



Exploiting Web and Social Networking Data

Pierre Senellart



16 February 2016



The World Wide Web

- Trillions of Web pages on the Web (only a fraction indexed by search engines)
- Numerous kinds:
 - Reference information
 - User-contributed data
 - E-commerce
 - Corporate sites
 - etc.



The World Wide Web

- Trillions of Web pages on the Web (only a fraction indexed by search engines)
- Numerous kinds:
 - Reference information
 - User-contributed data
 - E-commerce
 - Corporate sites
 - etc.
- Most of this **publicly accessible** and **easily retrievable**



The World Wide Web

- Trillions of Web pages on the Web (only a fraction indexed by search engines)
- Numerous kinds:
 - Reference information
 - User-contributed data
 - E-commerce
 - Corporate sites
 - etc.
- Most of this **publicly accessible** and **easily retrievable**
- Numerous applications for Web data:
 - General search
 - Knowledge base construction
 - Meta-analysis of scientific literature
 - Comparison shopping and marketing intelligence
 - etc.



Social data on the Web

Huge numbers of active users (2015):

Facebook 1.6 billion

QQ 860 million

Google+ 540 million

Instagram 400 million

Twitter 305 million

Weibo 220 million

LinkedIn 100 million



Social data on the Web

Huge numbers of **active** users (2015):

Facebook	1.6 billion
QQ	860 million
Google+	540 million
Instagram	400 million
Twitter	305 million
Weibo	220 million
LinkedIn	100 million

Huge volume of shared data:

500 million tweets per day on Twitter (6,000 per second on average!)...

... including statements by heads of states, revelations of political activists, etc.

Dmitry Medvedev @MedvedevRussiaE 12 Jul 10
Iran may soon acquire nuclear capability. The Non-Proliferation Treaty doesn't prohibit having such capability. That's one of the problems.

Voice of Tunisia @Voiceoftunisia 14 Jan 11
Be ready! RCD is preparing an attempt to steal the demonstration. Don't give him a chance! Ben Ali Out! #sidibouzid #tunisia #jasminrevolt



How to exploit Web and social data?

- First step: Collect relevant data by **crawling** (this talk)
- Second step: Extract structured and semantic information from raw collected data
- Third step: Use the extracted information for a given task



Plan

Introduction

Basics of Web Crawling

Crawling Complex Content

Conclusion



Web crawlers

- **crawlers, (Web) spiders, (Web) robots**: autonomous user agents that retrieve pages from the Web
- Basics of crawling:
 1. Start from a given URL or set of URLs
 2. Retrieve and process the corresponding page
 3. Discover new URLs on that page
 4. Repeat on each found URL
- No real termination condition (unlimited number of Web pages!)
- **Graph-browsing** problem
 - deep-first**: not very adapted, possibility of being lost in **robot traps**
 - breadth-first**
 - combination of both**: breadth-first with limited-depth deep-first on each discovered website



Scope of a crawler

- Web-scale
 - The Web is infinite! Avoid robot traps by putting depth or page number **limits** on each Web server
 - Focus on **important** pages [Abiteboul et al., 2003]
- Web servers under a list of **DNS domains**: easy filtering of URLs
- A given topic: **focused crawling** techniques [Chakrabarti et al., 1999, Diligenti et al., 2000, Gouriten et al., 2014] based on classifiers of Web page content and predictors of the interest of a link.
- The national Web (cf. **public deposit**, national libraries): what is this? [Abiteboul et al., 2002]
- A given Web site: what is a Web site? [Senellart, 2005]



Crawling ethics

- Standard for robot exclusion: `robots.txt` at the root of a Web server [Koster, 1994].

```
User-agent: *  
Allow: /searchhistory/  
Disallow: /search
```

- Per-page exclusion.

```
<meta name="ROBOTS" content="NOINDEX,NOFOLLOW">
```

- Per-link exclusion.

```
<a href="toto.html" rel="nofollow">Toto</a>
```

- Avoid **Denial Of Service** (DOS), wait $\approx 1s$ between two repeated requests to the same Web server



Legal aspects (France) – 1/3

- General principles:
 - to access or keep access to a “system for automated data processing” *in a fraudulent manner* is punished of two years of prison and 60,000 euros fine (Code pénal 323-1, modified by law 2015-912 on “Renseignement”)
 - to disrupt the functioning of a “system for automated data processing” is punished of five years of prison and 150,000 euros fine, extended to seven years and 300,000 euros when the system is a public one containing personal information (Code pénal 323-2, modified by law 2015-912 on “Renseignement”)
- A Web site hosted in a different country may invoke completely different legal principles, under a different jurisdiction
- Crawling content can be considered accessing and keeping access to a “system for automated data processing” (Cour d’appel de Paris, 5 February 2014, “Bluetouff case”)



Legal aspects (France) – 2/3

- robots.txt files de facto standard, and instructions in robots.txt files can be taken as a receivable way to specify what can be crawled (Cour d'appel de Paris, 26 January 2011, Google vs SAIF)
- Frequent requests to a Web site can be considered as a way to disrupt the functioning of a “system for automated data processing” (Cour d'appel de Bordeaux, 15 November 2011, Cédric M. vs C-Discount), but only if it reaches abusive levels and can be shown to have caused disruption



Legal aspects (France) – 3/3

- Web content is subject to “droit d’auteur” (Code de la propriété intellectuelle, Première partie, Livre Ier) and cannot generally be broadcast by third-parties; only transient copies are allowed (CJEU, 5 June 2014, PRCA vs NLA)
- Web content containing personal data is even more sensitive (Loi “Informatique et Libertés”): personal data should be collected for a specific purpose, and should be kept updated



