Intelligent Agents:
Web-mining Agents

Probabilistic Graphical Models

Approximate Inference: Sampling

Tanya Braun

UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR INFORMATIONSSYSTEME

Update: Only sampling techniques are covered (variational inf. scrapped).

- Outlines are amended to reflect the change.

Probabilistic Graphical Models (PGMs)

1. Recap: **Propositional** modelling
   - Factor model, Bayesian network, Markov network
   - Semantics, inference tasks + algorithms + complexity
2. **Probabilistic relational models** (PRMs)
   - Parameterised models, Markov logic networks
   - Semantics, inference tasks
3. **Lifted inference**
   - LVE, LJT, FOKC
   - Theoretical analysis
4. **Lifted learning**
   - Recap: propositional learning
   - From ground to lifted models
   - Direct lifted learning
5. **Approximate Inference: Sampling**
   - Importance sampling
   - MCMC methods
6. **Sequential models & inference**
   - Dynamic PRMs
   - Semantics, inference tasks + algorithms + complexity, learning
7. **Decision making**
   - (Dynamic) Decision PRMs
   - Semantics, inference tasks + algorithms, learning
8. **Continuous Models**
   - Probabilistic soft logic: modelling, semantics, inference tasks + algorithms

Approximations

- Approximate answers to queries such as the posterior $P(R|\boldsymbol{e})$
- Assume an intuitive of approximation:
  - The answer may be erroneous up to some amount
- Formally treated in PAC theory (Probably Approximately Correct) by parameters (δ, ε)
  - Confidence (quantified by δ) in that found solution maximally deviates from true solution up to ε
    - How many samples do you need to satisfy δ, ε

Based on Chapter 12.2, in "Probabilistic Graphical Models" by Koller & Friedman (2009), also includes information about PAC learning

Outline: 5. Approximate Inference

A. (Lifted) sampling

- Importance sampling: Likelihood weighting, importance sampling, lifted importance sampling (LIS)
- Markov Chain Monte Carlo (MCMC) sampling: Gibbs Sampling, Metropolis Hastings, Lifted MCMC

Sampling as Approximation: BNs

- Given a BN F, evidence $\boldsymbol{e}$, and query term R
- Generate a set of samples
  - Sometimes also called particles
  - Each sample contains an observation for each randvar, i.e., an event (in the model or in the sampling target)
  - So, generate an event for each randvar based on the model distribution
- Based on the samples, estimate $P(R|\boldsymbol{e})$ by counting (ML-like)

As maybe seen in the lecture "Advanced Topics Data Science and AI: Probabilistic and Differential Programming" by Özgür Özçep

Sampling: Generalisation

- Generalisation: Estimate expectation of some function $f(\boldsymbol{R})$ relative to a distribution $P(\boldsymbol{R})$

$$E_{P(\boldsymbol{R})}[f(\boldsymbol{R})] = \sum_{\boldsymbol{r} \in \mathcal{R}(\boldsymbol{R})} f(\boldsymbol{r}) P(\boldsymbol{r})$$

  - Generate a set of N samples, estimating value of f or its expectation
  - Aggregate the results

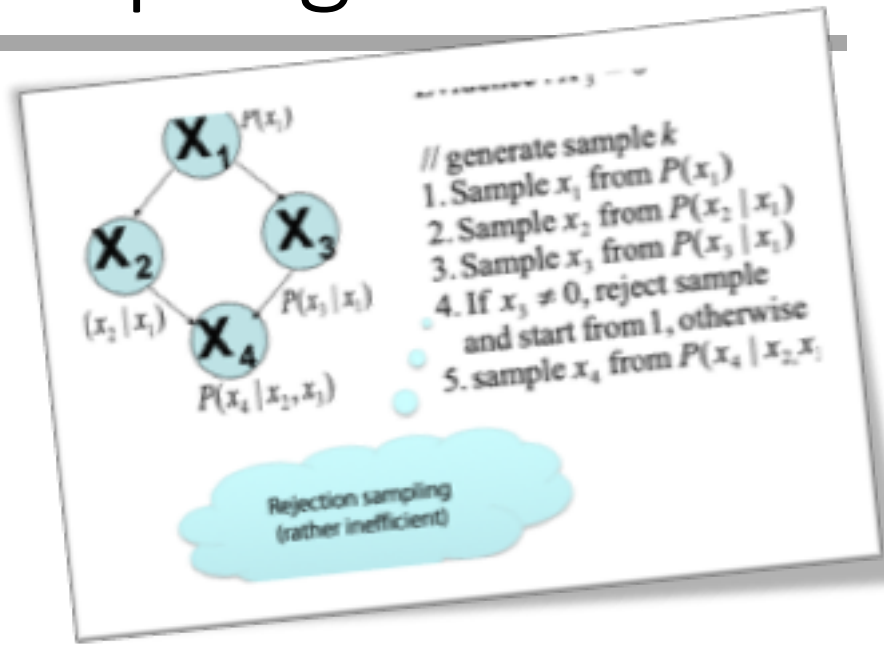
$$E_P[f(\boldsymbol{r})] \approx \frac{1}{N} \sum_{i=1}^{N} f(\boldsymbol{r}_i)$$

  - Can choose f to be indicator function $\mathbf{1}$ that is 1 if $\boldsymbol{R} = \boldsymbol{r}$ and 0 otherwise (which is what happens on the following slides)
- Accuracy usually depends on number of samples N
  - Because then the *law of large numbers* applies

Recap: Rejection Sampling

From Topic 3 in *IR part*:

- Rejection sampling for BNs
  - Given a topological order $\theta = (R_0, \dots R_n)$ for the randvars in the BN and some evidence $\boldsymbol{e}$
  - For each sample
    - Loop through θ
      - Sample a value r_i for R_i given the sampled values of the parent randvars
        - $parents(R_i)$ before R_i in θ, i.e., $parents(R_i) \subseteq \{R_0, \dots, R_{i-1}\}$
      - Drop current sample if r_i contradicts an event in $\boldsymbol{e}$
  - May generate many examples that are rejected
    - Made worse if evidence is very unlikely (unlikely to sample values that agree with evidence)



Likelihood Weighting

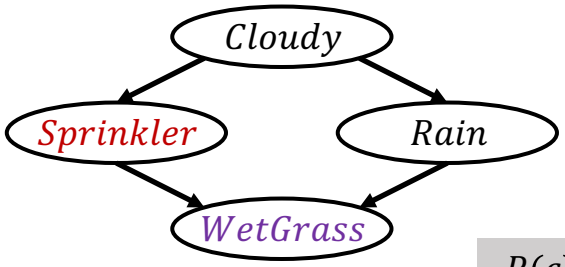
- Goal
  - Avoid inefficiency of rejection sampling
- Idea
  - Generate only events consistent with evidence $\boldsymbol{e}$
  - Each event is weighted by likelihood that the event accords to the evidence

R. Fung and K.-C. Chang. Weighing and integrating evidence for stochastic simulation in Bayesian networks. In *Uncertainty in AI*, 1989.
R. D. Shachter and M. A. Peot. Simulation approaches to general probabilistic inference on belief networks. In *Uncertainty in AI*, 1989.

Likelihood Weighting: Example

- $P(R|S = true, W = true)$?
  - Topological order: (C, S, R, W)
- Sampling (repeat N times)
  - Weight w of sample is set to 1.0
  - Sample from $P(C) = (0.5,0.5) \rightarrow true$
  - S is an evidence variable with value $true$
    $w \leftarrow w \cdot P(S = true|C = true) = 0.1$
  - Sample from $P(R|C = true) = (0.8,0.2) \rightarrow true$
  - W is an evidence variable with value $true$

  $w \leftarrow w \cdot P(W = true|S = true, R = true) = 0.099$
  - $[true, true, true, true]$ with weight 0.099
- Estimating
  - Accumulate weights to either $R = true$ or $R = false$
    - Above sample goes toward $R = true$ with weight 0.099
  - Normalise (= divide by sum of weights)



$P(c) = 0.5$

C	$P(s\|C)$
$true$	0.1
$false$	0.5

C	$P(r\|C)$
$true$	0.8
$false$	0.2

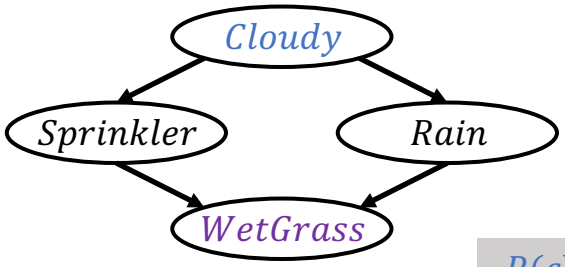
S	R	$P(w\|S,R)$
$true$	$true$	**0.99**
$true$	$false$	**0.9**
$false$	$true$	0.9
$false$	$false$	0.9

Likelihood Weighting: Example

- $P(R|C = true, W = true)$?
  - Topological order: (C, S, R, W)
- Sampling (repeat N times)
  - Weight w of sample is set to 1.0
  - C is an evidence variable with value $true$

    $w \leftarrow w \cdot P(C = true) = 0.5$
  - Sample from $P(S|C = true) = (0.1, 0.9) \rightarrow false$
  - Sample from $P(R|C = true) = (0.8, 0.2) \rightarrow true$
  - W is an evidence variable with value $true$

    $w \leftarrow w \cdot P(W = true|S = false, R = true) = 0.45$
  - $[true, false, true, true]$ with weight 0.45
- Estimating
  - Accumulate weights to either $R = true$ or $R = false$
    - Above sample goes toward $R = true$ with weight 0.45
  - Normalise (= divide by sum of weights)



$P(c) = 0.5$

C	$P(s\|C)$
$true$	0.1
$false$	0.5

C	$P(r\|C)$
$true$	0.8
$false$	0.2

S	R	$P(w\|S,R)$
$true$	$true$	0.99
$true$	$false$	0.9
$false$	$true$	0.9
$false$	$false$	0.9

