Lifted inference in relational graphical models and (potentially) probabilistic programs

David Poole

Department of Computer Science,
University of British Columbia
Leverhulme Trust visiting professor at the University of Oxford

March 2015
Outline

1. Relational Graphical Models

2. Exact Inference
   - Recursive Conditioning
   - Lifted Inference
   - Lifted Recursive Conditioning

3. Lifting Probabilistic Programs (?)
S, C logical variables representing students, courses
the set of individuals of a type is called a population
I(S), Gr(S, C), D(C) are parametrized random variables
Specify P(I(S)), P(D(C)), P(Gr(S, C) | I(S), D(C))
S, C logical variables representing students, courses
the set of individuals of a type is called a population
I(S), Gr(S, C), D(C) are parametrized random variables
Specify \( P(I(S)), P(D(C)), P(Gr(S, C) \mid I(S), D(C)) \)

Grounding:
- for every student \( s \), there is a random variable \( I(s) \)
- for every course \( c \), there is a random variable \( D(c) \)
- for every \( s, c \) pair there is a random variable \( Gr(s, c) \)
Plate Notation

- With 1000 students and 100 courses, grounding contains
  - 1000 $I(s)$ variables
  - 100 $D(C)$ variables
  - 100000 $Gr(s,c)$ variables
  total: 101100 variables

- Suppose $Gr$ has 3 possible values. Numbers to be specified to define the probabilities:
  1 for $I(s)$, 1 for $D(C)$, 8 for $Gr(S,C) = 10$ parameters.
### Example: Predicting Relations

<table>
<thead>
<tr>
<th>Student</th>
<th>Course</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_1$</td>
<td>$c_1$</td>
<td>$A$</td>
</tr>
<tr>
<td>$s_2$</td>
<td>$c_1$</td>
<td>$C$</td>
</tr>
<tr>
<td>$s_1$</td>
<td>$c_2$</td>
<td>$B$</td>
</tr>
<tr>
<td>$s_2$</td>
<td>$c_2$</td>
<td></td>
</tr>
<tr>
<td>$s_3$</td>
<td>$c_2$</td>
<td>$B$</td>
</tr>
<tr>
<td>$s_4$</td>
<td>$c_3$</td>
<td>$B$</td>
</tr>
<tr>
<td>$s_3$</td>
<td>$c_3$</td>
<td></td>
</tr>
<tr>
<td>$s_4$</td>
<td>$c_4$</td>
<td></td>
</tr>
</tbody>
</table>

- Students $s_3$ and $s_4$ have the same averages, on courses with the same averages.
- Which student would you expect to better?
Example: Predicting Relations

```
<table>
<thead>
<tr>
<th>Event</th>
<th>Observed Value</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>gr(s1,c1)</td>
<td>A</td>
<td>0.50</td>
</tr>
<tr>
<td>gr(s2,c1)</td>
<td>C</td>
<td>0.50</td>
</tr>
<tr>
<td>gr(s1,c2)</td>
<td>B</td>
<td>0.22</td>
</tr>
<tr>
<td>gr(s2,c3)</td>
<td>B</td>
<td>0.22</td>
</tr>
<tr>
<td>gr(s3,c2)</td>
<td>B</td>
<td>0.50</td>
</tr>
<tr>
<td>gr(s3,c4)</td>
<td>A</td>
<td>0.49</td>
</tr>
<tr>
<td>gr(s4,c3)</td>
<td>B</td>
<td>0.25</td>
</tr>
<tr>
<td>gr(s4,c4)</td>
<td>C</td>
<td>0.26</td>
</tr>
</tbody>
</table>
```

The diagram shows the relations between events `s1`, `s2`, `s3`, and `s4` and their corresponding observed values and probabilities.
Outline

