Context-specific independence
Parameter learning: MLE

Graphical Models – 10708
Carlos Guestrin
Carnegie Mellon University
October 5th, 2005
Announcements

- **Homework 2:**
  - Out today/tomorrow
  - Programming part in groups of 2-3

- **Class project**
  - Teams of 2-3 students
  - Ideas on the class webpage, but you can do your own

- **Timeline:**
  - 10/19: 1 page project proposal
  - 11/14: 5 page progress report (20% of project grade)
  - 12/2: poster session (20% of project grade)
  - 12/5: 8 page paper (60% of project grade)
  - All write-ups in NIPS format (see class webpage)
Clique trees versus VE

- Clique tree advantages
  - Multi-query settings
  - Incremental updates
  - Pre-computation makes complexity explicit

- Clique tree disadvantages
  - Space requirements – no factors are “deleted”
  - Slower for single query
  - Local structure in factors may be lost when they are multiplied together into initial clique potential
Clique tree summary

- Solve marginal queries for all variables in only twice the cost of query for one variable
- Cliques correspond to maximal cliques in induced graph
- Two message passing approaches
  - VE (the one that multiplies messages)
  - BP (the one that divides by old message)
- Clique tree invariant
  - Clique tree potential is always the same
  - We are only reparameterizing clique potentials
- Constructing clique tree for a BN
  - from elimination order
  - from triangulated (chordal) graph
- Running time (only) exponential in size of largest clique
  - Solve exactly problems with thousands (or millions, or more) of variables, and cliques with tens of nodes (or less)
Global Structure: Treewidth $w$

$O(n \exp(w))$
Local Structure 1:
Context specific independence
Local Structure 1: Context specific independence

Context Specific Independence (CSI)
After observing a variable, some vars become independent
CSI example: Tree CPD

- Represent $P(X_i|Pa_{Xi})$ using a decision tree
  - Path to leaf is an assignment to (a subset of) $Pa_{Xi}$
  - Leaves are distributions over $X_i$ given assignment of $Pa_{Xi}$ on path to leaf

- **Interpretation of leaf:**
  - For specific assignment of $Pa_{Xi}$ on path to this leaf – $X_i$ is independent of other parents

- Representation can be exponentially smaller than equivalent table
Tabular VE with Tree CPDs

- If we turn a tree CPD into table
  - “Sparsity” lost!
- Need inference approach that deals with tree CPD directly!
Local Structure 2: Determinism

If Battery Power = Dead, then Lights = OFF

<table>
<thead>
<tr>
<th>Battery Power</th>
<th>Lights</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>0.99</td>
</tr>
<tr>
<td>WEAK</td>
<td>0.20</td>
</tr>
<tr>
<td>DEAD</td>
<td>0</td>
</tr>
</tbody>
</table>

If Battery Power = Dead, then Lights = OFF.
Determinism and inference

- Determinism gives a little sparsity in table, but much bigger impact on inference.
- Multiplying deterministic factor with other factor introduces many new zeros.
  - Operations related to theorem proving, e.g., unit resolution.

<table>
<thead>
<tr>
<th>Battery Power</th>
<th>Lights</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ON</td>
</tr>
<tr>
<td>OK</td>
<td>.99</td>
</tr>
<tr>
<td>WEAK</td>
<td>.20</td>
</tr>
<tr>
<td>DEAD</td>
<td>0</td>
</tr>
</tbody>
</table>
Today’s Models …

- Often characterized by:
  - Richness in local structure (determinism, CSI)
  - Massiveness in size (10,000’s variables)
  - High connectivity (treewidth)

- Enabled by:
  - High level modeling tools: relational, first order
  - Advances in machine learning
  - New application areas (synthesis):
    - Bioinformatics (e.g. linkage analysis)
    - Sensor networks

- Exploiting local structure a must!
Exact inference in large models is possible...

- BN from a relational model
Recursive Conditioning

- Treewidth complexity (worst case)

- Better than treewidth complexity with local structure

- Provides a framework for time-space tradeoffs

- Only quick intuition today, details:
  - Koller & Friedman: 3.1-3.4, 6.4-6.6
The Computational Power of Assumptions
The Computational Power of Assumptions

- Battery Age
- Alternator
- Fan Belt
- Charge Delivered
- Battery
- Starter
- Gas Gauge
- Gas
- Leak
- Fuel Line
- Distributor
- Spark Plugs
- Engine Start
- Engine Turn Over
- Lights
- Radio
Decomposition

