Bayesian networks II

Advanced Artificial Intelligence: Lecture

Heriberto Cuayáhuitl
http://staff.lincoln.ac.uk/hcuayahuitl

School of Computer Science, University of Lincoln

UNIVERSITY OF LINCOLN

16 October 2019

Overview

- Last Week: Exact Inference in Bayes Nets
  - Inference by Enumeration
  - Inference by Variable Elimination
- This Week: Approximate Inference in Bayes Nets
  - Rejection Sampling
  - Likelihood Weighting
  - Gibbs Sampling

Reading of last week: Russell & Norvig, (2013). Artificial Intelligence: A Modern Approach, **chapter 14** (until 14.4)

Reading of this week: Russell & Norvig, (2013). Artificial Intelligence: A Modern Approach, **chapter 14** (from 14.5)

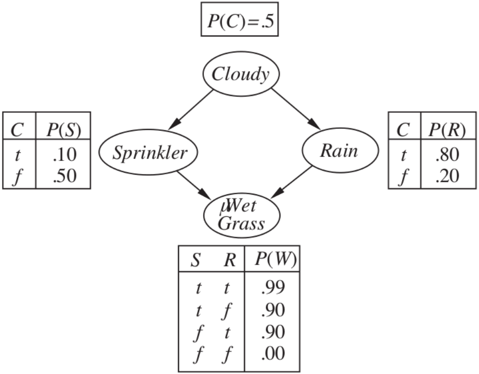
Probabilistic Sampling

- **Idea**: Draw N samples from prior/conditional distributions and compute approximate posterior probabilities
- **Why?** Approximate inference methods give reasonable answers and can be applied to large probability models
- **Example:** Sampling 8 times from $P(\mathrm{C}{=}\mathit{red}) = 0.6$, $P(\mathrm{C}{=}\mathit{green}) = 0.1$, and $P(\mathrm{C}{=}\mathit{blue}) = 0.3$,

gives the following distribution: $P(\mathrm{C}) = \langle 5/8, 1/8, 2/8 \rangle$

Sampling in Bayesian Networks

Sampling=generation of samples from a known probability distrib.



Assume topological order $[Cloudy, Sprinkler, Rain, WetGrass]$:

- Sample from $\mathbf{P}(Cloudy) = \langle 0.5, 0.5 \rangle$
- Sample from $\mathbf{P}(Sprinkler | Cloudy = true) = \langle 0.1, 0.9 \rangle$
- Sample from $\mathbf{P}(Rain | Cloudy = true) = \langle 0.8, 0.2 \rangle$
- Sample .. $\mathbf{P}(WetGrass | Sprinkler = false, Rain = true) = \langle 0.9, 0.1 \rangle$

Prior Sampling

- This function returns events like [*true*, *false*, *true*, *true*] — assuming the Bayesian network and variable ordering above

function PRIOR-SAMPLE(bn) **returns** an event sampled from the prior specified by bn
  inputs: bn, a Bayesian network specifying joint distribution $\mathbf{P}(X_1, \ldots, X_n)$

  $\mathbf{x} \leftarrow$ an event with n elements
  foreach variable X_i **in** $X_1, \ldots, X_n$ **do**
    $\mathbf{x}[i] \leftarrow$ a random sample from $\mathbf{P}(X_i \mid parents(X_i))$
  return x

Prior Sampling

- This function returns events like [*true*, *false*, *true*, *true*] — assuming the Bayesian network and variable ordering above

function PRIOR-SAMPLE(bn) **returns** an event sampled from the prior specified by bn
  inputs: bn, a Bayesian network specifying joint distribution $\mathbf{P}(X_1, \ldots, X_n)$

  $\mathbf{x} \leftarrow$ an event with n elements
  foreach variable X_i **in** $X_1, \ldots, X_n$ **do**
    $\mathbf{x}[i] \leftarrow$ a random sample from $\mathbf{P}(X_i \mid parents(X_i))$
  return x

