Siphoning Hidden-Web Data through Keyword-Based Interfaces

Luciano Barbosa*

Juliana Freire*!

* OGI/OHSU

OGI SCHOOL OF SCIENCE & ENGINEERING

! University of Utah

SBBD 2004

L. Barbosa, J. Freire
Hidden/Deep/Invisible Web

- Databases and document collections on the Web accessible through form interfaces
  - Data is published as the result of a user request
- Lots of data – much bigger than the visible Web [BrightPlanet, 2001; Lawrence and Gyles, 1998]
- High-quality content
- Several applications leverage (and are enabled by!) hidden data:
  - Web information integration, portals, data mining
Reconstructing Hidden Collections

- Form interfaces can be restrictive, and disallow interesting queries
- Some applications need all data hidden behind a form
  - Hidden-Web search engine
  - Data exploration and mining
  - Tracking changes to a gene database or your competitor’s Web site
- Can we automatically reconstruct a collection/database hidden behind a restrictive form interface?
Hidden Web: Issues

- Designed for human access
  - Fill out a form, get results
- Hard to find: Not accessible from search engines
- Hard to get:
  - Automation is hard – need to deal with form interface mechanics and restrictions
  - Expensive to retrieve data, e.g., a data mining application may require *too many* queries
Accessing Hidden Data: Challenges

- Automatically filling out forms

Text fields: open-ended attributes

Constraints on individual values: days of month

Constraints on set of values:
Month = Sep, day = 31

aircanada.ca
Accessing Hidden Data: State of the Art

- Wrappers can be very effective
  - Programs designed specifically to access data through a given form interface
  - Used in portals – small to medium scale
  - Require significant human input: write the program, specify input values

- Hidden-Web crawlers (HWC)
  - Attempt to automatically fill out any form encountered
  - More scalable than wrappers
  - Problems:
    - Not guaranteed to work – best effort...
    - Automatically filling out unknown forms is very hard, still need substantial human input
    - Can be very inefficient
Reconstructing Hidden Collections

- For reconstruction, need to generate a set of *valid inputs* that *retrieve all items* in the collection
- Wrappers are not a scalable solution
- HWCs focus on
  - Generating *valid inputs* – provide no guarantees all data will be retrieved
  - Structured (multi-attribute) forms – require deep understanding the semantics of forms and data

- **Idea:** Can we use keyword-based interfaces to automatically reconstruct hidden collections?
Keyword-based interfaces

- Intuitively, easier to fill out than structured forms
  - They have no structure
  - Domain = strings
- Widely used
  - Document collections
- ...even for structured data
  [BANKS – Bhalotia et al.,2002]
  - Web sites often support simple and advanced searches
  - Back-door access to structured databases!
Using Keywords to Siphon Data

- Naïve solution
  - Issue query for each word in the dictionary
  - Problem:
    - Large number of unnecessary queries with possibly overlapping results

- Our solution
  - Identify high-frequency words in the database
    - Intuition: High-frequency words result in high coverage
  - **Goal**: high coverage with a small number of queries
Reconstruction through Sampling and Probing

