Learning Markov Logic Networks Using Structural Motifs

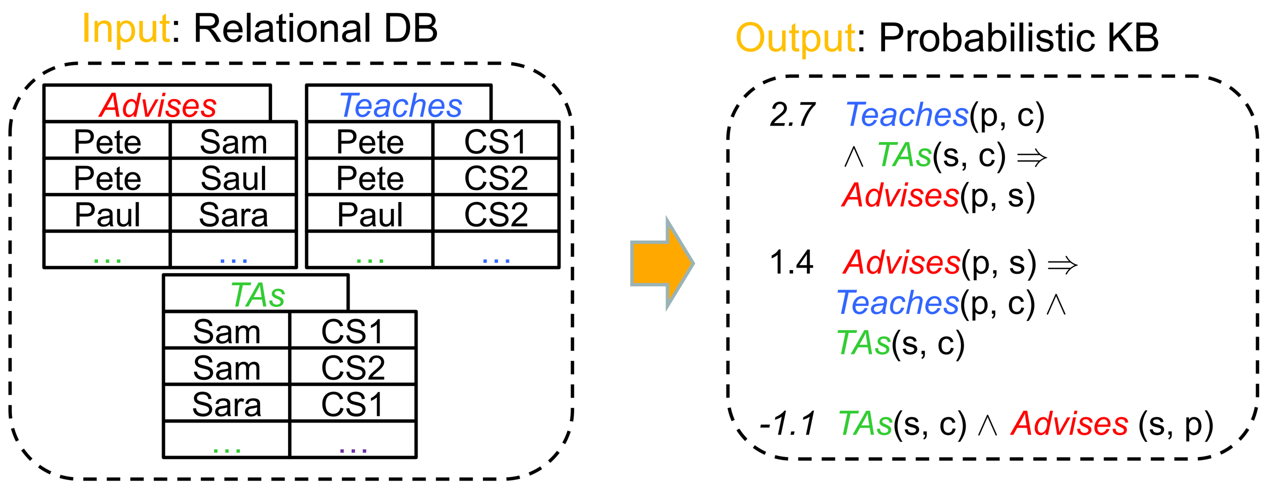
Stanley Kok & Pedro Domingos

Dept. of Computer Science and Engineering

University of Washington, Seattle, USA

Goal

- Learn probabilistic knowledge base (KB) from relational database (DB)



Main Idea

- Find recurring patterns in data (structural motifs)
- ↑Efficiency by restricting search to within structural motifs
  - Avoids spurious searches between motifs
  - Searches within a motif once, rather than in all occurrences
- Creates different motifs over same set of objects → captures different interactions among objects

Markov Logic

- A logical KB is a set of hard constraints on the set of possible worlds → brittle
- Let's make them soft constraints: When a world violates a formula, it becomes less probable, not impossible
- Give each formula a weight (Higher weight ⇒ Stronger constraint)
- A Markov logic network (MLN) is a set of pairs (F,w)
  - F is a formula in first-order logic
  - w is a real number

$$P(x) = \frac{1}{Z}\exp\left(\sum_{i=1}^{F} w_i n_i\right)$$

vector of truth assignments to ground atoms; partition function; weight of i^{th} formula; #true groundings of i^{th} formula

MLN Structure Learning

- MLN structure learning = learn formulas (and weights)
- Many previous systems use **generate-&-test** approach and/or have element of **greedy** search
  - e.g., MSL [Kok & Domingos, ICML'05] and BUSL [Mihalkova & Mooney, ICML'07]
  - Explore large search space → computationally expensive
  - Susceptible to local maxima
- LHL [Kok & Domingos, ICML'09] ameliorates above problems by clustering constants to form high-level concepts
  - But for long paths → search exponential space of paths.

