ProbKB Web-Scale Probabilistic Knowledge Base

Yang Chen

yang@cise.ufl.edu

Computer and Information Science and Engineering
University of Florida

Feb 22, 2012

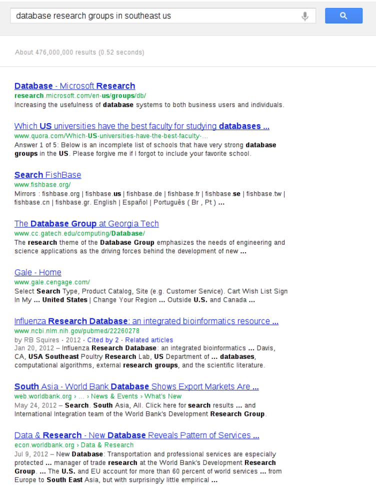
Outline

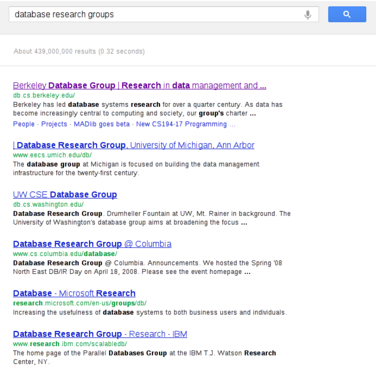
Knowledge bases–Introduction

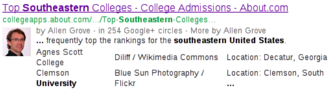
- A *knowledge base* [ND10] is a special kind of database for knowledge management. A knowledge base provides a means for information to be collected, organized, shared, searched and utilized.
- A knowledge base helps machines understand humans, languages, and the world.

DSR @ UF
Data Science Research

Knowledge bases–Introduction



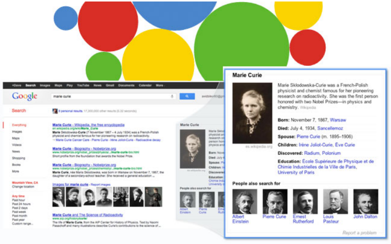




Outline

Examples

Google Knowledge Graph [Goo12]



Demo

DSR @ UF
Data Science Research

Examples

Nell

Nell [CBK+10] is a research project that attempts to create a computer system that learns over time to read the web.

Recently-Learned Facts twitter Refresh

instance	iteration	date learned	confidence
albert_iii is a monarch	623	13-aug-2012	99.9
kaseberg_creek is a river	620	01-aug-2012	99.1
senior_e_commerce_intelligence_consultant is a profession	622	09-aug-2012	98.4
deonna_anderson is a journalist	620	01-aug-2012	97.3
rose_city_comic_con is a convention	620	01-aug-2012	98.6
cancidas is a drug worked on by merck	622	09-aug-2012	93.8
softball is a sport with fans in the country new_zealand___wellington	623	13-aug-2012	100.0
mammals is an animal that eats berries	622	09-aug-2012	100.0
wccu is a TV affiliate of the network fox	625	21-aug-2012	100.0
cats is an animal that can develop respiratory_infection	622	09-aug-2012	100.0

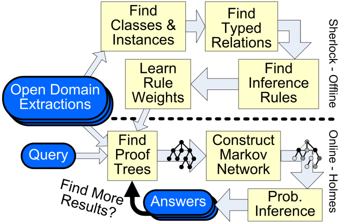
Note that Nell produces uncertain results. This is typical among automatic extraction systems and is our major motivation to develop a large-scale probabilistic knowledge base.

Examples

Sherlock-Holmes

The Sherlock-Holmes [Sch11] is an open information extraction system consisting of two components:

- Sherlock [SEWD10], which learns inference rules offline, and
- Holmes [SEW08], which uses inference rules to answer queries online.

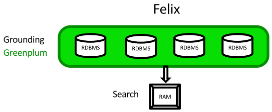


Examples

Tuffy-Felix

The Tuffy-Felix [NRDS11, NZRS11] system is an Markov logic network [RD06] implementation that does large-scale probabilistic inference using an RDBMS.

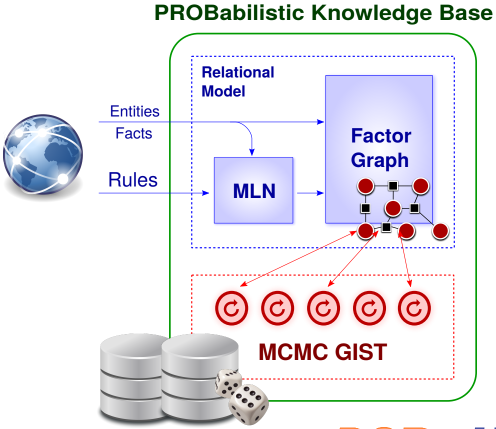
- A bottom-up approach to grounding using an RDBMS.
- A hybrid in-database grounding and in-memory inference architecture.
- Novel partitioning, loading, and parallel algorithms.
- Task decomposition to achieve web-scale.



