Web Scraping with R

Xiao Nan    @road2stat

6th China R Beijing
Outline

• Overview
• Toolkit
• Exception Handling
• Parallelization
• Outro
Part I
Overview
Two Types of Scrapers / Crawlers
The **REAL** ones and the ...
Fake ones?
I'm not a fake!
I can crawl the web, too!
Two Types of Scrapers

- General-purpose Crawlers
- Focused Crawlers (our focus today)
Browser Revisited

User

Request

Browser

Response

Server
Browser Revisited

User → Browser → Server

Request → Response

Analysis and Retrieve
Browser Revisited

User → Browser
Request
Browser → Server
Response
Analysis and Retrieve
User → Browser
Receive and Parse
Comparing to Other Languages

Pros & Cons

Pros

- Lightweight
- Easy to implement
- Easy to debug
- Seamless modeling integration: less I/O
Comparing to Other Languages
Pros & Cons

Pros
- Lightweight
- Easy to implement
- Easy to debug
- Seamless modeling integration: less I/O

Cons
- Fewer libraries (than Python & Ruby)
- Multi-Process Parallelization: forking is deficient ...
Choosing A Better Platform
Choosing A Better Platform

Why Linux?
Choosing A Better Platform

Why Linux?

- Network performance & mem. management → Faster
Why Linux?

- Network performance & mem. management → Faster
- Better parallelization support → Faster
Choosing A Better Platform

Why Linux?

- Network performance & mem. management → Faster
- Better parallelization support → Faster
- Unified encoding & locale → Faster (for coders)
Choosing A Better Platform

Why Linux?

- Network performance & mem. management $\rightarrow$ Faster
- Better parallelization support $\rightarrow$ Faster
- Unified encoding & locale $\rightarrow$ Faster (for coders)
- More recent third party libs $\rightarrow$ Faster (less bugs)
Part 2

Toolkit
Retrieve & Parse
### Toolkit

#### Available R Packages

<table>
<thead>
<tr>
<th>Pkg. Name</th>
<th>Retrieve?</th>
<th>Parse?</th>
<th>Must-Know?</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCurl</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>XML</td>
<td>Limited</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>rjson</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>RJSONIO</td>
<td>No</td>
<td>Yes</td>
<td>Optional</td>
</tr>
<tr>
<td>httr</td>
<td>Yes</td>
<td>Yes</td>
<td>Optional</td>
</tr>
<tr>
<td>selectr</td>
<td>No</td>
<td>Yes</td>
<td>Optional</td>
</tr>
<tr>
<td>ROAuth</td>
<td>No</td>
<td>No</td>
<td>Optional</td>
</tr>
</tbody>
</table>
Toolkit

Available R Packages

RCurl

Parsing

[JSON]

Lightweight

[rjson]

Retrieving

[XML]

Heavyweight

[XML]

ROAuth

RJSONIO

httr
Available R Packages

- **RCurl** → Header Configuration
  
  R 不务正业之 RCurl: [http://cos.name/cn/topic/17816](http://cos.name/cn/topic/17816)

- **XML** → XPath, 3 Critical Functions
  

- **rjson** → Oficially Listed / RJSONIO → by D.T.L.
  
- **httr** → Simplification version RCurl + XML + rjson
   Not Recommended for not discreet enough:

- **ROAuth** → Useful for APIs. see RWeibo of @lijian001

- **selectr** → Translate CSS Selectors to XPath Expressions
Available R Packages

- **RCurl** → Header Configuration
  
  R 不务正业之 RCurl: http://cos.name/cn/topic/17816

- **XML** → **XPath, 3 Critical Functions**
  
  http://www.road2stat.com/cn/r/rxml.html
Toolkit
Available R Packages

- **RCurl** → Header Configuration
  R 不务正业之 RCurl: http://cos.name/cn/topic/17816

- **XML** → XPath, 3 Critical Functions
  http://www.road2stat.com/cn/r/rxml.html

- **rjson** → Officially Listed / **RJSONIO** → by D.T.L.
Available R Packages

- **RCurl** → Header Configuration
  
  R 不务正业之 RCurl: [http://cos.name/cn/topic/17816](http://cos.name/cn/topic/17816)

- **XML** → XPath, 3 Critical Functions
  

- **rjson** → Officially Listed / **RJSONIO** → by D.T.L.