- The algorithm has two steps:
  1. **Sampling**: issue *sampling queries* to find high-frequency keywords – the *candidate keywords*
  2. **Probing**:
     - Combine the candidate keywords into a query, and *probe* the site to determine the query cardinality
     - Greedily select the query with largest cardinality

```
<table>
<thead>
<tr>
<th>Input: form page</th>
<th>Sampling</th>
<th>Probing</th>
<th>Output: high-coverage query</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Candidate keywords</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

SBBD 2004 - Siphoning Hidden-Web Data through Keyword-Based Interfaces

L. Barbosa, J. Freire
Sampling

- Building the keyword sample of the collection
  - Input: the form page
  - Output: candidate (high-frequency) keywords

**Start**

Get the most frequent term from form page

Submit a query with this word

Pick another term

Stop condition?

Error page?

Add new terms from results and increment the frequency of the existent ones

End

SBBD 2004 - Siphoning Hidden-Web Data through Keyword-Based Interfaces
L. Barbosa, J. Freire
Sampling: Issues and Solutions

- Choice of initial keyword submitted
  - Little effect on the final result as long as the query returns some answers [Callan and Connel, 2001]
  - Our approach: select terms in the form page
    - Probably related to the database content – likely to return some results
    - *Easy to obtain*

- Choice of stopping condition
  - Number of iterations: depends on the collection
    - Large/heterogeneous collections need a higher number of iterations
    - Larger number of iterations → higher cost
  - Our approach: try different values – if final coverage is low, iterate some more!
Sampling: Issues and Solutions

- Response page may contain content not related to the collection
  - May negatively impact the quality of the sample
  - E.g., ads and navigation bars
  - Error pages
  - Our solution: detect error pages – issue queries using dummy words [Doorebos et al]
  - Future: investigate smart techniques, such as automatic wrapper generation [Roadrunner, Crescenzi et al]
Sampling: Issues and Solutions

- Stopwords can lead to very high or very low coverage
  - Stopwords have the highest frequencies in collections
  - But they are not always indexed!
  - **Our solution:** detect whether stopwords are indexed – issue queries with stopwords before sampling
Greedily building the high-coverage query
- Input: candidate (high-frequency) keywords
- Output: high-coverage query
Probing: Issues and Solutions

- Determining the number of results
  - Locate and extract the number of results from result page
  - Our approach: use heuristics to locate number of results
  - Some search sites do not make this information available or only provide an approximation

- Stopping condition
  - Max number of requests
    - Avoid overloading the Web server and search engine
  - Max number of words in the query
    - Avoid overloading the search engine, and some interfaces limit this anyway!
  - Values used: fixed in the experiments
    - Maximum number of requests: 15
    - Maximum query size: 10
Experiments

Goals:
- Measure coverage
- Discover “good” parameters for algorithms
- Study the effectiveness of our choices
  - Stopwords
- Understand the limitations of the approach
  - Impact of query results: title only versus description; information-rich vs clutter-rich pages

Issue: collection sizes aren’t always available
## Experiments: Sites Used

<table>
<thead>
<tr>
<th>Site</th>
<th>Size (number of results)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nwfusion.com – Network World Fusion</td>
<td>60,000</td>
<td>News information about information technology</td>
</tr>
<tr>
<td>apsa.org – American Psychoanalytic Association</td>
