Relational Probabilistic Models

David Poole

Department of Computer Science,
University of British Columbia

Work with: `http://minervaintelligence.com`, `https://treatment.com/`

March 27, 2019

Outline

1. Knowledge Graphs
   - Tensor Factorization and Neural Network Models
2. Representation Issues
   - Desiderata
   - How do relational models relate to probabilistic graphical models
3. Unique properties of relational models
   - Learning general knowledge vs learning about a data set
   - Varying Populations
   - What can be observed?
4. Conclusions and Challenges
5. (Exact) Lifted Inference
   - Recursive Conditioning
   - Lifted Recursive Conditioning

Outline

1. Knowledge Graphs
   - Tensor Factorization and Neural Network Models
2. Representation Issues
   - Desiderata
   - How do relational models relate to probabilistic graphical models
3. Unique properties of relational models
   - Learning general knowledge vs learning about a data set
   - Varying Populations
   - What can be observed?
4. Conclusions and Challenges
5. (Exact) Lifted Inference
   - Recursive Conditioning
   - Lifted Recursive Conditioning

Choosing Individuals and Relations in Logic

First-order logical languages allow many different ways of representing facts.

E.g., How to represent: "Pen #7 is red."

Choosing Individuals and Relations in Logic

First-order logical languages allow many different ways of representing facts.

E.g., How to represent: "Pen #7 is red."

- $red(pen_7)$. It's easy to ask "What's red?"
  Can't ask "what is the color of pen_7?"

Choosing Individuals and Relations in Logic

First-order logical languages allow many different ways of representing facts.

E.g., How to represent: "Pen #7 is red."

- $red(pen_7)$. It's easy to ask "What's red?"
  Can't ask "what is the color of pen_7?"
- $color(pen_7, red)$. It's easy to ask "What's red?"
  It's easy to ask "What is the color of pen_7?"
  Can't ask "What property of pen_7 has value red?"

Choosing Individuals and Relations in Logic

First-order logical languages allow many different ways of representing facts.
E.g., How to represent: "Pen #7 is red."

- $red(pen_7)$. It's easy to ask "What's red?"
  Can't ask "what is the color of pen_7?"
- $color(pen_7, red)$. It's easy to ask "What's red?"
  It's easy to ask "What is the color of pen_7?"
  Can't ask "What property of pen_7 has value red?"
- $prop(pen_7, color, red)$. It's easy to ask all these questions.

Choosing Individuals and Relations in Logic

First-order logical languages allow many different ways of representing facts.
E.g., How to represent: "Pen #7 is red."

- $red(pen_7)$. It's easy to ask "What's red?"
  Can't ask "what is the color of pen_7?"
- $color(pen_7, red)$. It's easy to ask "What's red?"
  It's easy to ask "What is the color of pen_7?"
  Can't ask "What property of pen_7 has value red?"
- $prop(pen_7, color, red)$. It's easy to ask all these questions.

With a single relation it can be implicit $\longrightarrow$ triples:
$\langle pen_7, color, red \rangle$.

Universality of *prop*

To represent "a is a parcel"

Universality of *prop*

To represent "a is a parcel"

- $prop(a, type, parcel)$, where *type* is a special property and *parcel* is a class.
- $prop(a, parcel, true)$, where *parcel* is a Boolean property (characteristic function of the class *parcel*).

Triples

- To represent $tutorial("StarAI", nips2017, 1045, hallC)$. "the Star AI tutorial at NIPS 2017 is scheduled at 10:45 in Hall C."

Triples

- To represent $tutorial(\text{"}StarAI\text{"}, nips2017, 1045, hallC)$. "the Star AI tutorial at NIPS 2017 is scheduled at 10:45 in Hall C."
- Let $t123$ name the offering of the tutorial:

$$prop(t123, type, tutorial).$$

$$prop(t123, title, \text{"}StarAI\text{"}).$$

$$prop(t123, event, nips2017).$$

$$prop(t123, time, 1045).$$

$$prop(t123, room, hallC).$$

Triples

- To represent $tutorial(\text{"}StarAI\text{"}, nips2017, 1045, hallC)$. "the Star AI tutorial at NIPS 2017 is scheduled at 10:45 in Hall C."
- Let $t123$ name the offering of the tutorial:

$$prop(t123, type, tutorial).$$
$$prop(t123, title, \text{"}StarAI\text{"}).$$
$$prop(t123, event, nips2017).$$
$$prop(t123, time, 1045).$$
$$prop(t123, room, hallC).$$

- We have reified the booking.
- Reify means: to make into an individual.

Triples

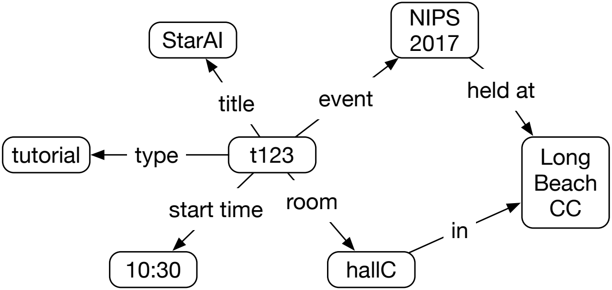
- To represent $tutorial(''StarAI'', nips2017, 1045, hallC)$. "the Star AI tutorial at NIPS 2017 is scheduled at 10:45 in Hall C."
- Let $t123$ name the offering of the tutorial:

$$prop(t123, type, tutorial).$$
$$prop(t123, title, ''StarAI'').$$
$$prop(t123, event, nips2017).$$
$$prop(t123, time, 1045).$$
$$prop(t123, room, hallC).$$

- We have reified the booking.
- Reify means: to make into an individual.
- How can we add extra arguments (e.g., presenters, chair)?

Triples and Knowledge Graphs

- Subject–verb–object
  Individual–property–value
  triples can be depicted as edges in graphs



- A modeller or learner needs to invent (reified) objects.

Triples are universal representations of relations

All relations can be represented in terms of triples:

$$\begin{array}{c|c|c|c|}
\cline{2-4}
 & \ldots & P_j & \ldots \\
\cline{2-4}
 & \ldots & \ldots & \ldots \\
r_i & \ldots & v_{ij} & \ldots \\
 & \ldots & \ldots & \ldots \\
\cline{2-4}
\end{array}$$

can be represented as

Triples are universal representations of relations

All relations can be represented in terms of triples:

$$
\begin{array}{c|c|c|c|}
\cline{2-4}
 & \dots & P_j & \dots \\
\cline{2-4}
 & \dots & \dots & \dots \\
r_i & \dots & v_{ij} & \dots \\
 & \dots & \dots & \dots \\
\cline{2-4}
\end{array}
$$

can be represented as the triple $\langle r_i, P_j, v_{ij} \rangle$.

- r_i becomes a reified individual.

Triples are universal representations of relations

All relations can be represented in terms of triples:

$$
\begin{array}{c|c|c|c|}
\cline{2-4}
 & \dots & P_j & \dots \\
\cline{2-4}
 & \dots & \dots & \dots \\
r_i & \dots & v_{ij} & \dots \\
 & \dots & \dots & \dots \\
\cline{2-4}
\end{array}
$$

can be represented as the triple $\langle r_i, P_j, v_{ij} \rangle$.

- r_i becomes a reified individual.
- Examples of reified individuals: a booking, a marriage, a talk, a lab report, an event, a party, a meeting, a drink

Triples are universal representations of relations

All relations can be represented in terms of triples:

	...	P_j	...
	...	...	...
r_i	...	v_{ij}	...
	...	...	...

can be represented as the triple $\langle r_i, P_j, v_{ij} \rangle$.

- r_i becomes a reified individual.
- Examples of reified individuals: a booking, a marriage, a talk, a lab report, an event, a party, a meeting, a drink
- Challenge for learning: each reified individual has limited data to learn from; (at most) one value for each property.

Triples are universal representations of relations

All relations can be represented in terms of triples:

	...	P_j	...
	...	...	...
r_i	...	v_{ij}	...
	...	...	...

can be represented as the triple $\langle r_i, P_j, v_{ij} \rangle$.

- r_i becomes a reified individual.
- Examples of reified individuals: a booking, a marriage, a talk, a lab report, an event, a party, a meeting, a drink
- Challenge for learning: each reified individual has limited data to learn from; (at most) one value for each property.

$prop(Individual, Property, Value)$ is the only relation needed: $\langle Individual, Property, Value \rangle$ triples, Semantic network, entity relationship model, knowledge graphs, ...

Outline

1. Knowledge Graphs
   - Tensor Factorization and Neural Network Models
2. Representation Issues
   - Desiderata
   - How do relational models relate to probabilistic graphical models
3. Unique properties of relational models
   - Learning general knowledge vs learning about a data set
   - Varying Populations
   - What can be observed?
4. Conclusions and Challenges
5. (Exact) Lifted Inference
   - Recursive Conditioning
   - Lifted Recursive Conditioning

Vector & Tensor Representations of Entities & Relations

- Knowledge graphs (semantic networks) are databases of triples
- We want to estimate $P(\langle h, r, t \rangle)$

Vector & Tensor Representations of Entities & Relations

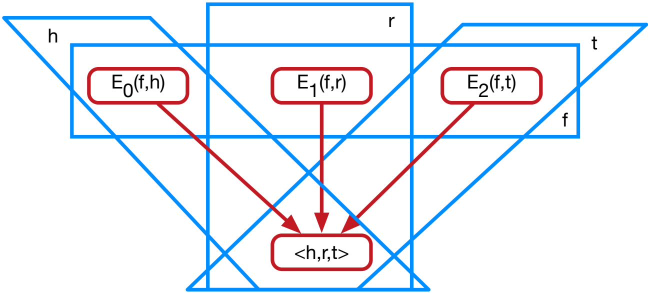
- Knowledge graphs (semantic networks) are databases of triples
- We want to estimate $P(\langle h, r, t\rangle)$
- Polyadic decomposition model (1927): two vector embeddings for each entity, and one for reach relation

$$P(\langle h, r, t\rangle) = sigmoid(\sum_f E_0(f, h) * E_1(f, r) * E_2(f, t))$$

Vector & Tensor Representations of Entities & Relations

- Knowledge graphs (semantic networks) are databases of triples
- We want to estimate $P(\langle h, r, t\rangle)$
- Polyadic decomposition model (1927): two vector embeddings for each entity, and one for reach relation

$$P(\langle h, r, t\rangle) = sigmoid(\sum_f E_0(f, h) * E_1(f, r) * E_2(f, t))$$



Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $\langle p123, \mathit{likes}, m53\rangle$ and $\langle m53, \mathit{directed\_by}, p534\rangle$.

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $\langle p123, \mathit{likes}, m53 \rangle$ and $\langle m53, \mathit{directed\_by}, p534 \rangle$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $\langle p123, \mathit{likes}, m53\rangle$ and $\langle m53, \mathit{directed\_by}, p534\rangle$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.
- DistMult: share same embedding for head and tail.

