WRAPUP: SRL & DBMS
Opportunities

- Applications
  - Web and Social Media
- Information Alignment
  - Entity Resolution
  - Ontology Alignment
- Modeling
  - Selectivity Estimation
  - Probabilistic Databases
Opportunities

- Applications
  - Web and Social Media
- Information Alignment
  - Entity Resolution
  - Ontology Alignment
- Modeling
  - Selectivity Estimation
  - Probabilistic Databases
Relational Info in Web & Query Logs

The Internet
Summary of Query Logs

- Clicked-For
- Shares-Terms
- Shares-Words
- Same-Topic
- Hyperlink
- Same-Session
- Precedes-In-Session
- Subset-URLs
- Identical-URLs
- Partial-Overlap-URLs
- Precedes-Temporally
- Prec-In-Logical-Sess.
- Same-Logical-Session
- Is-Represented-By
- Fulfills-Info-Need
- Targets-Info-Need
- Have-Info-Need
- Search-For
- Search-For-&-Click
- Similar Users
- More-Relevant-Than
Relational Info in Social Media
Summary of Social Media Relationships

User-User
- Friends
- Collaborators
- Family
- Fan/Follower
- Replies
- Co-Edits
- Co-Mentions, etc.

User-Doc
- Comments
- Edits, etc.

User-Group

User-Query-Click

User-Tag-Doc
Opportunities

- Applications
  - Web and Social Media

- **Information Alignment**
  - Entity Resolution
  - Ontology Alignment

- Modeling
  - Selectivity Estimation
  - Probabilistic Databases
Entity Resolution

- Entities
  - Author References
- Attributes
  - Name
- Relationships
  - Co-authorship
    - Symmetric
- Goal: Identify references that denote the same person
Entity Resolution

- References, names, coauthorship
- Use rules to express evidence
  - “If two people have the same name, they are probably identical”
  - “If two people have the same coauthors, they are probably identical”
  - “If A=B and B=C, then A and C must also denote the same person”
Entity Resolution

- References, names, coauthorship
- Use rules to express evidence
  - “If two people have the same name, they are probably identical”
  - “If two people have the same coauthors, they are probably identical”
  - “If A=B and B=C, then A and C must also denote the same person”

\[
\begin{align*}
A.\text{name} & \approx_{\text{Levenshtein}} B.\text{name} \\
\Rightarrow A \approx B : 0.5
\end{align*}
\]
Entity Resolution

- References, names, coauthorship
- Use rules to express evidence
  - “If two people have the same name, they are probably identical”
  - “If two people have the same coauthors, they are probably identical”
  - “If A=B and B=C, then A and C must also denote the same person”

\[
\{A.\text{coauthor}\} \approx \{\emptyset\} \{B.\text{coauthor}\} \implies A \approx B : 0.7
\]
Entity Resolution

- References, names, coauthorship
- Use rules to express evidence
  - “If two people have the same name, they probably identical”
  - “If two people have the same coauthors, they are probably identical”
  - “If A=B and B=C, then A and C must also denote the same person”

Domain Specific Constraint
Opportunities

- Applications
  - Web and Social Media
- Information Alignment
  - Entity Resolution
  - Ontology Alignment
- Modeling
  - Selectivity Estimation
  - Probabilistic Databases
Input: Unaligned Ontologies
similar(A,B) [A\approx B]
similar(Customer, Customers) [Customer\approx Customers]

**Input: Rules**

- **Service & Products**
  - Software
  - Hardware
  - IT Services

- **Customers**
  - Developer
  - Sales Person
  - Staff
  - helps
  - interacts
  - sells to

- **Employee**
  - work for

- **Organization**

- **Products & Services**
  - Software Dev
  - Hardware
  - Consulting
  - buys

- **Customer**
  - interacts with
  - helps
  - sells

- **Employee**
  - domain(work for, Employees)

- **domain(C,D)**

- **Similarity (A,B)**
  - [A\approx B]

- **Similarity (Customer, Customers)**
  - [Customer\approx Customers]
Input: Rules

\[ R \approx T \iff \text{domainOf}(R, A) \land \text{domainOf}(T, B) \land A \approx B \land R \not\approx T : 0.7 \]
Input: Rules