- The probability of an event can be estimated as $P(x_1, \ldots, x_n) \approx \frac{N_{PS}(x_1, \ldots, x_n)}{N}$

Prior Sampling

- This function returns events like [*true*, *false*, *true*, *true*] — assuming the Bayesian network and variable ordering above

function PRIOR-SAMPLE(bn) **returns** an event sampled from the prior specified by bn
  inputs: bn, a Bayesian network specifying joint distribution $\mathbf{P}(X_1, \ldots, X_n)$

  $\mathbf{x} \leftarrow$ an event with n elements
  foreach variable X_i **in** $X_1, \ldots, X_n$ **do**
    $\mathbf{x}[i] \leftarrow$ a random sample from $\mathbf{P}(X_i \mid parents(X_i))$
  return x

- The probability of an event can be estimated as $P(x_1, ..., x_n) \approx \frac{N_{PS}(x_1,...,x_n)}{N}$

- Example : generating 1000 samples from the sprinkler network, and observing that 511 of them have $Rain = true$, we get the estimated probability $\hat{P}(Rain = true) = 0.511$

Rejection Sampling: Concepts and Example

- Can be used to compute conditional probabilities of the form $\mathbf{P}(X|\mathbf{e})$, given query variable X and evidence $\mathbf{e}$

Rejection Sampling: Concepts and Example

- Can be used to compute conditional probabilities of the form $\mathbf{P}(X|\mathbf{e})$, given query variable X and evidence $\mathbf{e}$

- Idea: to generate samples from the prior distribution specified by the net and reject all those that do not match the evidence

Rejection Sampling: Concepts and Example

- Can be used to compute conditional probabilities of the form $\mathbf{P}(X|\mathbf{e})$, given query variable X and evidence $\mathbf{e}$

- Idea: to generate samples from the prior distribution specified by the net and reject all those that do not match the evidence

- Let $\hat{\mathbf{P}}(X|\mathbf{e}) = \frac{\mathbf{N}_{PS}(X,\mathbf{e})}{N_{PS}(\mathbf{e})} = \alpha \mathbf{N}_{PS}(X,\mathbf{e}) \approx \frac{\mathbf{P}(X,\mathbf{e})}{P(\mathbf{e})} = \mathbf{P}(X|\mathbf{e})$

Rejection Sampling: Concepts and Example

- Can be used to compute conditional probabilities of the form $\mathbf{P}(X|\mathbf{e})$, given query variable X and evidence $\mathbf{e}$

- Idea: to generate samples from the prior distribution specified by the net and reject all those that do not match the evidence

- Let $\hat{\mathbf{P}}(X|\mathbf{e}) = \frac{\mathbf{N}_{PS}(X,\mathbf{e})}{N_{PS}(\mathbf{e})} = \alpha \mathbf{N}_{PS}(X, \mathbf{e}) \approx \frac{\mathbf{P}(X,\mathbf{e})}{P(\mathbf{e})} = \mathbf{P}(X|\mathbf{e})$

- Example: estimate $\hat{\mathbf{P}}(Rain|Sprinkler{=}true)$ from 100 samples
  - 73 samples have $Sprinkler{=}false$ (rejected)
  - 27 samples: 8 have $Rain{=}true$ and 19 have $Rain{=}false$
  - $\hat{\mathbf{P}}(Rain|Sprinkler{=}true) \approx \alpha\langle 8, 19\rangle = \langle 0.296, 0.704\rangle$

Rejection Sampling: Algorithm

function REJECTION-SAMPLING(X, $\mathbf{e}$, bn, N) **returns** an estimate of $\mathbf{P}(X|\mathbf{e})$
  inputs: X, the query variable
    $\mathbf{e}$, observed values for variables $\mathbf{E}$
    bn, a Bayesian network
    N, the total number of samples to be generated
  local variables: $\mathbf{N}$, a vector of counts for each value of X, initially zero

  for j = 1 to N **do**
    $\mathbf{x} \leftarrow$ PRIOR-SAMPLE(bn)
    if $\mathbf{x}$ is consistent with $\mathbf{e}$ **then**
      $\mathbf{N}[x] \leftarrow \mathbf{N}[x]+1$ where x is the value of X in $\mathbf{x}$
  return NORMALIZE($\mathbf{N}$)

Likelihood Weighting: Concepts

- Motivation: Rejection sampling rejects so many samples!