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $\langle p123, likes, m53\rangle$ and $\langle m53, directed\_by, p534\rangle$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.
- DistMult: share same embedding for head and tail.
  Problem:

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $\langle p123, likes, m53\rangle$ and $\langle m53, directed\_by, p534\rangle$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.
- DistMult: share same embedding for head and tail.
  Problem: can only represent symmetric relations.

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $\langle p123, likes, m53\rangle$ and $\langle m53, directed\_by, p534\rangle$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.
- DistMult: share same embedding for head and tail.
  Problem: can only represent symmetric relations.
- CompleX: the embeddings are complex numbers, tail is the conjugate of the head embedding

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $\langle p123, \mathit{likes}, m53\rangle$ and $\langle m53, \mathit{directed\_by}, p534\rangle$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.
- DistMult: share same embedding for head and tail.
  Problem: can only represent symmetric relations.
- CompleX: the embeddings are complex numbers, tail is the conjugate of the head embedding
- SimpleE: have an embedding for r^{-1} and learn to predict both $\langle h, r, t\rangle$ and $\left\langle t, r^{-1}, h\right\rangle$

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom.

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...
  - learn specific knowledge about who is friends with who; e.g., which particular group of people are generally friends with each other.

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...
  - learn specific knowledge about who is friends with who; e.g., which particular group of people are generally friends with each other.
- The specific knowledge will tend to be more accurate on that population, but doesn't generalize to different populations.
- The general knowledge will tend to transfer better.
- Which is better depends on the goals and how success is measured.

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...
  - learn specific knowledge about who is friends with who; e.g., which particular group of people are generally friends with each other.
- The specific knowledge will tend to be more accurate on that population, but doesn't generalize to different populations.
- The general knowledge will tend to transfer better.
- Which is better depends on the goals and how success is measured.
- Many of the methods try to do both; learn about specific individuals and general knowledge.

Challenges of learning knowledge graphs

- Evaluating predictions when only positive examples are provided
  Consider the following relations:
  - Married to
  - Friend of
  - Knows about
  - Would get along with

Challenges of learning knowledge graphs

- Evaluating predictions when only positive examples are provided
  Consider the following relations:
  - Married to — each person related to 0 or 1 other persons (with a few exceptions)
  - Friend of — each person related to tens or hundreds of others
  - Knows about — each person might know about hundreds or thousands of others. Some people my be known by millions or billions of others.
  - Would get along with — almost everyone gets along with almost everyone else, but with some exceptions.

Challenges of learning knowledge graphs

- Evaluating predictions when only positive examples are provided
  Consider the following relations:
  - Married to — each person related to 0 or 1 other persons (with a few exceptions)
  - Friend of — each person related to tens or hundreds of others
  - Knows about — each person might know about hundreds or thousands of others. Some people my be known by millions or billions of others.
  - Would get along with — almost everyone gets along with almost everyone else, but with some exceptions.
- Often we compare rankings (ordering), but what if the answer is “no”?

Challenges of learning knowledge graphs

- Evaluating predictions when only positive examples are provided
  Consider the following relations:
  - Married to — each person related to 0 or 1 other persons (with a few exceptions)
  - Friend of — each person related to tens or hundreds of others
  - Knows about — each person might know about hundreds or thousands of others. Some people my be known by millions or billions of others.
  - Would get along with — almost everyone gets along with almost everyone else, but with some exceptions.
- Often we compare rankings (ordering), but what if the answer is "no"?
- Difficult to learn about reified entities.

Predicting Properties

- Tensor factorization models work well for predicting relations, but not for predicting properties.
  Tensor factorization relies on lower-dimensional representations, and there isn't one for properties.

Predicting Properties

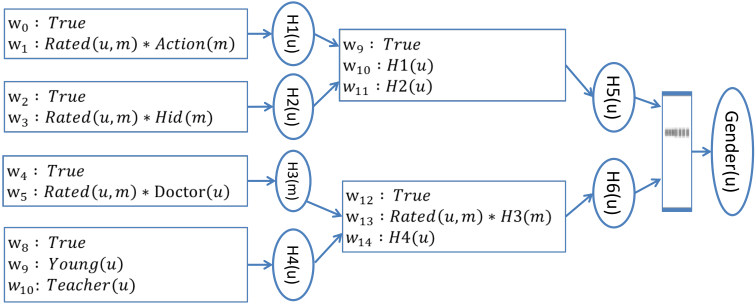
- Tensor factorization models work well for predicting relations, but not for predicting properties.
  Tensor factorization relies on lower-dimensional representations, and there isn't one for properties.
- Imagine trying to predict $age(P)$, the age of person P, and $rating(P, M)$ the rating of person P on movie M.
  One of the embeddings of each person can just memorize the age — no generalization!
  — there are too many parameters

Predicting Properties

- Tensor factorization models work well for predicting relations, but not for predicting properties.
  Tensor factorization relies on lower-dimensional representations, and there isn't one for properties.
- Imagine trying to predict $age(P)$, the age of person P, and $rating(P, M)$ the rating of person P on movie M.
  One of the embeddings of each person can just memorize the age — no generalization!
  — there are too many parameters
- We can build relational neural networks to solve this

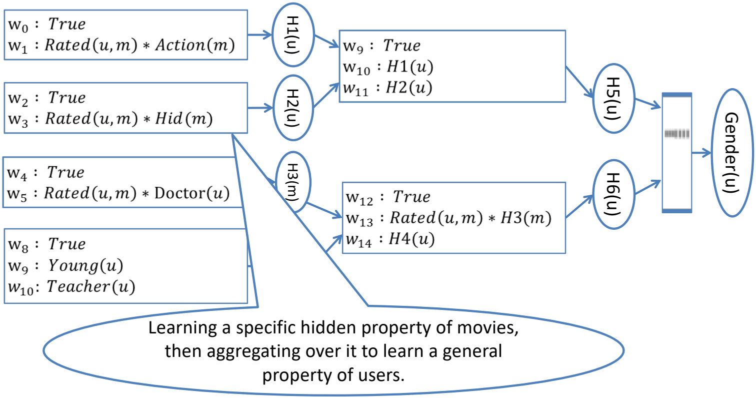
[Kazemi & Poole, AAAI 2017]

Relational Neural Nets (RelNNs)



[Kazemi & Poole AAAI 2017]

Relational Neural Nets (RelNNs)



[Kazemi & Poole AAAI 2017]

Outline

Desiderata for a Representation

- Expressiveness:
  Is it expressive enough to solve problem at hand?

Desiderata for a Representation

- Expressiveness:
  Is it expressive enough to solve problem at hand?
- Efficient Inference:
  Is it efficient in the worst case or average case?
  Can it exploit structure (e.g., independencies and symmetries)

Desiderata for a Representation

- Expressiveness:
  Is it expressive enough to solve problem at hand?
- Efficient Inference:
  Is it efficient in the worst case or average case?
  Can it exploit structure (e.g., independencies and symmetries)
- Understandability or explainability:
  Can people understand the model?
  Can a particular prediction be explained?

Desiderata for a Representation

- Expressiveness:
  Is it expressive enough to solve problem at hand?
- Efficient Inference:
  Is it efficient in the worst case or average case?
  Can it exploit structure (e.g., independencies and symmetries)
- Understandability or explainability:
  Can people understand the model?
  Can a particular prediction be explained?
- Learnability: Can it be learned from:
  - heterogenous data
  - prior knowledge

Desiderata for a Representation

- Expressiveness:
  Is it expressive enough to solve problem at hand?
- Efficient Inference:
  Is it efficient in the worst case or average case?
  Can it exploit structure (e.g., independencies and symmetries)
- Understandability or explainability:
  Can people understand the model?
  Can a particular prediction be explained?
- Learnability: Can it be learned from:
  - heterogenous data
  - prior knowledge
- Modularity:
  Can independently developed parts be combined to form larger model?
  Can a larger model be decomposed into smaller parts?

Outline

1. Knowledge Graphs
   - Tensor Factorization and Neural Network Models
2. Representation Issues
   - Desiderata
   - How do relational models relate to probabilistic graphical models
3. Unique properties of relational models
   - Learning general knowledge vs learning about a data set
   - Varying Populations
   - What can be observed?
4. Conclusions and Challenges
5. (Exact) Lifted Inference
   - Recursive Conditioning
   - Lifted Recursive Conditioning

Directed vs Undirected Probabilistic Graphical Models

- Undirected models (Markov networks, factor graphs) represent probability distributions in terms of factors.
  - a factor is a non-negative function of a set of variables
  - variables in a factor are neighbours of each other
  - each variable in independent of its non-neighbours given its neighbours.

Directed vs Undirected Probabilistic Graphical Models

- Undirected models (Markov networks, factor graphs) represent probability distributions in terms of factors.
  - a factor is a non-negative function of a set of variables
  - variables in a factor are neighbours of each other
  - each variable in independent of its non-neighbours given its neighbours.
- In directed models, factors represent conditional probabilities:
  - each variable in independent of its non-descendents given its parents.

Directed vs Undirected Probabilistic Graphical Models

- Undirected models (Markov networks, factor graphs) represent probability distributions in terms of factors.
  - a factor is a non-negative function of a set of variables
  - variables in a factor are neighbours of each other
  - each variable in independent of its non-neighbours given its neighbours.
- In directed models, factors represent conditional probabilities:
  - each variable in independent of its non-descendents given its parents.
- $\{directed\_models\} \subset \{undirected\_models\}$

Directed vs Undirected Probabilistic Graphical Models

- Undirected models (Markov networks, factor graphs) represent probability distributions in terms of factors.
  - a factor is a non-negative function of a set of variables
  - variables in a factor are neighbours of each other
  - each variable in independent of its non-neighbours given its neighbours.
- In directed models, factors represent conditional probabilities:
  - each variable in independent of its non-descendents given its parents.
- $\{directed\_models\} \subset \{undirected\_models\}$
- Directed (and undirected) models are universal: can represent any probability distribution over a finite set of variables

Directed vs Undirected Probabilistic Graphical Models

- Undirected models (Markov networks, factor graphs) represent probability distributions in terms of factors.
    - a factor is a non-negative function of a set of variables
    - variables in a factor are neighbours of each other
    - each variable in independent of its non-neighbours given its neighbours.
- In directed models, factors represent conditional probabilities:
    - each variable in independent of its non-descendents given its parents.
- $\{directed\_models\} \subset \{undirected\_models\}$
- Directed (and undirected) models are universal: can represent any probability distribution over a finite set of variables
- Algorithms developed for undirected models work for both.

Directed vs Undirected Probabilistic Graphical Models

- Undirected models (Markov networks, factor graphs) represent probability distributions in terms of factors.
  - a factor is a non-negative function of a set of variables
  - variables in a factor are neighbours of each other
  - each variable in independent of its non-neighbours given its neighbours.
- In directed models, factors represent conditional probabilities:
  - each variable in independent of its non-descendents given its parents.
- $\{directed\_models\} \subset \{undirected\_models\}$
- Directed (and undirected) models are universal: can represent any probability distribution over a finite set of variables
- Algorithms developed for undirected models work for both.
- That does not mean that representations for undirected models can represent directed models.