{A.subConcept} \approx \{B.subConcept\} \iff A \neq B \land A \approx B \land \text{type}(A, \text{concept}) \land \text{type}(B, \text{concept}) : 0.8
Input: Rules

similar := partial-functional
:= inverse partial-functional
Experiments: Ontology Alignment

- OAEI Ontology Alignment Benchmark (2008)
  - Real world ontologies (300s)
  - Synthetic ontology pairs
  - Approx 100 entities
  - 21 rules, modified standard string measures
OAEI comparison

Other results as reported by the benchmark participants.
Opportunities

- Applications
  - Web and Social Media
- Information Alignment
  - Entity Resolution
  - Ontology Alignment
- **Modeling**
  - Selectivity Estimation
  - Probabilistic Databases
Using SRL for DBMS

- Using SRL models for selectivity estimation and PrDB, the BIGGEST difference is the.....
  - Probabilistic semantics and query answering

- For selectivity estimation, rather than computing probabilities of possible worlds, we want to compute expected frequencies
- For probabilistic databases, in addition to computing probabilities of possible worlds, we want to compute the probabilities of the results of SQL queries over possible worlds
Opportunities

- Applications
  - Web and Social Media

- Information Alignment
  - Entity Resolution
  - Ontology Alignment

- Modeling
  - Selectivity Estimation
  - Probabilistic Databases
SRL for Selectivity Estimation

- Capture the \textit{frequency} information, rather than probabilistic information about individuals.

Query Result Size Estimation

- Crucial for
  - cost-based query optimization
  - query profilers
Traditional Approaches

- Approximate joint distribution by making several key independence assumptions:
  - Attribute Value Independence:
    - joint distribution is product of single attribute distributions
  - Join Uniformity Assumption:
    - tuple in one relation is equally likely to join with any tuple in the other relation
PRMs

Use graphical models to compactly represent joint distribution

- over single table for select selectivity
- over multiple tables for join selectivity

Provides a unified framework for estimating the selectivity of select-join queries over multiple tables
Estimation for Single Table

- Query: select * from R
  where R.A_1 = a_1 and … and R.A_k = a_k

  \[ \text{Size}(Q) = |R| \times P(a_1,\ldots,a_k) \]

- Use inference algorithm to compute \( P(a_1,\ldots,a_k) \)

- Algorithm complexity depends on FG connectivity; efficient in practice
Foreign-key Join Selectivity

Uniform Join Assumption

Assuming referential integrity

\[ \text{Size}(\text{Purchase} \rightarrow \text{Person}) = | \text{Purchase} | \]
Correlated Attributes

Type = luxury

Type = necessity

Income = high

Income = low

Purchase

Person
Skewed Join

Type = luxury

Type = necessity

Income = high

Income = low

Purchase

Person
Query: select * from R, S
where R.F = S.K
and R.A = a and S.B = b

\[ P(J_F) = \text{prob. randomly chosen tuple from } R \text{ joins with a randomly chosen tuple from } S \]

\[ \text{size}(Q) = |R| \cdot |S| \cdot P(J_F, a, b) \]
Universal Foreign Key Closure

- A DB schema is **table-stratified** if we can order the tables s.t. if R.F refers to S.K, S precedes R.F in the stratification ordering.

- The **universal foreign key closure** is the query constructed by introducing a tuple variable for each leaf in the stratification, and, introducing, for each foreign key, a new tuple variable.
Universal Foreign Key Closure

- Schema: R, S, T
  R.F refers to S, S.F refers to T
  stratification: T < S < R
  r ⇓ s ⇑ t

- Schema: R, S
  R.F₁ refers to S, R.F₂ refers to S
  stratification: S < R
  r ⇓ s₁ ⇑ s₂
PRMs for selectivity estimation

• Model distribution of attributes across multiple tables
• Allow attribute values to depend on attributes in the same table
• Allow attribute values to depend on attributes in other tables along a foreign key join
• Can model the join probability of two tuples using *join indicator variable*
Example PRM

