IBEX: Harvesting Entities from the Web Using Unique Identifiers

Aliaksandr Talaika$^1$, Joanna Biega$^1$, Antoine Amarilli$^2$, Fabian Suchanek$^2$

$^1$Max Planck Institute for Informatics, Germany

$^2$Télécom ParisTech, France

May 31st, 2015
Identifiers on the Web

- It is tricky to extract named entities from Web pages
Identifiers on the Web

- It is tricky to extract named entities from Web pages
Identifiers on the Web

- It is tricky to extract named entities from Web pages
- Some entities have identifiers with recognizable syntax
Identifiers on the Web

- It is tricky to extract named entities from Web pages
- Some entities have identifiers with recognizable syntax
Identifiers on the Web

- It is tricky to extract named entities from Web pages
- Some entities have identifiers with recognizable syntax
- We focus on the following id types:
  - GTINs (products): 8–14 digits
  - CAS (chemicals): 8 digits
  - DOIs (documents): numerical prefix, ‘/’
  - Email addresses (people)
Names for IDs

- We will extract identifiers from Web pages
- We also want a human-readable name
Names for IDs

- We will extract identifiers from Web pages
- We also want a human-readable name

→ Names for IDs often occur close to the IDs
Names for IDs

- We will extract *identifiers* from Web pages
- We also want a human-readable *name*
- Names for IDs often occur *close* to the IDs
Names for IDs

- We will extract identifiers from Web pages
- We also want a human-readable name

→ Names for IDs often occur close to the IDs
Names for IDs

- We will extract identifiers from Web pages
- We also want a human-readable name

→ Names for IDs often occur close to the IDs
→ Challenges:
  - Which text is the name?
Names for IDs

We will extract identifiers from Web pages

We also want a human-readable name

→ Names for IDs often occur close to the IDs

→ Challenges:
  • Which text is the name?
Names for IDs

- We will extract identifiers from Web pages
- We also want a human-readable name

→ Names for IDs often occur close to the IDs

→ Challenges:
  - Which text is the name?
  - Which name matches which ID?
We will extract identifiers from Web pages
We also want a human-readable name
→ Names for IDs often occur close to the IDs
→ Challenges:
  - Which text is the name?
  - Which name matches which ID?
Names for IDs

- We will extract identifiers from Web pages
- We also want a human-readable name

→ Names for IDs often occur close to the IDs

→ Challenges:
  - Which text is the name?
  - Which name matches which ID?
The problem

- Given a **Web crawl** (collection of pages) and **ID formats**: 
The problem

- Given a Web crawl (collection of pages) and ID formats:

  GTIN: nnnnnnnnnnnnnnnnnnnn
  CAS: nnnnn-pp-q
  Email: xxx@yyy.zzz
The problem

- Given a Web crawl (collection of pages) and ID formats:

  HTML
  <html>
  <title>
  <body> <p>...
  HTML
  <html>
  <title>
  <body> <p>...
  HTML
  <html>
  <title>
  <body> <p>...

  GTIN  nnnnnnnnnnnnnnn  CAS  nnnnn-pp-q  email  xxx@yyy.zzz

  → Find out the IDs that occur in the crawl
  → Find out the right name for each of them
The problem

- Given a Web crawl (collection of pages) and ID formats:

  HTML
  <html>
  <title>
  <body> <p>...
  HTML
  <html>
  <title>
  <body> <p>...
  HTML
  <html>
  <title>
  <body> <p>...
  HTML
  <html>
  <title>
  <body> <p>...
  HTML

  GTIN  nnnnnnnnnnnnnn
  CAS  nnnnn-pp-q
  email  xxx@yyy.zzz

  → Find out the IDs that occur in the crawl
  → Find out the right name for each of them

<table>
<thead>
<tr>
<th>GTIN</th>
<th>CAS</th>
<th>email</th>
</tr>
</thead>
<tbody>
<tr>
<td>8806085560352</td>
<td>Samsung I9505</td>
<td><a href="mailto:jd@applesaft.com">jd@applesaft.com</a></td>
</tr>
<tr>
<td>8806085601932</td>
<td>Samsung SM S24C770T</td>
<td><a href="mailto:ds@macrosoft.com">ds@macrosoft.com</a></td>
</tr>
<tr>
<td>10049-04-4</td>
<td>Chlorine dioxide</td>
<td></td>
</tr>
<tr>
<td>email</td>
<td>John Doe</td>
<td></td>
</tr>
<tr>
<td>email</td>
<td>David Smith</td>
<td></td>
</tr>
</tbody>
</table>
Related work