Example

Weighted formulae about a social situation:

$$-5 : \mathit{funFor}(X)$$

$$10 : \mathit{funFor}(X) \wedge \mathit{knows}(X, Y) \wedge \mathit{social}(Y)$$

Example

Weighted formulae about a social situation:

$$-5 : \mathit{funFor}(X)$$
$$10 : \mathit{funFor}(X) \land \mathit{knows}(X, Y) \land \mathit{social}(Y)$$

If Π includes observations for all $\mathit{knows}(X, Y)$ and $\mathit{social}(Y)$:

$$P(\mathit{funFor}(X) \mid \Pi) = \mathit{sigmoid}(-5 + 10 n_T)$$

n_T is the number of individuals Y for which $\mathit{knows}(X, Y) \land \mathit{social}(Y)$ is *True* in Π.

$$\mathit{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

Example

Weighted formulae about a social situation:

$$-5 : \mathit{funFor}(X)$$

$$10 : \mathit{funFor}(X) \wedge \mathit{knows}(X, Y) \wedge \mathit{social}(Y)$$

If Π includes observations for all $\mathit{knows}(X, Y)$ and $\mathit{social}(Y)$:

$$P(\mathit{funFor}(X) \mid \Pi) = \mathit{sigmoid}(-5 + 10 n_T)$$

n_T is the number of individuals Y for which $\mathit{knows}(X, Y) \wedge \mathit{social}(Y)$ is *True* in Π.

$$\mathit{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

- Using wighted formulae to define conditional probabilities $\rightarrow$ relational logistic regression (RLR).
- Using wighted formulae to define distributiuons $\rightarrow$ Markov logic networks (MLNs).

Abstract Example

$\alpha_0 : q$
$\alpha_1 : q \wedge \neg r(x)$
$\alpha_2 : q \wedge r(x)$
$\alpha_3 : r(x)$

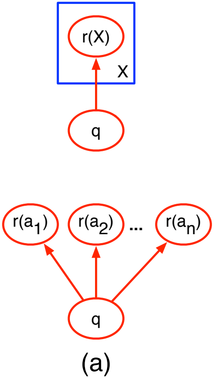
If $r(x)$ for every individual x is observed:

$$P(q \mid obs) = sigmoid(\alpha_0 + n_F\alpha_1 + n_T\alpha_2)$$

n_T is number of individuals for which $r(x)$ is true
n_F is number of individuals for which $r(x)$ is false

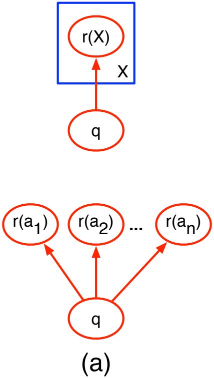
$$sigmoid(x) = \frac{1}{1 + e^{-x}}$$

Three Elementary Models



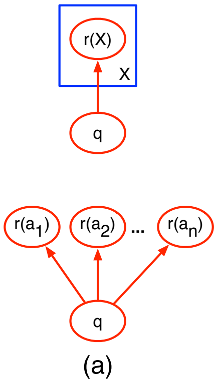
(a)

Three Elementary Models



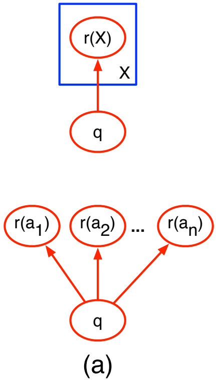
(a) Naïve Bayes
(b)

Three Elementary Models



(a) Naïve Bayes
(b) (Relational) Logistic Regression
(c)

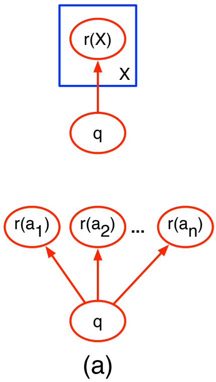
Three Elementary Models



(a) Naïve Bayes
(b) (Relational) Logistic Regression
(c) Markov (Logic) network

} described by unary and pairwise factors

Three Elementary Models

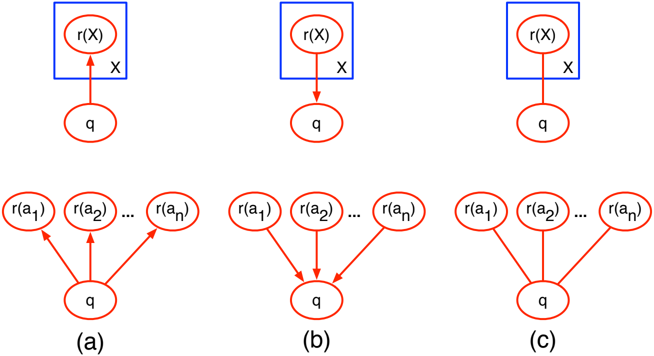


(a) Naïve Bayes
(b) (Relational) Logistic Regression
(c) Markov (Logic) network

} described by unary and pairwise factors

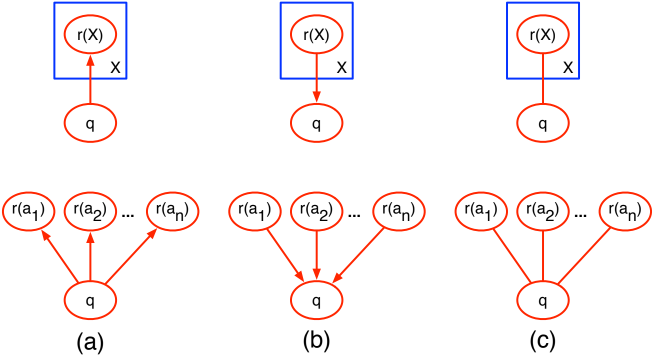
— They are identical models when all r's are observed.

Independence Assumptions



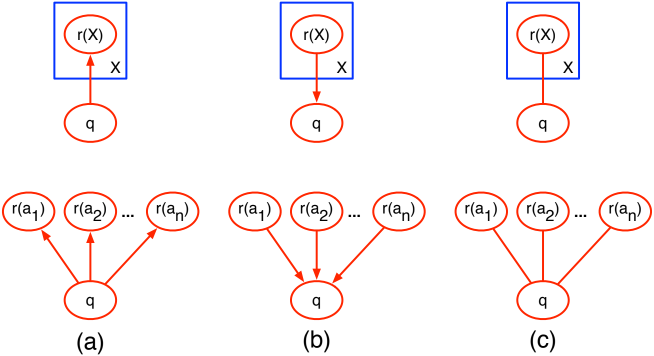
- Naïve Bayes (a) and Markov network (c): $r(a_i)$ and $r(a_j)$
  - are independent

Independence Assumptions



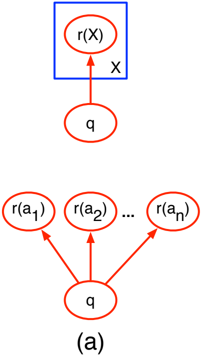
- Naïve Bayes (a) and Markov network (c): $r(a_i)$ and $r(a_j)$
  - are independent given Q
  - are dependent

Independence Assumptions



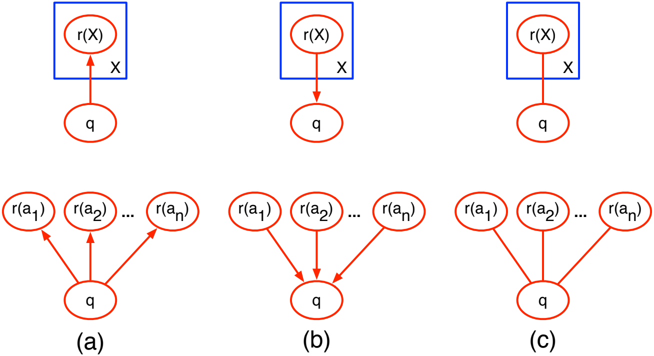
- Naïve Bayes (a) and Markov network (c): $r(a_i)$ and $r(a_j)$
  - are independent given Q
  - are dependent not given Q.
- Directed model with aggregation (b): $r(a_i)$ and $r(a_j)$
  - are dependent

Independence Assumptions



- Naïve Bayes (a) and Markov network (c): $r(a_i)$ and $r(a_j)$
  - are independent given Q
  - are dependent not given Q.
- Directed model with aggregation (b): $r(a_i)$ and $r(a_j)$
  - are dependent given Q,
  - are independent

Independence Assumptions



- Naïve Bayes (a) and Markov network (c): $r(a_i)$ and $r(a_j)$
  - are independent given Q
  - are dependent not given Q.
- Directed model with aggregation (b): $r(a_i)$ and $r(a_j)$
  - are dependent given Q,
  - are independent not given Q.

Modularity

- Directed models are inherently modular.
  $P(q \mid r(X))$ is defined so that distribution over $r(a_1) \ldots r(a_n)$ is not affected when q is summed out (and not observed)

Modularity

- Directed models are inherently modular.
  $P(q \mid r(X))$ is defined so that distribution over $r(a_1) \dots r(a_n)$ is not affected when q is summed out (and not observed)
- MLNs are provably not modular: If there is a distribution over $r(a_1) \dots r(a_n)$ (e.g., they are independent),
  $P(q \mid r(X))$ cannot be defined in an MLN so that
  - q depends on the r's ($P(q \mid r(X)) \neq P(q)$) and
  - if q is summed out, the distribution over $r(a_1) \dots r(a_n)$ is not changed.

Modularity

- Directed models are inherently modular.
  $P(q \mid r(X))$ is defined so that distribution over $r(a_1) \dots r(a_n)$ is not affected when q is summed out (and not observed)
- MLNs are provably not modular: If there is a distribution over $r(a_1) \dots r(a_n)$ (e.g., they are independent),
  $P(q \mid r(X))$ cannot be defined in an MLN so that
  - q depends on the r's ($P(q \mid r(X)) \neq P(q)$) and
  - if q is summed out, the distribution over $r(a_1) \dots r(a_n)$ is not changed.
  - Why? requires factors on arbitrary subsets of $r(a_1) \dots r(a_k)$
    — pairwise (or 3-wise or ...) interactions are not adequate
    — can't marry the parents

Cyclic Models

Whether people smoke depends on whether their friends smoke.

- MLN:

$$w : smokes(X) \leftarrow friends(X, Y) \land smokes(Y)$$

Cyclic Models

Whether people smoke depends on whether their friends smoke.

- MLN:

$$w : smokes(X) \leftarrow friends(X, Y) \wedge smokes(Y)$$

(where $\leftarrow$ is material implication) is equivalent to

$$\begin{aligned} w :& true(X) \wedge true(Y) \\ -w :& \neg smokes(X) \wedge friends(X, Y) \wedge smokes(Y) \end{aligned}$$

Cyclic Models

Whether people smoke depends on whether their friends smoke.

- MLN:

$$w : smokes(X) \leftarrow friends(X, Y) \wedge smokes(Y)$$

(where $\leftarrow$ is material implication) is equivalent to

$$\begin{aligned} w &: true(X) \wedge true(Y) \\ -w &: \neg smokes(X) \wedge friends(X, Y) \wedge smokes(Y) \end{aligned}$$

- ICL/Problog

$$w : smokes(X) \leftarrow \quad friends(X, Y) \wedge smokes(Y)$$

Cyclic Models

Whether people smoke depends on whether their friends smoke.