Likelihood Weighting: Algorithm

function LikelihoodWeighting$(R, \boldsymbol{e}, F, N)$ **returns** an estimate of $P(R|\boldsymbol{e})$
  local variables:
    W, a vector of weighted counts over the range
      values of R, initially 0
  for $j = 1$ **to** N **do**
    $\boldsymbol{r}, w \leftarrow \text{WeightedSample}(F, \boldsymbol{e})$
    $W[r] \leftarrow W[r] + w$ where r is the value of R in $\boldsymbol{r}$
  return $\text{Normalise}(W)$

function WeightedSample$(F, \boldsymbol{e})$ **returns** a compound event and a weight
  $\boldsymbol{r} \leftarrow$ an event with $n = |rv(F)|$ elements; $w \leftarrow 1$
  for $i = 1$ **to** n **do** ▹ follows topological order
    if R_i has a value r_i in $\boldsymbol{e}$ **then**
      $w \leftarrow w \cdot P\big(r_i|parents(R_i)\big)$; set r_i in $\boldsymbol{r}$
    else
      $r_i \leftarrow$ a random sample from $P\big(R_i|parents(R_i)\big)$
  return $\boldsymbol{r}, w$

Likelihood Weighting

- Normalise works as expected
  - Dividing each element by the sum over all elements

Likelihood Analysis

- Sampling probability for WeightedSample
  - i goes through $\boldsymbol{r} \setminus \boldsymbol{e}$
  - Takes only evidence in ancestors into consideration

$$S_{WS}(\boldsymbol{r}, \boldsymbol{e}) = \prod_{i=1}^{l} P\big(r_i | parents(R_i)\big)$$

- Weight for a given sample $\boldsymbol{r}, \boldsymbol{e}$
  - i goes through $\boldsymbol{e}$

$$w(\boldsymbol{r}, \boldsymbol{e}) = \prod_{j=1}^{k} P\left(e_j | parents\big(E_j\big)\right)$$

- Weighted sampling probability is

$$S_{WS}(\boldsymbol{r}, \boldsymbol{e}) \cdot w(\boldsymbol{r}, \boldsymbol{e}) = \prod_{i=1}^{l} P\big(r_i | parents(R_i)\big) \cdot \prod_{j=1}^{k} P\left(e_j | parents\big(E_j\big)\right) = P(\boldsymbol{r}, \boldsymbol{e})$$

  - Last step by semantics of BN
- Hence, likelihood weighting returns consistent estimates

But, performance still degrades with many evidence variables because few samples have nearly all the total weight

Importance Sampling

- Remember: Estimates expectation of a function $f(\boldsymbol{r})$ relative to some distribution $P(\boldsymbol{R})$

$$E_{P(\boldsymbol{R})}[f(\boldsymbol{R})] = \sum_{\boldsymbol{r} \in \mathcal{R}(\boldsymbol{R})} f(\boldsymbol{r}) P(\boldsymbol{r})$$

  - $P(\boldsymbol{R})$ typically called target distribution
  - Estimate this expectation by generating samples $\boldsymbol{r}_i$ from P and then estimating

$$E_P[f(\boldsymbol{r})] \approx \frac{1}{N} \sum_{i=1}^{N} f(\boldsymbol{r}_i)$$

    - What we have done so far

- If generating samples from P is hard, use a (simpler) proposal distribution Q instead

Herman Kahn. Random Sampling (Monte Carlo) Techniques in Neutron Attenuation Problems. Nucleonics, 1950.

Using a Proposal Distribution

- Condition: Proposal distribution Q dominates P
  - I.e., $Q(\boldsymbol{r}) > 0$ whenever $P(\boldsymbol{r}) > 0$
    - Q may not ignore any states that have non-zero probability with P
    - Specifically, support of Q has to include support of P
      - Support for a distribution S are all points $\boldsymbol{r}$ s. t. $S(\boldsymbol{r}) > 0$
  - In general, Q can be arbitrary but computational performance highly depends on how similar Q to P is
    - E.g., want probabilities close to zero in $Q(\boldsymbol{r})$ only if $f(\boldsymbol{r})P(\boldsymbol{r})$ also very small
      - Keep variance small

$$E_{P(\boldsymbol{R})}[f(\boldsymbol{R})] = \sum_{\boldsymbol{r} \in \mathcal{R}(\boldsymbol{R})} f(\boldsymbol{r})P(\boldsymbol{r})$$

- Generate samples from Q instead of P
  - Cannot average f-values of samples generated

  → Adjust estimator to compensate for incorrect sampling distribution

Unnormalised Importance Sampling

- How to adjust:

$$E_{P(\boldsymbol{R})}[f(\boldsymbol{R})] = \sum_{\boldsymbol{r}\in\mathcal{R}(\boldsymbol{R})} f(\boldsymbol{r})P(\boldsymbol{r})$$
$$= \sum_{\boldsymbol{r}\in\mathcal{R}(\boldsymbol{R})} Q(\boldsymbol{r})f(\boldsymbol{r})\frac{P(\boldsymbol{r})}{Q(\boldsymbol{r})}$$
$$= E_{Q(\boldsymbol{R})}\left[f(\boldsymbol{R})\frac{P(\boldsymbol{R})}{Q(\boldsymbol{R})}\right]$$

  - Adjustment: $\frac{P(\boldsymbol{R})}{Q(\boldsymbol{R})}$

Assumes that P is known

- Generate a set of samples from Q and then estimate

$$E_P[f(\boldsymbol{r})] \approx \frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{r}_i)\frac{P(\boldsymbol{r}_i)}{Q(\boldsymbol{r}_i)} = \frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{r}_i)w(\boldsymbol{r}_i)$$

$$w(\boldsymbol{r}_i) \stackrel{\text{def}}{=} \frac{P(\boldsymbol{r}_i)}{Q(\boldsymbol{r}_i)}$$

When P is known: Example

- Interpret likelihood weighting as importance sampling
  - Model without evidence set is P, model with evidence is Q
  - Samples are weighted according to probabilities in CPTs
    - Probability of a sample $\boldsymbol{r}_i$ (includes $\boldsymbol{e}$) in model F conformant with evidence $\boldsymbol{e}$, i.e., P:

$$P(\boldsymbol{r}_i|\boldsymbol{e}) = \frac{P(\boldsymbol{r}_i)}{P(\boldsymbol{e})} = \frac{\prod_{r_j \in \boldsymbol{r}_i} P\left(r_j|parents(R_j)\right)}{P(\boldsymbol{e})}$$

    - Probability of a sample $\boldsymbol{r}_i$ sampled in model F with evidence set, i.e., Q:

$$Q(\boldsymbol{r}_i) = \prod_{r_j \in (\boldsymbol{r}_i \setminus \boldsymbol{e})} P\left(r_j|parents(R_j)\right)$$

    - Weight

$$w(\boldsymbol{r}_i) = \frac{P(\boldsymbol{r}_i|\boldsymbol{e})}{Q(\boldsymbol{r}_i)} = \frac{\prod_{r_j \in \boldsymbol{r}_i} P\left(r_j|parents(R_j)\right)}{P(\boldsymbol{e}) \prod_{r_j \in (\boldsymbol{r}_i \setminus \boldsymbol{e})} P\left(r_j|parents(R_j)\right)} = \frac{\prod_{r_j \in \boldsymbol{e}} P\left(r_j|parents(R_j)\right)}{P(\boldsymbol{e})}$$

    - $P(\boldsymbol{e})$ identical for all samples → okay to ignore, i.e.,

$$w(\boldsymbol{r}_i) = \prod_{r_j \in \boldsymbol{e}} P\left(r_j|parents(R_j)\right)$$

When P is not known

- Most common reason for sampling from Q: P only known up to normalising constant Z
  - Access to a function $\tilde{P}$ that is not a normalised distribution but $\tilde{P}(\boldsymbol{R}) = P(\boldsymbol{R}) \cdot Z$

$$w(\boldsymbol{r}_i) \stackrel{\text{def}}{=} \frac{P(\boldsymbol{r}_i)}{Q(\boldsymbol{r}_i)}$$

- Normalised Importance Sampling
  - Define $w(\boldsymbol{r}_i)$ using $\tilde{P}$: $w(\boldsymbol{r}_i) \stackrel{\text{def}}{=} \frac{\tilde{P}(\boldsymbol{r}_i)}{Q(\boldsymbol{r}_i)}$
  - Estimation no longer works (no probability distribution)

$$E_P[f(\boldsymbol{r})] \approx \frac{1}{N} \sum_{i=1}^{N} f(\boldsymbol{r}_i) \frac{P(\boldsymbol{r}_i)}{Q(\boldsymbol{r}_i)}$$

$$E_{P(\boldsymbol{R})}[f(\boldsymbol{R})] \neq \sum_{\boldsymbol{r} \in \mathcal{R}(\boldsymbol{R})} f(\boldsymbol{r}) \tilde{P}(\boldsymbol{r})$$

$$= \sum_{\boldsymbol{r} \in \mathcal{R}(\boldsymbol{R})} Q(\boldsymbol{r}) f(\boldsymbol{r}) \frac{\tilde{P}(\boldsymbol{r})}{Q(\boldsymbol{r})}$$

$$\neq E_{Q(\boldsymbol{R})} \left[f(\boldsymbol{R}) \frac{\tilde{P}(\boldsymbol{r})}{Q(\boldsymbol{R})} \right]$$

Normalised Importance Sampling

- Trick: Consider $w(\boldsymbol{R})$ as a randvar with expected value Z

$$E_{Q(\boldsymbol{R})}[w(\boldsymbol{R})] = \sum_{\boldsymbol{r}\in\mathcal{R}(\boldsymbol{R})} Q(\boldsymbol{r})w(\boldsymbol{r}) = \sum_{\boldsymbol{r}\in\mathcal{R}(\boldsymbol{R})} Q(\boldsymbol{r})\frac{\tilde{P}(\boldsymbol{r})}{Q(\boldsymbol{r})} = \sum_{\boldsymbol{r}\in\mathcal{R}(\boldsymbol{R})} \tilde{P}(\boldsymbol{r}) = Z$$

