PROBKB Web-Scale Probabilistic Knowledge Base

Yang Chen, Xing Liu
{yang,xinliu}@cise.ufl.edu

Computer and Information Science and Engineering
University of Florida

Mar 12, 2013

Outline

Knowledge bases–Introduction

- A *knowledge base* is a collection of entities, facts, and relationships that conforms with a certain data model.
- A knowledge base helps machines understand humans, languages, and the world.

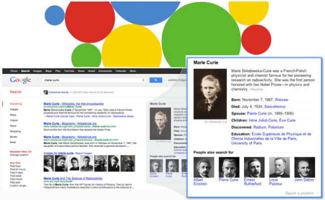


Figure: Google knowledge graph

DSR @ UF
Data Science Research

Challenges & Motivation
Knowledge Acquisition

- **Statistical Inference**
  - **Markov logic**
- Information extraction
  - NELL (CMU), OpenIE (UW)
  - Entities, relations, rules
- Human collaboration
  - Wikipedia
  - Freebase

Challenges & Motivation
Uncertainty Management

- **Statistical Inference**
  - **Probabilistic graphical models**
  - **Markov chain Monte Carlo**
- Data integration
  - Merging multiple data sources
  - Crowdsourcing/user feedback
- Data cleaning
  - Conflict, incomplete, outdated data

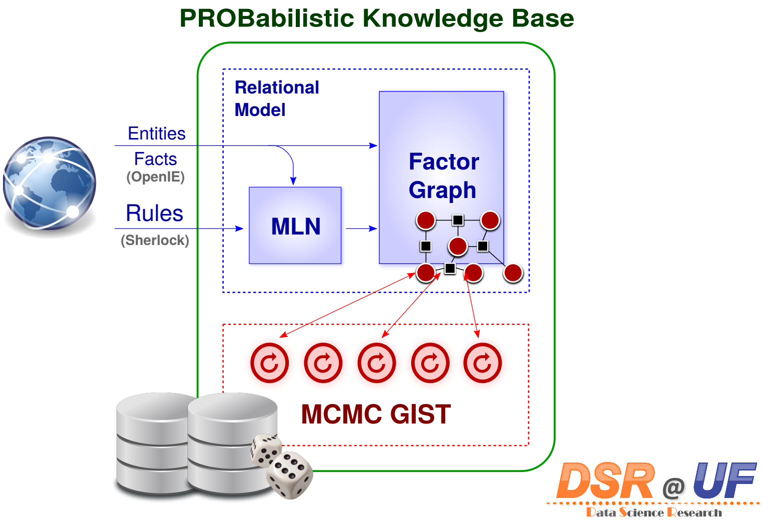
Challenges & Motivation
Scalability

- **Scalable data management systems**
  - **Relational DBMS**
  - Hadoop
  - Spark, GraphLab, **Datapath**, etc
- Scalable algorithms
  - Incremental inference
  - Query-driven inference

DSR @ UF
Data Science Research

Outline

Architecture



Markov Logic–Probabilistic Inference Framework

A *Markov logic network* (MLN) 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)

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

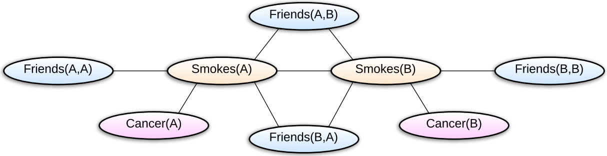


Figure: Grounded Markov network.

Outline

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: A Relational Point of View

- State-of-the-art (Tuffy, Nell): one table for each relation
- By considering only Horn clauses, we store the rules and relationships in a few tables:

Table: `MLN (M)`

head	body1	body2
p_1	q_1	r_1
p_2	q_2	r_2
p_3	q_3	r_3
p_4	q_4	r_4
	$\dots$	

Table: `Relationships (R)`

pred	ent1	ent2
p_1	x_1	y_1
p_1	x_2	y_2
p_2	x_1	y_1
p_2	x_2	y_2
	$\dots$	

Markov Logic: A Relational Pointer of View

The grounding of *ALL* rules of form

$$p(x,y) \leftarrow q(x,z), r(z,y)$$

is then expressed as a relational operation:

$$\begin{aligned} R \leftarrow\ & \rho_{R(\texttt{pred},\texttt{ent1},\texttt{ent2})}(\pi_{M.\texttt{head},R_2.\texttt{ent1},R_3.\texttt{ent2}} \\ & ((M \bowtie_{M.\texttt{body1}=R_2.\texttt{pred}} R_2) \\ & \bowtie_{M.\texttt{body2}=R_3.\texttt{pred AND } R_2.\texttt{ent2}=R_3.\texttt{ent1}} R_3)) \\ G \leftarrow\ & R_1 \bowtie R \end{aligned} \tag{1}$$