- MLN:

$$w : smokes(X) \leftarrow friends(X,Y) \land smokes(Y)$$

(where $\leftarrow$ is material implication) is equivalent to

$$\begin{aligned} & w : true(X) \land true(Y) \\ -& w : \neg smokes(X) \land friends(X,Y) \land smokes(Y) \end{aligned}$$

- ICL/Problog

$$w : smokes(X) \leftarrow \exists Y \; friends(X,Y) \land smokes(Y)$$

Cyclic Models

Whether people smoke depends on whether their friends smoke.

- MLN:

$$w : smokes(X) \leftarrow friends(X, Y) \wedge smokes(Y)$$

(where $\leftarrow$ is material implication) is equivalent to

$$\begin{aligned} w &: true(X) \wedge true(Y) \\ -w &: \neg smokes(X) \wedge friends(X, Y) \wedge smokes(Y) \end{aligned}$$

- ICL/Problog

$$w : smokes(X) \leftarrow \exists Y \ friends(X, Y) \wedge smokes(Y)$$

  - Probability of smokes goes up as the number of friends increases!

Cyclic Models

Whether people smoke depends on whether their friends smoke.

- MLN:

$$w : smokes(X) \leftarrow friends(X, Y) \wedge smokes(Y)$$

(where $\leftarrow$ is material implication) is equivalent to

$$w : true(X) \wedge true(Y)$$

$$-w : \neg smokes(X) \wedge friends(X, Y) \wedge smokes(Y)$$

- ICL/Problog

$$w : smokes(X) \leftarrow \exists Y \; friends(X, Y) \wedge smokes(Y)$$

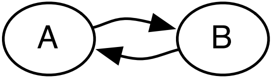
  - Probability of smokes goes up as the number of friends increases!
  - ICL/Problog cannot represent negative effects: someone is less likely to smoke if their friends smoke (without there being a non-zero probability of logical inconsistency)

Cyclic Directed Models

- Make model acyclic, by totally ordering variables.
  Destroys exchangeability. Symmetries are not preserved.

Cyclic Directed Models

- Make model acyclic, by totally ordering variables. Destroys exchangeability. Symmetries are not preserved.
- (Relational) dependency networks: directed model,



  - $P(A, B)$ has 3 degrees of freedom,
  - $P(A \mid B), P(B \mid A)$, uses 4 numbers; typically inconsistent.
  - resulting distribution means stationary (equilibrium) distribution of Markov chain.

Outline

1. Knowledge Graphs
   - Tensor Factorization and Neural Network Models
2. Representation Issues
   - Desiderata
   - How do relational models relate to probabilistic graphical models
3. Unique properties of relational models
   - Learning general knowledge vs learning about a data set
   - Varying Populations
   - What can be observed?
4. Conclusions and Challenges
5. (Exact) Lifted Inference
   - Recursive Conditioning
   - Lifted Recursive Conditioning

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom.

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...
  - learn specific knowledge about who is friends with who; e.g., which particular group of people are generally friends with each other.

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...
  - learn specific knowledge about who is friends with who; e.g., which particular group of people are generally friends with each other.
- The specific knowledge will tend to be more accurate on that population, but doesn't generalize to different populations.
- The general knowledge will tend to transfer better.
- Which is better depends on the goals and how success is measured.

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...
  - learn specific knowledge about who is friends with who; e.g., which particular group of people are generally friends with each other.
- The specific knowledge will tend to be more accurate on that population, but doesn't generalize to different populations.
- The general knowledge will tend to transfer better.
- Which is better depends on the goals and how success is measured.
- Many of the methods try to do both; learn about specific individuals and general knowledge.

Canonical polyadic tensor factorization model

- Tensor factorixation models — which learn vectors for individuals — tend to not learn generalized knowledge but about the particular individuals

Canonical polyadic tensor factorization model

- Tensor factorixation models — which learn vectors for individuals — tend to not learn generalized knowledge but about the particular individuals
- Lifted graphical models (MLNs, RLR, Problog) learn general knowledge through training weights (and structure) and specific knowledge through conditioning.

Canonical polyadic tensor factorization model

- Tensor factorixation models — which learn vectors for individuals — tend to not learn generalized knowledge but about the particular individuals
- Lifted graphical models (MLNs, RLR, Problog) learn general knowledge through training weights (and structure) and specific knowledge through conditioning.
- Still open research problem

Outline

What happens as Population size n Changes: Simplest case

$\alpha_0 : q$
$\alpha_1 : q \wedge \neg r(x)$
$\alpha_2 : q \wedge r(x)$
$\alpha_3 : r(x)$

Weighted formulae define distribution:

$$P_{MLN}(q \mid n) = sigmoid(\ \alpha_0 + n \log(e^{\alpha_2} + e^{\alpha_1 - \alpha_3})\ )$$

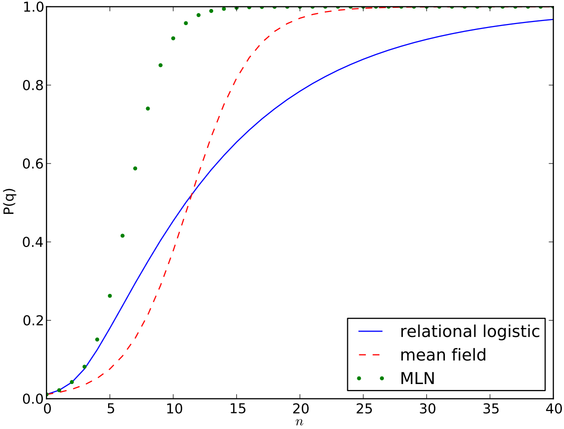
Weighted formulae define conditionals:

$$P_{RLR}(q \mid n) = \sum_{i=0}^{n} \binom{n}{i} sigmoid(\alpha_0 + i\alpha_1 + (n-i)\alpha_2)(1-p_r)^i p_r^{n-i}$$

Mean-field approximation:

$$P_{MF}(q \mid n) = sigmoid(\alpha_0 + np_r\alpha_1 + n(1-p_r)\alpha_2)$$

Population Growth: $P(q \mid n)$



All: $P(q \mid n) \rightarrow$ 0 or 1 as $n \rightarrow \infty$

Challenges of varying populations

- Example: The Movielens 100k dataset contains data about $rated(P, M, R, T)$ meaning person P gave movie M a rating of R at time T.
  Plus user demographic and movie information.
- Number of ratings per user is between 20 (arbitrary threshold) and 737; average of 106
- Number of ratings per movie is between 1 and 583; average of 59

Challenges of varying populations

- Example: The Movielens 100k dataset contains data about $rated(P, M, R, T)$ meaning person P gave movie M a rating of R at time T.
  Plus user demographic and movie information.
- Number of ratings per user is between 20 (arbitrary threshold) and 737; average of 106
- Number of ratings per movie is between 1 and 583; average of 59
- Predicting age from ratings is difficult:
  - With some high counts we can't assume movies produce independent evidence

Challenges of varying populations

- Example: The Movielens 100k dataset contains data about $rated(P, M, R, T)$ meaning person P gave movie M a rating of R at time T.

  Plus user demographic and movie information.
- Number of ratings per user is between 20 (arbitrary threshold) and 737; average of 106
- Number of ratings per movie is between 1 and 583; average of 59
- Predicting age from ratings is difficult:
  - With some high counts we can't assume movies produce independent evidence
  - With many low counts we need to regularize (and can't measure dependence)

Challenges of varying populations

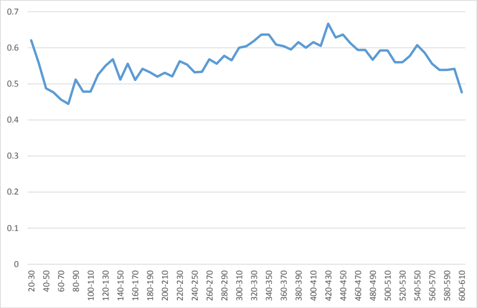
- Example: The Movielens 100k dataset contains data about $rated(P, M, R, T)$ meaning person P gave movie M a rating of R at time T.
  Plus user demographic and movie information.
- Number of ratings per user is between 20 (arbitrary threshold) and 737; average of 106
- Number of ratings per movie is between 1 and 583; average of 59
- Predicting age from ratings is difficult:
  - With some high counts we can't assume movies produce independent evidence
  - With many low counts we need to regularize (and can't measure dependence)
  - There is 100 times as much ratings information as age information

Challenges of varying populations

- Example: The Movielens 100k dataset contains data about *rated*(P, M, R, T) meaning person P gave movie M a rating of R at time T.

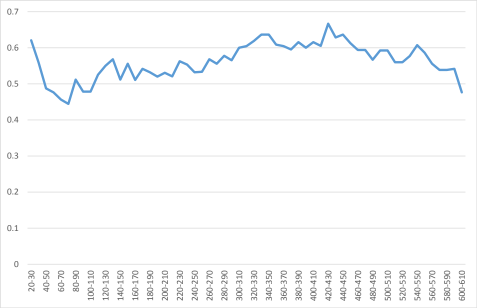
  Plus user demographic and movie information.
- Number of ratings per user is between 20 (arbitrary threshold) and 737; average of 106
- Number of ratings per movie is between 1 and 583; average of 59
- Predicting age from ratings is difficult:
  - With some high counts we can't assume movies produce independent evidence
  - With many low counts we need to regularize (and can't measure dependence)
  - There is 100 times as much ratings information as age information
- Bigger datasets have even more variability.

Real Data



Observed $P(25 < Age(u) < 45 \mid n)$, where n is number of movies watched from the Movielens dataset.

Real Data



Observed $P(25 < Age(u) < 45 \mid n)$, where n is number of movies watched from the Movielens dataset.
Dont use:

$$w : middle\_age(U) \leftarrow rated(U, M) \wedge foo(M)$$

then $P(middle\_age(U)) \rightarrow$ 0 or 1 as number of movies increases.