Outline

Architecture

- Extracted entities, facts, and rules stored as relational model.
- Efficient grounding via a few relational operators.
- Parallel MCMC inference implemented as GIST operations.
- Incremental inference: save computation by focusing on least convergent variables.



Outline

Markov Logic–Probabilistic Inference Framework

A *Markov logic network* (MLN) [RD06] is a set of formulae with weights. Together with a finite set of constants $C = \{c_1, \ldots, c_{|C|}\}$, it defines a Markov network.

Weight	First-Order Logic
0.7	Fr(x,y)$\wedge$Fr(y,z)$\to$Fr(x,z)
1.5	Sm(x)$\to$Ca(x)
1.1	Fr(x,y)$\wedge$Sm(x)$\to$Sm(y)

Table: Example Markov logic network.

A set of constants (entities, or objects)

$$C = \{A, B\}.$$

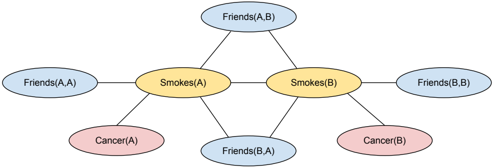


Figure: Grounded Markov network.

Markov Logic Inference

Grounding

Grounding is the process of substituting constants into MLN clauses.

The result of grounding is a *factor graph* (or *Markov network*) from which we can infer marginal probabilities for individual facts.

Key Challenges

- Time-consuming, especially if the numbers of rules and entites are large.
- Grounded network has an intractably large size, making inference tasks slow.

Markov Logic Inference

Scaling to the Web

- First-order Horn clauses:
  - Avoids the need to enumerate ground atoms.
  - Stored as *first-class* citizen in RDBMS, grounding expressed as a few `Join` s.
  - Easier to learn than general first-order clauses.
- Leveraging RDBMS query optimization techniques and possibly MPP frameworks (e.g. Greenplum).
- Ontology (typing): reducing the number of possible groundings and improving accuracy.

Grounding

A single `JOIN` operation handles all rules of type

$$p(x : c_1, y : c_2) \leftarrow q(x : c_1, z : c_3), r(z : c_3, y : c_2)$$

```
SELECT DISTINCT mln.head AS head, mln.body1 AS body1,
    mln.body2 AS body2, r1.ent1 AS ent1, r1.ent2 AS
    ent2, r2.ent2 AS ent3
FROM relations r1, mln, relations r2, relations r3,
    instances i1, instances i2, instances i3
WHERE r1.pred = mln.head AND r2.pred = mln.body1 AND
    r3.pred = mln.body2
AND r1.ent1 = r2.ent1 AND r1.ent2 = r3.ent2 AND r2.ent2
    = r3.ent1
AND i1.ent = r1.ent1 AND i1.class = mln.class1
AND i2.ent = r1.ent2 AND i2.class = mln.class2
AND i3.ent = r2.ent2 AND i3.class = mln.class3
```

Results

We can ground the whole Sherlock-Holmes dataset the first two rounds in 10 minutes using PostgreSQL, while the state-of-the-art implementation MLN (Tuffy [NRDS11]) crashes during its grouding phase.

#relations	10,672
#rules	31,000
#constants	1.1M
#evidence	250,000
#queries	10,672
Tuffy	**Crash**
ProbKB	10 min[1]

Table: Dataset statistics and performance.

[1] First two rounds.

DSR @ UF Data Science Research

Inference

Inference–Computing Probabilities

The Markov random field defines a probability distribution on all nodes in it:

$$P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(\mathbf{x})\right),$$

where n_i is the number of ground clauses that satisfy clause i in the MLN and w_i is the weight of that clause. Z is the normalization constant, also called the *partition function*.

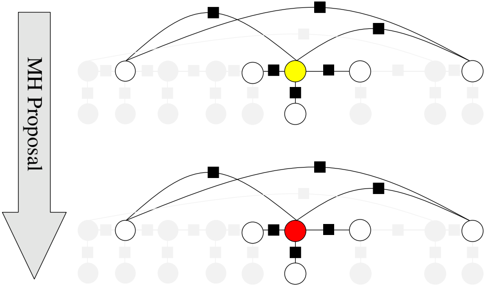
MCMC-MH

Metropolis-Hastings (MH)

- Exact inference: intractable due to Z.
- MCMC-MH: efficient since Z cancels out.
- MCMC-MH: only changed factors need to be considered [WMM10]:

$$\frac{\pi(\mathbf{x})}{\pi(\mathbf{x}')} = \frac{\frac{1}{Z}\prod_i \phi_i(\mathbf{x}_i)}{\frac{1}{Z}\prod_i \phi_i(\mathbf{x}'_i)} = \frac{\prod_{i \text{ having } x_{(k)}} \phi_i(x_{(k)}, \mathbf{x})}{\prod_{i \text{ having } x_{(k)}} \phi_i(x'_{(k)}, \mathbf{x})}.$$

MCMC-MH Efficiency



Outline

GraphLab

Data-Parallel	Graph-Parallel

Map Reduce

- Feature Extraction
- Cross Validation
- Computing Sufficient Statistics

GraphLab Carnegie Mellon

Graphical Models
- Gibbs Sampling
- Belief Propagation
- Variational Opt.