Likelihood Weighting: Concepts

- Motivation: Rejection sampling rejects so many samples!
- Key ideas:
  1. It avoids rejecting samples by generating only events that are consistent with evidence $\mathbf{e}$.
  2. It fixes the values for the evidence variables $\mathbf{E}$.
  3. It samples only the non-evidence variables.

Likelihood Weighting: Example[1]

Query: $\mathbf{P}(Rain | Cloudy = true, WetGrass = true)$

1. $w \leftarrow 1$

[1] Assume topological order $[Cloudy, Sprinkler, Rain, WetGrass]$

Likelihood Weighting: Example[1]

Query: $\mathbf{P}(Rain|Cloudy = true, WetGrass = true)$

1. $w \leftarrow 1$
2. $w \leftarrow w \times P(Cloudy = true) = 1 \times 0.5 = 0.5$

[1] Assume topological order $[Cloudy, Sprinkler, Rain, WetGrass]$

Likelihood Weighting: Example[1]

Query: $\mathbf{P}(Rain | Cloudy = true, WetGrass = true)$

1. $w \leftarrow 1$
2. $w \leftarrow w \times P(Cloudy = true) = 1 \times 0.5 = 0.5$
3. Sample from $\mathbf{P}(Sprinkler | Cloudy = true) = \langle 0.1, 0.9 \rangle$ because $Sprinkler \notin \mathbf{E}$; assume this returns *false*

[1] Assume topological order $[Cloudy, Sprinkler, Rain, WetGrass]$

Likelihood Weighting: Example[1]

Query: $\mathbf{P}(Rain | Cloudy = true, WetGrass = true)$

1. $w \leftarrow 1$
2. $w \leftarrow w \times P(Cloudy = true) = 1 \times 0.5 = 0.5$
3. Sample from $\mathbf{P}(Sprinkler | Cloudy = true) = \langle 0.1, 0.9 \rangle$ because $Sprinkler \notin \mathbf{E}$; assume this returns *false*
4. Sample from $\mathbf{P}(Rain | Cloudy = true) = \langle 0.8, 0.2 \rangle$ because $Rain \notin \mathbf{E}$; assume this returns *true*

[1] Assume topological order [*Cloudy*, *Sprinkler*, *Rain*, *WetGrass*]

Likelihood Weighting: Example[1]

Query: $\mathbf{P}(Rain|Cloudy = true, WetGrass = true)$

1. $w \leftarrow 1$
2. $w \leftarrow w \times P(Cloudy = true) = 1 \times 0.5 = 0.5$
3. Sample from $\mathbf{P}(Sprinkler|Cloudy = true) = \langle 0.1, 0.9 \rangle$ because $Sprinkler \notin \mathbf{E}$; assume this returns *false*
4. Sample from $\mathbf{P}(Rain|Cloudy = true) = \langle 0.8, 0.2 \rangle$ because $Rain \notin \mathbf{E}$; assume this returns *true*
5. $w \leftarrow w \times P(WetGrass = true|Sprinkler = false, Rain = true) = 0.5 \times 0.9 = 0.45$

[1] Assume topological order $[Cloudy, Sprinkler, Rain, WetGrass]$

Likelihood Weighting: Example[1]

Query: $\mathbf{P}(Rain|Cloudy = true, WetGrass = true)$

1. $w \leftarrow 1$
2. $w \leftarrow w \times P(Cloudy = true) = 1 \times 0.5 = 0.5$
3. Sample from $\mathbf{P}(Sprinkler|Cloudy = true) = \langle 0.1, 0.9 \rangle$ because $Sprinkler \notin \mathbf{E}$; assume this returns *false*
4. Sample from $\mathbf{P}(Rain|Cloudy = true) = \langle 0.8, 0.2 \rangle$ because $Rain \notin \mathbf{E}$; assume this returns *true*
5. $w \leftarrow w \times P(WetGrass = true|Sprinkler = false, Rain = true) = 0.5 \times 0.9 = 0.45$
6. Return event $[true, false, true, true]$ with weight 0.45