**Named Entity Recognition.** Cannot figure out the ID–name map
Related work

**Named Entity Recognition.** Cannot figure out the ID–name map

**Wrapper induction.** Assumes all pages are similar
Related work

**Named Entity Recognition.** Cannot figure out the ID–name map

**Wrapper induction.** Assumes all pages are similar

**Product extraction.** Usually completes existing databases
Related work

**Named Entity Recognition.** Cannot figure out the ID–name map

**Wrapper induction.** Assumes all pages are similar

**Product extraction.** Usually completes existing databases

**Knowledge bases.** Insufficient coverage
Related work

**Named Entity Recognition.** Cannot figure out the ID–name map

**Wrapper induction.** Assumes all pages are similar

**Product extraction.** Usually completes existing databases

**Knowledge bases.** Insufficient coverage

**Existing databases.** Not freely downloadable, and domain-specific
Related work

**Named Entity Recognition.** Cannot figure out the ID–name map

**Wrapper induction.** Assumes all pages are similar

**Product extraction.** Usually completes existing databases

**Knowledge bases.** Insufficient coverage

**Existing databases.** Not freely downloadable, and domain-specific

→ Relying on IDs will make our life easier!
Table of contents

1 Introduction
2 Extracting candidates
3 Cleaning up candidates
4 Experimental results
5 Conclusion
Task description

Extract candidate name–ID pairs from pages in parallel:

crawl

![HTML](<html><title><body><p>...<br>extractor<br>ID Name<br>... ...<br>ID Name<br>... ...<br>ID Name<br>... ...)

candidates
HTML parsing

```html
<body>
<h1>Galaxy S6</h1>
<p>Id: <b>8806</b></p>
<h1>Gear</h1>
<h2>S6 Cable</h2>
4047
</body>
```

- Custom DOM parser
- Knowledge on tag nestings
- Regroup headers and content
**HTML parsing**

```
<body>
<h1>Galaxy S6</h1>
<p>Id:  <b>8806</b></p>
<h1>Gear</h1>
<h2>S6 Cable</h2>
4047
</body>
```

- Custom DOM parser
- Knowledge on tag nestings
- Regroup headers and content
HTML parsing

- Custom DOM parser
- Knowledge on tag nestings
- Regroup headers and content
  - Fast (Web-scale)
  - Agnostic (no assumptions)
  - Resilient (real HTML sucks)
  - Simple (clean up later)
Extracting pairs

- Use the **pattern** to find **IDs**
- **Record**: maximal subtree containing *only one ID*
  - Detail record (one)
  - Free record (many)
- **Leaves** in each record are the *name candidates*
Extracting pairs

- Use the pattern to find IDs
- **Record**: maximal subtree containing **only one ID**
  - Detail record (one)
  - Free record (many)
- **Leaves** in each record are the name candidates
Extracting pairs

- Use the pattern to find IDs
- **Record**: maximal subtree containing only one ID
  - Detail record (one)
  - Free record (many)
- Leaves in each record are the name candidates
## Extracting pairs

### Extracting candidates

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>8806</td>
<td>Galaxy S6</td>
</tr>
<tr>
<td>8806</td>
<td>Id:</td>
</tr>
</tbody>
</table>

### Cleaning up candidates

- **Use the pattern** to find IDs
- **Record**: maximal subtree containing **only one ID**
  - Detail record (one)
  - Free record (many)
- **Leaves** in each record are the **name candidates**
Extracting pairs

Use the pattern to find IDs

**Record**: maximal subtree containing only one ID
   - Detail record (one)
   - Free record (many)

- Leaves in each record are the name candidates

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>8806</td>
<td>Galaxy S6</td>
</tr>
<tr>
<td>8806</td>
<td>Id:</td>
</tr>
<tr>
<td>4047</td>
<td>Gear</td>
</tr>
<tr>
<td>4047</td>
<td>S6 Cable</td>
</tr>
</tbody>
</table>
Table of contents

1. Introduction
2. Extracting candidates
3. Cleaning up candidates
4. Experimental results
5. Conclusion
Task description