- Given $E_{Q(\boldsymbol{R})}[w(\boldsymbol{R})] = Z$

$$\begin{aligned} E_{P(\boldsymbol{R})}[f(\boldsymbol{R})] &= \sum_{\boldsymbol{r}\in\mathcal{R}(\boldsymbol{R})} f(\boldsymbol{r})P(\boldsymbol{r}) = \frac{1}{Z}\sum_{\boldsymbol{r}\in\mathcal{R}(\boldsymbol{R})} Q(\boldsymbol{r})f(\boldsymbol{r})\frac{\tilde{P}(\boldsymbol{r})}{Q(\boldsymbol{r})} \\ &= \frac{1}{Z}E_{Q(\boldsymbol{R})}[f(\boldsymbol{R})w(\boldsymbol{R})] = \frac{E_{Q(\boldsymbol{R})}[f(\boldsymbol{R})w(\boldsymbol{R})]}{E_{Q(\boldsymbol{R})}[w(\boldsymbol{R})]} \end{aligned}$$

- Use estimator for numerator and denominator

$$E_P[f(\boldsymbol{r})] \approx \frac{\sum_{i=1}^{N} f(\boldsymbol{r}_i)w(\boldsymbol{r}_i)}{\sum_{i=1}^{N} w(\boldsymbol{r}_i)}$$

Sampling in Undirected Models

- Assuming we have a proposal distribution $Q(\boldsymbol{R})$, which allows for easy sampling, we can generate samples $\boldsymbol{r}_i$ and weight them accordingly by

$$w(\boldsymbol{r}_i) = \frac{\tilde{P}(\boldsymbol{r}_i)}{Q(\boldsymbol{r}_i)}$$

  - where $\tilde{P}(\boldsymbol{r}_i) = \prod_f \phi_f \left(\pi_{rv(f)}(\boldsymbol{r}_i)\right)$
    - Product of potentials where the arguments have values according to $\boldsymbol{r}_i$
- If we do this for all samples, we estimate Z (or $P(\boldsymbol{e})$)

$$E_{P(\boldsymbol{R})}[w(\boldsymbol{R})] \approx \sum_{\boldsymbol{r} \in \mathcal{R}(\boldsymbol{R})} \frac{\tilde{P}(\boldsymbol{r}_i)}{Q(\boldsymbol{r}_i)} = Z$$

- If are interested in $P(r|\boldsymbol{e})$

$$E_P[f(r)] \approx \frac{\sum_{i=1}^{N} f(r) \frac{\tilde{P}(r)}{Q(r)}}{\sum_{i=1}^{N} \frac{\tilde{P}(\boldsymbol{r}_i)}{Q(\boldsymbol{r}_i)}}$$

Where To Get Q From? – An Idea

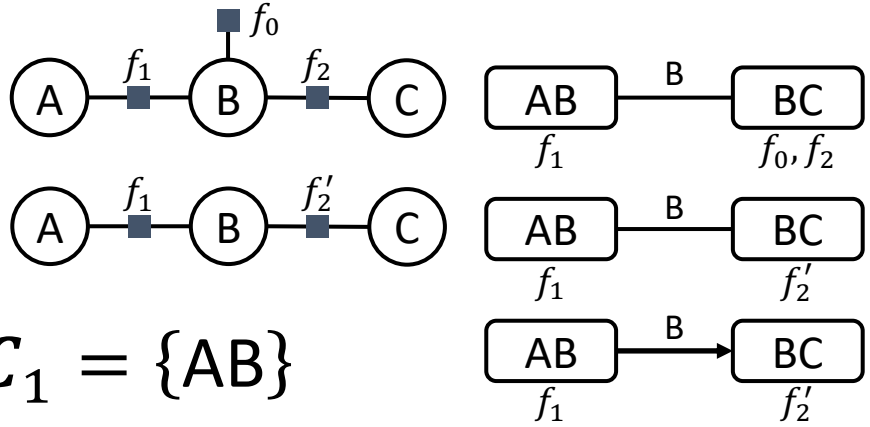
- Given a model F
- Turn F into a model F' s. t. factors are over maximal cliques
  - In jtree: Multiply all factors in each local model s. t. each cluster has a local factor $f_i = \phi_i(R_1, \dots, R_{k_i})$
- Basically turn F' into a "sort-of" BN F^{BN} by
  - Choosing one cluster as root and directed all edges away from that cluster → "directed jtree"
  - Normalise such that
    - Root cluster: marginal over all randvars (potentials sum to 1)
    - All other clusters: conditional over separator randvars of "parent" in directed jtree

    → Enforces a factorisation into (conditional) probability distributions with $Z = 1$ as seen during parameter estimation

    → Fixes a form of topological ordering on randvars in F
      - Root cluster randvars first, followed by randvars as they are visited in the directed jtree
- Sample from $Q = F^{BN}$, e.g., using likelihood weighting

Example

- Model F with jtree J
- Model F', max. cliques
  - $f_2' = f_2 \cdot f_0$
- Choose one as root, e.g., $\boldsymbol{C}_1 = \{\text{AB}\}$
- Normalise
  - $\boldsymbol{C}_1$ such that f_1 is a marginal distribution f_1^{BN}
  - $\boldsymbol{C}_2$ such that f_2' is a distribution f_2^{BN} conditional on B



A	B	f_1
0	0	1
0	1	2
1	0	3
1	1	4

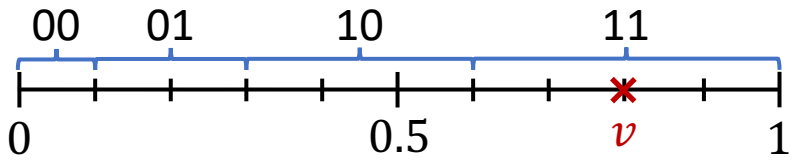
A	B	f_1^{BN}
0	0	0.1
0	1	0.2
1	0	0.3
1	1	0.4

B	C	f_2'
0	0	6
0	1	4
1	0	4
1	1	1

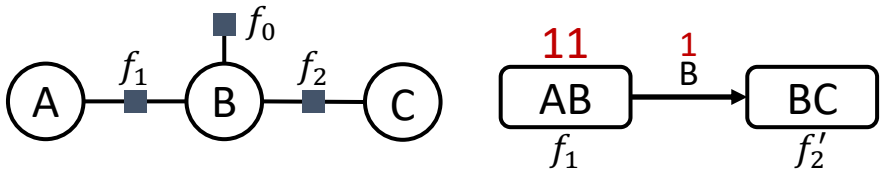
B	C	f_2^{BN}
0	0	0.6
0	1	0.4
1	0	0.8
1	1	0.2

Example

- Ordering of AB, C
- Sample values for AB from f_1^{BN}
  - Since it is a distribution
    - Generate a random number v between 0 and 1 and take values for AB where v lies in the corresponding interval



    - E.g., $v = 0.8 \rightarrow 11$
  - Map to separator randvars $\rightarrow$ B = 1



B	f_0
0	2
1	1

A	B	f_1
0	0	1
0	1	2
1	0	3
1	1	4

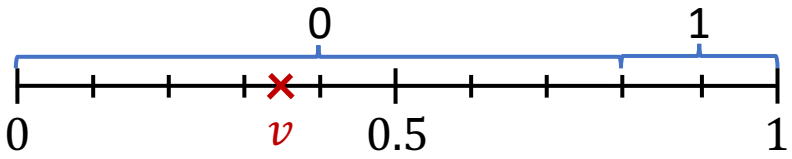
B	C	f_2
0	0	3
0	1	2
1	0	4
1	1	1

A	B	f_1^{BN}
0	0	0.1
0	1	0.2
1	0	0.3
1	1	0.4

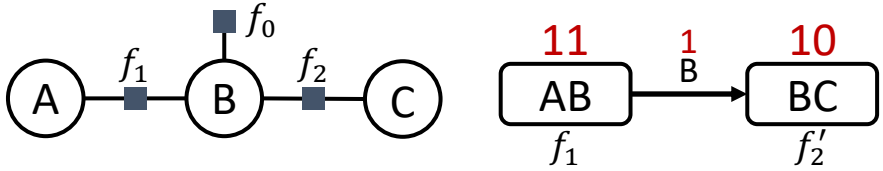
B	C	f_2^{BN}
0	0	0.6
0	1	0.4
1	0	0.8
1	1	0.2

Example

- Ordering of AB, C
- Sample values for BC\B conditioned on B = 1 from f_2^{BN}
  - With B = 1, it is a distribution



  - E.g., $v = 0.35 \rightarrow 0$
- Sample: $[1,1,0]$
  - Q weight: $Q([1,1,0]) = 0.4 \cdot 0.8$
  - $\tilde{P}$ weight: $\tilde{P}([1,1,0]) = 1 \cdot 4 \cdot 4$
  - $w([1,1,0]) = \frac{16}{0.32} = 50$



B	f_0
0	2
1	1

A	B	f_1
0	0	1
0	1	2
1	0	3
1	1	4

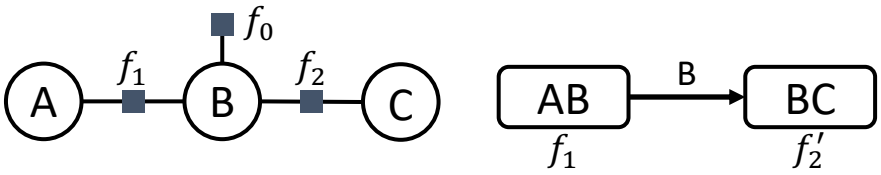
B	C	f_2
0	0	3
0	1	2
1	0	4
1	1	1

A	B	f_1^{BN}
0	0	0.1
0	1	0.2
1	0	0.3
1	1	0.4

B	C	f_2^{BN}
0	0	0.6
0	1	0.4
1	0	0.8
1	1	0.2

Example

- Ordering of AB, C
- Set of N samples $\boldsymbol{r}_i$ with weights $w(\boldsymbol{r}_i)$
  - E.g., sample $[1,1,0]$
    - $w([1,1,0]) = 50$
- Assume query for $P(C = 1)$
- Estimate



