Toward Large Scale Integration: Building a MetaQuerier over Databases on the Web

K. C.-C. Chang, B. He, and Z. Zhang
Presented by:
M. Hossein Sheikh Attar

1

Background

- Deep (hidden) Web
  - Searchable online databases
  - 450,000 databases on the Internet
  - Growing fast
  - Invisible to users and current crawlers
    - Accessible through query interfaces

2

Problem Statement

- Currently users have difficulties in
  - Finding the right sources
    - E.g., What is a good source for finding apartments in Waterloo?
  - Querying them
    - Each source supports different query capabilities
- The goal of MetaQuerier
  - Make deep Web systematically accessible
  - Make it uniformly usable

3

Challenges

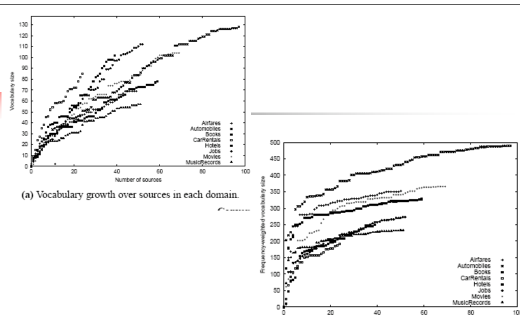
- Somewhat similar to the traditional information integration problem
- However
  - The scale is much larger
  - Dynamic discovery
    - *No pre-selected* sources
  - On-the-fly semantic discovery
    - Ad-hoc queries
    - No *pre-configured per-source* knowledge

4

Summary of Observations

- Survey and observe (do some "reality checks")
  - Helps make right assumptions
- Online databases are NOT arbitrarily complex
  - Convergence
  - Regularity
- Reason
  - Influence by peers
    - Amazon effect

5



(a) Vocabulary growth over sources in each domain.

(b) Frequency-weighted vocabulary growth.

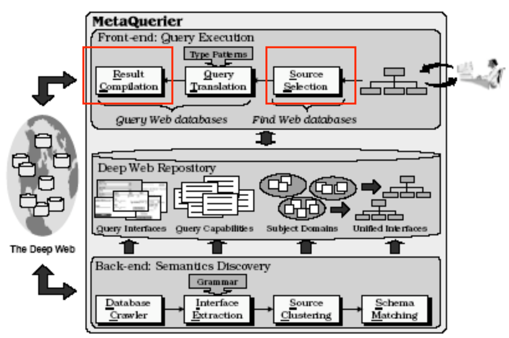
Convergence of schema vocabularies.

6

Architecture of MetaQuerier

- 7 Subsystems
- Plan
  - Study and implement each subsystem individually
  - Integrate them
- 5 subsystems implemented so far

7



8

Subsystem 1: Database Crawler

9

Database Crawler (DC)

- Focused crawler for finding query interfaces
- Survey shows that the query interfaces are close to root page of Web sites

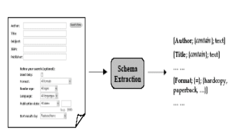
10

Subsystem 2: Interface Extraction

11

Interface Extraction (IE)

- Input: HTML query interface
- Output: query capabilities
  - constraint templates:
  - [attribute, *operator*, value]
  - E.g., [title, *contains*, $v]

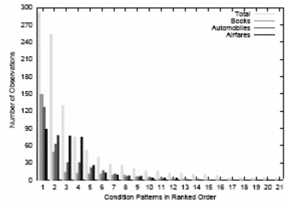


Subsystem *SE*: Schema extraction.

12

IE – Hidden Syntax Hypothesis

- Different interfaces share similar patterns
- Regularities
  - Presentation conventions



IE – Hidden Syntax Hypothesis

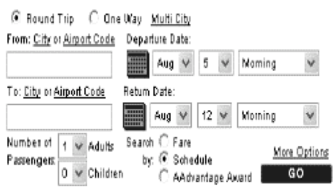
- Hypothetic hidden syntax across sources
  - Using this hidden syntax, we can interpret an interface unseen before
  - Principles algorithmic framework
    - Using a grammar for *pattern specification*
    - Using a parser for *pattern recognition*

Grammar

#	*Production Rules*	*Visual Patterns*
P1	QI← HQI | Above(QI, HQI)	
P2	HQI← CP | Left(HQI, CP)	
P3	CP← TextVal | TextOp | EnumRB	
P4	TextVal← Left(Attr,Val) | Above(Attr,Val) | Below (Attr, Val)	Attr Val
P5	TextOp← Left(Attr, Val) ∧ Below (Op, Val)	Attr Val
P6	Op← RBList	Op
P7	EnumRB← RBList	RBList
P8	RBList ← RBU | Left(RBList RBU)	RBList RBU
P9	RBU← Left(radiobutton, text)	radiobutton text
P10	Attr←text	
P11	Val←textbox	

Productions of the 2P grammar.