Random Walks & Hitting Times

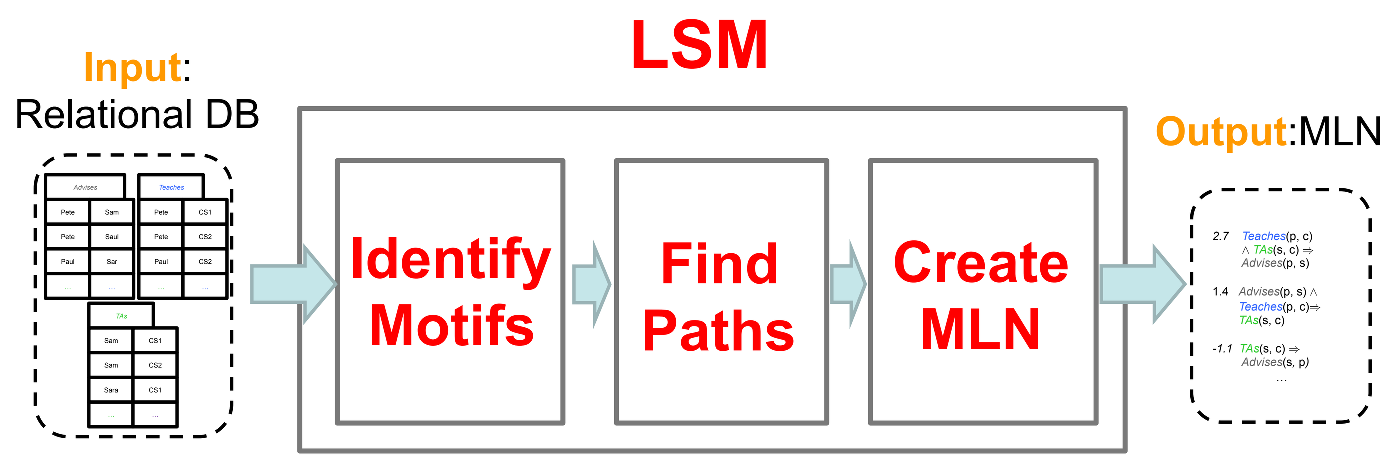
- **Random walk:** random traversal of a graph
  - When at a node, randomly select one neighbor to move to
- **Hitting time** btw node *i* and *j*: expected number of steps in a random walk starting from *i* to reach *j* for the first time
  - Smaller hitting time → node *i* and *j* are more densely connected → closer node *j* is to *i*
  - Expensive to compute for all pairs of nodes
- **Truncated** hitting time: random walk limited to T steps
  - Only visit vicinity of node *i*
  - Efficiently estimated by sampling [Sarkar, Moore & Prakash, ICML'08]