Example of polynomial dependence of population

$\alpha_0 : q$

$\alpha_1 : q \wedge true(X)$

$\alpha_2 : q \wedge r(X)$

$\alpha_3 : true(X)$

$\alpha_4 : r(X)$

$\alpha_5 : q \wedge true(X) \wedge true(Y)$

$\alpha_6 : q \wedge r(X) \wedge true(Y)$

$\alpha_7 : q \wedge r(X) \wedge r(Y)$

Example of polynomial dependence of population

$$
\begin{aligned}
\alpha_0 &: q \\
\alpha_1 &: q \wedge true(X) \\
\alpha_2 &: q \wedge r(X) \\
\alpha_3 &: true(X) \\
\alpha_4 &: r(X) \\
\alpha_5 &: q \wedge true(X) \wedge true(Y) \\
\alpha_6 &: q \wedge r(X) \wedge true(Y) \\
\alpha_7 &: q \wedge r(X) \wedge r(Y)
\end{aligned}
$$

In RLR and in MLN, if all $r(a_i)$ are observed:

Example of polynomial dependence of population

$\alpha_0 : q$
$\alpha_1 : q \wedge true(X)$
$\alpha_2 : q \wedge r(X)$
$\alpha_3 : true(X)$
$\alpha_4 : r(X)$
$\alpha_5 : q \wedge true(X) \wedge true(Y)$
$\alpha_6 : q \wedge r(X) \wedge true(Y)$
$\alpha_7 : q \wedge r(X) \wedge r(Y)$

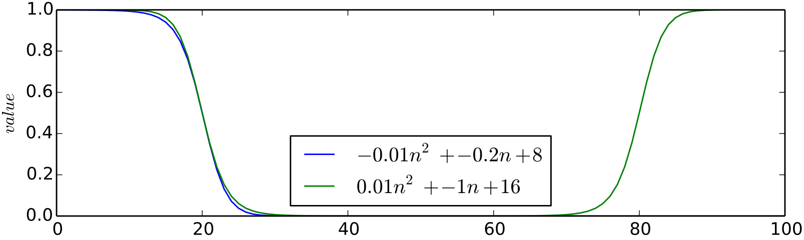
In RLR and in MLN, if all $r(a_i)$ are observed:

$$P(q \mid obs) = sigmoid(\alpha_0 + n\alpha_1 + n_1\alpha_2 + n^2\alpha_5 + n_1 n\alpha_6 + n_1^2\alpha_7)$$

$r(X)$ is true for n_1 individuals out of a population of n.

Danger of fitting to data without understanding the model

- RLR can fit sigmoid of any polynomial.
- Consider sigmoid of a polynomial of degree 2:



Outline

Observation Protocols

- Example: we want to predict the probability that Sam will like a apartment.
- Observation: there is a pink bedroom.

Observation Protocols

- Example: we want to predict the probability that Sam will like a apartment.
- Observation: there is a pink bedroom.

The protocol of how the observation was obtained matters:

- Was it a pink sensor that reported what was pink?

Observation Protocols

- Example: we want to predict the probability that Sam will like a apartment.
- Observation: there is a pink bedroom.

The protocol of how the observation was obtained matters:

- Was it a pink sensor that reported what was pink?
- Was it reporting the first thing it observed?

Observation Protocols

- Example: we want to predict the probability that Sam will like a apartment.
- Observation: there is a pink bedroom.

The protocol of how the observation was obtained matters:

- Was it a pink sensor that reported what was pink?
- Was it reporting the first thing it observed?
- Was it reporting the most unusual feature of the apartment?

Observation Protocols

- Example: we want to predict the probability that Sam will like a apartment.
- Observation: there is a pink bedroom.

The protocol of how the observation was obtained matters:

- Was it a pink sensor that reported what was pink?
- Was it reporting the first thing it observed?
- Was it reporting the most unusual feature of the apartment?
- Was it telling us the most positive aspect of the apartment?

Observation Protocols

- Example: we want to predict the probability that Sam will like a apartment.
- Observation: there is a pink bedroom.

The protocol of how the observation was obtained matters:

- Was it a pink sensor that reported what was pink?
- Was it reporting the first thing it observed?
- Was it reporting the most unusual feature of the apartment?
- Was it telling us the most positive aspect of the apartment?
- Did it know that Sam was interested in whether there was a pink bedroom?

Observation Protocols

- Example: we want to predict the probability that Sam will like a apartment.
- Observation: there is a pink bedroom.

The protocol of how the observation was obtained matters:

- Was it a pink sensor that reported what was pink?
- Was it reporting the first thing it observed?
- Was it reporting the most unusual feature of the apartment?
- Was it telling us the most positive aspect of the apartment?
- Did it know that Sam was interested in whether there was a pink bedroom?
- Was it reporting the colour of each bedroom?

Observation Protocols

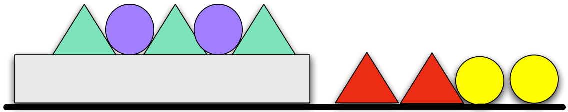
- Example: we want to predict the probability that Sam will like a apartment.
- Observation: there is a pink bedroom.

The protocol of how the observation was obtained matters:

- Was it a pink sensor that reported what was pink?
- Was it reporting the first thing it observed?
- Was it reporting the most unusual feature of the apartment?
- Was it telling us the most positive aspect of the apartment?
- Did it know that Sam was interested in whether there was a pink bedroom?
- Was it reporting the colour of each bedroom?

There are unboundedly many possible relations in a real-world object such as a house.

Observation Protocols

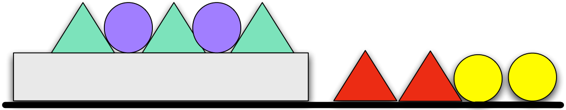


Observe a triangle and a circle touching. What is the probability the triangle is green?

$$P(green(x) \\ |triangle(x) \wedge \exists y \ circle(y) \wedge touching(x, y))$$

The answer depends on how the x and y were chosen!

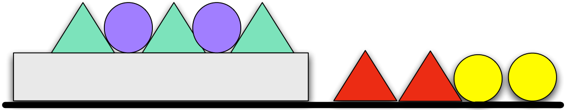
Protocol for Observing



$$P(green(x) \mid triangle(x) \wedge \exists y\ circle(y) \wedge touching(x, y))$$

$select(x)$	$select(y)$	$select(x, y)$
$select(y)$	$select(x)$	
$3/4$	$2/3$	$4/5$

Protocol for Observing



$$P(green(x) \mid triangle(x) \wedge \exists y \ circle(y) \wedge touching(x, y))$$

$select(x)$	$select(y)$	$select(x, y)$
$select(y)$	$select(x)$	
$3/4$	$2/3$	$4/5$

A logical formula does not provide enough information to determine the probabilities.

Data

Real data is messy!

- Multiple levels of abstraction
- Multiple levels of detail
- Sometimes observations are abstract and lifted
  e.g., "3 people out of 300 in the audience asked a question".
- Uses the vocabulary from many ontologies
- Rich meta-data:
  - Who collected each datum? (identity and credentials)
  - Who transcribed the information?
  - What was the protocol used to collect the data? (Chosen at random or chosen because interesting?)
  - What were the controls — what was manipulated, when?
  - What sensors were used? What is their reliability and operating range?
  - What is the provenance of the data; what was done to it when?
- Errors, forgeries, . . .

Outline

1. Knowledge Graphs
   - Tensor Factorization and Neural Network Models
2. Representation Issues
   - Desiderata
   - How do relational models relate to probabilistic graphical models
3. Unique properties of relational models
   - Learning general knowledge vs learning about a data set
   - Varying Populations
   - What can be observed?
4. Conclusions and Challenges
5. (Exact) Lifted Inference
   - Recursive Conditioning
   - Lifted Recursive Conditioning

Take Home

- Exchangeability and dependence on population size distinguish relational models from non-relational models

Take Home

- Exchangeability and dependence on population size distinguish relational models from non-relational models
- Relational models are different from normal graphical models

Take Home

- Exchangeability and dependence on population size distinguish relational models from non-relational models
- Relational models are different from normal graphical models

Challenges that relational models tackle:

- Heterogeneity: information about individuals varies greatly in kind and amount

Take Home

- Exchangeability and dependence on population size distinguish relational models from non-relational models
- Relational models are different from normal graphical models

Challenges that relational models tackle:

- Heterogeneity: information about individuals varies greatly in kind and amount
- Representations should
  - let people state their prior knowledge,
  - let them understand what they stated, and
  - let them understand the posterior models (given evidence).

Take Home

- Exchangeability and dependence on population size distinguish relational models from non-relational models
- Relational models are different from normal graphical models

Challenges that relational models tackle:

- Heterogeneity: information about individuals varies greatly in kind and amount
- Representations should
  - let people state their prior knowledge,
  - let them understand what they stated, and
  - let them understand the posterior models (given evidence).
- Learn general knowledge as well as about particular individuals

Take Home

- Exchangeability and dependence on population size distinguish relational models from non-relational models
- Relational models are different from normal graphical models

Challenges that relational models tackle:

- Heterogeneity: information about individuals varies greatly in kind and amount
- Representations should
  - let people state their prior knowledge,
  - let them understand what they stated, and
  - let them understand the posterior models (given evidence).
- Learn general knowledge as well as about particular individuals
- Use the meta-data of how data was collected
- Model protocol used to generate the observations
- Also model what is not observed (e.g., because it was redundant information, unimportant, false or unknown)

1. Representations: Problog & MLNs
2. Representation Issues
3. **(Exact) Lifted Inference**
4. Lifted Approximate Inference and Optimization
5. Learning
6. Applications

(Exact) Lifted Inference

Outline

1. Knowledge Graphs
   - Tensor Factorization and Neural Network Models
2. Representation Issues
   - Desiderata
   - How do relational models relate to probabilistic graphical models
3. Unique properties of relational models
   - Learning general knowledge vs learning about a data set
   - Varying Populations
   - What can be observed?
4. Conclusions and Challenges
5. (Exact) Lifted Inference
   - Recursive Conditioning
   - Lifted Recursive Conditioning

Why Exact Inference?

Why do we care about exact inference?

- Gold standard

Why Exact Inference?

Why do we care about exact inference?

- Gold standard
- Size of problems amenable to exact inference is growing

Why Exact Inference?

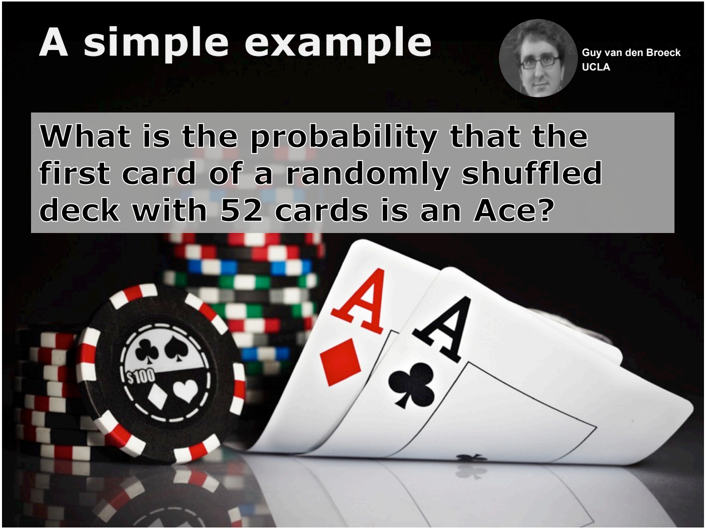
Why do we care about exact inference?