- **httr** → Simplification version RCurl + XML + rjson
  
  Not Recommended for not discreet enough:

**Toolkit**

Available R Packages

- **RCurl** → Header Configuration
  
  R 不务正业之 RCurl: http://cos.name/cn/topic/17816

- **XML** → XPath, 3 Critical Functions
  
  [Link](http://www.road2stat.com/cn/r/rxml.html)

- **rjson** → Officially Listed / **RJSONIO** → by D.T.L.

- **httr** → Simplification version RCurl + XML + rjson
  
  Not Recommended for not discreet enough:
  
  [Link](http://randyzwitch.com/r-error-message-fun/)

- **ROAuth** → Useful for APIs. see RWeibo of @lijian001
Available R Packages

- **RCurl** → Header Configuration
  
  R 不务正业之 RCurl: [http://cos.name/cn/topic/17816](http://cos.name/cn/topic/17816)

- **XML** → XPath, 3 Critical Functions
  

- **rjson** → Officially Listed / **RJSONIO** → by D.T.L.

- **httr** → Simplification version RCurl + XML + rjson
  
  Not Recommended for not discreet enough:
  

- **ROAuth** → Useful for APIs. see RWeibo of @lijian001

- **selectr** → Translate CSS Selectors to XPath Expressions
Toolkit
Front-End and Miscellaneous

• Chrome Developer Tools / FireBug →
  Analyzing AJAX Requests: http://cos.name/cn/topic/107729
Toolkit
Front-End and Miscellaneous

• Chrome Developer Tools / FireBug →
  Analyzing AJAX Requests: http://cos.name/cn/topic/107729

• JSONView → Output Formatted JSON
**Toolkit**

Front-End and Miscellaneous

- **Chrome Developer Tools / FireBug →** Analyzing AJAX Requests: [http://cos.name/cn/topic/107729](http://cos.name/cn/topic/107729)
- **JSONView →** Output Formatted JSON
- **Visual Event →** Bounded event on DOM elements
Toolkit
Front-End and Miscellaneous

• Chrome Developer Tools / FireBug → Analyzing AJAX Requests: http://cos.name/cn/topic/107729
• JSONView → Output Formatted JSON
• Visual Event → Bounded event on DOM elements
• tcpdump + Wireshark → Packet Capture & Protocol Analysis
Part 3
Exception Handling
More than 70%
Exception Handling

Coding Strategy

- Dirty HTML & XML: Preprocess with htmltidy
Exception Handling

Coding Strategy

- Dirty HTML & XML: Preprocess with htmltidy
- Build-in Condition/Error Handler Function: XML::xmlStructuredStop
Exception Handling

Coding Strategy

- Dirty HTML & XML: Preprocess with htmltidy
- Build-in Condition/Error Handler Function: XML::xmlStructuredStop
- Coding Strategy: Interactive until fault-tolerant.
Exception Handling
FAQ on COS BBS

- Cookie Operation → http://cos.name/cn/topic/108806
- Referer Validation → http://cos.name/cn/topic/109407
- Session Validation → http://cos.name/cn/topic/107802
- Encoding Errors → Identify the Problem Source

Google: 编码 site:cos.name/cn/
Exception Handling
The Various Data Source

- Choose Official API first: NCBI with rOpenSci
Exception Handling

The Various Data Source

- Choose Official API first: NCBI with rOpenSci
- Restricted API usage: Private API key
Twitter公式クライアントのコンシューマキー