Grounding Results

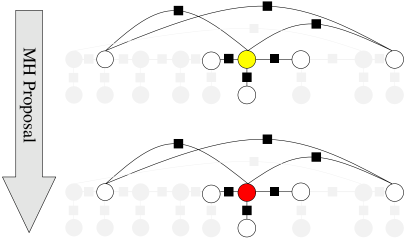
The relational Markov logic model saves us from managing thousands of tables as in previous approaches. As a result, We grounded the whole Sherlock-Holmes dataset in about 1.5 hours using Greenplum, while the state-of-the-art implementation MLN crashes during its grouding phase.

#relations	10,672
#rules	31,000
#constants	1.1M
#evidence	250,000
#queries	10,672
Tuffy	**Crash**
ProbKB	1.5 h

Table: Dataset statistics and performance.

Outline

Inference: MCMC-MH



Markov locality property allows for parallel computing.

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.

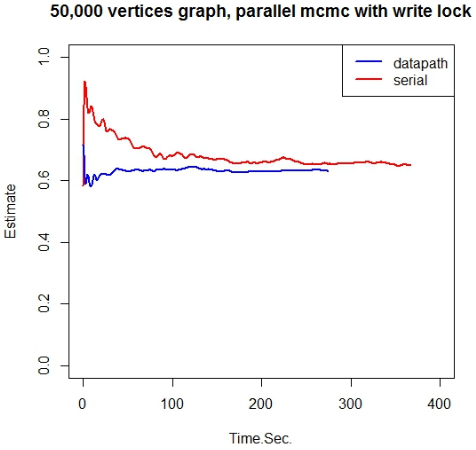
- A user-defined local scheduler allows general MCMC proposal implementation.
- The GIST state keeps track of the inference result.

Datapath GIST

Parallel Mcmc implementation modification on datapath

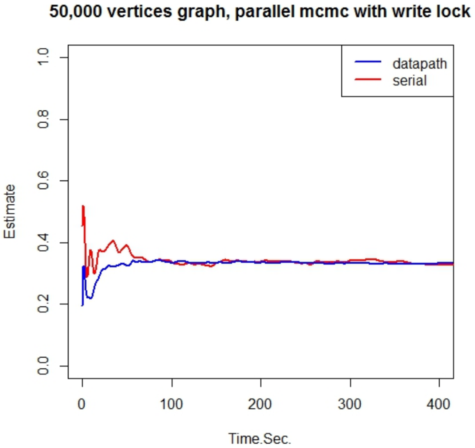
- Previous approach: cut the graph to smaller graphs, run mcmc on each partition.
    - Lost information when a factor is accross multiple subgraphs. Inaccurate, but the faster performance.
- Modified approach: Do not cut the graph. Add write lock on each variables.
    - More accurate, but lower performance.

Parallel mcmc with write lock in datapath



DSR @ UF
Data Science Research

Parallel mcmc with write lock in datapath



Preliminary Results

# Vertices	5000	10,000	25,000	250,000	2,500,000
Single Thread	7 sec	28 sec	228 sec	Hours	N/A
Datapath	7 sec	13 sec	30 sec	2661 sec	N/A

Table: Time to generate 5000 joint samples for different graphs.

Outline

Next Steps

- Grounding/inference connection
  - Construct graph structures from relational format.
  - Partition and Merge: opportunity of GLA parallelism.
- DB-memory synchronization
  - Determine an update and write-back policy to synchronize DB and in-memory data.
  - Buffer manager; IO efficiency
- Incremental inference
  - Schedule Datapath execution so that computation is focused on the least convergent portion of the factor graph.
- Evaluation
  - MCMC evaluation: multi-chain convergence test
  - Result evaluation: Manual check (AMT)

Responsibilities

Yang Grounding

Xing Inference

Questions?

Thank you!