1. Relational Graphical Models

2. Exact Inference
   - Recursive Conditioning
   - Lifted Inference
   - Lifted Recursive Conditioning

3. Lifting Probabilistic Programs (?)
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
Inference via factorization in graphical models

\[ P(E \mid g) = \frac{P(E \land g)}{\sum_E P(E \land g)} \]

\[ P(E \land g) = \sum_F \sum_B \sum_C \sum_A \sum_D P(A)P(B \mid AC)P(C)P(D \mid C)P(E \mid B)P(F \mid E)P(g \mid ED) \]
Inference via factorization in graphical models

\[ P(E \mid g) = \frac{P(E \land g)}{\sum_E P(E \land g)} \]

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

\[ = \left( \sum_F P(F \mid E) \right) \left( \sum_B P(E \mid B) \sum_C \left( P(C) \left( \sum_A P(A)P(B \mid AC) \right) \left( \sum_D P(D \mid C)P(g \mid ED) \right) \right) \]
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 $t$. 
Recursive Conditioning

procedure $rc(Con : \text{ context}, Fs : \text{ set of factors})$:
  if $\exists v$ such that $\langle\langle Con, Fs \rangle, v \rangle \in cache$
    return $v$
  else if $\text{vars}(Con) \not\subseteq \text{vars}(Fs)$
    return $rc(\{X = v \in Con : X \in \text{vars}(Fs)\}, Fs)$
  else if $\exists F \in Fs$ such that $\text{vars}(F) \subseteq \text{vars}(Con)$
    return $\text{eval}(F, Con) \times rc(Con, Fs \setminus \{F\})$
  else if $Fs = Fs_1 \cup Fs_2$ where $\text{vars}(Fs_1) \cap \text{vars}(Fs_2) \subseteq \text{vars}(Con)$
    return $rc(Con, Fs_1) \times rc(Con, Fs_2)$
  else select variable $X \in \text{vars}(Fs)$
    $sum \leftarrow 0$
    for each $v \in \text{domain}(X)$
      $sum \leftarrow sum + rc(Con \cup \{X = v\}, Fs)$
    $cache \leftarrow cache \cup \{\langle\langle Con, Fs \rangle, sum \rangle\}$
    return $sum$
Recursive Conditioning

procedure $rc(\text{Con} : \text{context}, Fs : \text{set of factors})$:
  if $\exists v$ such that $\langle\langle\text{Con}, Fs\rangle, v\rangle \in \text{cache}$
    return $v$
  else if $\text{vars}(\text{Con}) \not\subseteq \text{vars}(Fs)$
    return $rc(\{X = v \in \text{Con} : X \in \text{vars}(Fs)\}, Fs)$
  else if $\exists F \in Fs$ such that $\text{vars}(F) \subseteq \text{vars}(\text{Con})$
    return $\text{eval}(F, \text{Con}) \times rc(\text{Con}, Fs \setminus \{F\})$
  else if $Fs = Fs_1 \cup Fs_2$ where $\text{vars}(Fs_1) \cap \text{vars}(Fs_2) \subseteq \text{vars}(\text{Con})$
    return $rc(\text{Con}, Fs_1) \times rc(\text{Con}, Fs_2)$
  else select variable $X \in \text{vars}(Fs)$
    $sum \leftarrow 0$
    for each $v \in \text{domain}(X)$
      $sum \leftarrow sum + rc(\text{Con} \cup \{X = v\}, Fs)$
    $\text{cache} \leftarrow \text{cache} \cup \{\langle\langle\text{Con}, Fs\rangle, sum\rangle\}$
    return $sum$
Recursive Conditioning

procedure $rc(Con : \text{context}, Fs : \text{set of factors})$:

- if $\exists v$ such that $\langle\langle Con, Fs \rangle, v \rangle \in cache$
  return $v$
- else if $\text{vars}(Con) \not\subseteq \text{vars}(Fs)$
  return $rc(\{X = v \in Con : X \in \text{vars}(Fs)\}, Fs)$
- else if $\exists F \in Fs$ such that $\text{vars}(F) \subseteq \text{vars}(Con)$
  return $\text{eval}(F, Con) \times rc(Con, Fs \setminus \{F\})$
- else if $Fs = Fs_1 \uplus Fs_2$ where $\text{vars}(Fs_1) \cap \text{vars}(Fs_2) \subseteq \text{vars}(Con)$
  return $rc(Con, Fs_1) \times rc(Con, Fs_2)$
- else select variable $X \in \text{vars}(Fs)$
  
  \[ sum \leftarrow 0 \]
  for each $v \in \text{domain}(X)$
  
  \[ sum \leftarrow sum + rc(Con \cup \{X = v\}, Fs) \]