Twitter for iPhone

Consumer key: IQKbtAYlXLripLGFPWd0HUA
Consumer secret: GgDY1kSvaPxGxC4X8liwpUcqRwwr3lCADbz8A7ADU

Twitter for Android

Consumer key: 3nVuSo8Zzx6U4vzUxf5w
Consumer secret: Bcs59EFbbsdF6s19Ng71smgStWECwXXKSjYvPVt7qys

Twitter for Google TV

Consumer key: iAlYJ4HqUVfIuoNhif1DA
Consumer secret: 172fOpzu2cZyNyA3mMYvE8m8MEEyLbztOdrbUOlU

Twitter for iPad

Consumer key: CjulERSDeqhhjSme66ECg
Consumer secret: IQWdVqvFxghAtURHGeGiWAsmCAGndW3WmbEx6Hck

Twitter for Mac
Exception Handling

The Various Data Source

- Choose Official API first (NCBI with rOpenSci)
- Restricted API usage: Private API key
- SSL and SSL Decryption: Trusted MITM

Part 4
Parallelization
Parallelization

The best solution is?

A Conventional Way: `RCurl::getURIAsynchronous()`
Parallelization

The best solution is?

A Conventional Way: `RCurl::getURIAsynchronous()`

- Native. Extremely easy to use.
Parallelization

The best solution is?

A Conventional Way: `RCurl::getURIAsynchronous()`

- Native. Extremely easy to use.
- Pitfalls: Have to control the process number by hand. This seems weird!
Parallelization

The best solution is?

A Better Way: \textcolor{red}{doMC + foreach}
Parallelization

The best solution is?

A Better Way: doMC + foreach

• Full control / Easy to migrate / Natural to code:

```r
require(doMC)
registerDoMC(20)
x <- foreach(i = 1:1e+5, ...) %dopar% {
  xxx <- getURL(urls[i])
}
```
Parallelization
The best solution is?

A Better Way: \texttt{doMC} + \texttt{foreach}

- Full control / Easy to migrate / Natural to code:

```r
require(doMC)
registerDoMC(20)
x <- foreach(i = 1:1e+5, ...) %dopar% {
  xxx <- getURL(urls[i])
}
```

- Single machine, \texttt{registerDoMC}(20), 10 min, 1e+5 pages.
Parallelization

The best solution is?

A Better Way: doMC + foreach

• Full control / Easy to migrate / Natural to code:

```r
require(doMC)
registerDoMC(20)
x <- foreach(i = 1:1e+5, ...) %dopar% {
  xxx <- getURL(urls[i])
}
```

• Single machine, registerDoMC(20), 10 min, 1e+5 pages.
• Pitfalls: (Almost) Linux only.
Parallelization

More Pitfalls

- Requires high-perf storage → Redis (rredis) or MongoDB (RMongo)?
- Memory leak (RCurl & XML) → Avoid long exec. time
- Intensive testing before run → Minimize errors
Part 5

Outro
Remarks
Web Crawler Ethics

- Web Crawler Ethics
Remarks
Web Crawler Ethics

- Web Crawler Ethics
- Honor robots.txt
Remarks
Web Crawler Ethics

- Web Crawler Ethics
- Honor robots.txt
- A Balanced Crawling Rate
Remarks

Web Crawler Ethics

- Web Crawler Ethics
- Honor robots.txt
- A Balanced Crawling Rate
- Spammer Shame
Web Crawler Ethics

- Web Crawler Ethics
- Honor robots.txt
- A Balanced Crawling Rate
- Spammer Shame
- With great power comes great responsibility.
Further Reading

1. XML & JSON Specification (esp. XPath)
2. RCurl & XML Documentation
3. Web Data Mining (Chapter 8) by Bing Liu
4. XML and Web Technologies for Data Sciences with R by Duncan Temple Lang, et al. (Due Sep. 2013)
Q & A