Web Data Management - Some Issues
Properties of Web Data

- Lack of a schema
  - Data is at best “semi-structured”
  - Missing data, additional attributes, “similar” data but not identical

- Volatility
  - Changes frequently
  - May conform to one schema now, but not later

- Scale
  - Does it make sense to talk about a schema for Web?
  - How do you capture “everything”?

- Querying difficulty
  - What is the user language?
  - What are the primitives?
  - Aren’t search engines or metasearch engines sufficient?
Outline

- Distribution Models
- Modeling Issues
- Web Data Integration
- Web Querying
- Web Caching
Data Delivery on the Internet

- Properties of information supply
  - It is very large in volume
  - It is highly heterogeneous
  - May not have a properly defined schema
  - Data available from too many devices and in streaming fashion
    - Data stream systems

- Properties of information consumption
  - It is data intensive
    - Use of large data sets is common
  - It requires access to diverse data sources
    - Existing databases and/or repositories must somehow be “glued” together
    - Application integration
Data Delivery Alternatives

Delivery Mode
- Hybrid
- Push-only
- Pull-only

Frequency
- Periodic
- Conditional
- Irregular

Communication Method
- Unicast
- One-to-many
- One-to-many
Web Data Modeling

- Can’t depend on a strict schema to structure the data
- Data are self-descriptive
  
  ```json
  {name: {first: “Tamer”, last: “Ozsu”}, institution: “University of Waterloo”, salary: 300000}
  ```

- Usually represented as an edge-labeled graph
  
  XML can also be modeled this way

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```

```
```
Web Data Integration

- What is being integrated?
- Integration or interoperation?
- More flexible architectures
  - Barriers to joining and leaving federations should be minimum
  - Participants should be able to maintain their own environments as much as possible
  - Application as well as data integration
- Role of XML
  - “Data model”
  - Exchange format
- Flexible operation
  - Ability to deal with data inconsistencies as well as schema inconsistencies
  - Systems should be able to deal with failures and incomplete federations
Approaches to Web Querying

- Search engines and metasearchers
  - Keyword-based
  - Category-based
- Information integration
- Semistructured data querying
- Special Web query languages
- Learning-based systems
- Question-Answering
Information Integration

- Basic principle: Integrate part of the Web data into a database as either virtual or materialized views and query over these views

- Example systems:
  - Information Manifold [Levy et al., 1996]
  - Araneus [Atzeni et al., 1997]
  - WSQ/DSQ [Goldman & Widom, 2000]
Evaluation

■ Advantages
  ➞ Well-understood
  ➞ Well-known database techniques can be brought to bear

■ Disadvantages
  ➞ Not querying the *entire* Web; more querying some data on the Web
  ➞ Does not scale well; integration methodology should be low overhead
Semistructured Data Querying

- Basic principle: Consider Web as a collection of semistructured data and use those techniques
- Uses an edge-labeled graph model of data
- Example systems & languages:
  - Lore/Lorel [Abiteboul et al., 1997]
  - UnQL [Buneman et al., 1996]
  - StruQL [Fernandez et al., 1997]
Lorel Example

Select zip codes of all cheap restaurants
Select Guide.restaurant(.address)?.zipcode
Where Guide.restaurant.% grep "cheap"
**Evaluation**

- **Advantages**
  - Simple and flexible
  - Fits the natural link structure of Web pages

- **Disadvantages**
  - Data model too simple (no record construct or ordered lists)
  - Graph can become very complicated
    - Aggregation and typing combined
    - DataGuides
  - No differentiation between connection between documents and subpart relationships
Web Query Languages

- Basic principle: Take into account the documents’ content and internal structure as well as external links
- The graph structures are more complex
- Examples
  - WebSQL [Mendelzon et al., 1996]
  - W3QS [Kanopnicki & Shmueli, 1995]
  - WebLog [Lakshmanan et al., 1993]
  - WebOQL [Arocena & Mendelzon, 1999]
  - StruQL [Fernandez et al., 1997]

First Generation

Second Generation
WebOQL Example

Find, in the csPapers database, all the papers authored by "Smith" and extract their title and URL of the full version of the papers.

\[
\text{select } [y.Title, y'.Url] \\
\text{from } x \text{ in csPapers, } y \text{ in } x' \\
\text{where } y.Authors \sim \text{"Smith"}
\]
Evaluation

Advantages

- More powerful data model - Hypertree
  - Ordered edge-labeled tree
  - Internal and external arcs
- Language can exploit different arc types (structure of the Web pages can be accessed)
- Languages can construct new complex structures.