  $cache \leftarrow cache \cup \{\langle\langle Con, Fs \rangle, sum \rangle\}$
  return $sum$
Recursive Conditioning

procedure $rc(\text{Con} : \text{context}, Fs : \text{set of factors})$:

if $\exists v$ such that $\langle \langle \text{Con}, Fs \rangle, v \rangle \in \text{cache}$

return $v$

else if $\text{vars(Con)} \not\subseteq \text{vars(Fs)}$

return $rc(\{X = v \in \text{Con} : X \in \text{vars(Fs)}\}, Fs)$

else if $\exists F \in Fs$ such that $\text{vars(F)} \subseteq \text{vars(Con)}$

return $\text{eval}(F, \text{Con}) \times rc(\text{Con}, Fs \setminus \{F\})$

else if $Fs = Fs_1 \cup Fs_2$ where $\text{vars(Fs}_1) \cap \text{vars(Fs}_2) \subseteq \text{vars(Con)}$

return $rc(\text{Con}, Fs_1) \times rc(\text{Con}, Fs_2)$

else select variable $X \in \text{vars(Fs)}$

$sum \leftarrow 0$

for each $v \in \text{domain}(X)$

$sum \leftarrow sum + rc(\text{Con} \cup \{X = v\}, Fs)$

$cache \leftarrow cache \cup \{\langle \langle \text{Con}, Fs \rangle, sum \rangle\}$

return $sum$
Outline

1. Relational Graphical Models

2. Exact Inference
   - Recursive Conditioning
   - Lifted Inference
   - Lifted Recursive Conditioning

3. Lifting Probabilistic Programs (?)
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).
Suppose we observe:

- Joe has purple hair, a purple car, and has big feet.
- A person with purple hair, a purple car, and who is very tall was seen committing a crime.

What is the probability that Joe is guilty?
Background parametrized belief network
Observing information about Joe
Observing Joe and the crime
A **parametric factor** (parfactor) is a triple \( \langle C, V, t \rangle \) where

- \( C \) is a set of inequality constraints on parameters,
- \( V \) is a set of parametrized random variables
- \( t \) is a table representing a factor from the random variables to the non-negative reals.

\[
\langle \{ X \neq \text{sue} \}, \{ \text{interested}(X), \text{boring} \} , \begin{array}{c|c|c}
\text{interested} & \text{boring} & \text{Val} \\
\hline
\text{yes} & \text{yes} & 0.001 \\
\text{yes} & \text{no} & 0.01 \\
\text{...} & & \\
\end{array}
\rangle
\]
Factored Parametric Factors

A factored parametric factor is a triple $\langle C, V, t \rangle$ where

- $C$ is a set of inequality constraints on parameters,
- $V$ an assignment to parametrized random variables
- $t$ number

Parfactor:

\[
\langle \{ X \neq \text{sue} \}, \{ \text{interested}(X), \text{boring} \}, \rangle
\]

becomes

\[
\langle \{ X \neq \text{sue} \}, \text{interested}(X) \land \text{boring}, 0.001 \rangle
\]

\[
\langle \{ X \neq \text{sue} \}, \text{interested}(X) \land \neg \text{boring}, 0.01 \rangle
\]

\[
\ldots
\]
Outline

1. Relational Graphical Models

2. Exact Inference
   - Recursive Conditioning
   - Lifted Inference
   - Lifted Recursive Conditioning

3. Lifting Probabilistic Programs (?)
Lifted Recursive Conditioning

\( lrc(Con, Fs) \)

