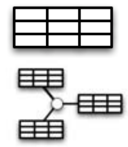
# WebTables & Octopus

Michael J. Cafarella
University of Washington

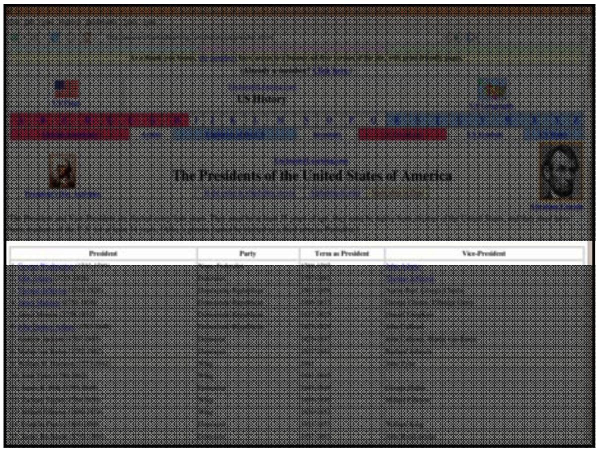
CSE454
April 30, 2009

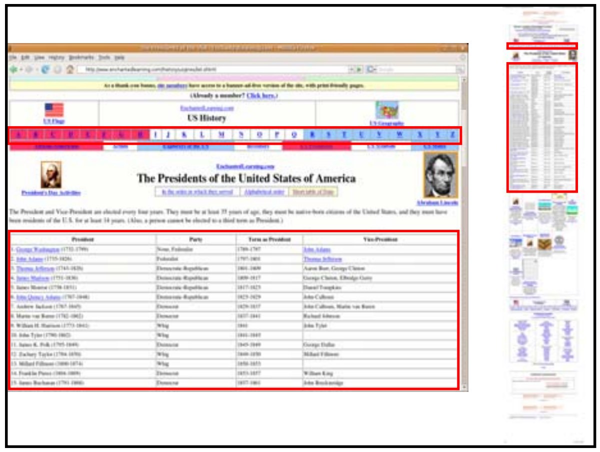
## Outline

- WebTables
- Octopus



2

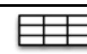




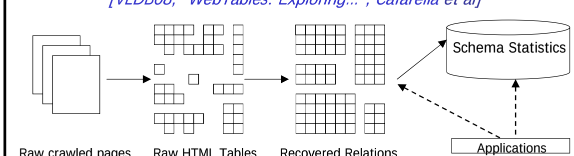

This page contains 16 distinct HTML tables, but only one relational database

Each relational database has its own schema, usually with labeled columns.

## WebTables



- WebTables system automatically extracts dbs from web crawl
  *[WebDB08, “Uncovering…”, Cafarella et al]*
  *[VLDB08, “WebTables: Exploring…”, Cafarella et al]*




- An extracted relation is one table plus labeled columns
- Estimate that our crawl of 14.1B raw HTML tables contains ~154M good relational dbs

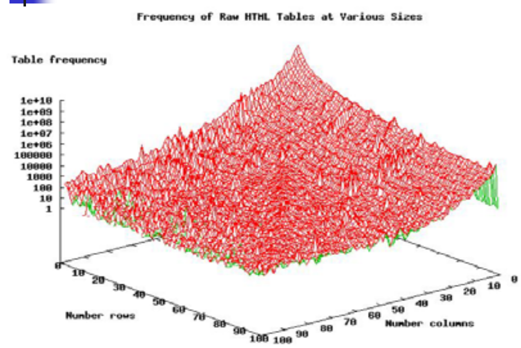
6

## The Table Corpus

| Table type | % total | count |
|---|---|---|
| Small tables | 88.06 | 12.34B |
| HTML forms | 1.34 | 187.37M |
| Calendars | 0.04 | 5.50M |
| Obvious non-rel | 89.44 | 12.53B |
| Other non-rel (est.) | 9.46 | 1.33B |
| Rel (est.) | 1.10 | 154.15M |

7

## The Table Corpus




8

## Relation Recovery

**Step 1. Relational Filtering**
Recall 81%, Precision 41%

**Step 2. Metadata Detection**
Recall 85%, Precision 89%

- Output
  - 271M databases, about 125M are good
  - Five orders of magnitude larger than previous largest corpus *[WWW02, "A Machine Learning...", Wang & Hu]*
  - 2.6M unique relational schemas
- What can we do with those schemas?