Disadvantages

- You still need to know the graph structure
- Complexity issue
Learning-Based Approaches

- Basic principle: Learn what the user’s intent is from the query and find the data
- Some based on NLP, others metasearch systems; agent technology and mining-based
- Examples:
  - InfoSpider [Menczer & Below, 1998]
  - WebWatcher [Joachims & Freitag, 1997]
  - Fab [Balabanovic, 1997]
  - Syskill & Webert [Pazzani et al., 1996]
  - WebSifter II [Kerschberg et al., 2001]
Question-Answer Approach

- Basic principle: Web pages that could contain the answer to the user query are retrieved and the answer extracted from them.
- NLP and information extraction techniques
- Used within IR in a closed corpus; extensions to Web
- Examples
  - QASM [Radev et al., 2001]
  - Ask Jeeves
  - Mulder [Kwok et al, 2001]
  - WebQA [Lam & Özsu, 2002]
WebQA Objectives

- Query the *entire* Web
- Return actual answers, not URLs
- Scale with additional data sources
- Accept fuzziness (precision/recall)
- Do not depend on existence of a schema
Interaction with Web Server
Query Parser

- Query can be NL or WebQAL (internal language)
  - `<Category>` [-output <output option>] -keywords <keyword list>
- Elimination of stopwords
- Categorization
  - Name
  - Place
  - Time
  - Quantity
  - Abbreviation
  - Weather
  - Other
- Very light weight NLP
  - Rule-based
  - Only categorization
  - 99% accurate on TREC-9 questions
Summary Retriever

Keyword generator: a set of keywords for sources

Source ranker: identify more promising sources for this query

Record consolidator/ranker: submits keywords and consolidates results

Record retriever: retrieves records from wrappers

Wrapper: uniform API

Keyword list

Source Name Snippet Local Rank

Google 1

Submit
Next
Answer Extraction

- Candidate identification
  - Rule-based based on the category
  - Candidates are scored based on frequency of occurrence in records

- Rearranger takes care of anomalies
  - Score(Bell) > Score(Alexander Graham Bell)

List of ranked records

Candidate Retriever

Rearranger

Output Converter

Top ten answers (user readable)
Evaluation

- Using TREC-9
  - List of 693 questions and a list of documents
  - Answers should be 50-byte or 250-byte passage, not exact answers
  - Ranked score between 0 (worst) and 1 (best)
    - Score = $1/n$ where $n$ is the rank of the correct answer

- Two measures
  - Accuracy
  - Efficiency
Experiment 1

- Run TREC-9 queries directly against the search engines

![Graph showing performance of search engines on different query categories]
Experiment 2

- Run TREC-9 queries through WebQA; use CIA Fact Book 2001 as secondary
Experiment 3

- Run TREC-9 queries using best combination
WebQA Performance

- Local Execution Time
- Remote Execution Time

<table>
<thead>
<tr>
<th>Name</th>
<th>Place</th>
<th>Time</th>
<th>Quantity</th>
<th>Abbreviation</th>
<th>Other</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(Bar chart showing execution times for different categories.)
Web Caching

- Storing objects at places through which a user's request passes.
  - browser cache, proxy cache, server cache

- Merits of Web caching:
  - reduces network traffic
  - reduces client latency
  - reduces server load

- Caching architectures
  - With respect to location
    - Hierarchical, distributed

- Cache consistency issues
  - Weak cache consistency algorithms
  - Strong cache consistency algorithms
# Classification

<table>
<thead>
<tr>
<th></th>
<th>Client-Validation</th>
<th>Server Invalidation</th>
<th>C/S Interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong</td>
<td>Polling-every-time</td>
<td>Invalidation</td>
<td>Lease</td>
</tr>
<tr>
<td>Weak</td>
<td>TTL,PCV</td>
<td>PSI</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Why Strong Consistency? Motivating Examples

- Online Shopping Store
- Travel Tickets Reservation
- Stock Quotes
- Online Auction