Database Principles in Text Analytics

Benny Kimelfeld
Technion Data & Knowledge Lab
Faculty of Computer Science, Technion, Israel
What did I do at Almaden?

For one, permanent committee member of the annual CS picnic

Thanks Almaden for providing pictures!
Outline

• Enterprise Search
• Information Extraction
• Prioritized Repairing
# Concepts of Search over Structured Data

<table>
<thead>
<tr>
<th>Query</th>
<th>Data</th>
<th>Answer</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keywords</td>
<td>(Structured) DB</td>
<td>Connected set of tuples/items</td>
<td><em>Data = graph; answer = subtree; kws = leaves</em></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="Structured DB" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DB + schema</td>
<td>DB query</td>
<td><em>Answer = CQ that connects the keywords</em></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="DB + Schema" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tree data (XML)</td>
<td>Tree Node</td>
<td><em>Each subtree treated as a separate document</em></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="Tree Data" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Docs + aux. DB</td>
<td>Document</td>
<td><em>DB indexes entities and relationships inside the documents</em></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="Document + Aux DB" /></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Concepts of Search over Structured Data

<table>
<thead>
<tr>
<th>Query</th>
<th>Data</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keywords (sequence of terms, no restrictions)</td>
<td>(Structured) DB</td>
<td>Connected set of tuples/items</td>
</tr>
<tr>
<td></td>
<td>DB + schema</td>
<td>DB query</td>
</tr>
<tr>
<td></td>
<td>Tree data (XML)</td>
<td>Tree Node</td>
</tr>
<tr>
<td>Docs + aux. DB</td>
<td>Document</td>
<td></td>
</tr>
</tbody>
</table>

Explored in my PhD w/ Shuky Sagiv

Work w/ Ron @Almaden
Enterprise Search Projects @Almaden

- OmniFind
  - Personal email search

- Gumshoe
  - Enterprise (internal Web) search
Example: Email Search

**from sara john number**

**Search**

**Interpretation:**
Find emails that contain the words “from” “sara” “john” and “number”

![Sad emoji]

**Interpretation:**
Find emails from Sara, where some phone# and “john” is included

![Sad emoji]

**Interpretation:**
Find emails from Sara, s.t. the phone# of the person “John” is included

![Smiling emoji]
A schema is a partially ordered set of concepts + subtyping

employee \subseteq person
Database: Instances of Concepts

A database is a set of records
(atomic & compound records)

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Data</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>r3087</td>
<td>company</td>
<td>IBM Corp.</td>
<td>doc15</td>
</tr>
<tr>
<td>r1034</td>
<td>person</td>
<td>John</td>
<td>doc23</td>
</tr>
<tr>
<td>r309</td>
<td>phone</td>
<td>(713) 853-4145</td>
<td>doc23</td>
</tr>
<tr>
<td>r2437</td>
<td>prph</td>
<td>r1034, r309</td>
<td>doc23</td>
</tr>
</tbody>
</table>
From Search Queries to DB Queries

**from sara john number**

Each of the four keywords should occur in the document

- **sender**
- **phone**
- **sara**
- **john**

The sender is “sara,” “john” is included, some phone# is included

**prph**

- **sender**
- **person**
- **phone**
- **sara**
- **john**

The sender is “sara,” person “john” is included along w/ his phone#
Rewrite Rules

Rewrite Rules

Interpretations

laura haas email

person

laura haas email

prEmail

laura haas

prEmail

domain

title

bluepages

laura haas

bluepages

laura haas
Research

• **Framework** [Fagin, K, Li, Raghavan, Vaithyanathan, PODS10]
  – “Search database systems”
  – Specificity (or containment) of interpretations
  – How to produce (top-specific, nonempty) interpretations?

• **Convergence** [Fagin, K, Li, Raghavan, Vaithyanathan, PODS11]
  – *How to apply rewrite rules to the search query?*
  – Simple way: each rule applied once, predefined order
  – Thorough way: least fixpoint (apply repeatedly)
    • Problem: “bad” rule sets lead to **non-termination**
      – Real problem: termination is **undecidable**
    • Robust & tractable **safety** guarantees termination
Sources of Auxiliary Data

- **Information extraction**
  - Signature, person, phone, person-phone, ...

- **Domain knowledge**
  - Email search: email headers (metadata), user’s address book, etc.
  - Enterprise search: business data, HR data, etc.
  - Online store search: product database, etc.