*[VLDB08, "WebTables: Exploring...", Cafarella et al]*

9

## Schema Statistics

Recovered Relations

| make | model | year |
|---|---|---|
| Toyota | Camry | 1984 |

| make | model | year |
|---|---|---|
| Mazda | Protégé | 2003 |
| Chevrolet | Impala | 1979 |

| make | model | year | color |
|---|---|---|---|
| Chrysler | Volare | 1974 | yellow |
| Nissan | Sentra | 1994 | red |

| name | addr | city | state | zip |
|---|---|---|---|---|
| Dan S | 16 Park | Seattle | WA | 98195 |
| Alon H | 129 Elm | Belmont | CA | 94011 |

| name | size | last-modified |
|---|---|---|
| Readme.txt | 182 | Apr 26, 2005 |
| cac.xml | 813 | Jul 23, 2008 |

| Schema | Freq |
|---|---|
| {make, model, year} | 2 |
| {make, model, year, color} | 1 |
| {name, addr, city, state, zip} | 1 |
| {name, size, last-modified} | 1 |

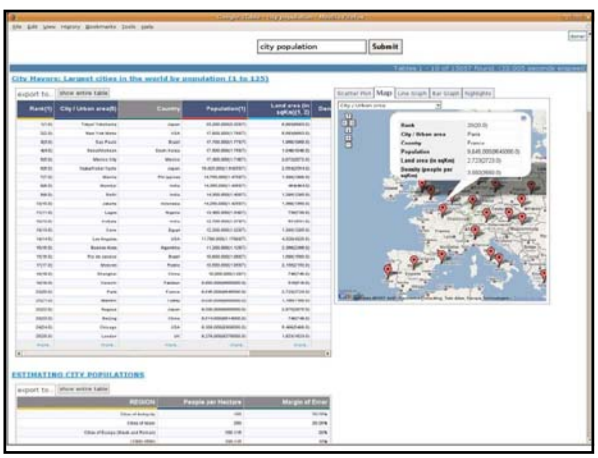
- Schema stats useful for computing attribute probabilities
  - p("make"), p("model"), p("zipcode")
  - p("make" | "model"), p("make" | "zipcode")

10

## App #1: Relation Search

- Problem: keyword search for high-quality extracted databases
- Output depends on both quality of extracted tables and the ranking function

11



## App #1: Relation Search

- Schema statistics can help improve both:
  - Relation Recovery (Metadata detection)
  - Ranking
- By computing a schema coherency score *S(R)* for relation *R*, and adding it to feature vector
- Measures how well a schema “hangs together”
  - High: {make, model}
  - Low: {make, zipcode}
- Average pairwise Pointwise Mutual Information score for all attributes in schema

$$S(R) = \frac{\Sigma_{A,B \in R, A \neq B} log(\frac{p(A,B)}{p(A)p(B)})}{|R|(|R|-1)}$$

13

## App #1: Experiments

- Metadata detection, when adding schema stats scoring
  - Precision 0.79 ⇒ 0.89
  - Recall 0.84 ⇒ 0.85
- Ranking: compared 4 rankers on test set
  - **Naïve**: Top-10 *pages* from google.com
  - **Filter**: Top-10 *good tables* from google.com
  - **Rank**: Trained ranker
  - **Rank-Stats**: Trained ranker with coherency score
- What fraction of top-k are relevant?

| k | Naïve | Filter | Rank | Rank-Stats |
|---|---|---|---|---|
| **10** | 0.26 | 0.35 (34%) | 0.43 (65%) | 0.47 (80%) |
| **20** | 0.33 | 0.47 (42%) | 0.56 (70%) | 0.59 (79%) |
| **30** | 0.34 | 0.59 (74%) | 0.66 (94%) | 0.68 (100%) |

14

## App #2: Schema Autocomplete

- Input: topic attribute (*e.g.*, make)
- Output: relevant schema {make, model, year, price}
  - “tab-complete” for your database

- For input set *I*, output *S*, threshold *t*
  - while p(*S-I* | *I*) > *t*
    - *newAttr* = max p(*newAttr*, *S-I* | *I*)
    - *S* = *S* ∪ *newAttr*
    - emit *newAttr*

15

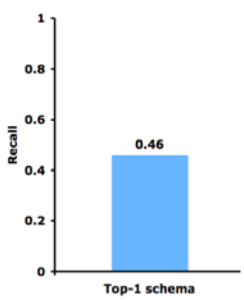
## App #2: Schema Autocomplete

| | |
|---|---|
| name | name, size, last-modified, type |
| instructor | instructor, time, title, days, room, course |
| elected | elected, party, district, incumbent, status, … |
| ab | ab, h, r, bb, so, rbi, avg, lob, hr, pos, batters |
| sqft | sqft, price, baths, beds, year, type, lot-sqft, … |

16

## App #2: Experiments

- Asked experts for schemas in 10 areas
- What was autocompleter’s recall?