- \( Con \) is a set of assignments to random variables and counts to assignments of instances of relations. e.g.:

\[
\{ \neg A, \#_x F(x) \land G(x) = 7, \\
\neg F(x) \land \neg G(x) = 5, \\
\neg F(x) \land G(x) = 18, \\
\neg F(x) \land \neg G(x) = 0 \}
\]

- \( Fs \) is a set of factored parametrized factors, e.g.,

\[
\{ \langle \{ \}, \neg A \land \neg F(x) \land G(x), 0.1 \rangle, \\
\langle \{ \}, A \land \neg F(x) \land G(x), 0.2 \rangle, \\
\langle \{ \}, F(x) \land G(y), 0.3 \rangle, \\
\langle \{ \}, F(x) \land H(x), 0.4 \rangle \}
\]
Evaluating ParFactors

\textbf{Con}:

\[ \{ \neg A, \ #_x F(x) \land G(x) = 7, \]
\[ \ #_x F(x) \land \neg G(x) = 5, \]
\[ \ #_x \neg F(x) \land G(x) = 18, \]
\[ \ #_x \neg F(x) \land \neg G(x) = 0 \} \]

\textbf{Fs}:

\[ \{ \langle \{\}, \neg A \land \neg F(x) \land G(x), 0.1 \rangle, \]
\[ \langle \{\}, A \land \neg F(x) \land G(x), 0.2 \rangle, \]
\[ \langle \{\}, F(x) \land G(y), 0.3 \rangle, \]
\[ \langle \{\}, F(x) \land H(x), 0.4 \rangle \} \]

\textit{lrc}(\textit{Con}, \textit{Fs}) \textit{returns}: 
Evaluating ParFactors

Con:

$$\{ \neg A, \ #_xF(x) \land G(x) = 7, \ #_xF(x) \land \neg G(x) = 5, \ #_x\neg F(x) \land G(x) = 18, \ #_x\neg F(x) \land \neg G(x) = 0 \}$$

Fs:

$$\{ \langle \{\}, \neg A \land \neg F(x) \land G(x), 0.1 \rangle, \langle \{\}, A \land \neg F(x) \land G(x), 0.2 \rangle, \langle \{\}, F(x) \land G(y), 0.3 \rangle, \langle \{\}, F(x) \land H(x), 0.4 \rangle \}$$

$$lrc(Con, Fs) \text{ returns:}$$

$$0.1^{18} \times 0.3^{12 \times 25} \times lrc(Con, \{ \langle \{\}, F(x) \land H(x), 0.4 \rangle \})$$
Branching

Con:

\[
\{ \neg A, \ \#_x F(x) \land G(x) = 7, \\
\#_x F(x) \land \neg G(x) = 5, \\
\#_x \neg F(x) \land G(x) = 18, \\
\#_x \neg F(x) \land \neg G(x) = 0 \}
\]

Fs:

\[
\{ \langle \{ \}, F(x) \land H(x), 0.4 \rangle, \ldots \}
\]

Branching on $H$ for the 7 “$x$” individuals s.th. $F(x) \land G(x)$:

\[
lrc(\text{Con}, Fs) =
\]
Branching

**Con:**

\[\{ \neg A, \ \#_x F(x) \land G(x) = 7, \]
\[\#_x F(x) \land \neg G(x) = 5, \]
\[\#_x \neg F(x) \land G(x) = 18, \]
\[\#_x \neg F(x) \land \neg G(x) = 0 \}\]

**Fs:**

\[\{ \langle \{ \}, F(x) \land H(x), 0.4 \rangle, \ldots \}\]

Branching on \( H \) for the 7 “\( x \)” individuals s.th. \( F(x) \land G(x) \):

\[lrc(Con, Fs) = \]

\[\sum_{i=0}^{7} \binom{7}{i} lrc(\{ \neg A, \ \#_x F(x) \land G(x) \land H(x) = i, \]
\[\#_x F(x) \land G(x) \land \neg H(x) = 7 - i, \]
\[\#_x F(x) \land \neg G(x) = 5, \ldots \}, Fs)\]
Recognizing Disconnectedness