B	f_0
0	2
1	1

A	B	f_1
0	0	1
0	1	2
1	0	3
1	1	4

B	C	f_2
0	0	3
0	1	2
1	0	4
1	1	1

$$P(C) \approx \frac{\sum_{i=1}^{N} \mathbf{1}(\boldsymbol{r}_i, C = 1) w(\boldsymbol{r}_i)}{\sum_{i=1}^{N} w(\boldsymbol{r}_i)}$$

$$\mathbf{1}(\boldsymbol{r}_i, C = 1) = \begin{cases} 1 & \pi_C(\boldsymbol{r}_i) = 1 \\ 0 & otherwise \end{cases}$$

A	B	f_1^{BN}
0	0	0.1
0	1	0.2
1	0	0.3
1	1	0.4

B	C	f_2^{BN}
0	0	0.6
0	1	0.4
1	0	0.8
1	1	0.2

Lifted Importance Sampling (LIS)

- Consider an MLN $\Psi = \{(w_i, \psi_i)\}_{i=1}^{n}$
  - Probability of a world ω : $P_\Psi(\omega) = \frac{1}{Z_\Psi} \exp(\sum_{i=1}^{n} w_i n_i(\omega))$
  - Normalisation: $Z_\Psi = \sum_\omega \exp(\sum_{i=1}^{n} w_i n_i(\omega))$
  - *Normal* form:
    - No constants in any formula
    - If any distinct atoms with the same predicate symbol have variables x, y in the same position, then x, y have the same domain
- Idea
  - Sample a value for one predicate
    - Value applies to all instances of predicate under the same evidence (group)
  - Use value to estimate quantities defined over the group

Vibhav Gogate, Abhay Jha, and Deepak Venugopal: Advances in Lifted Importance Sampling. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

Lifted Importance Sampling (LIS)

- Consider estimating $Z_\Psi = \sum_\omega \exp(\sum_{i=1}^n w_i n_i(\omega))$
  - Remember $E_{Q(\boldsymbol{R})}[w(\boldsymbol{R})] = Z$
- Then

$$Z_\Psi = \sum_\omega \exp\left(\sum_{i=1}^n w_i n_i(\omega)\right) \frac{Q(\omega)}{Q(\omega)} = E_{Q(\boldsymbol{R})}\left[\frac{\exp(\sum_{i=1}^n w_i n_i(\omega))}{Q(\omega)}\right]$$

- Given N sampled worlds $\omega^{(t)}$, sampled independently from Q, then

$$Z \approx \hat{Z} = \frac{1}{N}\sum_{t=1}^{N} \frac{\exp\left(\sum_{i=1}^n w_i n_i\left(\omega^{(t)}\right)\right)}{Q(\omega^{(t)})}$$

- LIS uses different lifting rules to handle instances as groups
  - Reduce variance for indistinguishable instances

Vibhav Gogate, Abhay Jha, and Deepak Venugopal: Advances in Lifted Importance Sampling. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

LIS: Lifting Rules – Power Rule

- Given a normal MLN Ψ, a set of logical variables $\boldsymbol{x}$ is called a *decomposer* if it satisfies the following two conditions
  1. Every atom in Ψ contains exactly one variable from $\boldsymbol{x}$
  2. For any predicate symbol R, there exists a position s. t. variables from $\boldsymbol{x}$ only appear at that position in atoms of R
     - Any $x, y \in \boldsymbol{x}$ have the same domain because of normality

  and given a decomposer $\boldsymbol{x}$, rewrite Z_Ψ as

  Compare DPGs

$$Z_\Psi = \left(Z_{\Psi|\boldsymbol{x}\to \mathrm{x}}\right)^{|\mathcal{D}(X)|}$$

  - $\Psi|\boldsymbol{x} \to \mathrm{x}$ denoting that all occurrences of $\boldsymbol{x}$ are replaced with the same constant $\mathrm{x} \in \mathcal{D}(x)$ and the resulting MLN is converted into a normal MLN

Vibhav Gogate, Abhay Jha, and Deepak Venugopal: Advances in Lifted Importance Sampling. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

LIS: Lifting Rules – Generalised Binomial Rule

- Given a normal MLN Ψ and a singleton atom $R(x)$ not involved in self-joins (does not appear more than once in same formula), rewrite Z_Ψ as

Compare counting

$$Z_\Psi = \sum_{j=0}^{|\mathcal{D}(x)|} \binom{|\mathcal{D}(x)|}{j} Z_{\Psi|r^j} w(j) 2^{p(j)}$$

  - $\Psi|r^j$ denotes that in Ψ, truth values are assigned to $R(x)$ such that j instances are set to $true$; specifically
    - Ground all $R(x)$ and assign truth values to the groundings
    - Delete all formulas that evaluate to either $true$ or $false$
    - Delete all groundings of $R(x)$
    - Convert the resulting MLN into a normal one
  - $w(j)$ is the exponentiated sum of the weights of formulas that evaluate to $true$
  - $p(j)$ is the number of ground atoms that are removed from the MLN as a result of removing formulas
    - Don't-care propositional atoms, which can be set to $true$ or $false$
- Can be relaxed by not requiring singleton atoms but then no longer exact

Vibhav Gogate, Abhay Jha, and Deepak Venugopal: Advances in Lifted Importance Sampling. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

LIS: Lifting Rules – Isolated Variable Rule

- For predicate symbol R of an MLN Ψ, a logical variable x at position m in its arguments is called *isolated*
  - if it is exclusive to R in all formulas containing R
- Let $\boldsymbol{x}$ denote the set of all isolated variables of R and let $\boldsymbol{y}$ denote the set of remaining variables in R
  - $\mathcal{D}(\boldsymbol{y})$ cartesian product of the domains of $\boldsymbol{y}$ and $\mathbf{y}_i$ denotes the i'th element
- Then, estimate Z_Ψ as

$$Z_\Psi = Z_{\Psi|\boldsymbol{x}} w(R) 2^{p(R)} \prod_{i=1}^{|\mathcal{D}(\boldsymbol{y})|} \frac{\binom{|\mathcal{D}(\boldsymbol{x})|}{j_i}}{Q_i(j_i|j_1, \ldots, j_{i-1})}$$

  - $\Psi|\boldsymbol{x}$ an MLN obtained from Ψ by applying the following steps:
    1. For $i = 1$ to $|\mathcal{D}(\boldsymbol{y})|$, sample number j_i from a distribution $Q_i(j_i|j_1, \ldots, j_{i-1})$ and set j_i arbitrarily selected groundings of $R(x, \mathbf{y}_i)$ to $true$ and the remaining to $false$,
    2. Delete all formulas that evaluate to either $true$ or $false$
    3. Delete all groundings of R
    4. Convert the MLN to a normal one
  - $w(R)$ exponentiated sum of the weights of formulas that evaluate to $true$
  - $p(R)$ number of ground atoms that are removed from Ψ as a result of (2)

Vibhav Gogate, Abhay Jha, and Deepak Venugopal: Advances in Lifted Importance Sampling. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

LIS tries to apply the power rule, followed by the generalised binomial rule, followed by the isolated variable rule. If all fail, then LIS grounds an atom and samples for the groundings.

function LIS$(\Psi \ -in\ normal\ form, Q)$ **returns** an estimate of Z

if Ψ is empty **then**

return 1

if there exists a decomposer $\boldsymbol{x}$ **then**

return $\left(\mathrm{LIS}(\Psi|\boldsymbol{x} \to \mathrm{x}, Q)\right)^{|\mathcal{D}(x)|}$

if there exists a singleton atom $R(x)$ without self-joins **then**

Use Q to sample an integer $j \in \{0, \ldots, |\mathcal{D}(x)|\}$

return $\frac{\mathrm{LIS}(\Psi|r^j, Q) w(j) 2^{p(j)}}{Q(j)} \binom{|\mathcal{D}(x)|}{j}$

if there exists isolated variables $\boldsymbol{x}$ in a predicate R **then**

return $\mathrm{LIS}(\Psi|\boldsymbol{x}, Q) w(R) 2^{p(R)} \prod_{j=1}^{|\mathcal{D}(\boldsymbol{y})|} \frac{\binom{|\mathcal{D}(\boldsymbol{x})|}{j_i}}{Q_i(j_i|j_1,\ldots,j_{i-1})}$

Choose an atom A and sample all of its groundings from Q

Let a be the sampled assignment

return $\frac{\mathrm{LIS}(\Psi|a, Q) w(a) 2^{p(a)}}{Q(a)}$

LIS for Z

Vibhav Gogate, Abhay Jha, and Deepak Venugopal: Advances in Lifted Importance Sampling. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

LIS: Constructing Q

- General ideas used
  - 4: disjoint parts
    - Handle independently
  - 8: choose an ordering for an atom
    - Assume parent-child relationship
- Lifting rules used for constructing Q
  - 2: power rule
    - Simplifies the MLN
  - 12: approximate generalised bin. rule
  - 13: isolated variable rule

Algorithm 2: Construct Proposal (CP)

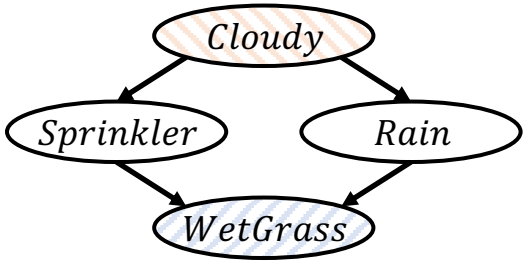
Input: An MLN $\mathcal{M}$, an integer k and a set of atoms $\mathbf{R}$
Output: The structure of the proposal distribution Q