School
  Prestige

Person
  Age
  Income

Purchase
  Type

Attended

Bought-by

Type=necessity
  false
  true

Income = high
  0.9998, 0.0002
  0.99, 0.01

0.999, 0.001
Path Dependency Graph

- Construct **path dependency graph**
PRM for Selectivity Semantics

\[ P(Q) = \prod_{V \in Q} P(V \mid Pa(V), J^* = T) \]
Answering Selectivity Estimation Queries

- Construct Query Evaluation FG for Query:
  
  ```sql
  select * from Person, Purchase
  where Person.id = Purchase.buyer-id
  and Person.Income = high and Purchase.Type = luxury
  ```

- Compute upward closure of query attributes by including all parents as well
SRL for Selectivity Estimation

- Syntax - (almost) same
- Learning - (almost) same
- Semantics - (very) different
- Inference - (very) different
Opportunities

- Applications
  - Web and Social Media
- Information Alignment
  - Entity Resolution
  - Ontology Alignment
- **Modeling**
  - Selectivity Estimation
  - **Probabilistic Databases**
Probabilistic Databases*

- Goal of Probabilistic Databases: Managing and querying large volumes of data annotated with probabilities
- Much work in recent years, leading up to many systems

  Mystiq (University of Washington)  ProbView (Maryland)
  Trio (Stanford)  Orion (Purdue University)
  MayBMS (Cornell)  Bayesstore (Berkeley)
  PrDB (Maryland)  Lahar (University of Washington)
  MCDB (Univ. of Florida, IBM)  ..... 

- Other work on approximations, answering various types of queries (ranking, nearest neighbors etc.), summarization...

Slides courtesy of Amol Deshpande; PrDB work joint w/ Prithviraj Sen, Bhargav Kanagal, Jian Li
Probabilistic Databases

- Types of uncertainties typically supported
  - *Tuple-existence* uncertainty
    - A tuple may or may not exist in the database
  - *Attribute-value* uncertainty
    - The *value* of an attribute not known precisely
    - Instead a distribution over possible values is provided

- **PrDB Goals:**
  - Increase representational power to support:
    - Correlations among the data items
    - Uncertainties at different abstraction levels and granularities
  - Scale reasoning and querying to large-scale uncertain data while supporting the above
PrDB Framework

- Flexible uncertainty model (based on probabilistic graphical models)
  - Support for representing rich correlation structures [ICDE’07]
  - Support for specifying uncertainty at multiple abstraction levels [DUNE’07]

- Declarative constructs for interacting with the database
  - Manipulating and updating uncertainty as a first class citizen

- Rich querying semantics
  - SQL queries; Inference, reasoning, and what-if queries

- New techniques for scaling reasoning and query processing
  - Inference techniques to exploit the structure in the data [VLDB’08, UAI’09]
  - Novel techniques for minimizing the complexity of query evaluation [VLDB’10]
  - Index structures for handling large volumes of data [SIGMOD’09,’10]
  - Efficient algorithms for ranking [VLDB’09,’10], consensus answers [PODS’09]
A Simple Example

- Represent tuple uncertainty and correlations using factors

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>‘m’</td>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td>s2</td>
<td>‘n’</td>
<td>1</td>
<td>0.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>B</th>
<th>C</th>
<th>prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>1</td>
<td>‘p’</td>
<td>0.4</td>
</tr>
</tbody>
</table>
A Simple Example

- Represent tuple uncertainty and correlations using *factors*

```
<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>‘m’</td>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td>s2</td>
<td>‘n’</td>
<td>1</td>
<td>0.5</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th></th>
<th>s1</th>
<th>f₁(s₁)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.4</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.6</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th></th>
<th>B</th>
<th>C</th>
<th>prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>1</td>
<td>‘p’</td>
<td>0.4</td>
</tr>
</tbody>
</table>

```
<table>
<thead>
<tr>
<th></th>
<th>s2</th>
<th>t₁</th>
<th>f₂(s₂, t₁)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.4</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