Semi-Supervised Learning
- Label Propagation
- CoEM

Collaborative Filtering
- Tensor Factorization

Data-Mining
- PageRank
- Triangle Counting

DSR @ UF Data Science Research

GraphLab Execution Model

Algorithm 1 GraphLab Execution Model

Input: Data graph $G = (V, E, D)$
Input: Initial vertex set $\mathcal{T} = \{v_1, v_2, \ldots\}$
while $\mathcal{T}$ is not empty **do**
    $v \leftarrow$`RemoveNext`$(\mathcal{T})$
    $(\mathcal{T}', \mathcal{S}_v) \leftarrow f(v, \mathcal{S}_v)$
    $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{T}'$
Output: Modified data graph $G = (V, E, D')$

- Vertexes schedule execution of their neighbors.
- Not applicable to general MCMC algorithms.

DSR @ UF
Data Science Research

Datapath GIST

Generalized Iterable State Transforms (GIST)

- GIST Performs *transitions* upon a *state* until that state has converged to the desired result.
  - Transition: MCMC Proposal function.
  - State: Factor graph with its samples.
- A user-defined local scheduler allows general MCMC proposal implementation.
- The GIST state keeps track of the inference result.

DSR @ UF
Data Science Research

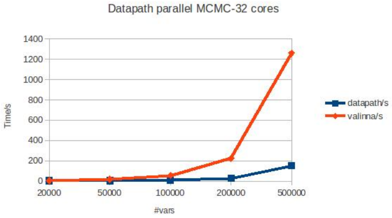
GraphLab Preliminary Results

#samples	10	100	200	500	State-of-the-art
IE	0.2s	2s	4s	10.2s	25.216s
ER	90.8s	181.5s	373s	>600s	225s
RC	5.2s	52.7s	111.3s	297.8s	**Crashed**
Sherlock-600	1.2s	12.6s	28.3s	65.1s	55min

Table: GraphLab-based parallel inference vs the state-of-the-art.

Datapath Preliminary Results

Figure: Inference over simulated factor graphs



Outline

Incremental MCMC

Goal

- EFBP [ND10] in a MCMC setting

Incremental MCMC

The incremental maintenance algorithm adopts the Query-aware MCMC [WM11] technique:

- Assumption: not much new information is added to the knowledge base each time.
- Newly extracted knowledge serves as the query node.
- Maintaining samples for both old and new nodes.

Incremental Maintenance of MCMC Samples

The incremental MCMC proposal function T employs the following steps:

1. Sample the variable index space according to some distribution p reflecting influence of new nodes and recent sample behaviors.
2. Sample the selected variable according to some distribution q over that variable's domain, leaving all other variables unchanged.

We adjust the distribution p so that it focuses on newly added variables.

Conclusions

- ProbKB is a web-scale **PROB**abilistic **K**nowledge **B**ase with Markov logic network as the primary data model.
- All extractions, including entities and rules, are stored in RDBMS as relational model, allowing efficient grounding algorithms.
- In-database GIST operator allows parallel MCMC inference.
- Incremental MCMC focuses computation on most recently added variables, speeding up convergence.

Questions?

Thank you!

Outline

References I

A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E.R. Hruschka Jr, and T.M. Mitchell.
Toward an architecture for never-ending language learning.
In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, pages 1306–1313, 2010.

Google.
Introducing the knowledge graph: things, not strings, 2012.

A. Nath and P. Domingos.
Efficient belief propagation for utility maximization and repeated inference.
In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, 2010.

DSR @ UF
Data Science Research

References II

Feng Niu, Christopher Ré, AnHai Doan, and Jude W. Shavlik.
Tuffy: Scaling up statistical inference in markov logic networks using an rdbms.
PVLDB, 4(6):373–384, 2011.

F. Niu, C. Zhang, C. Ré, and J. Shavlik.
Felix: Scaling Inference for Markov Logic with an Operator-based Approach.
ArXiv e-prints, August 2011.

M. Richardson and P. Domingos.
Markov logic networks.
Machine learning, 62(1):107–136, 2006.

References III

S. Schoenmackers.
Inference Over the Web.
PhD thesis, University of Washington, 2011.

S. Schoenmackers, O. Etzioni, and D.S. Weld.
Scaling textual inference to the web.
In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 79–88. Association for Computational Linguistics, 2008.

S. Schoenmackers, O. Etzioni, D.S. Weld, and J. Davis.
Learning first-order horn clauses from web text.
In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1088–1098. Association for Computational Linguistics, 2010.

DSR @ UF
Data Science Research

References IV

Michael L Wick and Andrew McCallum.
Queryaware mcmc.
In *proceedings of the 25th Conference on Neural Information Processing Systems (NIPS)*, pages 2564–2572, 2011.

M. Wick, A. McCallum, and G. Miklau.
Scalable probabilistic databases with factor graphs and mcmc.
Proceedings of the VLDB Endowment, 3(1-2):794–804, 2010.

DSR @ UF
Data Science Research