Relational Model

- \(S(x, y)\)
- \(R(x, y)\)
- \(Q(x)\)

 grounding

- \(S(A_1, A_1)\)
- \(S(A_1, A_n)\)
- \(S(A_n, A_1)\)
- \(S(A_n, A_n)\)

- \(R(A_1, A_1)\)
- \(R(A_1, A_n)\)
- \(R(A_n, A_1)\)
- \(R(A_n, A_n)\)

- \(Q(A_1)\)
- \(Q(A_n)\)

Parfactors \(Fs\):

\[
\{ \langle \emptyset \rangle, \{ S(x, y), R(x, y) \}, t_1 \} \\
\{ \langle \emptyset \rangle, \{ Q(x), R(x, y) \}, t_2 \} \\
\]

\(lrc(Con, Fs) = \)
Recognizing Disconnectedness

Parfactors $Fs$:

\[
\{ \langle \{ \}, \{ S(x, y), R(x, y) \}, t_1 \rangle \}
\]

\[
\langle \{ \}, \{ Q(x), R(x, y) \}, t_2 \rangle \}
\]

\[lrc(\text{Con}, Fs) = lrc(\text{Con}, Fs\{x/C\})^n\]

...now we only have unary predicates
Observations and Queries

- Observations become the initial context. Observations can be ground or lifted.
- \[ P(q|obs) = \frac{rc(q \land obs, Fs)}{rc(q \land obs, Fs) + rc(\neg q \land obs, Fs)} \]
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 $r$)
- Otherwise, for unary relations, grounding is exponential and lifted inference is polynomial.
- If non-unary relations become unary, above holds.
- Otherwise, ground an argument.
  Always exponentially better than grounding everything.
What we can and cannot lift

We can lift a model that consists just of

\[ \langle \{x, z\}, \{F(x), \neg G(z)\}, \alpha_4 \rangle \]

or just of

\[ \langle \{x, y, z\}, \{F(x, z), G(y, z)\}, \alpha_2 \rangle \]

or just of

\[ \langle \{x, y, z\}, \{F(x, z), G(y, z), H(y)\}, \alpha_3 \rangle \]

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

\[ \langle \{x, y, z, w\}, \{F(x, z), G(y, z), H(y, w)\}, \alpha_3 \rangle \]

or

\[ \langle \{x, y, z\}, \{F(x, z), G(y, z), H(y, x)\}, \alpha_3 \rangle \]
Outline

1. Relational Graphical Models

2. Exact Inference
   - Recursive Conditioning
   - Lifted Inference
   - Lifted Recursive Conditioning

3. Lifting Probabilistic Programs (?)
Example: Predicting Relations

Fred has unusual shoe size. Someone with unusual shoe size shot Joe. What is the probability Fred shot Joe?
amERICA := draw(0.2)
for x in range(0,1000000):
    size_23_shoe[x] := draw(0.00001)
    if amERICA: has_gun[x] := draw(0.7)
    else: has_gun[x] := draw(0.02)
for y in range(0,1000000):
    has_motive[x,y] := draw(0.001)
    has_opp[x,y] := draw(0.05)
    if has_motive[x,y] and has_gun[x] and has_opp[x,y]:
        actually_shot[x,y] := draw(0.1)
    if actually_shot[x,y]:
        someone_shot[y] := True
observe someone_shot[joe]
observe size_23_shoe[fred]
query actually_shot[fred,joe]
Lifting probabilistic programs?

- When we create many instances of one object, just create the “generic object”
- When we have to branch on a value; just count the qualitatively different answers
- If caching states in MCMC, assignments with the same counts can be treated as the same
- If computing some parts analytically, this provides one more technique in the toolbox
Conclusion

- Often probabilities depend on the number of individuals (even if not observed).
- Lifting exploits symmetry / exchangeability in relational models.
- Unary relations (properties) can be lifted. Binary relations cannot all be.
- Approximate lifted inference looks for cases that are approximately exchangeable or uses lifting in approximate algorithms.
- Probabilistic logic programs use lifted inference. Can other probabilistic programming languages?