Clean up the junk in **ID–name pairs**

<table>
<thead>
<tr>
<th>Page</th>
<th>ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>page1.html</td>
<td>9780261102361</td>
<td>The Two Towers</td>
</tr>
<tr>
<td>page1.html</td>
<td>9780261102361</td>
<td>J. R. R. Tolkien</td>
</tr>
<tr>
<td>page1.html</td>
<td>9780261102354</td>
<td>The Fellowship of the Ring</td>
</tr>
<tr>
<td>page1.html</td>
<td>9780261102354</td>
<td>J. R. R. Tolkien</td>
</tr>
<tr>
<td>page2.html</td>
<td>9780261102354</td>
<td>The Lord of the Rings (Part 1)</td>
</tr>
<tr>
<td>page3.html</td>
<td>9780261102354</td>
<td>The Fellowship of the Ring</td>
</tr>
</tbody>
</table>

→ Idea: unlike real names, bad names are **not specific** to an ID
Filtering names

- Group by name
- Consider the IDs for each name

<table>
<thead>
<tr>
<th>Name</th>
<th>ID</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>J. R. R. Tolkien</td>
<td>9780261102361</td>
<td>page1.html</td>
</tr>
<tr>
<td>J. R. R. Tolkien</td>
<td>9780261102354</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Fellowship of the Ring</td>
<td>9780261102354</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Fellowship of the Ring</td>
<td>9780261102354</td>
<td>page3.html</td>
</tr>
<tr>
<td>The Two Towers</td>
<td>9780261102361</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Lord of the Rings (Part 1)</td>
<td>9780261102354</td>
<td>page2.html</td>
</tr>
</tbody>
</table>
Filtering names

- Group by name
- Consider the IDs for each name

<table>
<thead>
<tr>
<th>Name</th>
<th>ID</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>J. R. R. Tolkien</td>
<td>9780261102361</td>
<td>page1.html</td>
</tr>
<tr>
<td>J. R. R. Tolkien</td>
<td>9780261102354</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Fellowship of the Ring</td>
<td>9780261102354</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Fellowship of the Ring</td>
<td>9780261102354</td>
<td>page3.html</td>
</tr>
<tr>
<td>The Two Towers</td>
<td>9780261102361</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Lord of the Rings (Part 1)</td>
<td>9780261102354</td>
<td>page2.html</td>
</tr>
</tbody>
</table>
Deciding specificity

For each name, consider the histogram of ID occurrences:

→ most frequent ID $id_1$ must be much more frequent than $id_2$
→ $id_1$ must be sufficiently frequent overall
Putting it together

- We have eliminated *unspecific names*

<table>
<thead>
<tr>
<th>Name</th>
<th>ID</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>J. R. R. Tolkien</td>
<td>9780261102361</td>
<td>page1.html</td>
</tr>
<tr>
<td>J. R. R. Tolkien</td>
<td>9780261102354</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Fellowship of the Ring</td>
<td>9780261102354</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Fellowship of the Ring</td>
<td>9780261102354</td>
<td>page3.html</td>
</tr>
<tr>
<td>The Two Towers</td>
<td>9780261102361</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Lord of the Rings (Part 1)</td>
<td>9780261102354</td>
<td>page2.html</td>
</tr>
</tbody>
</table>
Putting it together

- We have eliminated unspecified names
- Some IDs may still have multiple names
  - Group by ID
  - Keep the most popular name

<table>
<thead>
<tr>
<th>Name</th>
<th>ID</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>J. R. R. Tolkien</td>
<td>9780261102361</td>
<td>page1.html</td>
</tr>
<tr>
<td>J. R. R. Tolkien</td>
<td>9780261102354</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Fellowship of the Ring</td>
<td>9780261102354</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Fellowship of the Ring</td>
<td>9780261102354</td>
<td>page3.html</td>
</tr>
<tr>
<td>The Two Towers</td>
<td>9780261102361</td>
<td>page1.html</td>
</tr>
<tr>
<td>The Lord of the Rings (Part 1)</td>
<td>9780261102354</td>
<td>page2.html</td>
</tr>
</tbody>
</table>
Putting it together

- We have eliminated *unspecific names*
- Some IDs may still have *multiple names*
  - Group by ID
  - Keep the *most popular name*