1 **if** $\mathcal{M}$ *is empty* **then return** 1
2 **if** *there exists a decomposer* $\boldsymbol{x}$ **then**
3 Let $x \in \mathbf{x}$ and $X \in \Delta_x$. **return** $\mathrm{CP}(\mathcal{M}[X/\mathbf{x}], k, \mathbf{R})$
4 **if** $\mathcal{M}$ *can be decomposed into* m *MLNS* $\mathcal{M}_1, \ldots, \mathcal{M}_k$ *such that no two MLNs share any atoms* **then**
5 **for** $i = 1$ *to* m **do**
6 $\mathrm{CP}(\mathcal{M}_i, k, \mathbf{R})$
7 **return** 1
8 Heuristically select an atom R from $\mathcal{M}$
9 Heuristically select k atoms from $\mathbf{R}$ as parents of R
// Construct Proposal over R
10 **for** *every assignment to the groundings of* $pa(\mathrm{R})$ *index by* i **do**
11 **if** R *contains no isolated variables* **then**
12 Use the approximate generalized binomial rule to construct $Q_i(\mathrm{R})$
13 **else**
14 Use the isolated variables rule to construct $Q_i(\mathrm{R})$
15 Add R to $\mathbf{R}$
16 Ground R and then remove it from all formulas of $\mathcal{M}$
17 **return** $\mathrm{CP}(\mathcal{M}, k, \mathbf{R})$

Vibhav Gogate, Abhay Jha, and Deepak Venugopal: Advances in Lifted Importance Sampling. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

Problems with Importance Sampling

- Requires a reasonably fitting proposal distribution Q
  - Can be hard to construct/find if we deal with something other than directed models
- Cannot estimate distributions well for evidence in leaves
  - Independent of whether we deal with directed or undirected models
  - Consider two extreme cases in BNs (the easy model type)
    - All evidence at roots
      - → Proposal distribution = posterior distribution
      - → No weighting necessary (for all, $w = P(\boldsymbol{e})$)
    - All evidence at leaves
      - → Proposal distribution = prior distribution
      - → Correction purely by weights, yielding high variance
      - → Will only work well if prior similar to posterior distribution; otherwise most samples are irrelevant, evidenced by a low weight



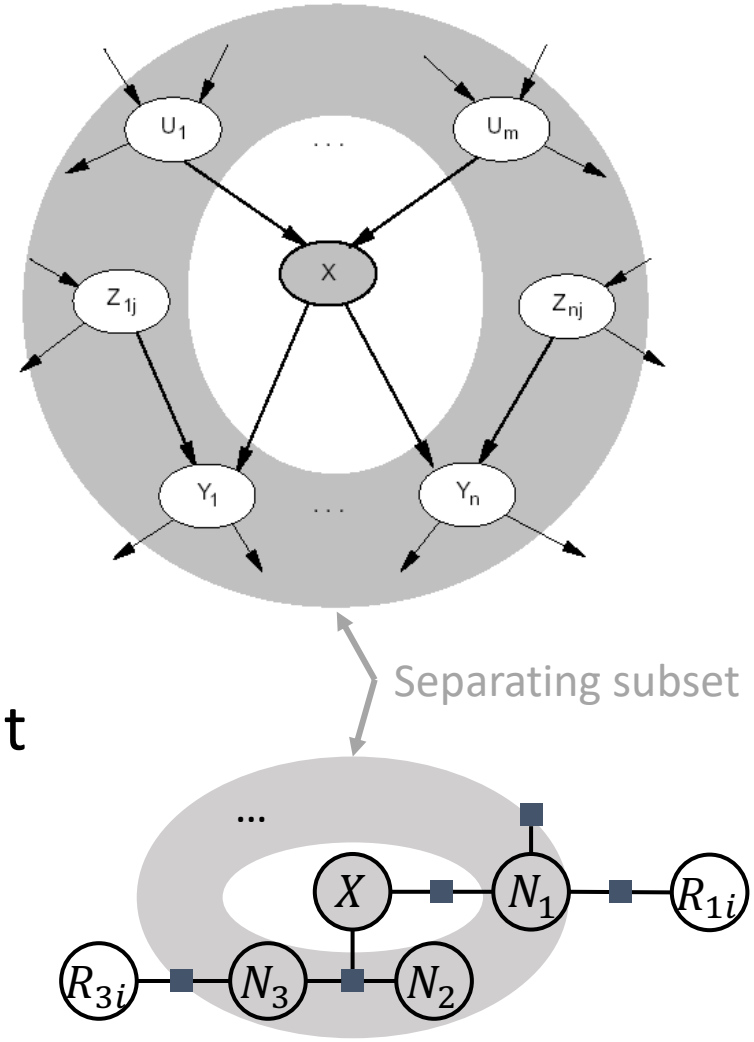
Markov Chain Monte Carlo (MCMC)

- Monte Carlo methods
  - Repeated random sampling to get to some numerical result

- Let us think of the model as being in a particular current state specifying a value for every variable
- MCMC generates each compound event by making a random change to the preceding event
  - Next state generated by randomly sampling a value for one non-evidence variable R_i conditioned on the current values of the variables in Markov blanket of R_i
  - Simplest form called Gibbs sampling, which the next slides build towards

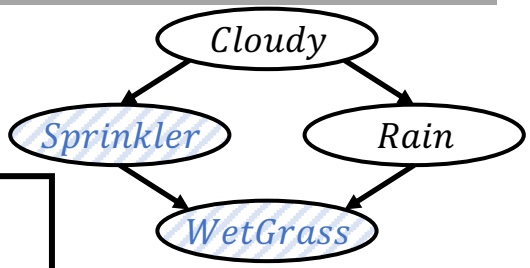
Markov Blanket

- Directed model:
  - Markov blanket of a node X:
    Parents U_k of X
    + *children* Y_i of X
    + *children's parents* Z_{ij}
    - Parents Z_{ij} of Y_i that are not X
- Undirected model:
  - Markov blanket of a node R:
    All direct neighbours of R
    - Direct connection via a factor
- Node is conditionally independent of all other nodes in network, given its Markov Blanket
  - Global Markov property with the Markov blanket as the separating subset

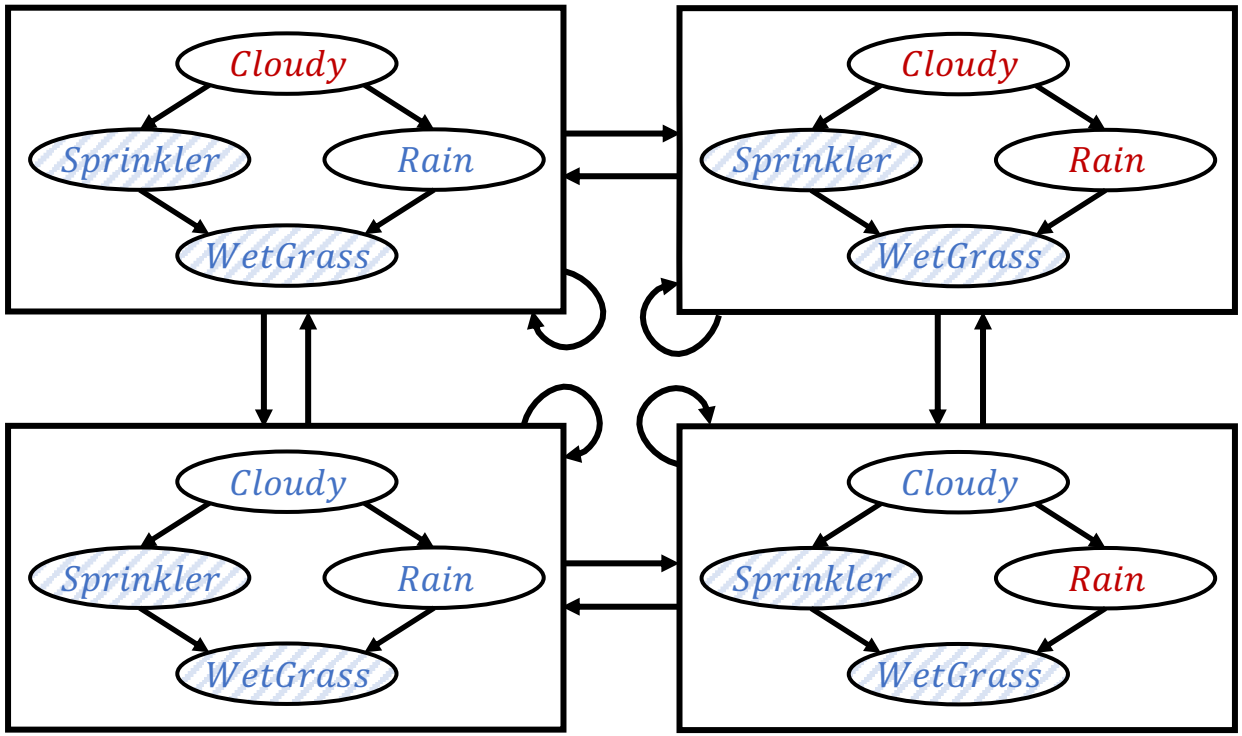


MCMC: Example

- Given $S = true, W = true$, four states (boxes)
  - Four possible combinations of range values for C, R



State-transition system



Fixed evidence

Colour coding

true

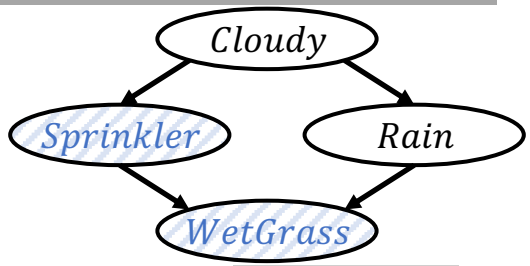
false

  - Arrows between states describe transition probabilities
    - Leads to a (the Markov) chain of states
- Wander about for a while, average what you see