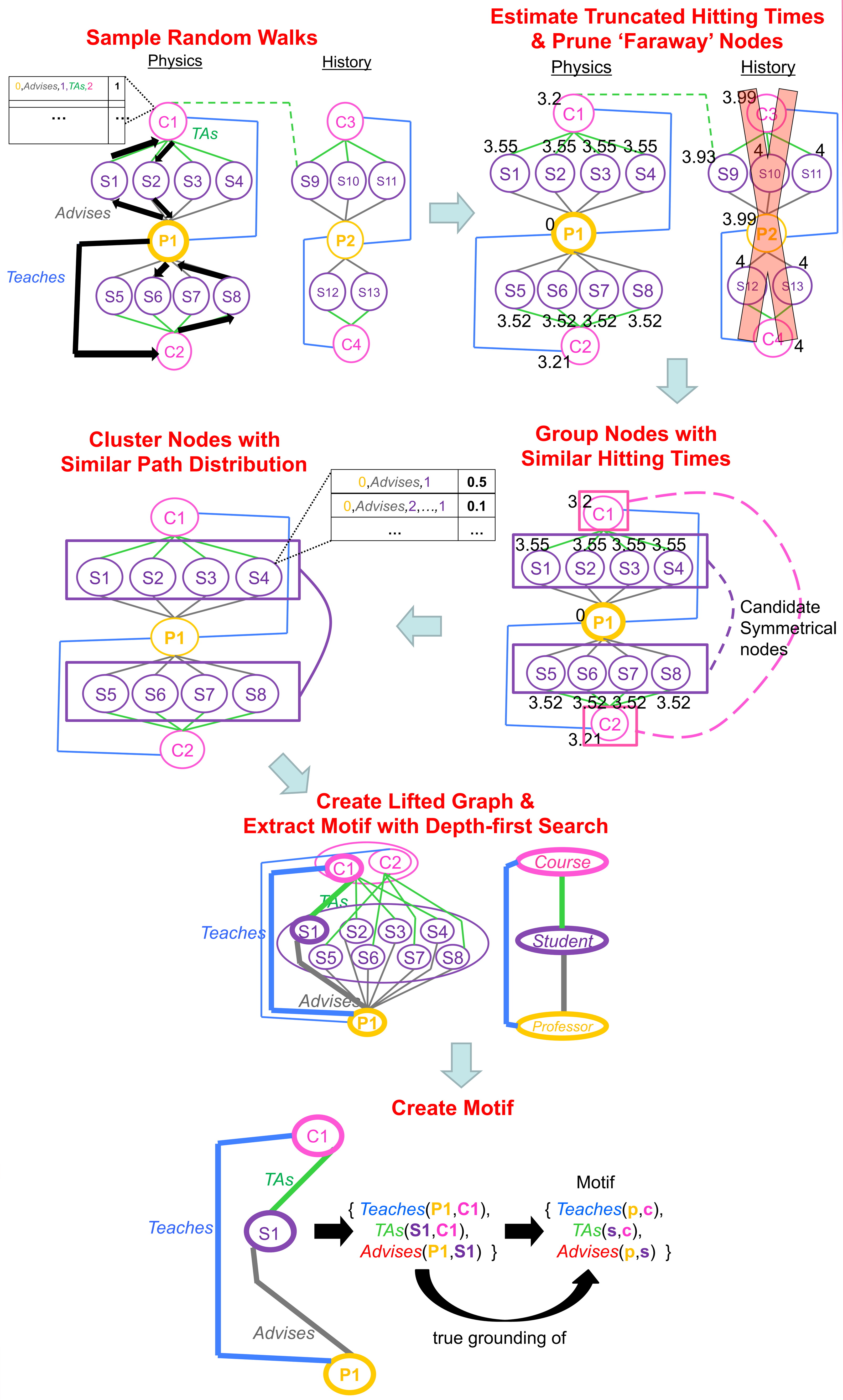
Symmetrical Paths & Nodes

- In a graph, two paths are symmetrical iff the strings created by replacing the nodes with integers indicating the order in which the nodes are visited are identical
- Two nodes *v* and *w* are symmetrical wrt. to a node *s* iff each path from *s* to *v* is symmetrical to some path from *s* to *w* and vice versa
  - Intuition: *v* and *w* are indistinguishable wrt. *s*

Learning Using Structural Motifs (LSM)

- First MLN structure learner that can learn **long clauses**
  - Long clauses capture more complex dependencies than short clauses
  - Typically want to set max. clause length to large value so as not to *a priori* preclude good clauses
- Finds literals that are **densely** connected by arguments
  - Using **random walks & truncated hitting times**
- Clusters constants into high-level concepts
  - Using **symmetrical paths & nodes**
- **Structural Motifs** = a set of literals
  - Defines a set of clauses that can be created from one or more of the literals, i.e., a sub-space of clauses
- Represents relational data as a graph
  - Nodes = constants; edges = true ground atoms





FindPaths

- Trace paths in motifs using variant of depth-first search

CreateMLN

- Conjoin literals in paths found by FindPaths
- Convert conjunction to clauses
- Create new clauses by flipping signs of literals

Advises(p,s) ∧ Teaches(p,c) ∧ TAs(s,c)

¬Advises(p,s) ∨ ¬Teaches(p,c) ∨ ¬TAs(s,c)

¬Advises(p,s) ∨ Teaches(p,c) ∨ TAs(s,c), …

- Score clauses according to pseudo-likelihood
- Retain clause if it does better than all sub-clauses (taken individually)
- Add all retained clauses to MLN
- Trains weights of clauses
- Remove clauses with absolute weight less than threshold