0 = Tuple does not exist
1 = Tuple exists

s2 and t1 mutually exclusive
A Simple Example

- Represent tuple uncertainty and correlations using factors

<table>
<thead>
<tr>
<th>S</th>
<th>A</th>
<th>B</th>
<th>prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>'m'</td>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td>s2</td>
<td>'n'</td>
<td>1</td>
<td>0.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>T</th>
<th>B</th>
<th>C</th>
<th>prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>1</td>
<td>'p'</td>
<td>0.4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>s1</th>
<th>f₁(s₁)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.4</td>
</tr>
<tr>
<td>1</td>
<td>0.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>s2</th>
<th>t1</th>
<th>f₂(s₂, t₁)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.4</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Factor graph!
A Simple Example

- Represent tuple uncertainty and correlations using *factors*

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>'m'</td>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td>s2</td>
<td>'n'</td>
<td>1</td>
<td>0.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>s1</th>
<th>f₁(s₁)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.6</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>s2</th>
<th>t1</th>
<th>f₂(s₂, t₁)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

And, if we have attributed uncertainty and/or shared correlations, then *parfactor* graph
Prob of Possible Worlds

- Distribution specified completely by:
  - A set of random variables
  - A set of factors over the random variables

- Joint pdf obtained by multiplying all the factors and normalizing:
  \[ \Pr(s_1, s_2, t_1) \propto f_1(s_1) f_2(s_2, t_1) \]

  For example:
  \[
  \Pr(s_1 = 0, s_2 = 0, t_1 = 0) = \frac{1}{Z} f_1(s_1 = 0) f_2(s_2 = 0, t_1 = 0)
  \]

- An *Inference* task: Finding a marginal prob. distribution over subset of variables
  - e.g. \( \Pr(t_1) \)
During query processing, add new deterministic factors (hard constraints) corresponding to intermediate tuples.

- Encode the dependencies between base tuples and intermediate tuples.

Example query: $\pi_c(S \bowtie_B T)$

- IF: $s_1$ and $t_1$ are 1
- THEN: $Pr(i_1 = 1) = 1$, $Pr(i_1 = 0) = 0$
- ELSE: $Pr(i_1 = 1) = 0$, $Pr(i_1 = 0) = 1$
Probabilistic Graphical Models

- A PGM can compactly represent a joint probability distribution over a large number of random variables with complex correlations.
- Specified completely by:
  - A set of random variables
  - A set of factors over the random variables
- Joint pdf obtained by multiplying all the factors and normalizing

\[
\text{Pr}(s_1, s_2, t_1, i_1, i_2, r_1) \propto \sum f_1(s_1) f_2(s_2, t_1) f^>(s_1, t_1, i_1) f^>(s_2, t_1, i_2) f^\vee(i_1, i_2, r_1)
\]
A Simple Example

- **Query evaluation**: Find the result tuple probabilities
  - Can use standard techniques like variable elimination, junction trees (exact), message passing, loopy Belief propagation, Gibbs Sampling (approx)

\[
\Pr(r_1) \propto \sum_{s_1, t_1, i_1, i_2} f_1(s_1) f_2(s_2, t_1) f^\wedge(s_1, t_1, i_1) f^\wedge(s_2, t_1, i_2) f^\vee(i_1, i_2, r_1)
\]
Efficient Inference with Shared Factors

- **Option 1**: “Ground out” (propositionalize) the random variables, and use standard techniques
  - Would need to create a model with a few million nodes

- **Option 2**: Directly operate on the shared factors
  - “Lifted inference”: Much work in recent years in the ML community

- We developed a general purpose lifted inference technique for PrDB based on *bisimulation* [VLDB’08, UAI’09]
### Inference with Shared Factors