MCMC: Example

- $P(R|S = true, W = true)$?
  - Topological order: (C, S, R, W)
- Random initial state: $[true, true, false, true]$
- Sampling (repeat N times)
  - C is sampled given the current values of its Markov blanket, i.e., sample from $P(C|S = true, R = false)$
    - Suppose result is $\underline{false}$
  - Current state: $[\underline{false}, true, false, true]$
    - Update counts: $R = false \rightarrow +1$
  - R is sampled given the current values of its Markov blanket, i.e., sample from $P(R|C = false, S = true, W = true)$
    - Suppose result is $\underline{true}$
  - Current state: $[false, true, \underline{true}, true]$
    - Update counts: : $R = true \rightarrow +1$
- Assume after $N = 80$ iterations, the process visited 20 states where $R = true$ and 60 states where $R = false$, then answer to query is

$$\text{Normalise}\big((20,60)\big) = (0.25,0.75)$$



$P(c) = 0.5$

C	$P(s\|C)$
$true$	0.1
$false$	0.5

C	$P(r\|C)$
$true$	0.8
$false$	0.2

S	R	$P(w\|S,R)$
$true$	$true$	0.99
$true$	$false$	0.9
$false$	$true$	0.9
$false$	$false$	0.9

Gibbs Sampling: Algorithm

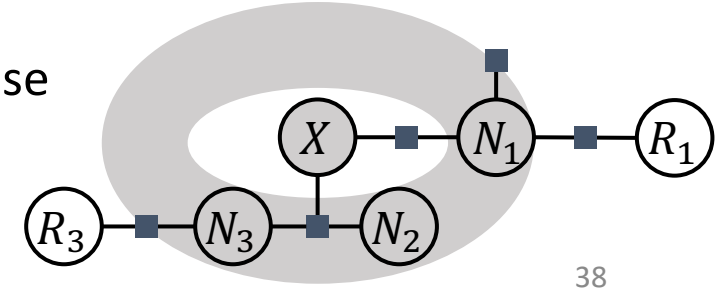
function Gibbs$(R, \boldsymbol{e}, F, N)$ **returns** an estimate of $P(R|\boldsymbol{e})$
  local variables:
    C, a vector of counts over the range values of R, initially 0
    $\boldsymbol{S}$, the non-evidence randvars in F
    $\boldsymbol{r}$, the current state of whole network, initially copied from $\boldsymbol{e}$ with random values for $\boldsymbol{S}$
  for $j = 1$ **to** N **do**
    for each S in $\boldsymbol{S}$ **do**
      Sample the value s of S to replace in $\boldsymbol{r}$ from $P\big(S|mb(S)\big)$ given the values of $MB(S)$ in $\boldsymbol{r}$
      $C[r] \leftarrow C[r] + 1$ where r is the value of R in $\boldsymbol{r}$
  return $\mathrm{Normalise}(W)$

Gibbs Sampling

- State of network = current assignment to all randvars $\boldsymbol{r}$
- Generate next state by sampling one randvar given Markov blanket $MB(S)$ with values $mb(S)$ in current state
  - Sample each randvar in turn, keeping evidence fixed
    - Can also choose a randvar to sample at random each time

Gibbs and Undirected Models

- We have assignments for each randvar in F
- We need to sample from $P\big(S|mb(S)\big)$
  - In example below,
    sample from $P\big(X|mb(X)\big) = P(X|n_1, n_2, n_3)$
- Since S independent from all other randvars given $MB(S)$ and we have values for $MB(S)$, i.e., $mb(S)$, we only need to consider the normalised product $P\big(S, mb(S)\big)$ of the factors between X and $MB(S)$ set to $mb(S)$
  - E.g., $\phi(X, N_1), \phi(X, N_2, N_3)$
    - $\phi(X, n_1) \cdot \phi(X, n_2, n_3)$ and normalise



Gibbs and Undirected Models

X	N_1	f_1
~~0~~	~~0~~	~~1~~
0	1	2
~~1~~	~~0~~	~~3~~
1	1	4

X	N_2	N_3	f_2
~~0~~	~~0~~	~~0~~	~~3~~
~~0~~	~~0~~	~~1~~	~~2~~
0	1	0	4
~~0~~	~~1~~	~~1~~	~~1~~
~~1~~	~~0~~	~~0~~	~~5~~
~~1~~	~~0~~	~~1~~	~~6~~
1	1	0	8
~~1~~	~~1~~	~~1~~	~~1~~

X	N_1	f_1
0	1	2
1	1	4

X	N_2	N_3	f_2
0	1	0	4
1	1	0	8

(lines dropped)

X	f_1
0	2
1	4

X	f_2
0	4
1	8

(columns dropped)

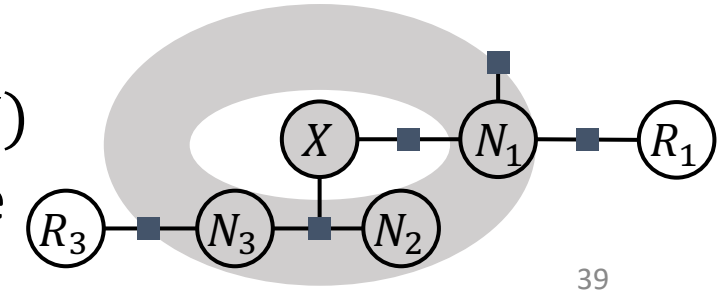
multiply

X	f_{12}
0	8
1	32

normalise

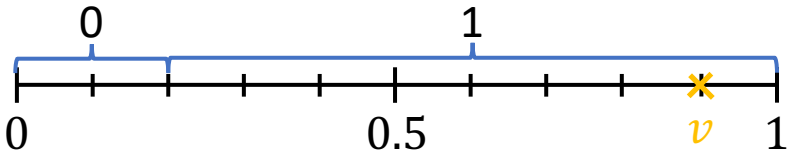
X	f'_{12}
0	0.2
1	0.8

- E.g., $N_1 = 1, N_2 = 1, N_3 = 0$
  - $f_{12} = \phi(X, 1) \cdot \phi(X, 1, 0) = \phi(X)$
  - f'_{12} used to sample a new X value



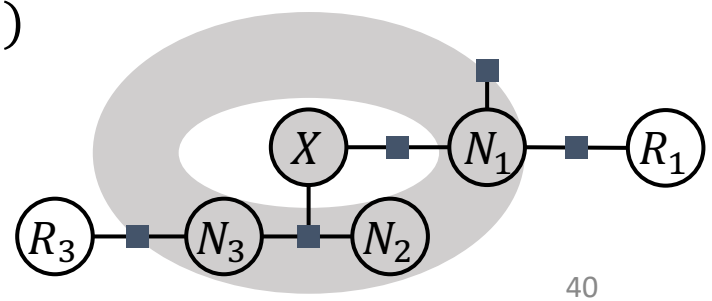
Gibbs and Undirected Models

- Sample a value for X from f'_{12}
  - E.g., $v = 0.9 \rightarrow X = 1$



- New state
  - $N_1 = 1, N_2 = 1, N_3 = 0, R_3 = r_3, R_1 = r_1, X = 1$
- Assuming that we are interested in $P(R_3)$,
  - Add 1 to the counter of r_3
- Continue with sampling next randvar
  - E.g., N_1 with $mb(N_1) = \{X = 1, R_1 = r_1\}$
    - $\phi(X = 1, N_1), \phi(N_1), \phi(N_1, R_1 = r_1)$
      - Multiply and normalise, yielding ϕ'
    - Sample a new value for N_1 using ϕ'

X	f'_{12}
0	0.2
1	0.8

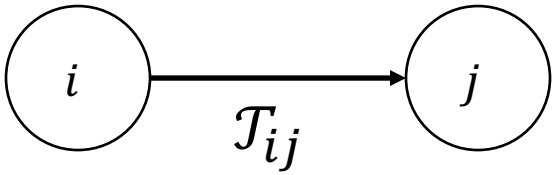


Some Basics for MCMC

- A Markov chain consists of n *states*, plus an $n \times n$ *transition matrix* $\mathcal{T}$
  - At each step, we are in exactly one of the states
  - For $1 \leq i, j \leq n$, matrix entry $\mathcal{T}_{ij}$ tells us the relative frequency of j being the next state, given we are currently in state i

$$\sum_{j=1}^{n} \mathcal{T}_{ij} = 1$$

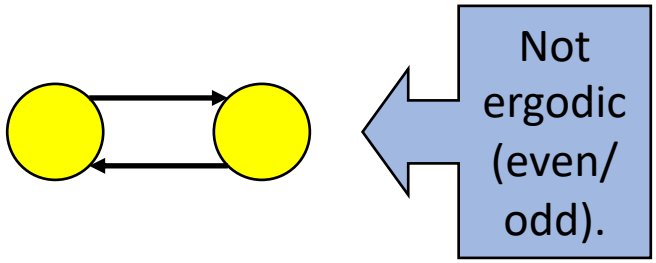
  - $\mathcal{T}_{ij} \triangleq \mathcal{T}(i \to j)$



$\mathcal{T}_{ii} > 0$ is OK (self loops)

Some Basics for MCMC

- Markov chain has to be ergodic for MCMC to work
- Markov chain is ergodic if
  - You have a path from any state to any other state (irreducibility)
    - No part of the system wanders off
  - Returns to states occur at irregular times (aperiodicity)
    - Periodicity: Returns to a state are only possible every $c > 1$ steps
  - For any start state, after a finite transient time T_0, the probability of being in any state at a fixed time $T > T_0$ is nonzero (positive recurrence)
    - Given a finite state space: Positive recurrence follows from irreducibility



Some Basics for MCMC

- Ergodic theory: about dynamical systems that are ergodic
  - System must be measure-preserving
    - Measure on a set: assign a number to each suitable subset of that set
    - Axioms of probability theory correspond to axioms of measure theory (*Kolmogorov axioms*)

    → Some ergodic theorems can be applied to probabilistic setting

Kolmogorov axioms

1. Probability of an event is a non-negative real number
2. Assumption of unit measure: ~probabilities add up to 1
3. Assumption of σ-additivity: Probability of a set of disjoint events equals the sum over the individual probabilities (independence)

- Some differences
  - In ergodic theory
    - irreducible + positive recurrent = ergodic and
    - irreducible + positive recurrent + aperiodic = mixing
  - Whereas in probability theory
    - irreducible + aperiodic + positive recurrent = ergodic

Some Basics for MCMC

- For any finite-state ergodic Markov chain, there is a unique long-term visit rate for each state
  - "*Steady-state*" or *stationary* distribution
    - Over long time-period, each state visited in proportion to this rate
    - It does not matter where we start
  - → Reason why sampling works with a large enough sampling size
  - → *Stationarity*: Transition probabilities between states do not change over time
  - Well-known application that you might have seen: *PageRank*, original ranking principle of Google
    - Rank set of relevant web pages for a query according to the probabilities they have in the steady-state distribution (ranking is query independent)
    - Markov chain:
      - Web pages = states (i.e., being on one and not the others)
      - Arrows from one state/webpage to the next if outgoing link from one to the next
      - Transition model $\mathcal{T}$: for each state, uniformly distributed over all outgoing links
    - Compute steady state distribution λ (as vector): λ has to fulfil $\lambda^T \mathcal{T} = \lambda^T$
      - Eigenvector corresponding to eigenvalue 1

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd: The PageRank Citation Ranking: Bringing Order to the Web. In *Proceedings of the 7th International World Wide Web Conference*, 1998.