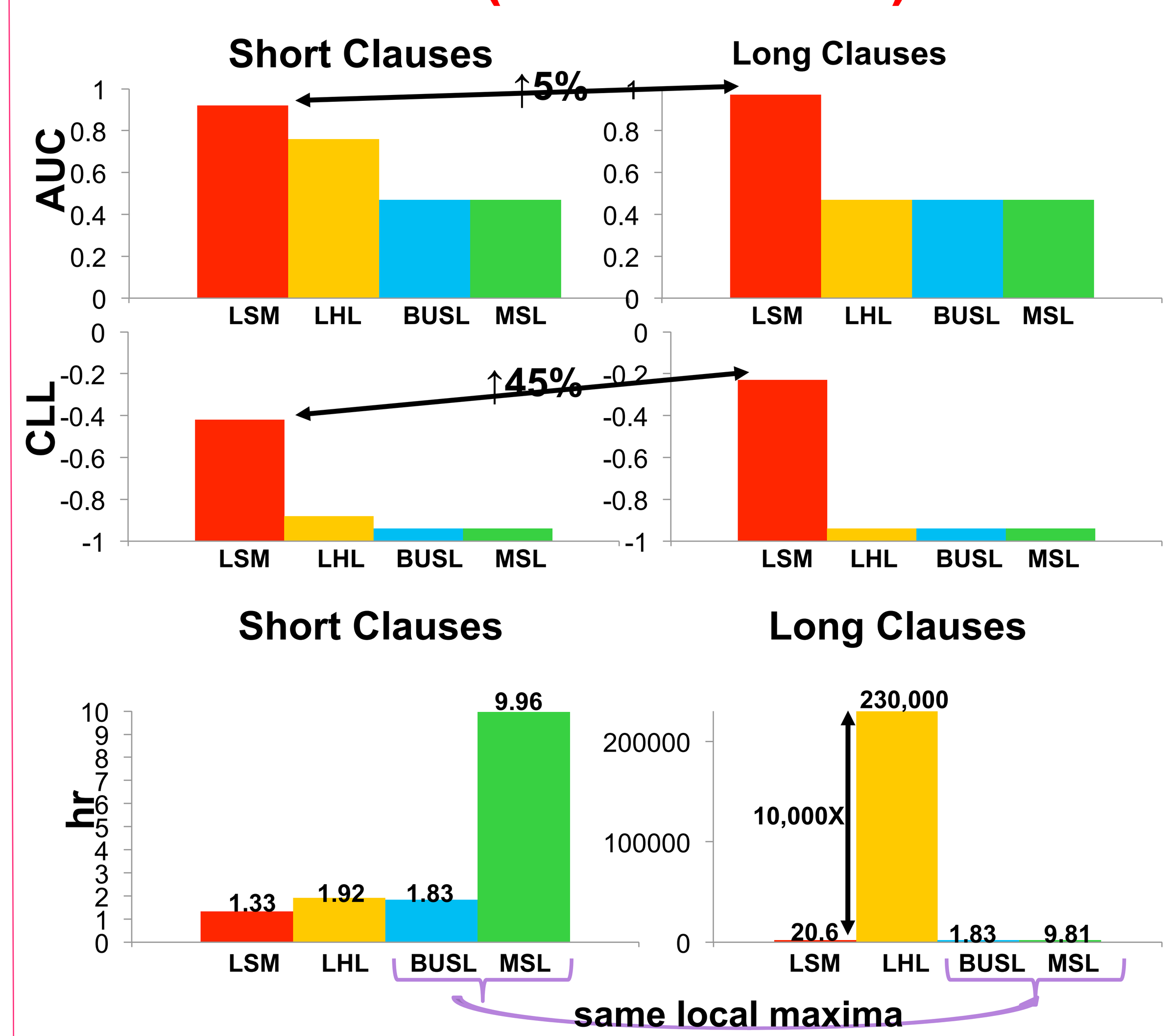
Datasets

- Cora
  - Citations to computer science papers
  - Papers, author, titles, etc., & their relationships
  - 687,422 ground atoms; 42,558 true ones
- Two other publicly-available datasets: IMDB, UW-CSE

Methodology

- Five-fold cross validation
- Inferred prob. true for groundings of each pred.
  - Groundings of all other predicates as evidence
- For **Cora**, inferred **four predicates jointly** too
  - **SameCitation, SameTitle, SameAuthor, SameVenue**
- MCMC to eval test atoms: 10^6 samples or 24 hrs
- Evaluate area under precision-recall curve (**AUC**)
- Evaluate average conditional log-likelihood (**CLL**)
- Compared against state-of-the-art MLN structure learners: LHL, BUSL, MSL
- Two clause lengths per system: short length of **4**, and long length of **10**

Cora (4 Predicates)



Examples of Clauses Learned

VenueOfCit(v,c) ∧ VenueOfCit(v,c') ∧ AuthorOfCit(a,c) ∧ AuthorOfCit(a',c') ∧ SameAuthor(a,a') ∧ TitleOfCit(t,c) ∧ TitleOfCit(t',c') ⇒ SameTitle(t,t')

SameCitation(c,c') ∧ TitleOfCit(t,c) ∧ TitleOfCit(t',c') ∧ HasWordTitle(t,w) ∧ HasWordTitle(t',w) ∧ AuthorOfCit(a,c) ∧ AuthorOfCit(a',c') ∧ SameAuthor(a,a')