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9780261102361</td>
<td>The Two Towers</td>
<td>page1.html</td>
</tr>
<tr>
<td>9780261102354</td>
<td>The Fellowship of the Ring</td>
<td>page1.html</td>
</tr>
<tr>
<td>9780261102354</td>
<td>The Fellowship of the Ring</td>
<td>page3.html</td>
</tr>
<tr>
<td>9780261102354</td>
<td>The Lord of the Rings (Part 1)</td>
<td>page2.html</td>
</tr>
</tbody>
</table>
Putting it together

- We have eliminated *unspecific names*
- Some IDs may still have *multiple names*
  - Group by ID
  - Keep the *most popular name*

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9780261102361</td>
<td>The Two Towers</td>
<td>page1.html</td>
</tr>
<tr>
<td>9780261102354</td>
<td>The Fellowship of the Ring</td>
<td>page1.html</td>
</tr>
<tr>
<td>9780261102354</td>
<td>The Fellowship of the Ring</td>
<td>page3.html</td>
</tr>
<tr>
<td>9780261102354</td>
<td>The Lord of the Rings (Part 1)</td>
<td>page2.html</td>
</tr>
</tbody>
</table>
Putting it together

- We have eliminated *unspecific names*
- Some IDs may still have *multiple names*
  - Group by ID
  - Keep the most popular name

→ We have our **final result**: IDs and their name

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9780261102361</td>
<td>The Two Towers</td>
<td>page1.html</td>
</tr>
<tr>
<td>9780261102354</td>
<td>The Fellowship of the Ring</td>
<td>page1.html</td>
</tr>
<tr>
<td>9780261102354</td>
<td>The Fellowship of the Ring</td>
<td>page3.html</td>
</tr>
<tr>
<td>9780261102354</td>
<td>The Lord of the Rings (Part 1)</td>
<td>page2.html</td>
</tr>
</tbody>
</table>
# Table of contents

1. Introduction
2. Extracting candidates
3. Cleaning up candidates
4. Experimental results
5. Conclusion
Experimental setup

- English portions of ClueWeb09 and ClueWeb12
  → 35 TB of data
  → 1.2 billion Web pages
Experimental setup

- English portions of ClueWeb09 and ClueWeb12
  - → 35 TB of data
  - → 1.2 billion Web pages
- ID types: GTINs, CAS numbers, DOIs, emails
Experimental setup

- English portions of ClueWeb09 and ClueWeb12
  - 35 TB of data
  - 1.2 billion Web pages
- ID types: GTINs, CAS numbers, DOIs, emails
- Implemented as a MapReduce task with Hadoop
  - 10 nodes in the cluster
  - 8 tasks per node
Experimental setup

- English portions of ClueWeb09 and ClueWeb12
  - 35 TB of data
  - 1.2 billion Web pages
- ID types: GTINs, CAS numbers, DOIs, emails
- Implemented as a MapReduce task with Hadoop
  - 10 nodes in the cluster
  - 8 tasks per node
- Only 10 hours processing time
Evaluation

- Take **200 random ids** for each type
- Manually extract the **correct name** (gold standard)
Evaluation

- Take **200 random ids** for each type
- Manually extract the **correct name** (gold standard)
- Measure:
  - **Recall**: which proportion of gold id–name pairs were kept
  - **Accuracy**: among the gold ids that were kept, which proportion has the right name
Evaluation

- Take **200 random ids** for each type
- Manually extract the **correct name** (gold standard)
- Measure:
  - **Recall**: which proportion of gold id–name pairs were kept
  - **Accuracy**: among the gold ids that were kept, which proportion has the right name