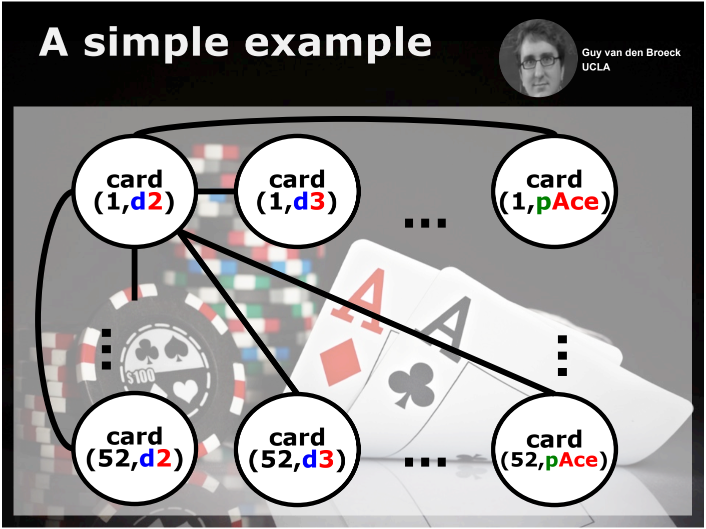
- Gold standard
- Size of problems amenable to exact inference is growing
- Learning for inference

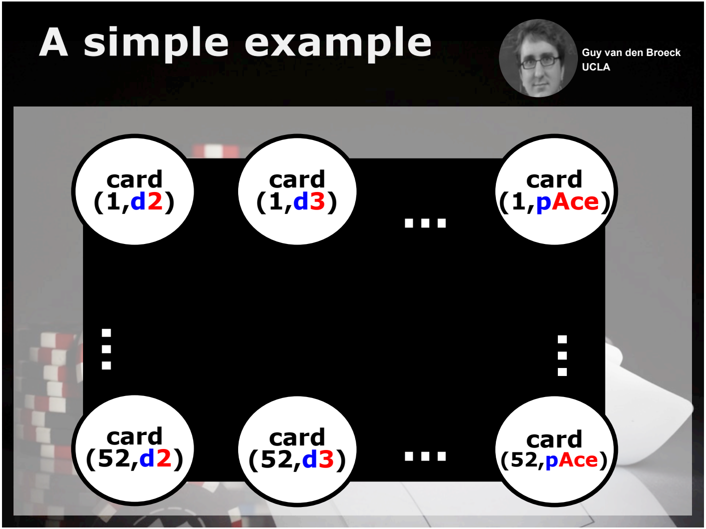
Why Exact Inference?

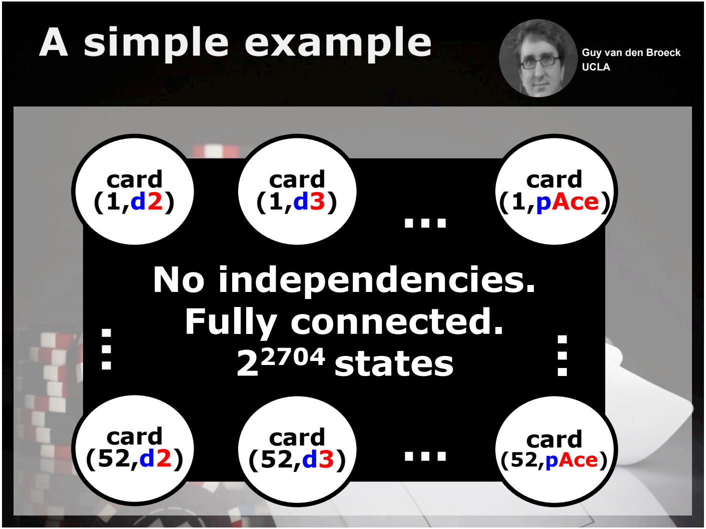
Why do we care about exact inference?

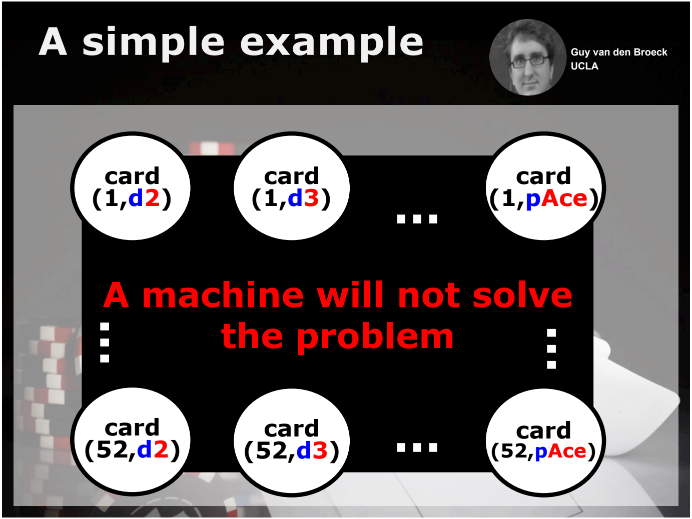
- Gold standard
- Size of problems amenable to exact inference is growing
- Learning for inference
- Basis for efficient approximate inference:
    - Rao-Blackwellization
    - Variational Methods

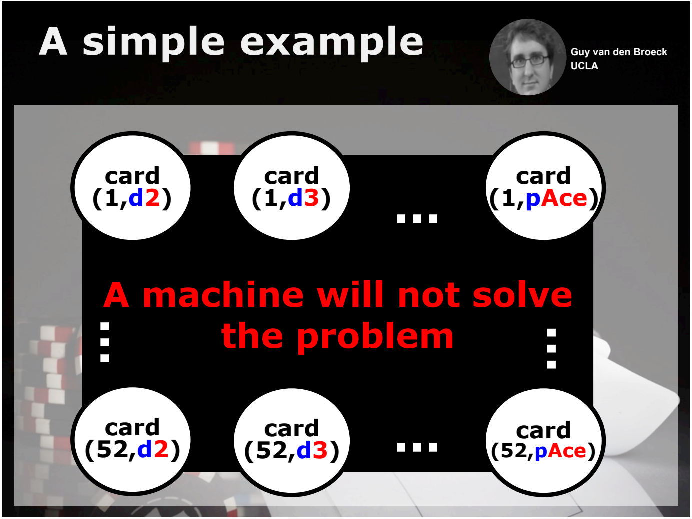

A simple example
Guy van den Broeck
UCLA
What is the probability that the first card of a randomly shuffled deck with 52 cards is an Ace?
A
A
$100


A simple example
Guy van den Broeck
UCLA
card (1,d2)
card (1,d3)
...
card (1,pAce)
card (52,d2)
card (52,d3)
...
card (52,pAce)


A simple example
Guy van den Broeck
UCLA
card (1,d2)
card (1,d3)
...
card (1,pAce)
card (52,d2)
card (52,d3)
...
card (52,pAce)


A simple example
Guy van den Broeck
UCLA
card (1,d2)
card (1,d3)
...
card (1,pAce)
No independencies.
Fully connected.
2^{2704} states
card (52,d2)
card (52,d3)
...
card (52,pAce)


A simple example
Guy van den Broeck
UCLA
card (1,d2)
card (1,d3)
...
card (1,pAce)
A machine will not solve the problem
card (52,d2)
card (52,d3)
...
card (52,pAce)



... unless it can represent and exploit symmetry.

Example: lifted inference

- Consider determining the guilt of someone fitting the description of a person who committed a crime.
- Suppose the probability of someone at random matching the description is one in a million.

Example: lifted inference

- Consider determining the guilt of someone fitting the description of a person who committed a crime.
- Suppose the probability of someone at random matching the description is one in a million.
- The probability this person committed the crime depends on how many people there are:
  - If there were a thousand other people,

Example: lifted inference

- Consider determining the guilt of someone fitting the description of a person who committed a crime.
- Suppose the probability of someone at random matching the description is one in a million.
- The probability this person committed the crime depends on how many people there are:
  - If there were a thousand other people, it is very unlikely there was someone else who committed the crime.
  - If there was a population of 10 million,

Example: lifted inference

- Consider determining the guilt of someone fitting the description of a person who committed a crime.
- Suppose the probability of someone at random matching the description is one in a million.
- The probability this person committed the crime depends on how many people there are:
  - If there were a thousand other people, it is very unlikely there was someone else who committed the crime.
  - If there was a population of 10 million, then we would expect that there would be 10 people who fit the description, and so the probability that this suspect was guilty would be around 10%.

Example: lifted inference

- Consider determining the guilt of someone fitting the description of a person who committed a crime.
- Suppose the probability of someone at random matching the description is one in a million.
- The probability this person committed the crime depends on how many people there are:
  - If there were a thousand other people, it is very unlikely there was someone else who committed the crime.
  - If there was a population of 10 million, then we would expect that there would be 10 people who fit the description, and so the probability that this suspect was guilty would be around 10%.
- We don't need to reason about all of the other individuals separately, but can count over them.

Example: lifted inference

- Suppose someone is giving a presentation, and three people out of 100 people in the audience asked a question (so 97 people were observed to not ask a question).

Example: lifted inference

- Suppose someone is giving a presentation, and three people out of 100 people in the audience asked a question (so 97 people were observed to not ask a question).
- A reasonable model about the eloquence of the speaker might depend on the questions asked
- each of the people who didn't ask a question, their silence might depend on the questions asked, but not on the questions not asked.
- Rather than reasoning separately about each person who was observed to not ask a question, it is reasonable to just count over them.

Example: lifted inference

- The spread of a malaria (or other diseases) may depend on the number of people and the number of mosquitoes.
- Individual mosquitoes are important in such a model, but we don't want to model each mosquito separately.

Lifted Inference

- Idea: treat those individuals about which you have the same information as a block; just count them.

Lifted Inference

- Idea: treat those individuals about which you have the same information as a block; just count them.
- Use the ideas from lifted theorem proving - no need to ground.

Lifted Inference

- Idea: treat those individuals about which you have the same information as a block; just count them.
- Use the ideas from lifted theorem proving - no need to ground.
- Potential to be exponentially faster in the number of non-differentialed individuals.

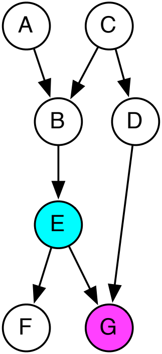
Lifted Inference

- Idea: treat those individuals about which you have the same information as a block; just count them.
- Use the ideas from lifted theorem proving - no need to ground.
- Potential to be exponentially faster in the number of non-differentialed individuals.
- Relies on knowing the number of individuals (the population size).