---

[1] Assume topological order $[Cloudy, Sprinkler, Rain, WetGrass]$

Likelihood Weighting: Algorithm

function LIKELIHOOD-WEIGHTING(X, $\mathbf{e}$, bn, N) **returns** an estimate of $\mathbf{P}(X|\mathbf{e})$
  inputs: X, the query variable
    $\mathbf{e}$, observed values for variables $\mathbf{E}$
    bn, a Bayesian network specifying joint distribution $\mathbf{P}(X_1, \ldots, X_n)$
    N, the total number of samples to be generated
  local variables: $\mathbf{W}$, a vector of weighted counts for each value of X, initially zero

  for j = 1 to N **do**
    $\mathbf{x}, w \leftarrow$ WEIGHTED-SAMPLE(bn, $\mathbf{e}$)
    $\mathbf{W}[x] \leftarrow \mathbf{W}[x] + w$ where x is the value of X in $\mathbf{x}$
  return NORMALIZE($\mathbf{W}$)

---

function WEIGHTED-SAMPLE(bn, $\mathbf{e}$) **returns** an event and a weight

  $w \leftarrow 1$; $\mathbf{x} \leftarrow$ an event with n elements initialized from $\mathbf{e}$
  foreach variable X_i **in** $X_1, \ldots, X_n$ **do**
    if X_i is an evidence variable with value x_i in $\mathbf{e}$
      then $w \leftarrow w \times P(X_i = x_i \mid parents(X_i))$
      else $\mathbf{x}[i] \leftarrow$ a random sample from $\mathbf{P}(X_i \mid parents(X_i))$
  return $\mathbf{x}$, w

Gibbs Sampling: Concepts

- Motivation:
  While Likelihood Weighting can be more efficient than Rejection sampling, its performance degrades as the number of evidence variables increases (e.g. due to low weights)

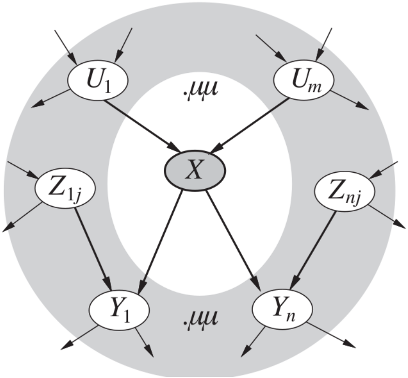
Gibbs Sampling: Concepts

- Motivation:
  While Likelihood Weighting can be more efficient than Rejection sampling, its performance degrades as the number of evidence variables increases (e.g. due to low weights)

- Key ideas:
  1. Instead of generating each sample from scratch, MCMC (Markov Chain Monte Carlo) algorithms generate each sample by making a random change to the preceeding sample.
  2. Start with an arbitrary event (called *state*), with evidence $\mathbf{e}$ fixed, and generate a next state by sampling a value from one of the non-evidence variables X_i given its *Markov blanket*.
  3. Wander randomly around the state space by flipping one variable at a time, but keeping the evidence fixed!