<td>34,000</td>
<td>Bibliographies of psychoanalytic literature</td>
</tr>
<tr>
<td>cdc.gov – Centers for Disease Control and Prevention</td>
<td>461,194</td>
<td>Health-related documents</td>
</tr>
<tr>
<td>epa.gov – Environment Protection Agency</td>
<td>550,134</td>
<td>Environment-related documents</td>
</tr>
<tr>
<td>georgetown.edu – Georgetown University</td>
<td>61,265</td>
<td>Search interface to the site</td>
</tr>
<tr>
<td>chid.nih.gov – Combined Health Information Database</td>
<td>127,211</td>
<td>Health-related documents</td>
</tr>
<tr>
<td><a href="http://www.ncbi.nlm.nih.gov/pubmed">www.ncbi.nlm.nih.gov/pubmed</a> – NCBI PubMed</td>
<td>14,000,000</td>
<td>Citations for biomedical articles</td>
</tr>
</tbody>
</table>
Experiments: Coverage

- High coverage and quick convergence in sampling

<table>
<thead>
<tr>
<th>Site</th>
<th>5 iterations</th>
<th>10 iterations</th>
<th>15 iterations</th>
<th>Use stopwords</th>
</tr>
</thead>
<tbody>
<tr>
<td>nwfusion.com</td>
<td>94.8</td>
<td>94.4</td>
<td>94.4</td>
<td>true</td>
</tr>
<tr>
<td>apsa.org</td>
<td>86.6</td>
<td>88.5</td>
<td>91.6</td>
<td>true</td>
</tr>
<tr>
<td>cdc.gov</td>
<td>90.4</td>
<td>90.4</td>
<td>90.4</td>
<td>true</td>
</tr>
<tr>
<td>epa.gov</td>
<td>94.2</td>
<td>94.2</td>
<td>94.2</td>
<td>true</td>
</tr>
<tr>
<td>georgetown.edu</td>
<td>98.3</td>
<td>97.9</td>
<td>97.9</td>
<td>true</td>
</tr>
<tr>
<td>chid</td>
<td>35.9</td>
<td>22.8</td>
<td>22.8</td>
<td>true</td>
</tr>
<tr>
<td>gsfc.nasa.gov</td>
<td>99.9</td>
<td>99.9</td>
<td>99.9</td>
<td>false</td>
</tr>
<tr>
<td>pubmed</td>
<td>33.8</td>
<td>34.6</td>
<td>48.9</td>
<td>false</td>
</tr>
</tbody>
</table>

- Pubmed: after 50 iterations, 79.8% coverage
  - Collection is very large and heterogeneous
  - Does not index stopwords
Collection Idiosyncrasies

- The algorithm assumes that all items in the collection are uniformly indexed.
- CHID: different fields are indexed differently.
- The lowest coverage: 35.9%.

1. Preventing Cryptosporidiosis.

Subfile: AIDS Education
Format (FM): 08 - Brochure.
Language(s) (LG): English.
Year Published (YR): 2003.
Audience code (AC): 084 - HIV Positive Persons. 157 - Persons with HIV/AIDS.
Corporate Author (CN): Project Inform, National HIV/AIDS Treatment Hotline.
Physical description (PD): 4 p.: b&w.

Abstract (AB): This information sheet discusses cryptosporidiosis (Crypto), a diarrheal disease caused by a parasite that can live in the intestines of humans and animals. This disease can be very serious, even fatal, in people with weakened immune systems. The information sheet describes, the symptoms, transmission, diagnosis, treatment, and prevention of Crypto, and gives examples of people who might be immuno-compromised or have a weakened immune system, such as people with AIDS or cancer, and transplant patients on immunosuppressive drugs. The information sheet also explains how crypto affects such people.

Verification/Update Date (VE): 200304
Notes (NT): This material is available in the following languages: AD0031721 Spanish.
Accession Number (AN): AD0031720.
Experiments: Effectiveness of stopwords

Document Descriptions

<table>
<thead>
<tr>
<th>Collections</th>
<th>Coverage with stopwords</th>
<th>Coverage no stopwords</th>
</tr>
</thead>
<tbody>
<tr>
<td>nwfusion</td>
<td>94</td>
<td>83</td>
</tr>
<tr>
<td>apsa*</td>
<td>90.4</td>
<td>82</td>
</tr>
<tr>
<td>cdc</td>
<td>94.2</td>
<td>92.5</td>
</tr>
<tr>
<td>epa</td>
<td>99.9</td>
<td>97.9</td>
</tr>
<tr>
<td>nasa*</td>
<td>94.6</td>
<td>84.6</td>
</tr>
<tr>
<td>pubmed*</td>
<td>22.8</td>
<td></td>
</tr>
<tr>
<td>georgetown</td>
<td>58.8</td>
<td></td>
</tr>
<tr>
<td>chid</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Experiments: Effectiveness of stopwords