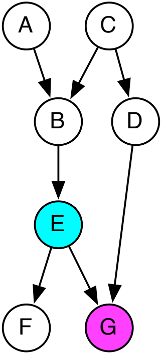
Outline

Inference via factorization in graphical models

$$P(E \mid g) = \frac{P(E \wedge g)}{\sum_E P(E \wedge g)}$$



Inference via factorization in graphical models



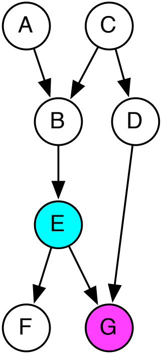
$$P(E \mid g) = \frac{P(E \wedge g)}{\sum_E P(E \wedge g)}$$

$$P(E \wedge g)$$

$$= \sum_F \sum_B \sum_C \sum_A \sum_D P(A)P(B \mid AC) P(C)P(D \mid C)P(E \mid B)P(F \mid E)P(g \mid ED)$$

$$=$$

Inference via factorization in graphical models



$$P(E \mid g) = \frac{P(E \wedge g)}{\sum_E P(E \wedge g)}$$

$$P(E \wedge g)$$

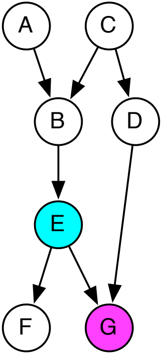
$$= \sum_F \sum_B \sum_C \sum_A \sum_D P(A)P(B \mid AC)$$

$$P(C)P(D \mid C)P(E \mid B)P(F \mid E)P(g \mid ED)$$

$$=$$

$$\left(\sum_D P(D \mid C)P(g \mid ED)\right)\Bigg)\Bigg)$$

Inference via factorization in graphical models



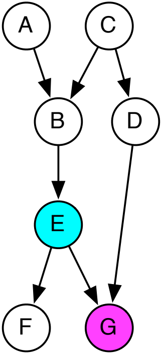
$$P(E \mid g) = \frac{P(E \wedge g)}{\sum_E P(E \wedge g)}$$

$$P(E \wedge g)$$
$$= \sum_F \sum_B \sum_C \sum_A \sum_D P(A)P(B \mid AC) P(C)P(D \mid C)P(E \mid B)P(F \mid E)P(g \mid ED)$$

$$=$$

$$\left(\sum_A P(A)P(B \mid AC)\right)$$
$$\left(\sum_D P(D \mid C)P(g \mid ED)\right)\Big)$$

Inference via factorization in graphical models



$$P(E \mid g) = \frac{P(E \wedge g)}{\sum_E P(E \wedge g)}$$

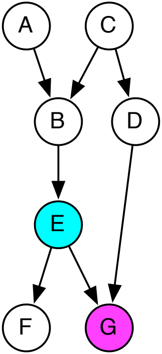
$$P(E \wedge g)$$

$$= \sum_F \sum_B \sum_C \sum_A \sum_D P(A)P(B \mid AC) P(C)P(D \mid C)P(E \mid B)P(F \mid E)P(g \mid ED)$$

$$=$$

$$\sum_C \left(P(C) \left(\sum_A P(A)P(B \mid AC) \right) \left(\sum_D P(D \mid C)P(g \mid ED) \right) \right)$$

Inference via factorization in graphical models



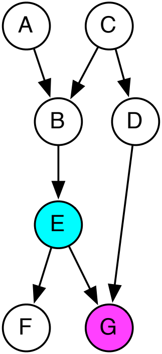
$$P(E \mid g) = \frac{P(E \wedge g)}{\sum_E P(E \wedge g)}$$

$$P(E \wedge g)$$

$$= \sum_F \sum_B \sum_C \sum_A \sum_D P(A)P(B \mid AC) P(C)P(D \mid C)P(E \mid B)P(F \mid E)P(g \mid ED)$$

$$= \sum_B P(E \mid B) \sum_C \left(P(C) \left(\sum_A P(A)P(B \mid AC) \right) \left(\sum_D P(D \mid C)P(g \mid ED) \right) \right)$$

Inference via factorization in graphical models



$$P(E \mid g) = \frac{P(E \wedge g)}{\sum_E P(E \wedge g)}$$

$$\begin{aligned}
P(E \wedge g) & \\
&= \sum_F \sum_B \sum_C \sum_A \sum_D P(A)P(B \mid AC) \\
&\quad P(C)P(D \mid C)P(E \mid B)P(F \mid E)P(g \mid ED) \\
&= \left(\sum_F P(F \mid E)\right) \\
&\quad \sum_B P(E \mid B) \sum_C \left(P(C)\left(\sum_A P(A)P(B \mid AC)\right)\right. \\
&\quad \left.\left(\sum_D P(D \mid C)P(g \mid ED)\right)\right)
\end{aligned}$$

Recursive Conditioning

- Computes sum (partition function) from outside in

Input:

- Context - assignment of values to variables
- Set of factors

Output: value of summing out other variables (partition function)

- Evaluate a factor as soon as all its variables are assigned
- Cache values already computed
- Recognize disconnected components
- Recursively branch on a variable

Variable Elimination and Recursive Conditioning

- Variable elimination is the dynamic programming variant of recursive conditioning.
- Recursive Conditioning is the search variant of variable elimination
- They do the same additions and multiplications.
- Complexity $O(nr^t)$, for n variables, range size r, and treewidth t.

Outline

1. Knowledge Graphs
   - Tensor Factorization and Neural Network Models
2. Representation Issues
   - Desiderata
   - How do relational models relate to probabilistic graphical models
3. Unique properties of relational models
   - Learning general knowledge vs learning about a data set
   - Varying Populations
   - What can be observed?
4. Conclusions and Challenges
5. (Exact) Lifted Inference
   - Recursive Conditioning
   - Lifted Recursive Conditioning

Weighted Formula

A Weighted formula is a pair $\langle F, v \rangle$ where

- F a formula on parametrized random variables
- v number

Example:

$\langle X \neq Y \wedge likes(X, Y) \wedge rich(Y), 0.001 \rangle$
$\langle likes(X, X) \wedge rich(X), 0.7 \rangle$

...

Lifted Recursive Conditioning

LiftedRC(*Context*, *WeightedFormulas*)

- *Context* is a set of assignments to random variables and counts to assignments of instances of relations. e.g.:

$$\begin{aligned}\{\neg a,\ & \#_X f(X) \wedge g(X) = 7,\\ & \#_X f(X) \wedge \neg g(X) = 5,\\ & \#_X \neg f(X) \wedge g(X) = 18,\\ & \#_X \neg f(X) \wedge \neg g(X) = 0\}\end{aligned}$$

- *WeightedFormulas* is a set of weighted formulae, e.g.,

$$\begin{aligned}\{ & \langle \neg a \wedge \neg f(X) \wedge g(X), 0.1\rangle,\\ & \langle a \wedge \neg f(X) \wedge g(X), 0.2\rangle,\\ & \langle f(X) \wedge g(Y), 0.3\rangle,\\ & \langle f(X) \wedge h(X), 0.4\rangle\}\end{aligned}$$

Evaluating Weighted Formulae

Context:

$$\{\neg a, \quad \#_X f(X) \land g(X) = 7,$$
$$\#_X f(X) \land \neg g(X) = 5,$$
$$\#_X \neg f(X) \land g(X) = 18,$$
$$\#_X \neg f(X) \land \neg g(X) = 0\}$$

WeightedFormulas:

$$\{ \quad \langle \neg a \land \neg f(X) \land g(X), 0.1\rangle\,,$$
$$\langle a \land \neg f(X) \land g(X), 0.2\rangle\,,$$
$$\langle f(X) \land g(Y), 0.3\rangle\,,$$
$$\langle f(X) \land h(X), 0.4\rangle\}$$

$LiftedRC(Context, WeightedFormulas)$ returns:

Evaluating Weighted Formulae

Context:

$$\{\neg a, \quad \#_X f(X) \wedge g(X) = 7, \quad \#_X f(X) \wedge \neg g(X) = 5, \quad \#_X \neg f(X) \wedge g(X) = 18, \quad \#_X \neg f(X) \wedge \neg g(X) = 0\}$$

WeightedFormulas:

$$\{ \langle \neg a \wedge \neg f(X) \wedge g(X), 0.1 \rangle, \quad \langle a \wedge \neg f(X) \wedge g(X), 0.2 \rangle, \quad \langle f(X) \wedge g(Y), 0.3 \rangle, \quad \langle f(X) \wedge h(X), 0.4 \rangle \}$$

$LiftedRC(Context, WeightedFormulas)$ returns:

$$0.1^{18} *$$

Evaluating Weighted Formulae

Context:

$$\begin{aligned}\{\neg a, \quad & \#_X f(X) \land g(X) = 7, \\ & \#_X f(X) \land \neg g(X) = 5, \\ & \#_X \neg f(X) \land g(X) = 18, \\ & \#_X \neg f(X) \land \neg g(X) = 0\}\end{aligned}$$

WeightedFormulas:

$$\begin{aligned}\{ \quad & \langle \neg a \land \neg f(X) \land g(X), 0.1 \rangle, \\ & \langle a \land \neg f(X) \land g(X), 0.2 \rangle, \\ & \langle f(X) \land g(Y), 0.3 \rangle, \\ & \langle f(X) \land h(X), 0.4 \rangle\}\end{aligned}$$

LiftedRC(*Context*, *WeightedFormulas*) returns:

$$0.1^{18} * 1 *$$

Evaluating Weighted Formulae

Context:

$$\{\neg a, \quad \#_X f(X) \wedge g(X) = 7,$$
$$\#_X f(X) \wedge \neg g(X) = 5,$$
$$\#_X \neg f(X) \wedge g(X) = 18,$$
$$\#_X \neg f(X) \wedge \neg g(X) = 0\}$$

WeightedFormulas:

$$\{ \quad \langle \neg a \wedge \neg f(X) \wedge g(X), 0.1 \rangle ,$$
$$\langle a \wedge \neg f(X) \wedge g(X), 0.2 \rangle ,$$
$$\langle f(X) \wedge g(Y), 0.3 \rangle ,$$
$$\langle f(X) \wedge h(X), 0.4 \rangle \}$$

$LiftedRC(Context, WeightedFormulas)$ returns:

$$0.1^{18} * 1 * 0.3^{12*}$$

Evaluating Weighted Formulae

Context:

$$\{\neg a, \quad \#_X f(X) \land g(X) = 7, \quad \#_X f(X) \land \neg g(X) = 5, \quad \#_X \neg f(X) \land g(X) = 18, \quad \#_X \neg f(X) \land \neg g(X) = 0\}$$

WeightedFormulas:

$$\{ \quad \langle \neg a \land \neg f(X) \land g(X), 0.1\rangle, \quad \langle a \land \neg f(X) \land g(X), 0.2\rangle, \quad \langle f(X) \land g(Y), 0.3\rangle, \quad \langle f(X) \land h(X), 0.4\rangle\}$$

$LiftedRC(Context, WeightedFormulas)$ returns:

$$0.1^{18} * 1 * 0.3^{12*25} *$$

Evaluating Weighted Formulae

Context:

$$\begin{aligned}\{\neg a, \quad & \#_X f(X) \wedge g(X) = 7, \\ & \#_X f(X) \wedge \neg g(X) = 5, \\ & \#_X \neg f(X) \wedge g(X) = 18, \\ & \#_X \neg f(X) \wedge \neg g(X) = 0\}\end{aligned}$$

WeightedFormulas:

$$\begin{aligned}\{ \quad & \langle \neg a \wedge \neg f(X) \wedge g(X), 0.1 \rangle, \\ & \langle a \wedge \neg f(X) \wedge g(X), 0.2 \rangle, \\ & \langle f(X) \wedge g(Y), 0.3 \rangle, \\ & \langle f(X) \wedge h(X), 0.4 \rangle \}\end{aligned}$$