- **Global knowledge**
  - WordNet, DBPedia, YAGO, GeoNames, …
Outline

• Enterprise Search

• Information Extraction

• Prioritized Repairing
Information Extraction (IE)

data-in-text $\rightarrow$ data-in-db
(unstructured) $\rightarrow$ (structured)

"Information Extraction (IE) is the name given to any process which selectively structures and combines data which is found, explicitly stated or implied, in one or more texts. The final output of the extraction process varies; in every case, however, it can be transformed so as to populate some type of database."

J. Cowie and Y. Wilks., Handbook of Natural Language Processing, 2000
IE with IBM’s SystemT

```
create view Caps as
extract regex /[A-Z](\w-\w)+/ on D.text as name from Document D;

create view Last as
extract dictionary LastGaz on D.text as name from Document D;

create view CapsLast as
select CombineSpans(C.name, L.name) as name
from Caps C, Last L
where FollowsTok(C.name, L.name, 0, 0);

... regex + join w/ previous views

create view PersonAll as
(select R.name from FirstLast R) union all ...
... union all (select R.name from CapsLast R);

create view Person as select * from PersonAll R
consolidate on R.name using 'ContainedWithin';

output view Person;
```

[Chiticariu, Krishnamurthy, Li, Raghavan, Reiss, Vaithyanathan, ACL 2010]
**Document Spanners**

[Fagin, K, Reiss, Vansummeren, JACM15]

**Document Spanner:** a function that maps every doc (string) into a relation over the doc’s spans

More formally:

- Finite alphabet $S$ of symbols
- A spanner maps each doc. $d \in S^*$ into a relation over the spans $[i,j)$ of $d$
- The relation has a fixed signature (set of attributes)
  - The attributes come from an infinite domain of variables $x, y, z, \ldots$

\[Fagin, K, Reiss, Vansummeren, JACM15\]
Kaspersky Lab CEO Eugene Kaspersky said Intel CEO Paul Otellini and the Intel board had no idea what they were in for when the company announced it was acquiring McAfee on August 19, 2010.
What expressive power does relational QL add?

We began with a basic setup:

- Basic extraction by REGEX formulas
- Relational Algebra (RA)
Spanners as Regex Formulas

• Regular expression with embedded variables

\[ \gamma := \emptyset \mid \varepsilon \mid \sigma \mid \gamma \vee \gamma \mid \gamma \cdot \gamma \mid \gamma^* \mid x\{\gamma\} \]

  **Ordinary regex**
  **Span variable**

• Examples:
  - .* x{\d\d\d} .*
  - .* in w{Alabama | Alaska | Arizona | ...} .*
  - (.* z{[A-Z][a-z]*}, y{[A-Z][a-z]*}) .*) | ...

• Restriction: each “evaluation” (parse tree) assigns one span to each variable (see [Fagin+, JACM15])
Spanners as Datalog w/ Regex

Token(\(x\)) := \([\epsilon | \.*\_] \times ([a-zA-Z]* \{([,V_].*) | \epsilon\})\]

State(\(x\)) := \(\text{Token}(x) \times [\.* \times \{\text{Georgia|Virginia|Washington}\}.\ast]\)

Cap1st(\(x\)) := \(\text{Token}(x) \times [\.* \times ([A-Z].\ast).\ast]\)

CommaSp(\(x,y,z\)) := [\.* z\{\.*\}, _ y\{\.*\}.\ast]

Loc(\(z\)) := \(\text{CommaSp}(x,y,z) \times \text{Cap1st}(x) \times \text{State}(y)\)

\[\text{RETURN}(x,z) := \text{Cap1st}(x) \times [\.* x\{\.*\}_\text{from}_z\{\.*\}.\ast]\times \text{Loc}(z)\]

Another representation system for spanners

<table>
<thead>
<tr>
<th>(X)</th>
<th>(Z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,7)</td>
<td>[13,28)</td>
</tr>
<tr>
<td>(\text{Carter})</td>
<td>(\text{Plains, Georgia})</td>
</tr>
<tr>
<td>[30,40)</td>
<td>[46,69)</td>
</tr>
<tr>
<td>(\text{Washington})</td>
<td>(\text{Westmoreland, Virginia})</td>
</tr>
</tbody>
</table>

Carter_from_Plains,_Georgia,_Washington
_from_Westmoreland,_Virginia
Spanners as Automata

• In an accepting run, each variable opens and later closes exactly once
  ⇒ Each accepting run defines an assignment to the variables

• Nondeterministic ⇒ multiple runs ⇒ multiple tuples

Another representation system for spanners
Fundamental Result

Spanners definable by spanner automata = Spanners definable by RA over regex formulas