Markov Blanket

The Markov blanket of node (or variable) X consists of its parents, children, and children's parents (gray area)



$$P(X|mb(X) = \alpha P(X|Parents(X)) \times \prod_{Y_i \in Children(X)} P(Y_i|Parents(Y_i))$$

Gibbs Sampling: Example

Query: $\mathbf{P}(Rain|Splinker = true, WetGrass = true)$

1. state [Cloudy=true,Splinkler=true,Rain=false,WetGrass=true]

Gibbs Sampling: Example

Query: $\mathbf{P}(Rain|Splinker = true, WetGrass = true)$

1. state [Cloudy=true,Splinkler=true,Rain=false,WetGrass=true]
2. *Cloudy* is sampled given its Markov blanket, i.e. we sample from $\mathbf{P}(Cloudy|Sprinkler = true, Rain = false)$. If $Cloudy = false$ then the new state is $[false, true, false, true]$

Gibbs Sampling: Example

Query: $\mathbf{P}(Rain|Splinker = true, WetGrass = true)$

1. state [Cloudy=true,Splinkler=true,Rain=false,WetGrass=true]
2. *Cloudy* is sampled given its Markov blanket, i.e. we sample from $\mathbf{P}(Cloudy|Sprinkler = true, Rain = false)$. If $Cloudy = false$ then the new state is $[false, true, false, true]$
3. *Rain* is sampled given its Markov blanket, i.e. we sample from $\mathbf{P}(Rain|Cloudy = false, Sprinkler = true, WetGrass = true)$. If $Rain = true$ then the new state is $[false, true, true, true]$

Gibbs Sampling: Example

Query: $\mathbf{P}(Rain|Splinker = true, WetGrass = true)$

1. state [Cloudy=true,Splinkler=true,Rain=false,WetGrass=true]
2. *Cloudy* is sampled given its Markov blanket, i.e. we sample from $\mathbf{P}(Cloudy|Sprinkler = true, Rain = false)$. If $Cloudy = false$ then the new state is $[false, true, false, true]$
3. *Rain* is sampled given its Markov blanket, i.e. we sample from $\mathbf{P}(Rain|Cloudy = false, Sprinkler = true, WetGrass = true)$. If $Rain = true$ then the new state is $[false, true, true, true]$
4. If this process observes 20 states with $Rain = true$ and 60 states with $Rain = false$, then the answer to our example query is $\alpha < 20, 60 >=< 0.25, 0.75 >$

Gibbs Sampling: Algorithm

function GIBBS-ASK(X, $\mathbf{e}$, bn, N) **returns** an estimate of $\mathbf{P}(X|\mathbf{e})$
local variables: $\mathbf{N}$, a vector of counts for each value of X, initially zero
$\mathbf{Z}$, the nonevidence variables in bn
$\mathbf{x}$, the current state of the network, initially copied from $\mathbf{e}$

initialize $\mathbf{x}$ with random values for the variables in $\mathbf{Z}$
for j = 1 to N **do**
for each Z_i in $\mathbf{Z}$ **do**
set the value of Z_i in $\mathbf{x}$ by sampling from $\mathbf{P}(Z_i|mb(Z_i))$
$\mathbf{N}[x] \leftarrow \mathbf{N}[x] + 1$ where x is the value of X in $\mathbf{x}$
return NORMALIZE($\mathbf{N}$)

Keywords

- **Bayesian Network**
- **Approximate Inference**
- **Probabilistic Sampling**
- **Prior Sampling**
- **Rejection Sampling**
- **Estimated Probability Distribution**
- **Likelihood Weighting**
- **Markov Chain Monte Carlo**
- **Markov Blanket**
- **Gibbs Sampling**

Summary

- Probabilistic inference in Bayes nets means computing a probability distribution of a set of query variables, given a set of evidence and hidden variables
- Exact inference algorithms include inference by enumeration and variable elimination
- Approximate inference algorithms include rejection sampling, likelihood weighting, & Gibbs sampling
- Approximate inference can be used to answer queries in large Bayes—but the accuracy depends on the number of samples
- These algorithms combined with parameter learning (e.g. MLE) allow you to equip a system with probabilistic reasoning

Next

Workshop:

- Excercises about probabilistic inference, Q&A
- Wednesday from 10:30-12hrs, room=INB1302

Reading of this week: Russell & Norvig, (2013). Artificial Intelligence: A Modern Approach, **chapter 14** (until Section 14.5)

Reading of next week: Russell & Norvig, (2013). Artificial Intelligence: A Modern Approach, **chapter 15** (with Nicola)