17

## App #3: Synonym Discovery

- Input: topic attribute (*e.g.*, address)
- Output: relevant synonym pairs (telephone = tel-#)
  - Used for schema matching *[VLDB01, “Generic Schema Matching…”, Madhavan et al]*
  - Linguistic thesauri are incomplete; hand-made thesauri are burdensome

- For attributes *a*, *b* and input domain *C*, when *p(a,b)=0*

$$syn(a,b) = \frac{p(a)p(b)}{\epsilon + \Sigma_{z \in A}(p(z|a,C) - p(z|b,C))^2}$$

18

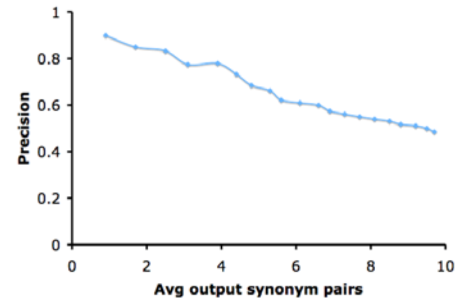
## App #3: Synonym Discovery

| | |
|---|---|
| name | e-mail\|email, phone\|telephone, e-mail_address\|email_address, date\|last_modified |
| instructor | course-title\|title, day\|days, course\|course-#, course-name\|course-title |
| elected | candidate\|name, presiding-officer\|speaker |
| ab | k\|so, h\|hits, avg\|ba, name\|player |
| sqft | bath\|baths, list\|list-price, bed\|beds, price\|rent |

19

## App #3: Experiments

- For each input attr, repeatedly emit best synonym pair (until min threshold reached)




20

## WebTables Contributions

- Largest collection of databases and schemas, by far
- Large-scale extracted schema data for first time; enables novel applications

21

## Outline

- WebTables
- Octopus

22

## Multiple Tables

- Can we combine tables to create new data sources?
- Data integration for the Structured Web
- Many existing "mashup" tools, which ignore realities of Web data
  - A lot of useful data is not in XML
  - User cannot know all sources in advance
  - Transient integrations

23

## Integration Challenge

- Try to create a database of all "VLDB program committee members"


24

## Octopus

- Provides "workbench" of data integration operators to build target database
  - Most operators are not correct/incorrect, but high/low quality (like search)
  - Also, prosaic traditional operators

25

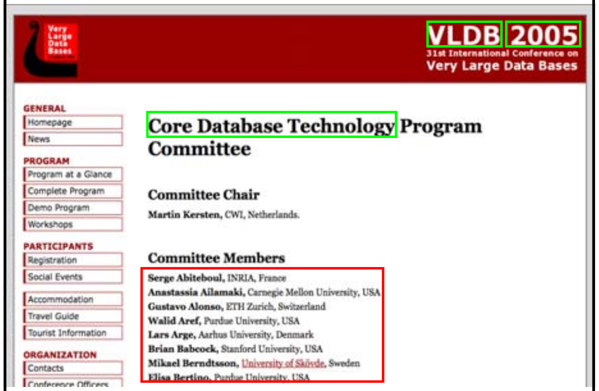
## Walkthrough - Operator #1

- **SEARCH**("VLDB program committee members")


| | |
|---|---|
| serge abiteboul | inria |
| michael adiba | …grenoble |
| antonio albano | …pisa |
| … | … |


| | |
|---|---|
| serge abiteboul | inria |
| anastassia ail… | carnegie… |
| gustavo alonso | etz zurich |
| … | … |

26




27

## Walkthrough - Operator #2

- Recover relevant data

**CONTEXT()**

| | | |
|---|---|---|
| serge abiteboul | inria | 1996 |
| michael adiba | …grenoble | 1996 |
| antonio albano | …pisa | 1996 |
| … | … | … |

**CONTEXT()**

| | | |
|---|---|---|
| serge abiteboul | inria | 2005 |
| anastassia ail… | carnegie… | 2005 |
| gustavo alonso | etz zurich | 2005 |
| … | … | … |

28

## Walkthrough - Union

- Combine datasets

Union()

| | | |
|---|---|---|
| serge abiteboul | inria | 1996 |
| michael adiba | …grenoble | 1996 |
| antonio albano | …pisa | 1996 |
| serge abiteboul | inria | 2005 |
| anastassia ail… | carnegie… | 2005 |
| gustavo alonso | etz zurich | 2005 |
| … | … | … |