Ambiguities - 2P Grammar



16

Ambiguities - 2P Grammar

- *2P grammar*
  - A set of ***productions*** to capture conventional hidden patterns
  - A set of ***preferences*** to capture hidden priority conventions
- Best effort parser
  - multiple parse trees
  - incomplete parse trees
  - Merging trees at the end

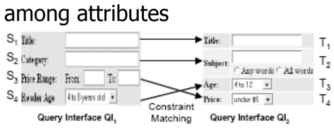
17

Subsystem 3: Schema Matching

18

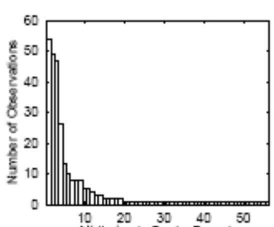
Schema Matching

- Input:
  - Query capabilities from several extracted forms in a domain
- Output:
  - semantic correspondence (matching) among attributes



Schema Matching

- Existing method do not scale well to our problem
- Large scale is both a challenge and an opportunity
  - Holistic schema matching
  - Explore context information across all schemas
  - Assumes the existence of a hidden generative schema model



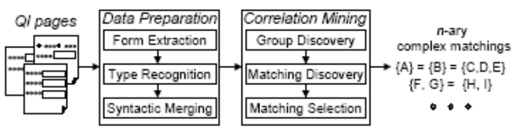
Attribute frequencies in Books domain.

Schema Matching

- Abstract the problem as *correlation mining*
  - Mining for *positive and negative* correlations
- Examples:
  - {first name, last name}, author

22

Schema Matching – DCM framework



From matching to mining: the DCM framework.

domain	*the MGS framework*	*the DCM framework*
Books	{author} = {last name} (P) {author} = {first name} (P) {subject} = {category} (Y)	{author} = {last name, first name} (Y) {publisher} = {last name} (N) {subject} = {category} (Y)
Movies	{artist} = {actor} = {star} (Y) {genre} = {category} (Y)	{artist} = {actor} (P) {genre} = {category} (Y) {rating} = {keyword} (N) {price} = {format} (N)
MusicRecords	{title} = {album} (Y) {artist} = {band} (Y) {genre} = {soundtrack} (N) {keyword} = {catalog} (N)	{title} = {album} (Y) {artist} = {band} (Y) {genre} = {label} (N)
Automobiles	{style} = {type} = {category} (Y) {state} = {mileage} (N) {zip code} = {color} (N)	{style} = {type} = {category} (Y) {state} = {mileage} (N)

Integrating Subsystems

25

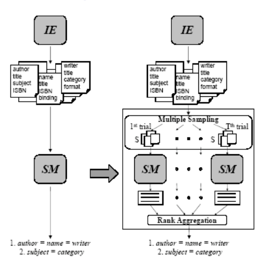
System Integration

- Challenges
  - Accuracy problems
    - IE delivers 85-90% accuracy
    - Not accurate enough for SM
- Opportunities
  - feedback

26

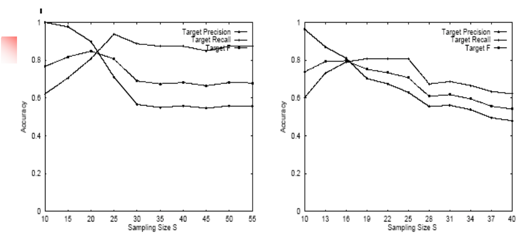
Ensemble Framework

- Ensemble Framework
  - Accuracy problems mainly because of noisy input
  - ***Sampling*** and ***voting*** techniques



(a) The base framework (b) The ensemble framework

27



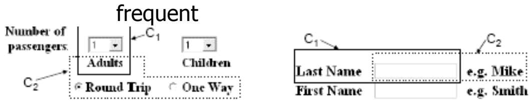
(a) Books (T=41). (b) Airfares (T=41).

The target accuracy under various sampling sizes.

28

Feedback

- Feedback: Domain Statistics
- Example
  - Conflict between
    - [last name; contain; $val]
    - [e.g. Mike; contain; $val]
  - SM notices that the first one is much more frequent



Feedback

- IE processes one interface at a time
- SM has *holistic* domain statistics
- Feedback from SM can help IE resolve conflicts
- Another example that large scale is both curse and blessing

30

Feedback

- 3 types of domain statistics
  - Type of attributes
  - Frequency of attributes
  - Correlation of attributes

31

Summary

- Large scale integration involves challenges and opportunities
- Integrating subsystems also involves challenges and opportunities
- Holistic Integration insights
  - Hidden regularity
  - Peer majority

32