= Spanners definable by NR-Datalog over regex formulas

Token(x) := \[(\varepsilon | .*)_x{[a-zA-Z]+}((,V_.*)_| \varepsilon)\]
State(x) := \[.\ x{\text{Georgia}\mid\text{Virginia}\mid\text{Washington}}\].
Cap1st(x) := \[.\ x{[A-Z].}].
CommaSp(x,y,z) := \[.\ z{x{.}_,\ y{.}}].
Loc(z) := CommaSp(x,y,z), Cap1st(x), State(y)
RETURN(x,z) := Cap1st(x), \[.*x{.}\_\text{from}_z{.}.*\]}, Loc(z)
Consequences & Follow Ups

• Analysis of language extensions
  - Expressiveness, closure, difference, string operators
    [Fagin+, PODS14, JACM15]

• Principles of declarative cleaning in IE
  - [Fagin+, PODS14, TODS16]

• Complexity analysis
  - [Freydenberger & Holldack, ICDT16, ICDT17]

• Uniform structured/unstructured DB
  - [Nahshon, Peterfreund, Vansummeren, WebDB16]
Outline

• Enterprise Search
• Information Extraction
• Prioritized Repairing
Cleaning IE Inconsistencies

• Extractors may produce inconsistent results
  ▪ Data artifacts
  ▪ Developer limitations

• Rather than repairing the existing extractors, common practice is to clean (intermediate) results
  – GATE/JAPE “controls” [Cunningham02]
  – POSIX regex disambiguation [Fowler03]
  – SystemT “consolidators” [Chiticariu+10]
  – Implicit in other rule systems, e.g., WHISK [Soderland99]
Implementation in IBM SystemT

```sql
create view Caps as 
extract regex /[A-Z](\w|-)+/ on D.text as name from Document D;

create view Last as 
extract dictionary LastGaz on D.text as name from Document D;

create view CapsLast as 
select CombineSpans(C.name, L.name) as name
from Caps C, Last L
where FollowsTok(C.name, L.name, 0, 0);
...

create view PersonAll as 
(select R.name from FirstLast R) union all ...
... union all (select R.name from CapsLast R);

create view Person as select * from PersonAll R
consolidate on R.name using 'ContainedWithin';

output view Person;

[Chiticariu, Krishnamurthy, Li, Raghavan, Reiss, Vaithyanathan, ACL 2010]
Five GATE/JAPE Controls

Sequence 12345 and sequence 12.

Document

.* x\{d\d+\} .* 

Spanner

Context: Sequence 1 2 3 4 5 and sequence 1 2.
Match

All

Brin

Screenshots from GATE UI

Once

First

Appelt

general architecture for text engineering
Declarative Cleaning

• Problem: existing policies are ad-hoc; how to expose a language for user declaration?

• We proposed a framework for declarative cleaning in IE [PODS14,TODS16]

• Can state rules like:

  - x and y are overlapping spans → not [ Person(x) & Location(y) ]
  - x and y separated by “and/or,” → not [ Person(x) & Location(y) ]
  - y strictly contains x → Prefer Person(y) to Person(x)
  - true → Prefer Location(y) to Person(x)

“denial constraints”

“priority relation”
Research Outcomes

- Framework based on:
  - Consistent query answering [Arenas+99]
  - Prioritized database repairs [Staworko+12]

- The framework captures, unifies, generalizes the policies of SystemT, GATE, POSIX, ...

- In addition, studied:
  - When do the rules make sense?
  - When are the rules unambiguous?
  - Do cleaning rules add expressive power?

Static analysis: quickly becomes undecidable
Prioritized Repairing

• We are given an **inconsistent database**, and a **preference relation** among tuples
  ▪ Reliability, timestamps, semantics (divorced > single), ...

• Wish to lift preferences from **tuples** to **repairs**
  ▪ Repair = maximal consistent subset of the database

• Several lifting alternatives [Staworko+12]

• We investigated complexity aspects:
  ▪ **Repair checking**: *Is a given repair optimal?*
    - [Fagin, K, Kolaitis, PODS15]
  ▪ **Categoricity**: *Is repairing ambiguous?*
    - [K, Livshits, Peterfreund, ICDT17]
Concluding Remarks

• Described 3 lines of research with Ron @Almaden
  ▪ Enterprise search via search database systems
  ▪ Foundations of IE via document spanners
  ▪ Declarative cleaning in IE via prioritized repairing

• Current effort: stronger document spanners; uniform structured/unstructured; further prioritized repairing; …

• Takeaway: Again and again, “annoying details” led to fruitful fundamental research!