→ In phase 1 and 2, we choose **one random name** per id
Overall results

GTIN

Number
Accuracy
Recall

CAS

Number
Accuracy
Recall

DOI

Number
Accuracy
Recall

Phase 1
Phase 2
Phase 3

Emails

Number
Accuracy
Recall

0/25
## Results: richest sources by type and email domains

<table>
<thead>
<tr>
<th>Product sources</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>www2.loot.co.za</td>
<td>304,431</td>
</tr>
<tr>
<td><a href="http://www.books-by-isbn.com">www.books-by-isbn.com</a></td>
<td>50,683</td>
</tr>
<tr>
<td>gtin13.com</td>
<td>26,834</td>
</tr>
<tr>
<td>en.wikipedia.org</td>
<td>21,873</td>
</tr>
<tr>
<td><a href="http://www.buchhandel.de">www.buchhandel.de</a></td>
<td>18,264</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>442,261</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chemical sources</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.chembuyersguide.com">www.chembuyersguide.com</a></td>
<td>129,211</td>
</tr>
<tr>
<td><a href="http://www.chemnet.com">www.chemnet.com</a></td>
<td>22,061</td>
</tr>
<tr>
<td><a href="http://www.lookchem.com">www.lookchem.com</a></td>
<td>12,354</td>
</tr>
<tr>
<td><a href="http://www.seekchemicals.com">www.seekchemicals.com</a></td>
<td>7,326</td>
</tr>
<tr>
<td><a href="http://www.tradingchem.com">www.tradingchem.com</a></td>
<td>4,769</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>171,561</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Document sources</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>wwwtest.soils.org</td>
<td>20,635</td>
</tr>
<tr>
<td><a href="http://www.plosone.org">www.plosone.org</a></td>
<td>19,261</td>
</tr>
<tr>
<td><a href="http://www.citeulike.org">www.citeulike.org</a></td>
<td>13,491</td>
</tr>
<tr>
<td><a href="http://www.astm.org">www.astm.org</a></td>
<td>10,020</td>
</tr>
<tr>
<td>bja.oxfordjournals.org</td>
<td>9,030</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>62,427</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Domain name</th>
<th>Email addresses</th>
</tr>
</thead>
<tbody>
<tr>
<td>gmail.com</td>
<td>304,236</td>
</tr>
<tr>
<td>yahoo.com</td>
<td>290,292</td>
</tr>
<tr>
<td>hotmail.com</td>
<td>281,498</td>
</tr>
<tr>
<td>aol.com</td>
<td>259,769</td>
</tr>
<tr>
<td>comcast.net</td>
<td>95,983</td>
</tr>
</tbody>
</table>
Results: first and last names

- John: 240k
- David: 220k
- Michael: 160k
- Mark: 120k
- Smith: 80k
- Johnson: 40k
- Brown: 40k
- Jones: 40k
Results: full names

John Smith: 2000
David Smith: 1400
John Doe: 1000
Michael Smith: 800
David Brown: 600
Number of products by country, by company

- **US**: 1M
- **UK**: 800k
- **DE**: 750k
- **JP**: 450k
- **FR**: 100k

- **Bernat**: 1.15M
- **Panasonic**: 1.05M
- **Lion**: 1.0M
- **Nikon**: 0.05M

Graph showing the number of products by country and company.
Analyses: world trade

Products *produced* somewhere (GTIN) but *sold* elsewhere (URL).
Table of contents

1. Introduction
2. Extracting candidates
3. Cleaning up candidates
4. Experimental results
5. Conclusion
Summary

- Harvest **IDs** and **names** at Web scale
- **10 hours** to process **35 TB** with **10 nodes**
- **Our catch:**
  - **13M** emails
  - **235k** chemicals
  - **1M** documents
  - **1.4M** books
  - **1.1M** products
Summary

- Harvest IDs and names at Web scale
- 10 hours to process 35 TB with 10 nodes
- Our catch:
  - 13M emails
  - 235k chemicals
  - 1M documents
  - 1.4M books
  - 1.1M products
- Freely available online!
  http://resources.mpi-inf.mpg.de/d5/ibex/
- Accuracy from 73% to 96%
- Many fun measurements: people names, world trade, etc.
Summary

- Harvest **IDs** and **names** at Web scale
- **10 hours** to process **35 TB** with **10 nodes**
- **Our catch:**
  - 13M emails
  - 235k chemicals
  - 1M documents
  - 1.4M books
  - 1.1M products
- Freely available online!
  - http://resources.mpi-inf.mpg.de/d5/ibex/
- Accuracy from **73%** to **96%**
- Many fun **measurements**: people names, world trade, etc.
  - How to generalize this to **attributes**?
  - Find more **uses** for the dataset?
Summary

- Harvest **IDs** and **names** at Web scale
- **10 hours** to process **35 TB** with **10 nodes**
- **Our catch:**
  - 13M emails
  - 235k chemicals
  - 1M documents
  - 1.4M books
  - 1.1M products
- **Freely available online!**
- Accuracy from **73%** to **96%**
- Many fun **measurements**: people names, world trade, etc.
- How to generalize this to **attributes**?
- Find more **uses** for the dataset?

Thanks for your attention!

These slides are inspired from an earlier presentation by Aliaksandr Talaika.