$LiftedRC(Context, WeightedFormulas)$ returns:

$$0.1^{18} * 1 * 0.3^{12*25} * LiftedRC(Context, \{\langle f(X) \wedge h(X), 0.4 \rangle\})$$

Branching

Context:

$$\{\neg a,\ \#_X f(X) \wedge g(X) = 7,$$
$$\#_X f(X) \wedge \neg g(X) = 5,$$
$$\#_X \neg f(X) \wedge g(X) = 18,$$
$$\#_X \neg f(X) \wedge \neg g(X) = 0\}$$

WeightedFormulas: $\{\langle f(X) \wedge h(X), 0.4\rangle, \dots\}$
Branching on H for the 7 "X" individuals s.th. $f(X) \wedge g(X)$:
$LiftedRC(Context, WeightedFormulas) =$

Branching

Context:

$$\{\neg a,\ \#_X f(X) \wedge g(X) = 7,$$
$$\#_X f(X) \wedge \neg g(X) = 5,$$
$$\#_X \neg f(X) \wedge g(X) = 18,$$
$$\#_X \neg f(X) \wedge \neg g(X) = 0\}$$

WeightedFormulas: $\{\langle f(X) \wedge h(X), 0.4\rangle, \dots\}$
Branching on H for the 7 "X" individuals s.th. $f(X) \wedge g(X)$:
$LiftedRC(Context, WeightedFormulas) =$

$$\sum_{i=0}^{7} \binom{7}{i} LiftedRC(\{\neg a,\ \begin{array}{l} \#_X f(X) \wedge g(X) \wedge h(X) = i, \\ \#_X f(X) \wedge g(X) \wedge \neg h(X) = 7 - i, \\ \#_X f(X) \wedge \neg g(X) = 5, \dots\}, \end{array}$$
$$WeightedFormulas)$$

Branching

Context:

$$\{\neg a,\ \#_X f(X) \wedge g(X) = 7,$$
$$\#_X f(X) \wedge \neg g(X) = 5,$$
$$\#_X \neg f(X) \wedge g(X) = 18,$$
$$\#_X \neg f(X) \wedge \neg g(X) = 0\}$$

WeightedFormulas: $\{\langle f(X) \wedge h(X), 0.4\rangle, \ldots\}$
Branching on H for the 7 "X" individuals s.th. $f(X) \wedge g(X)$:
$LiftedRC(Context, WeightedFormulas) =$

$$\sum_{i=0}^{7} \binom{7}{i} LiftedRC(\{\neg a,\ \begin{array}{l} \#_X f(X) \wedge g(X) \wedge h(X) = i, \\ \#_X f(X) \wedge g(X) \wedge \neg h(X) = 7 - i, \\ \#_X f(X) \wedge \neg g(X) = 5, \ldots\}, \end{array}$$
$$WeightedFormulas)$$

Branching

Context:

$$\begin{aligned}\{\neg a,\ & \#_X f(X) \land g(X) = 7,\\ & \#_X f(X) \land \neg g(X) = 5,\\ & \#_X \neg f(X) \land g(X) = 18,\\ & \#_X \neg f(X) \land \neg g(X) = 0\}\end{aligned}$$

WeightedFormulas: $\{\langle f(X) \land h(X), 0.4\rangle, \dots\}$
Branching on H for the 7 "X" individuals s.th. $f(X) \land g(X)$:
$LiftedRC(Context, WeightedFormulas) =$

$$\sum_{i=0}^{7} \binom{7}{i} LiftedRC(\{\neg a,\ \begin{aligned}& \#_X f(X) \land g(X) \land h(X) = i,\\ & \#_X f(X) \land g(X) \land \neg h(X) = 7 - i,\\ & \#_X f(X) \land \neg g(X) = 5, \dots\},\end{aligned}$$
$$WeightedFormulas)$$

Branching

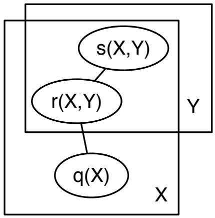
Context:

$$\{\neg a,\ \#_X f(X) \wedge g(X) = 7,$$
$$\#_X f(X) \wedge \neg g(X) = 5,$$
$$\#_X \neg f(X) \wedge g(X) = 18,$$
$$\#_X \neg f(X) \wedge \neg g(X) = 0\}$$

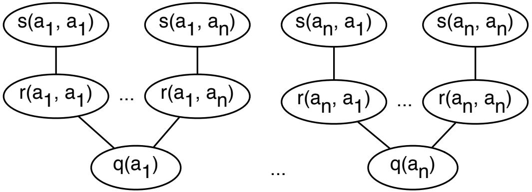
WeightedFormulas: $\{\langle f(X) \wedge h(X), 0.4\rangle, \dots\}$
Branching on H for the 7 "X" individuals s.th. $f(X) \wedge g(X)$:
$LiftedRC(Context, WeightedFormulas) =$

$$\sum_{i=0}^{7} \binom{7}{i} LiftedRC(\{\neg a,\ \#_X f(X) \wedge g(X) \wedge h(X) = i,$$
$$\#_X f(X) \wedge g(X) \wedge \neg h(X) = 7 - i,$$
$$\#_X f(X) \wedge \neg g(X) = 5, \dots\},$$
$$WeightedFormulas)$$

Recognizing Disconnectedness



Relational Model

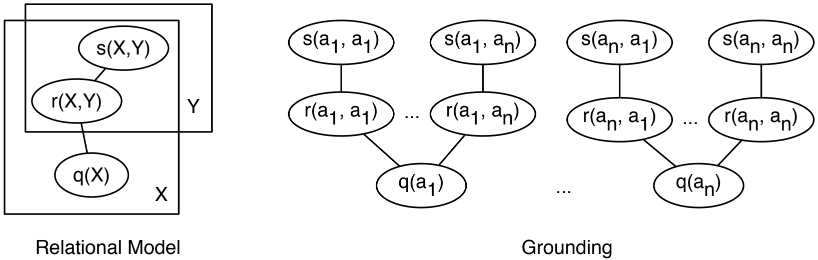


Grounding

Weighted formulae *WeightedFormulas*:

$$\{\ \langle\{s(X,Y)\wedge r(X,Y)\},t_1\rangle$$
$$\langle\{q(X)\wedge r(X,Y)\},t_2\rangle\}$$

Recognizing Disconnectedness



Weighted formulae *WeightedFormulas*:

$$\{\langle\{s(X,Y)\wedge r(X,Y)\}, t_1\rangle$$
$$\langle\{q(X)\wedge r(X,Y)\}, t_2\rangle\}$$

$$LiftedRC(Context, WeightedFormulas)$$
$$= LiftedRC(Context, WeightedFormulas\{X/c\})^n$$

...now we only have unary predicates

Observations and Queries

- Observations become the initial context.
  Observations can be ground or lifted.
-
$$P(q|obs) = \frac{LiftedRC(q \wedge obs, WFs)}{LiftedRC(q \wedge obs, WFs) + LiftedRC(\neg q \wedge obs, WFs)}$$

  calls can share the cache
- "How many?" queries are also allowed

Complexity

As the population size n of undifferentiated individuals increases:

- If grounding is polynomial — instances must be disconnected — lifted inference is constant in n (taking r^n for real r)

Complexity

As the population size n of undifferentiated individuals increases:

- If grounding is polynomial — instances must be disconnected — lifted inference is constant in n (taking r^n for real r)
- Otherwise, for unary relations, grounding is exponential and lifted inference is polynomial.

Complexity

As the population size n of undifferentiated individuals increases:

- If grounding is polynomial — instances must be disconnected — lifted inference is constant in n (taking r^n for real r)
- Otherwise, for unary relations, grounding is exponential and lifted inference is polynomial.
- If non-unary relations become unary, above holds.

Complexity

As the population size n of undifferentiated individuals increases:

- If grounding is polynomial — instances must be disconnected — lifted inference is constant in n (taking r^n for real r)
- Otherwise, for unary relations, grounding is exponential and lifted inference is polynomial.
- If non-unary relations become unary, above holds.
- Otherwise, ground one individual from population, recurse. Sometimes this domain recursion is linear, but is typically exponential (as is grounding the population).

Complexity

As the population size n of undifferentiated individuals increases:

- If grounding is polynomial — instances must be disconnected — lifted inference is constant in n (taking r^n for real r)
- Otherwise, for unary relations, grounding is exponential and lifted inference is polynomial.
- If non-unary relations become unary, above holds.
- Otherwise, ground one individual from population, recurse. Sometimes this domain recursion is linear, but is typically exponential (as is grounding the population).

Always exponentially faster than grounding everything.

What we can and cannot lift

We can lift a model that consists just of

$$\langle\{f(X) \land g(Z)\}, \alpha_4\rangle$$

What we can and cannot lift

We can lift a model that consists just of

$$\langle \{ f(X) \wedge g(Z) \}, \alpha_4 \rangle$$

or just of

$$\langle \{ f(X,Z) \wedge g(Y,Z) \}, \alpha_2 \rangle$$

What we can and cannot lift

We can lift a model that consists just of

$$\langle\{f(X) \wedge g(Z)\}, \alpha_4\rangle$$

or just of

$$\langle\{f(X,Z) \wedge g(Y,Z)\}, \alpha_2\rangle$$

or just of

$$\langle\{f(X,Z) \wedge g(Y,Z) \wedge h(Y)\}, \alpha_3\rangle$$

What we can and cannot lift

We can lift a model that consists just of

$$\langle\{f(X) \land g(Z)\}, \alpha_4\rangle$$

or just of

$$\langle\{f(X,Z) \land g(Y,Z)\}, \alpha_2\rangle$$

or just of

$$\langle\{f(X,Z) \land g(Y,Z) \land h(Y)\}, \alpha_3\rangle$$

We cannot lift (still exponential) a model that consists just of:

$$\langle\{f(X,Z) \land g(Y,Z) \land h(Y,W)\}, \alpha_3\rangle$$

What we can and cannot lift

We can lift a model that consists just of

$$\langle\{f(X) \land g(Z)\}, \alpha_4\rangle$$

or just of

$$\langle\{f(X,Z) \land g(Y,Z)\}, \alpha_2\rangle$$

or just of

$$\langle\{f(X,Z) \land g(Y,Z) \land h(Y)\}, \alpha_3\rangle$$

We cannot lift (still exponential) a model that consists just of:

$$\langle\{f(X,Z) \land g(Y,Z) \land h(Y,W)\}, \alpha_3\rangle$$

or

$$\langle\{f(X,Z) \land g(Y,Z) \land h(Y,X)\}, \alpha_3\rangle$$

Compilation

- The computation reduces to products and sums
- The structure can be determined at compile time
- Orders of magnitude faster than lifted recursive conditioning
- Often abstracted as weighted model counting (WMC)

Take Home

- Lifted inference exploits symmetries ("for all")
- Instead of considering which individuals a predicate is true for, count how many individuals it is true for, and determine appropriate probabilities.
- Always exponentially better in the number of undifferentiated individuals than grounding everything.
- Open problem: finding a dichotomy of those problems we know we can lift and those we know it is impossible to lift.