Stationary Distribution Formally

- A Markov chain is regular if there exists some number k such that, for every $\boldsymbol{r}, \boldsymbol{r}' \in \mathcal{R}(\boldsymbol{R})$, the probability of getting from $\boldsymbol{r}$ to $\boldsymbol{r}'$ in exactly k steps is > 0
  - For finite state spaces: Condition on regularity equivalent to condition on ergodicity
    - Sometimes easier to verify
  - In factor models/MNs:
    If all potentials are strictly positive, then the Gibbs-sampling Markov chain is regular

Stationary Distribution Formally

- Markov chain with transition model $\mathcal{T}$ is reversible if there exists a unique distribution λ such that, for all $\boldsymbol{r}, \boldsymbol{r}' \in \mathcal{R}(\boldsymbol{R})$:

$$\lambda(\boldsymbol{r})\mathcal{T}(\boldsymbol{r} \to \boldsymbol{r}') = \lambda(\boldsymbol{r}')\mathcal{T}(\boldsymbol{r}' \to \boldsymbol{r})$$

  - Equation is called *detailed balance*
    - Pick a starting state at random according to λ
    - Take a random transition from the chosen state according to $\mathcal{T}$
  - Asserts that, using this process, probability of a transition from $\boldsymbol{r} \to \boldsymbol{r}'$ is the same as probability of transition from $\boldsymbol{r}' \to \boldsymbol{r}$

If $\mathcal{T}$ is regular and satisfies the detailed balance equation relative to λ, then λ is the unique stationary distribution of $\mathcal{T}$.

Parallelisation

- Run Gibbs independently on full copies of the same model
  - More samples in the same time

  or
  - Same samples in fewer time
- Combine individual counters in one

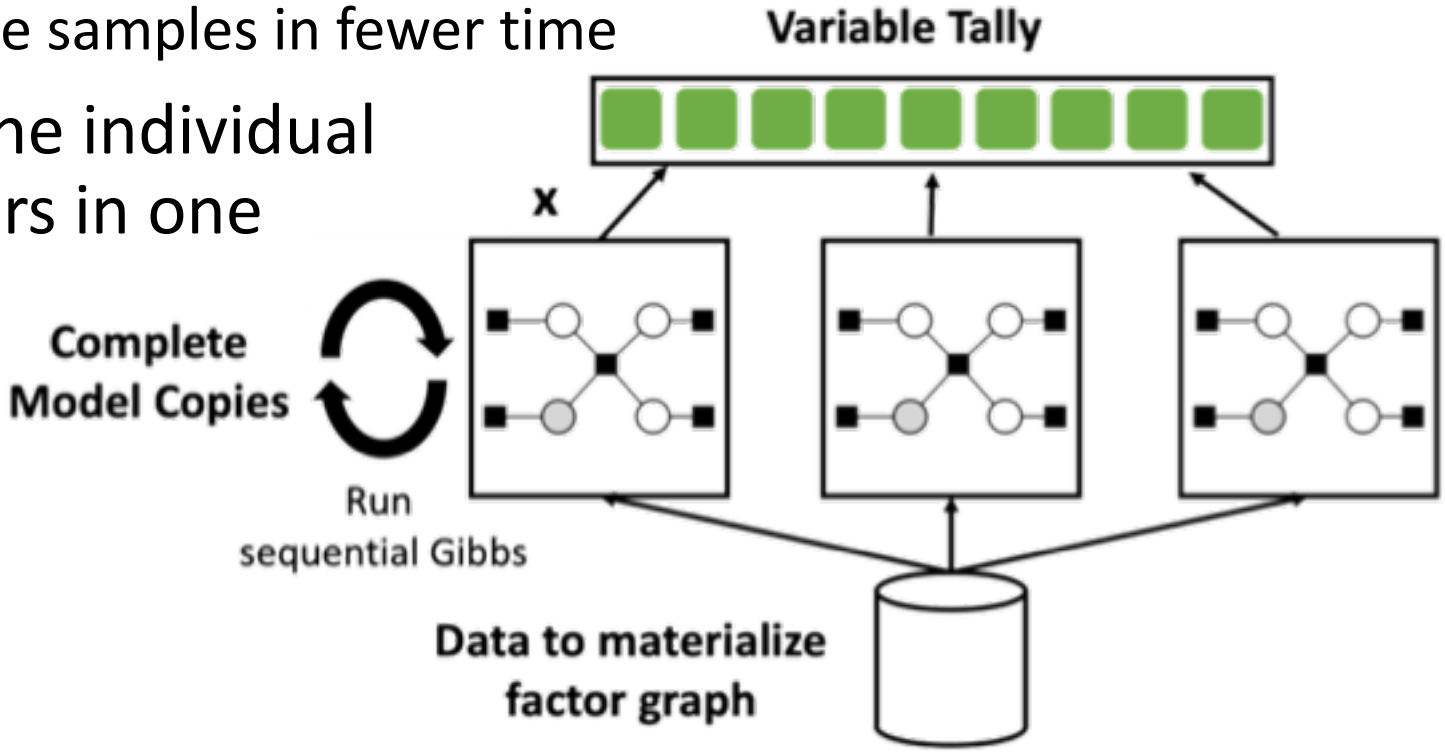
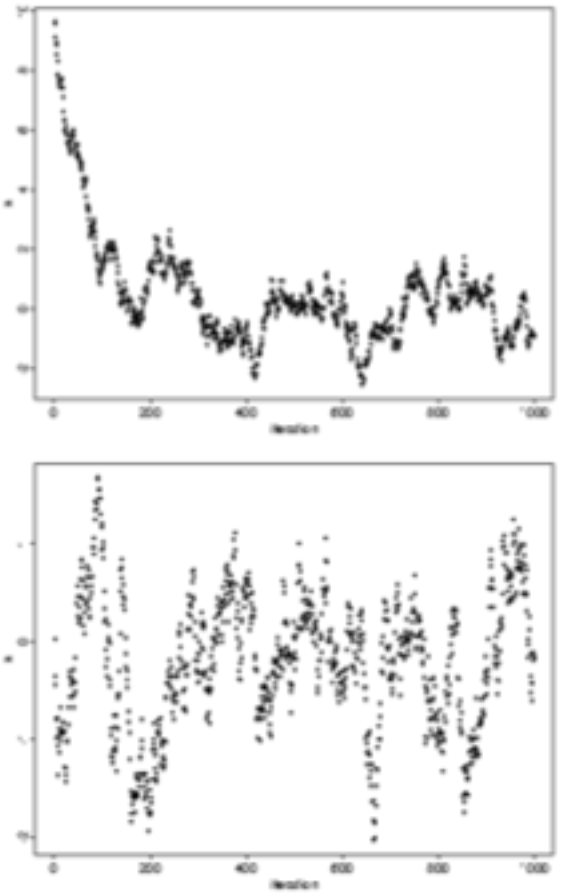


Figure: Theo Rekatsinas, https://thodrek.github.io/CS839_fall18/lectures/lecture_14/Lecture_14.pdf

Burn-in & Thinning

- Controversial techniques that each try to solve a problem
- Problem 1: samples start at a random state that might be highly unlikely and skew the distribution
  - *Burn-in*/warm-up: tossing the first $N' < N$ samples
  - Alternatives
    - Start at highly likely state if known
    - Start at state that a previous run ended in
- Problem 2: as the next state depends on the previous one, the samples are no longer independent (autocorrelation)
  - *Thinning*/subsampling: only taking every k'th sample
    - Does not really solve problem



A set of random variables following a mean-zero normal distribution; started at $x = 10$ and $x = 0$

Figures: Charles Geyer: Burn-in Is Unnecessary. http://users.stat.umn.edu/~geyer/mcmc/burn.html
William A. Link and Mitchell J. Eaton: On Thinning of Chains in MCMC. In Methods in Ecology and Evolution, 2011. https://besjournals.onlinelibrary.wiley.com/doi/10.1111/j.2041-210X.2011.00131.x

Other Problems with Gibbs Sampling

- Only very local moves over the state space
  - One randvar at a time
- In models with tightly correlated randvars, such moves can lead from highly likely states to states with very low probability
  - With a high probability of moving back to the high-probability state
  - Chain is unlikely to move away from such a state
    - Chain will mix slowly

→ Consider chains that allow broader range of moves including larger steps

→ Have to construct such a Markov chain with the same/desired stationary distribution

Metropolis-Hastings Algorithm (MH)

- Construct a Markov chain that is reversible with a particular stationary distribution
- Does not assume that we can generate next-state samples from a particular target distribution but uses the idea of a proposal distribution
  - c.f. importance sampling for proposal distribution
    - Target distribution: next-state sampling distribution at a desired state
    - Sample from proposal distribution and correct for error
  - But: do not keep track of importance weights
    - Are going to decay exponentially with number of transitions
  - Instead: randomly choose whether to accept a proposed transition with a probability that corrects for the difference between proposal and target distribution

W.K. Hastings: Monte Carlo Sampling Methods Using Markov Chains and Their Applications. In *Biometrika*, 1970.

