu succes!
Timpul de executie total este: 36 s
```

Capitolul 3. Explicarea modulelor aplicației

1. DNS Client

În clasa DnsClient am creat următoarele metode: createRequest, printArrayByte, getResponse şi getPointer. În medota createRequest creez cererea DNS:

- Setez dimensiuea vectorului pentru request ca fiind lungimea domeniului plus 18: 12 reprezintă dimensiunea exactă a header-ului, 2 byte pentru codificarea domeniului (unul de inceput şi unu de final), încă 4 bytes pentru setări suplimentare;
- Primii doi byte sunt pentru codul unic de identificare al cererii, acesta este un număr generat aleator mai mic sau egal cu $2^{16} - 1$;
- Recursion Desired este setat pe 1;
- Numărul de întrebări este setat pe 1;
- Question Type este setat pe 1 pentru tipul adresă IP;
- Question Class este setat pe 1 pentru clasa Internet;

În metoda getResponse se creează conexiunea prin inițializarea unui DatagramSocket. IP de pe care se face cererea este preluat din clasa Utility prin variabila Utility.dnsValue. Este trimisă cererea şi preluat raspunsul, după este închisă conexiunea.

În checkResponse este prelucrat răspunsul primit de la serverul DNS. Prima dată este verificată potrivirea identificatorului unuic al cererii, cel din răspuns trebuie să coincidă cu cel din cerere. După este verificat RCode dacă este 0 ceea ce înseamnă că nu s-a produs o eroare, în caz contrar este afişat codul de eroare. Ulterior este verificat numărul de raspunsuri primite, urmând să fie parcurs fiecare raspuns. Este preluat numele de particulă recursiv. După acesta se va sări peste un număr de octeţi dependent de tipul de particulă: în cazul in care este pointer se va sări cu 2 octeți sau dacă este nume de particulă se va sări cu dimensiunea acestuia.

Ulterior se va verifica Type Record: valoarea 1 pentru Type Record tip IPv4 sau valoarea 5 pentru tip Nume canonic. Pentru Record Class se va verifica daca valoarea acestuia este 1 pentru tipul Internet. Se va sări peste 4 Bytes care reprezintă bytes pentru TTL şi va fi preluat DataLength. Dacă acesta este 4 şi valoare Type Record este 1 atunci este tip IPv4 şi este preluată valoarea acestuia. Dacă Type Reocrd este 5 atunci este tip nume canonic şi se va reapela funcţia getPointer. După parcurgerea tuturor răspunsurilor se va returna adresa IP pentru domeniul respectiv.

2. HTTP Client

Această clasă cuprinde următoarele metode: getResource şi isAllowed. Prin metoda getResource se creează cererea HTTP şi se obţine resursa dorită. Cererea este de forma:

- GET nume_resursă HTTP/1.1\r\n
- Host: nume_domeniu\r\n
- User-Agent: nume_agent\r\n

Numele resursei, respectiv numele domeniului sunt preluate din adresa URL, forma generală fiind: **protocol_de_acces://nume_domeniu[:port][/cale/locală]**. În caz că portul nu este specificat va fi utilizat implicit portul 80. Este inițializat Socket-ul și este trimisă cererea către serverul HTTP unde este procesată și conceput un răspuns.

Răspunsurile de interes de la serverul HTTP sunt:

- **200 OK** – cererea a fost tratată și serverul a trimis resursa
- **301 Moved Permanently** – resursa a fost mutată permanent, caz în care domeniul este actualizat și cererea este refăcută către locația indicată de antetul „Location" din răspuns.
- **302 Found** – metodă clasică de redirecționare, se reface pur și simplu cererea
- **307 Moved Temporarily** – resursa este mutată temporar, se reface pur și simplu cererea.
- **404 Not Found** – resursă negăsită, ceea ce poate indica un link invalid sau o încercare de spider trap.

Resursele sunt salvate sub formă de fișiere HTML în structură arborescentă de foldere, folosind calea completă de tipul: **nume_de_domeniu/cale/locală/către/resursă.**

3. Robot Crawler

În această clasă sunt următoarele metode: proccesseQueue, getIP, getLinks. Inițial este introdus în coadă domeniul care urmează să fie analizat. Este setat numărul maxim de pagini ca fiind 100.

Este analizat primul domeniu din coadă. Se află adresa IP prin apelul metodei getIP. Aceasta realizează cererea DNS cu ajutorul metodelor din DNSClient, trimite cererea şi primeşte adresa IP. După pentru adresa URL respectivă se vor extrage portul, domeniul şi resursa care urmează să fie cerută. Dacă portul nu există va fi setat implicit pe 80. Se verifică dacă protocolul adresei este http, după aceea este creată cererea HTTP pentru fişierul robots.txt .

Aplicația este construită în așa fel încât să respecte directivele de permisiune / respingere a unui robot web pe baza semnăturii acestuia, directive aflate în folder-ul rădăcină al domeniului, în fișierul „robots.txt". Dacă s-a ajuns pe un domeniu neexplorat, se preia mai întâi acest fișier, se parsează setul de reguli și se stochează într-o structură de date pentru utilizare ulterioară.

Regulile sunt folosite pentru a stabili dacă avem voie sau nu să explorăm o anumită porţiune din website.

Exemple de reguli:

- User-agent: SEMNĂTURĂ_ROBOT
- Disallow:

- User-agent: *
- Disallow: /cale/locală/nepermisă
- User-agent: SEMNĂTURĂ_ROBOT_2
- Allow: /cale/locală/permisă

După ce este preluat fişierul robots.txt este salvat într-un fişier index.html în directorul **nume_de_domeniu/**. Odată descărcat fişierul, este apelată metoda isAllowed pentru a verifica dacă ar trebui să accesze resursa dorită sau nu.

Sunt verificate urmatoarele cazuri:

- Dacă User-Agent este „*” şi nu are nici o regulă în cadrul Disallow, atunci flag-ul este setat pe **TRUE**;
- Dacă User-Agent este identic cu numele agentului care efectuează cererea şi nu are nici o regulă în cadrul Disallow, atunci web Crawler va avea automat acces la resursa respectivă;
- Dacă User-Agent este identic cu numele agentului care efectuează cererea şi regula din cadrul Disallow este „/” atunci automat web Crawler nu va accesa acea resursă;
- Dacă User-Agent este „*” şi regula din cadrul Disallow este „/” atunci automat web Crawler nu va accesa acea resursă;
- Dacă Disallow conţine o adresă şi User-Agent-ul este „*” sau identic cu numele agentului care efectuează cererea atunci automat nu se va accesa resursa respectivă;

După este verificat codul HTTP pentru resursa respectivă:

- Dacă este 200 atunci adresa este memorată într-o coadă cu URL vizitate, este extrasă resursa HTML, iar din resursa respectivă sunt extrase toate Link-urile cu ajutoru funcţiei getLinks. Pentru fiecare link extras este verificat dacă există deja resursa cu calea respectivă, iar în caz că nu există este adăugat link-ul respectiv în coadă pentru a fi procesat:
- Dacă este 301 sau 307 atunci pentru vechiul domeniu este memorat noua cale către resursa respectivă.

Autoevaluare

Consider ca proiectul este de nota 9!