29

## Walkthrough - Operator #3

- Add column to data
- Similar to "join" but join target is a topic

**EXTEND**( "publications", col=0) "publications"

| | | | | |
|---|---|---|---|---|
| serge abiteboul | inria | 1996 | "Large Scale P2P Dist…" | 1996 |
| michael adiba | …grenoble | 1996 | "Exploiting bitemporal…" | 1996 |
| antonio albano | …pisa | 1996 | "Another Example of a…" | 1996 |
| serge abiteboul | inria | 2005 | "Large Scale P2P Dist…" | 2005 |
| anastassia ail… | carnegie… | 2005 | "Efficient Use of the…" | 2005 |
| gustavo alonso | etz zurich | 2005 | "A Dynamic and Flexible…" | 2005 |
| … | … | … | … | … |

- User has integrated data sources with little effort
- No wrappers; data was never intended for reuse

30

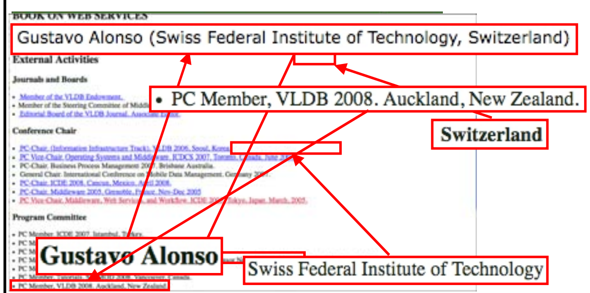
## CONTEXT Algorithms

- Input: table and source page
- Output: data values to add to table
- *SignificantTerms* sorts terms in source page by “importance” (tf-idf)



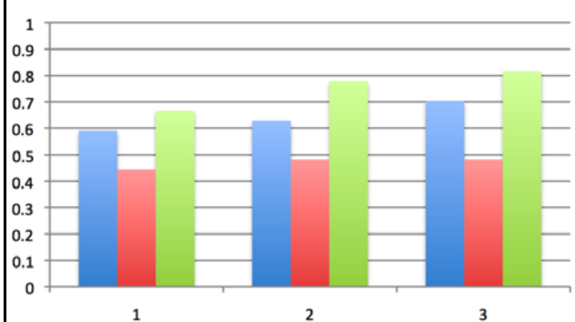
## Related View Partners

- Looks for different “views” of same data






## CONTEXT Experiments






## EXTEND Algorithms

- Recall: **EXTEND**( “publications”, col=0)
- JoinTest looks for single “joinable” table
- E.g., extend a column of US cities with “mayor” data
- Algorithm:
  1. SEARCH for a table that is relevant to topic (e.g., “mayors”); rank by relevance
  2. Retain results with a joinable column (to col=0). Use Jaccardian distance fn between columns



## EXTEND Algorithms

- MultiJoin finds compatible “joinable tuples”; tuples can come from many different tables
- E.g., extend PC members with “publications”
- Algorithm:
  1. SEARCH for each source tuple (using cell text and “publications”)
  2. Cluster results, rank by weighted mix of topic-relevance and source-table coverage
  3. Choose best cluster, then apply join-equality test as in JoinTest
- Algorithms reflect different data ideas: found in one table or scattered over many?



## EXTEND Experiments

- Many test queries not EXTENDable
- We chose column and query to EXTEND

| Join column desc. | Topic query |
|---|---|
| Countries | Universities |
| Us states | Governors |
| Us cities | Mayors |
| Film titles | Characters |
| UK political parties | Member of parliament |
| Baseball teams | Players |
| Musical bands | albums |

- TestJoin: 60% of src tuples for three topics; avg 1 correct extension per src tuple
- MultiJoin: 33% of src tuples for *all topics; avg 45.5 correct extensions per src tuple*

## CollocSplit Algorithm

| |
|---|
| Mike Cafarella Univ of Washington |
| Alon Halevy Google, Inc. |
| Oren Etzioni Univ of Washington |
| H.V. Jagadish Univ of Michigan |

→

| | |
|---|---|
| Mike Cafarella | Univ of Washington |
| Alon Halevy | Google, Inc. |
| Oren Etzioni | Univ of Washington |
| H.V. Jagadish | Univ of Michigan |

1. For i = 1..MAX, find breakpoints, ranked by co-location score

| |
|---|
| Mike Cafarella Univ of Washington |
| Alon Halevy Google, Inc. |
| Oren Etzioni Univ of Washington |
| H.V. Jagadish Univ of Michigan |

