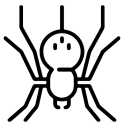
ALGORITMICA PER IL WEB

Prof. Sebastiano Vigna
6 CFU

Luca Cappelletti

Lecture Notes
Year 2017/2018



Magistrale Informatica
Università di Milano
Italy
October 10, 2017

Contents

Chapter 1

Chapter

1.1 Robots.txt

Si tratta di un file standard, inserito nella root di siti web, con il quale è possibile inserire rule che i crawler devono rispettare. Solitamente quando una persona vuole crawlare un sito va a leggere e parsare questo file una volta all'ora.

```
# Notice: Crawling Facebook is prohibited unless you have express written
# permission. See: http://www.facebook.com/apps/site_scraping_tos_terms.php

User-agent: Applebot
Disallow: /ajax/
Disallow: /album.php
Disallow: /checkpoint/
Disallow: /contact_importer/
Disallow: /feeds/
Disallow: /file_download.php
Disallow: /hashtag/
Disallow: /l.php
Disallow: /live/
Disallow: /moments_app/
Disallow: /p.php
Disallow: /photo.php
Disallow: /photos.php
Disallow: /sharer/

User-agent: baiduspider
Disallow: /ajax/
Disallow: /album.php
Disallow: /checkpoint/
Disallow: /contact_importer/
Disallow: /feeds/
Disallow: /file_download.php
Disallow: /hashtag/
Disallow: /l.php
Disallow: /live/
Disallow: /moments_app/
Disallow: /p.php
Disallow: /photo.php
Disallow: /photos.php
```

```
Disallow: /sharer/

User-agent: Bingbot
Disallow: /ajax/
Disallow: /album.php
Disallow: /checkpoint/
Disallow: /contact_importer/
Disallow: /feeds/
Disallow: /file_download.php
Disallow: /hashtag/
Disallow: /l.php
Disallow: /live/
Disallow: /moments_app/
Disallow: /p.php
Disallow: /photo.php
Disallow: /photos.php
Disallow: /sharer/

User-agent: Googlebot
Disallow: /ajax/
Disallow: /album.php
Disallow: /checkpoint/
Disallow: /contact_importer/
Disallow: /feeds/
Disallow: /file_download.php
Disallow: /hashtag/
Disallow: /l.php
Disallow: /live/
Disallow: /moments_app/
Disallow: /p.php
Disallow: /photo.php
Disallow: /photos.php
Disallow: /sharer/

User-agent: Applebot
Allow: /ajax/pagelet/generic.php/PagePostsSectionPagelet
Allow: /safetycheck/

User-agent: baiduspider
Allow: /ajax/pagelet/generic.php/PagePostsSectionPagelet
Allow: /safetycheck/

User-agent: Bingbot
Allow: /ajax/pagelet/generic.php/PagePostsSectionPagelet
Allow: /safetycheck/

User-agent: Googlebot
Allow: /ajax/pagelet/generic.php/PagePostsSectionPagelet
Allow: /safetycheck/

User-agent: ia_archiver
Allow: /about/privacy
```

```
Allow: /ajax/pagelet/generic.php/PagePostsSectionPagelet
Allow: /full_data_use_policy
Allow: /legal/terms
Allow: /policy.php
Allow: /safetycheck/

User-agent: msnbot
Allow: /ajax/pagelet/generic.php/PagePostsSectionPagelet
Allow: /safetycheck/

User-agent: *
Disallow: /
```

1.2 Multi-threading

Siccome l'IO della scheda di rete è a 2-3 decadi di tempo superiore rispetto all'analisi effettiva dei dati, è necessario creare una grande parallelizzazione (anche 5000 thread), con **crivello** o struttura condivisa.

1.2.1 Design sincrono

Un approccio è creare un thread immaginandolo come solitario, quindi avviarne 10000 e affidare il lavoro di sincronizzazione al sistema operativo. Questa sarà la versione che andremo a realizzare in questo corso. Con questo approccio abbiamo i seguenti problemi:

1. Ogni thread deve lavorare su un unico url, altrimenti incontriamo problemi con i siti. Viene risolto con la **coda con delay**, che aspetta che l'elemento in cima alla coda abbia un timestamp inferiore al timestamp corrente e blocca il thread fino a che non riceve l'ok dalla coda.

1.2.2 Design asincrono

Un altro approccio è lavorare alla sincronizzazione tra i thread avviando tante richieste e lasciando lavorare i thread mano a mano che arrivano le risposte delle richieste.

È dibattuto quale dei due approcci sia il più veloce.

1.3 Visit State (VS)

Per ogni sito abbiamo una coda fifo di url ed un timestamp, e quando un thread incontra un timestap minore di quello corrente acquisisce il **visit state** ed esegue la richiesta, scaricando alcuni degli url presenti alla lista connessa a quel nodo. Ad ogni visit state corrisponde un host, per cui nessun thread può fare richieste allo stesso host.

#	Timestamp	Host
0	1507621197	https://google.com
1	1507621097	https://facebook.com
i	...	...
n	1507601097	https://twitter.com

Figure 1.1: Coda di host del Visit State

#	Timestamp	URL
0	1507621197	https://google.com/search
1	1507601197	https://google.com/images
i	...	...
n	1504621197	https://google.com/maps

Figure 1.2: Coda degli url del primo Visit State

1.4 Problemi comuni

1. I server DNS centrali raramente reggono grossi carichi, per cui è necessario realizzare un **server ricorsivo locale**, per agente, separato. Le richieste a questo modo risultano apparire da IP diversi.
2. In aggiunta, per rispettare l'**IP politeness** è possibile realizzabile una coda di IP, contenente un timestamp che è i primo momento in cui si intende visitare quell'ip e un puntatore alla coda di visit state, di cui ogni nodo ha come priorità il valore pari al massimo tra il timestamp dell'ip e quello dei vhost.