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>s1</strong></td>
<td><strong>f₁(s₁)</strong></td>
</tr>
<tr>
<td>0</td>
<td>0.2</td>
</tr>
<tr>
<td>1</td>
<td>0.8</td>
</tr>
<tr>
<td><strong>s2</strong></td>
<td><strong>f₂(s₂)</strong></td>
</tr>
<tr>
<td>0</td>
<td>0.2</td>
</tr>
<tr>
<td>1</td>
<td>0.8</td>
</tr>
<tr>
<td><strong>s3</strong></td>
<td><strong>f₃(s₃)</strong></td>
</tr>
<tr>
<td>0</td>
<td>0.4</td>
</tr>
<tr>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td><strong>s₄</strong></td>
<td><strong>f₄(s₄)</strong></td>
</tr>
<tr>
<td>0</td>
<td>0.21</td>
</tr>
<tr>
<td>1</td>
<td>0.79</td>
</tr>
<tr>
<td><strong>t₁</strong></td>
<td><strong>g(t₁)</strong></td>
</tr>
<tr>
<td>0</td>
<td>0.5</td>
</tr>
<tr>
<td>1</td>
<td>0.5</td>
</tr>
</tbody>
</table>

---

Augmented FG constructed for a query.

Inference goal: compute probability distributions over variables: i₁, i₂, i₃, i₄.
Inference with Shared Factors

(Near-)identical answers because of the symmetry

How to identify such opportunities in general?
Bisimulation-based Lifted Inference

1: Capture a (simulated) inference run as a graph

2: Run exact or approximate graph bisimulation (to identify symmetries)

3: Compress

4: Run inference on the compressed graph
Example

[[ 3 relation join with 3 tuples each, attribute and tuple uncertainty ]]

Original RV-Elim graph, 1170 vertices

Compressed RV-Elim graph, 78 vertices
Bisimulation-based Lifted Inference

- Orders of magnitude performance improvements with symmetry
- Bisimulation can be done in linear time on DAGs
  - Somewhat more involved here
    - Need to keep track of the order in which factors were multiplied
    - Must construct labels on-the-fly as opposed to standard bisimulation
      - $O(|E| \log(D) + |V|)$
- Choice of elimination order during inference crucial
  - Dictates the amount of compression possible
  - We choose it by running bisimulation on the graphical model itself
- Our technique works on the ground (propositionalized) model
  - Enables approximations [UAI’09]
- Many open challenges in effectively exploiting symmetry and first order representations
Some Things We Skipped

- Other lifted inference techniques, e.g.,
  - Lifted variable elimination: [Poole, IJCAI03; de Salvo Braz et al IJCAI05, AAAI06; Milch et al., AAAI08]
  - Lifted belief propagation [Jaimovich et al., UAI07; Singla & Domingos, AAAI08; Kersting et al., UAI09; de Salvo Braz et al, SRL-09]

- SRL Models based on probabilistic programming languages
  - E.g., IBAL [Pfeffer, IJCAI01], BLOG [Milch et al., IJCAI05], Church [Goodman et al., UAI08], Factorie [McCallum et al., NIPS09]
Conclusion

- We’ve covered:
  - SRL Basics:
    - Relational Classifiers
    - Collective Classification
    - SRL Approaches: PRMs, MLNs, PSL
  - Examples of how SRL can be applied to several DBMS problems
- Many things not covered
- Hopefully you are now
  - excited about applying SRL to your DB research!
  - excited about applications of your DB research to SRL!
Thank you!!

Acknowledgements:
- The linqs group at UMD [http://www.cs.umd.edu/linqs](http://www.cs.umd.edu/linqs) is supported by ARO, DARPA, KDD, NSF, MIPS, Microsoft Research, Google, Yahoo!
- Lily is supported by a CI Fellowship under an NSF CRA grant
References

- [Antova et al., VLDBJ09] 10^10^6 Worlds and Beyond: Efficient Representation and Processing of Incomplete Information. VLDB Journal on Probabilistic Databases 2009.
References

References

References

- [Lu & Getoor, ICML03] Link-Based Classification. Q. Lu & L. Getoor. ICML 2003.
References

References

- [Riedel, UAI08] Improving the accuracy and Efficiency of MAP Inference for Markov Logic. S. Riedel. UAI 2008.
References

- [Singla & Richardson, WWW08] Yes, There is a Correlation - From Social Networks to Personal Behavior on the Web. P. Singla & M. Richardson. WWW 2008.