Introduction

Basics of Web Crawling

Crawling Complex Content

Modern Web Sites

Social Networking Sites

Conclusion



Introduction

Basics of Web Crawling

Crawling Complex Content

Modern Web Sites

Social Networking Sites

Conclusion



Crawling modern Web sites

- Some modern Web sites only work when cookies are activated (**session cookies**), or when **JavaScript code** is interpreted
- Regular Web crawlers (**wget**, **Heritrix**, **Apache Nutch**) do not usually perform any cookie management and do not interpret JavaScript code
- Crawling of some Websites therefore require more **advanced tools**



Advanced crawling tools

Web scraping frameworks such as **scrapy** (Python) or **WWW::Mechanize** (Perl) simulate a Web browser interaction and cookie management (but no JS interpretation)

Headless browsers such as **htmlunit** simulate a Web browser, including simple JavaScript processing

Browser instrumentors such as **Selenium** allow full instrumentation of a regular Web browser (Chrome, Firefox, Internet Explorer)

OXPath: a **full-fledged navigation and extraction language** for complex Web sites [Sellers et al., 2011]



Introduction

Basics of Web Crawling

Crawling Complex Content

Modern Web Sites

Social Networking Sites

Conclusion



Crawling social networks

- Theoretically possible to crawl social networking sites using a **regular Web crawler**
- Sometimes not allowed: `https://www.facebook.com/robots.txt`
- Often **very inefficient**, considering politeness constraints
- Better solution: Use provided social networking APIs
 - `https://dev.twitter.com/docs/api/1.1`
 - `https://developers.facebook.com/docs/graph-api/reference/v2.1/`
 - `https://developer.linkedin.com/apis`
 - `https://developers.google.com/youtube/v3/`
- Also possible to buy access to the data, directly from the social network or from brokers such as `http://gnip.com/`



Social networking APIs

- Most social networking Web sites (and some other kinds of Web sites) provide **APIs** to effectively access their content
- Usually require a **token** identifying the application using the API, sometimes a cryptographic signature as well
- May access the API as an authenticated user of the social network, or as an **external party**
- APIs seriously limit the **rate of requests**:

`https://dev.twitter.com/docs/api/1.1/get/search/tweets`



The case of Twitter

- Two main APIs:
 - **REST APIs**, including search, getting information about a user, a list, followers, etc. <https://dev.twitter.com/docs/api/1.1>
 - **Streaming API**, providing real-time result
- **Very limited history** available
- Search can be on **keywords**, **language**, **geolocation** (for a small portion of tweets)



Cross-network crawling

- Often useful to combine results from **different social networks**
- Numerous libraries facilitating SN API accesses (twipy, Facebook4J, FourSquare VP C++ API...) **incompatible with each other**... Some efforts at generic APIs (OneAll, APIBlender [Gouriten and Senellart, 2012])
- **Example use case:** No API to get all check-ins from FourSquare, but a number of check-ins are available on Twitter; given results of Twitter Search/Streaming, use FourSquare API to get information about check-in locations.



Plan

Introduction

Basics of Web Crawling

Crawling Complex Content

Conclusion



Conclusion

In Brief

- Crawling as a **graph-browsing** problem.
- Number of **de facto rules** to respect when crawling the Web
- Crawling modern Web content is **not as easy** as launching a traditional Web crawler
- Ideally: a traditional large-scale crawler that knows **when to delegate** to more specialized crawling mechanisms (tools querying social networking APIs, JS-aware crawlers, etc.)
- Huge variety of tools, techniques, suitable for different needs



References

Free software

wget simple yet effective Web spider

Heritrix Web-scale highly configurable Web crawler, used by the Internet Archive

Beautiful Soup Python module for parsing real-world Web pages

Scrapy rich Python module for Web crawling and content extraction

Selenium browser instrumentor, with API in several languages

To go further

- A good textbook [Chakrabarti, 2003]

Serge Abiteboul, Grégory Cobena, Julien Masanès, and Gerald Sedrati. A first experience in archiving the French Web. In *Proc. ECDDL*, Roma, Italie, September 2002.

Serge Abiteboul, Mihai Preda, and Gregory Cobena. Adaptive on-line page importance computation. In *Proc. WWW*, May 2003.

Soumen Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann, San Fransisco, USA, 2003.

Soumen Chakrabarti, Martin van den Berg, and Byron Dom. Focused crawling: A new approach to topic-specific Web resource discovery. *Computer Networks*, 31(11–16):1623–1640, 1999.

Michelangelo Diligenti, Frans Coetzee, Steve Lawrence, C. Lee Giles, and Marco Gori. Focused crawling using context graphs. In *Proc. VLDB*, Cairo, Egypt, September 2000.

Georges Gouriten and Pierre Senellart. API Blender: A uniform interface to social platform APIs. In *Proc. WWW*, Lyon, France, April 2012. Developer track.

Georges Gouriten, Silviu Maniu, and Pierre Senellart. Scalable, generic, and adaptive systems for focused crawling. In *Proc. Hypertext*, Santiago, Chile, September 2014. Douglas Engelbart Best Paper Award.

Martijn Koster. A standard for robot exclusion.

<http://www.robotstxt.org/orig.html>, June 1994.

Andrew Sellers, Tim Furche, Georg Gottlob, Giovanni Grasso, and Christian Schallhart. Exploring the Web with XPath. In *LWDM*, 2011.

Pierre Senellart. Identifying Websites with flow simulation. In *Proc. ICWE*, pages 124–129, Sydney, Australia, July 2005.