- Battery Age
- Alternator
- Fan Belt

Battery Power
- Battery
- Starter

Charge Delivered
- Gas Gauge
- Radio
- Lights
- Engine Turn Over

Gas
- Leak
- Fuel Line
- Distributor
- Spark Plugs

Engine Start

A. Darwiche
Case Analysis
Case Analysis

\[ p_l \ast p_r + p \]
Case Analysis

Battery Age → Alternator → Fan Belt → Battery

Battery → Charge Delivered → Battery Power

Starter → Engine Turn Over → Gas Gauge

Fuel Line → Distributor → Spark Plugs → Engine Start

Gas Leak → Gas

\[ p_l \ast p_r + p_l \ast p_r \]
Case Analysis

Battery Age
Alternator
Fan Belt

Charge Delivered

Battery Power
Gas Gauge
Radio
Lights
Engine Turn Over

Gas Leak
Fuel Line
Distributor
Spark Plugs
Engine Start

Gas

\[ p_l \times p_r + p_l \times p_r \]
Case Analysis

Battery Age

Alternator

Fan Belt

Charge Delivered

Battery

Radio

Lights

Engine Turn Over

Gas Gauge

Starter

Gas Leak

Fuel Line

Distributor

Spark Plugs

Engine Start

Gas

Leak

+$\ p_l \ p_r$

+$\ p_l \ p_r$

A. Darwiche
Decomposition Tree

A → B → C → D → E

Cutset

A → B → C → D → E

A

B

C

D

E

f(A) → f(A,B) → f(B,C) → f(C,D) → f(B,D,E)
Decomposition Tree

Time: $O(n \exp(w \log n))$
Space: Linear
(using appropriate dtree)
RC1

RC1(T,e)
   // compute probability of evidence e on dtree T

   If T is a leaf node
   Return Lookup(T,e)
   Else
      p := 0
      for each instantiation c of cutset(T)-E do
         p := p + RC1(Tl,ec) RC1(Tr,ec)
      return p
Lookup(T, e)

θ_{x|u} : CPT associated with leaf T

If X is instantiated in e, then
  x: value of X in e
  u: value of U in e

Return \theta_{x|u}

Else return 1 = \sum_x \theta_{x|u}
Caching

Context

A. Darwiche
Caching

Recursive Conditioning
An any-space algorithm with treewidth complexity
Darwiche AIJ-01

Time: $O(n \exp(w))$
Space: $O(n \exp(w))$
(using appropriate dtree)
**RC2**

**RC2(T, e)**

- If T is a leaf node, return Lookup(T, e)
- \( y := \) instantiation of context(T)
- If cache\(_T[y]\) <> nil, return cache\(_T[y]\)
- \( p := 0 \)
- For each instantiation \( c \) of cutset(T) - E do
  - \( p := p + RC2(T^l, ec) \) RC2(T\(_r\), ec)
- cache\(_T[y]\) := p

Return \( p \)
Decomposition with Local Structure

\[ X \text{ Independent of } B, C \text{ given } A \]
Decomposition with Local Structure

X Independent of B, C given A

A, B, C
Decomposition with Local Structure

\[ X \text{ Independent of } B, C \text{ given } A \]

No need to consider an exponential number of cases (in the cutset size) given local structure.
Caching with Local Structure
Caching with Local Structure

A. Darwiche
Caching with Local

No need to cache an exponential number of results (in the context size) given local structure
Determinism...

\[-A \land \neg B \land \neg C \Rightarrow \neg X\]

\[A \Rightarrow X\]

\[B \Rightarrow X\]

\[C \Rightarrow X\]

A natural setup to incorporate SAT technology:

- Unit resolution to:
  - Derive values of variables
  - Detect/skip inconsistent cases
- Dependency directed backtracking
- Clause learning
CSI Summary

- Exploit local structure
  - Context-specific independence
  - Determinism
- Significantly speed-up inference
  - Tackle problems with tree-width in the thousands

Acknowledgements

- Recursive conditioning slides courtesy of Adnan Darwiche
- Implementation available:
  - http://reasoning.cs.ucla.edu/ace
Where are we?

- Bayesian networks
  - Represent exponentially-large probability distributions compactly
- Inference in BNs
  - Exact inference very fast for problems with low tree-width
  - Exploit local structure for fast inference
- Now: Learning BNs
  - Given structure, estimate parameters
Thumbtack – Binomial Distribution

- $P(\text{Heads}) = \theta$, $P(\text{Tails}) = 1-\theta$

- Flips are i.i.d.:
  - Independent events
  - Identically distributed according to Binomial distribution

- Sequence $D$ of $\alpha_H$ Heads and $\alpha_T$ Tails

$$P(D \mid \theta) = \theta^{\alpha_H} (1 - \theta)^{\alpha_T}$$
Maximum Likelihood Estimation

- **Data:** Observed set $D$ of $\alpha_H$ Heads and $\alpha_T$ Tails
- **Hypothesis:** Binomial distribution
- Learning $\theta$ is an optimization problem
  - What’s the objective function?

- MLE: Choose $\theta$ that maximizes the probability of observed data:
  \[
  \hat{\theta} = \arg\max_\theta P(D \mid \theta) \\
  = \arg\max_\theta \ln P(D \mid \theta)
  \]
Your first learning algorithm

\[ \hat{\theta} = \arg \max_{\theta} \ln P(D | \theta) \]

\[ = \arg \max_{\theta} \ln \theta^\alpha H (1 - \theta)^\alpha T \]

- Set derivative to zero:

\[ \frac{d}{d\theta} \ln P(D | \theta) = 0 \]
MLE for conditional probabilities

- MLE estimate of $P(X=x) =$

- MLE estimate of $P(X=x|Y=y)$
  - Only consider subset of data where $Y=y$
Learning the CPTs

MLE: \[ \hat{p}(X_i = x_i \mid X_{i-1} = x_{i-1}) = \frac{\text{Count}(X_i = x_i, X_{i-1} = x_{i-1})}{\text{Count}(X_{i-1} = x_{i-1})} \]
MLE learning CPTs for general BN

- Vars $X_1, \ldots, X_n$ and BN structure given
  \[ P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | \text{Pa}_{X_i}) \]

- Each i.i.d. data point assigns a value all vars

- Likelihood of the data:

- MLE for CPT $P(X_i | \text{Pa}_{X_i})$:
  \[ \hat{P}(X_i = x_i | \text{Pa}_{X_i} = u) = \frac{\text{Count}(X_i = x_i, \text{Pa}_{X_i} = u)}{\text{Count}(\text{Pa}_{X_i} = u)} \]