37

## CollocSplit Algorithm

| |
|---|
| Mike Cafarella Univ of Washington |
| Alon Halevy Google, Inc. |
| Oren Etzioni Univ of Washington |
| H.V. Jagadish Univ of Michigan |

→

| | |
|---|---|
| Mike Cafarella | Univ of Washington |
| Alon Halevy | Google, Inc. |
| Oren Etzioni | Univ of Washington |
| H.V. Jagadish | Univ of Michigan |

1. For i = 1..MAX, find breakpoints, ranked by co-location score

i=1

<table>
<tr><td>Mike Cafarella || Univ of Washington</td></tr>
<tr><td>Alon Halevy || Google, Inc.</td></tr>
<tr><td>Oren Etzioni || Univ of Washington</td></tr>
<tr><td>H.V. Jagadish || Univ of Michigan</td></tr>
</table>

38

## CollocSplit Algorithm

| |
|---|
| Mike Cafarella Univ of Washington |
| Alon Halevy Google, Inc. |
| Oren Etzioni Univ of Washington |
| H.V. Jagadish Univ of Michigan |

→

| | |
|---|---|
| Mike Cafarella | Univ of Washington |
| Alon Halevy | Google, Inc. |
| Oren Etzioni | Univ of Washington |
| H.V. Jagadish | Univ of Michigan |

1. For i = 1..MAX, find breakpoints, ranked by co-location score

i=2

<table>
<tr><td>Mike Cafarella || Univ of || Washington</td></tr>
<tr><td>Alon || Halevy || Google, Inc.</td></tr>
<tr><td>Oren || Etzioni || Univ of Washington</td></tr>
<tr><td>H.V. || Jagadish || Univ of Michigan</td></tr>
</table>

39

## CollocSplit Algorithm

| |
|---|
| Mike Cafarella Univ of Washington |
| Alon Halevy Google, Inc. |
| Oren Etzioni Univ of Washington |
| H.V. Jagadish Univ of Michigan |

→

| | |
|---|---|
| Mike Cafarella | Univ of Washington |
| Alon Halevy | Google, Inc. |
| Oren Etzioni | Univ of Washington |
| H.V. Jagadish | Univ of Michigan |

1. For i = 1..MAX, find breakpoints, ranked by co-location score

i=3

<table>
<tr><td>Mike || Cafarella || Univ of || Washington</td></tr>
<tr><td>Alon || Halevy || Google, || Inc.</td></tr>
<tr><td>Oren || Etzioni || Univ of || Washington</td></tr>
<tr><td>H.V. || Jagadish || Univ of || Michigan</td></tr>
</table>

40

## CollocSplit Algorithm

| |
|---|
| Mike Cafarella Univ of Washington |
| Alon Halevy Google, Inc. |
| Oren Etzioni Univ of Washington |
| H.V. Jagadish Univ of Michigan |

→

| | |
|---|---|
| Mike Cafarella | Univ of Washington |
| Alon Halevy | Google, Inc. |
| Oren Etzioni | Univ of Washington |
| H.V. Jagadish | Univ of Michigan |

2.Choose *i* that yields *most-consistent* columns

<table>
<tr><td>Mike Cafarella || Univ of Washington</td></tr>
<tr><td>Alon Halevy || Google, Inc.</td></tr>
<tr><td>Oren Etzioni || Univ of Washington</td></tr>
<tr><td>H.V. Jagadish || Univ of Michigan</td></tr>
</table>

Our current consistency measure is avg std-dev of cell strlens

41

## Octopus Contributions

- Basic operators that enable Web data integration with very small user burden
- Realistic and useful implementations for all three operators

42

## Future Work

- WebTables
  - Schema autocomplete & synonyms just few of many possible *semantic services*
    - Input: schema; Output: tuples
      *database autopopulate*
    - Input: tuples; Output: schema
      *schema autogenerate*
- Octopus
  - Index support for interactive speeds



## Future Work (2)

- "The Database of Everything"
  *[CIDR09, "Extracting and Querying...", Cafarella]*
  - Is domain-independence enough?

| Text-embedded |
|---|
| be/is |
| ask/call |
| arrive/come/go |
| join/lead |
| born-in |

| Table-embedded |
|---|
| <web access log> |
| <file listing> |
| <forum posts> |
| <album listing> |
| <phone numbers> |

  - Multi-model, multi-extractor approach probable
  - Vast deduplication challenge
  - New tools needed for user/extractor interaction