Stopwords *often* give higher coverage
Regardless of the *presence or absence* descriptions

Using 15 iterations in sampling phase
Experiments: Result Contents

No Stopwords

<table>
<thead>
<tr>
<th>Collections</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>nwfusion</td>
<td>83</td>
</tr>
<tr>
<td>apsa*</td>
<td>35</td>
</tr>
<tr>
<td>cdc</td>
<td>82</td>
</tr>
<tr>
<td>epa</td>
<td>92.5</td>
</tr>
<tr>
<td>nasa*</td>
<td>99.9</td>
</tr>
<tr>
<td>pubmed*</td>
<td>65</td>
</tr>
<tr>
<td>georgetown</td>
<td>84.6</td>
</tr>
<tr>
<td>chid</td>
<td>58.8</td>
</tr>
</tbody>
</table>

Legend:
- Description
- Title
Experiments: Result Contents

Stopwords

The presence of descriptions lead to *slightly* larger coverage
Regardless of the presence or absence of indexing of stopwords
Experiments: Selecting Keywords

- Wrappers lead to marginal increases in coverage
  - Collections have content-rich pages
  - Exception: pubmed - result pages follow a template that contains a large percentage of extraneous information.

Extra info is relevant!
Coverage Queries: Examples

<table>
<thead>
<tr>
<th>Site</th>
<th>With stopword</th>
<th>Without stopword</th>
</tr>
</thead>
<tbody>
<tr>
<td>nwfusion</td>
<td>the,03,and</td>
<td>definition, data, latest, news, featuring</td>
</tr>
<tr>
<td>apsa</td>
<td>of, the, and, in, a, s, j</td>
<td>psychoanal, amer, review, psychoanalytic, int, new, study, family</td>
</tr>
<tr>
<td>cdc</td>
<td>cdc, search, health, of, the, to</td>
<td>health, department, texas, training, public, file, us, services</td>
</tr>
<tr>
<td>epa</td>
<td>epa, search, region, for, to, 8</td>
<td>epa, environmental, site, data</td>
</tr>
<tr>
<td>georgetown</td>
<td>georgetown, the, and, of, to</td>
<td>university, georgetown, description, information</td>
</tr>
<tr>
<td>chid</td>
<td>chid, nih, hiv, for, aids, the, prevention, of, to, health</td>
<td>aids, health, disease, author, number, education, english</td>
</tr>
<tr>
<td>nasa</td>
<td>n/a</td>
<td>nasa</td>
</tr>
<tr>
<td>pubmed</td>
<td>n/a</td>
<td>nlm, nih, cells, cell, effects, expression, virus, after, proteins, human</td>
</tr>
</tbody>
</table>

- Words that are relevant for the site, e.g., “nasa” \(\rightarrow\) 99.9\% coverage!
- Reveal indexing strategies
Related: Web Wrappers

- Automate navigation: WebVCR
- Information dissemination: WebViews
- Data extraction: Roadrunner, LiXto, DEByE
- Data integration: Web Integrator

How:
  - Manual
  - Semi-automatic generation

Benefits
  - Flexibility, efficiency

Drawbacks
  - Site-specific, not scalable
Related Work: Hidden-Web Crawlers

- Crawling: HiWe
  - Match labels to values

- Schema matching: Metaquerier
  - Statistical approach to form matching

- Benefits
  - More scalable than wrappers

- Drawbacks
  - Require substantial human input
  - Ignore forms with few fields
  - Not guaranteed to work...
Conclusions and Future Work

- A simple, yet effective, solution to siphon data hidden behind keyword-based search interfaces
  - Completely automatic
  - High-coverage
- Also works for Web services!
- Protect your data:
  - You may be *unknowingly* providing more access than intended
- Future work
  - Deal with limited number of returned results
  - Automatically derive page-cleaning wrappers
  - Automatically set parameters, e.g., stopping condition, ...
  - Experiment with more sites, esp. structured sites
  - Structured forms (Vinit Kalra)
  - Characterize search interfaces wrt data protection guarantees
Tatu: Sifting through the Hidden Web

- **Goal:** Mine, query, integrate data
- **Provide infrastructure for:**
  - Automatically filling out forms
  - Searching and crawling through hidden data
- **Applications**
  - Hidden-Web search engine
  - Integrating hidden content
  - *Reconstructing hidden databases*