Proposal Distribution in MH

- Proposal distribution $\mathcal{T}^Q$ defines a transition model over state space $\mathcal{R}(\boldsymbol{R})$
  - For each state $\boldsymbol{r}$, $\mathcal{T}^Q$ defines a distribution over possible successor states in $\mathcal{R}(\boldsymbol{R})$, from which one randomly selects a candidate next state $\boldsymbol{r}'$
    - Either accept proposal and transition to $\boldsymbol{r}'$
    - Or reject proposal and stay at $\boldsymbol{r}$
  - For each pair of states $\boldsymbol{r}, \boldsymbol{r}'$, there exists an *acceptance* probability $\mathcal{A}(\boldsymbol{r} \rightarrow \boldsymbol{r}')$
  - Actual transition model of Markov chain:

$$\mathcal{T}(\boldsymbol{r} \rightarrow \boldsymbol{r}') = \begin{cases} \mathcal{T}^Q(\boldsymbol{r} \rightarrow \boldsymbol{r}')\mathcal{A}(\boldsymbol{r} \rightarrow \boldsymbol{r}') & \boldsymbol{r} \neq \boldsymbol{r}' \\ \mathcal{T}^Q(\boldsymbol{r} \rightarrow \boldsymbol{r}) + \sum_{\boldsymbol{r}' \neq \boldsymbol{r}} \mathcal{T}^Q(\boldsymbol{r} \rightarrow \boldsymbol{r}')\big(1 - \mathcal{A}(\boldsymbol{r} \rightarrow \boldsymbol{r}')\big) & otherwise \end{cases}$$

  - Choice of proposal distribution arbitrary as long as it induces a regular chain

Acceptance Probabilities

- Given a proposal distribution $\mathcal{T}^Q$, select acceptance probabilities to obtain desired stationary distribution
  - Detailed balance equation that has to hold

$$\lambda(\boldsymbol{r})\mathcal{T}^Q(\boldsymbol{r} \to \boldsymbol{r}')\mathcal{A}(\boldsymbol{r} \to \boldsymbol{r}') = \lambda(\boldsymbol{r}')\mathcal{T}^Q(\boldsymbol{r}' \to \boldsymbol{r})\mathcal{A}(\boldsymbol{r}' \to \boldsymbol{r})$$

  - Set $\mathcal{A}$ to be

$$\mathcal{A}(\boldsymbol{r} \to \boldsymbol{r}') = \min\left[1, \frac{\lambda(\boldsymbol{r}')\mathcal{T}^Q(\boldsymbol{r}' \to \boldsymbol{r})}{\lambda(\boldsymbol{r})\mathcal{T}^Q(\boldsymbol{r} \to \boldsymbol{r}')}\right]$$

Let $\mathcal{T}^Q$ be any proposal distribution. Consider the Markov chain $\mathcal{T}$ defined by

$$\mathcal{T}(\boldsymbol{r} \to \boldsymbol{r}') = \begin{cases} \mathcal{T}^Q(\boldsymbol{r} \to \boldsymbol{r}')\mathcal{A}(\boldsymbol{r} \to \boldsymbol{r}') & \boldsymbol{r} \neq \boldsymbol{r}' \\ \mathcal{T}^Q(\boldsymbol{r} \to \boldsymbol{r}) + \sum_{\boldsymbol{r}' \neq \boldsymbol{r}} \mathcal{T}^Q(\boldsymbol{r} \to \boldsymbol{r}')\big(1 - \mathcal{A}(\boldsymbol{r} \to \boldsymbol{r}')\big) & otherwise \end{cases}$$

with

$$\mathcal{A}(\boldsymbol{r} \to \boldsymbol{r}') = \min\left[1, \frac{\lambda(\boldsymbol{r}')\mathcal{T}^Q(\boldsymbol{r}' \to \boldsymbol{r})}{\lambda(\boldsymbol{r})\mathcal{T}^Q(\boldsymbol{r} \to \boldsymbol{r}')}\right].$$

If $\mathcal{T}$ is regular, then it has the stationary distribution λ.

MH: Algorithm

- Follows the same procedure as Gibbs sampling except
  - Generate a new state $\boldsymbol{r}_i$ from proposal distribution $\mathcal{T}^Q$ instead of target distribution $\mathcal{T}$
  - Pick or discard $\boldsymbol{r}_i$ based on acceptance probability $\mathcal{A}$

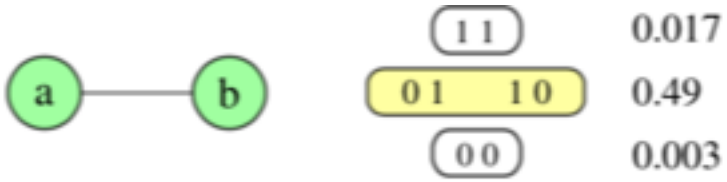
function Gibbs$(R, \boldsymbol{e}, F, N)$ **returns** an estimate of $P(R|\boldsymbol{e})$
  local variables:
    C, a vector of counts over the range values of R, initially 0
    $\boldsymbol{S}$, the non-evidence randvars in F
    $\boldsymbol{r}$, the current state of whole network, initially copied from $\boldsymbol{e}$ with random values for $\boldsymbol{S}$
  for $j = 1$ **to** N **do**
    for each S in $\boldsymbol{S}$ **do**
      Sample the value s of S to replace in $\boldsymbol{r}$ from $P(S|mb(S))$ given the values of $MB(S)$ in $\boldsymbol{r}$
      $C[r] \leftarrow C[r] + 1$ where r is the value of R in $\boldsymbol{r}$
  return Normalise(W)

Gibbs Sampling

Lifted MCMC

As far as I know at this point

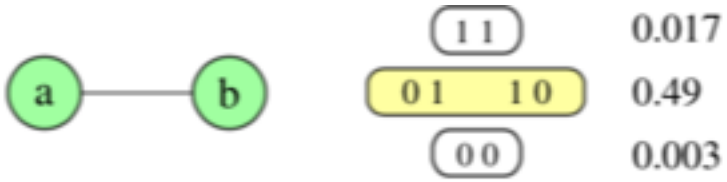
- No direct transformation of MCMC to lifted models
  - But: application of the lifting idea to Markov chains
- Exchangeable Boolean randvars $\boldsymbol{R}$
  - If for every assignment to all randvars in $\boldsymbol{R}$, i.e., $\boldsymbol{r} \in \{0,1\}^k$, and every permutation $\mathfrak{g}$ on $\{0,1\}^k$,
$$P(\boldsymbol{R} = \boldsymbol{r}) = P(\boldsymbol{R} = \boldsymbol{r}^{\mathfrak{g}})$$
  - Can find these so called automorphism groups using *colour passing*
- Then, there are $k + 1$ orbits each containing the randvar assignments
  - Here: Orbit ≈ equivalence class where elements within each class are mapped to the same probability



Mathias Niepert: Markov Chains on Orbits of Permutation Groups. In *UAI-12 Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, 2012.

Why Do Orbits help?

- Two Boolean random variables and a symmetric potential function
  - Probabilities of states 01 and 10 both 0.49
    - States 01 and 10 part of the same orbit
- Assume a standard Gibbs sampler is in state 10
  - Probability to transition to 11 or 00 is only 0.02
  - Cannot transition directly to state 01 (two changes)
  - Chain is "stuck" in 10 until it is able to move to 11 or 00
- With orbital Gibbs sampler, intuitively, while it is "waiting" to move to one of the low probability states, it samples the two high probability states horizontally uniformly at random from the orbit $\{01, 10\}$
  - Converges faster than standard Gibbs sampler
    - Can show analytically



Mathias Niepert: Markov Chains on Orbits of Permutation Groups. In *UAI-12 Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, 2012.

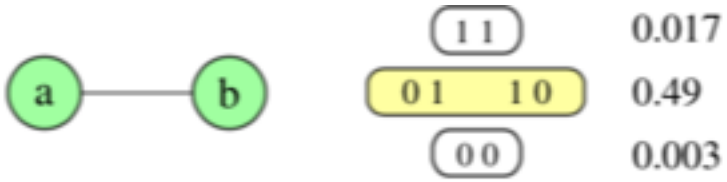
Orbital Markov Chain

- Assume a standard Markov chain M' over state space $\mathcal{R}(\boldsymbol{R})$ with stationary distribution λ
- Let $\mathfrak{G}$ be an automorphism group on $(\mathcal{R}(\boldsymbol{R}), \lambda)$
- Orbital Markov chain M for M' performs:
  - Let $\boldsymbol{r}'$ be the state of M' at time t
  - Sample $\boldsymbol{r}$, the state of M at time t, uniformly at random from the orbit ${\boldsymbol{r}'}^{\mathfrak{G}}$ of $\boldsymbol{r}'$
- If M' is aperiodic/irreducible/reversible, then M also aperiodic/irreducible/reversible
- So, we can build a Gibbs sampler that converges to stationary distribution λ at least as fast or faster

Mathias Niepert: Markov Chains on Orbits of Permutation Groups. In *UAI-12 Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, 2012.

Orbital Gibbs Sampling

- Two Markov chains,
  - One ordinary M'
  - One orbital M (based on symmetry groups)
- In each sampling iteration
  1. Run a step of traditional MCMC, chain M'
     - Select a randvar R uniformly at random
     - Sample a value for R based on the current states of M
  2. Sample the state of M uniformly at random from the orbit of the new state of R,
     i.e., select an equivalent state uniformly at random



Mathias Niepert: Markov Chains on Orbits of Permutation Groups. In *UAI-12 Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, 2012.

Lifted MH

Given an orbital Metropolis chain A:

- Given symmetry group G (approx. symmetries)
- Orbit $\boldsymbol{x}^G$ contains all states approx. symmetric to $\boldsymbol{x}$
- In state $\boldsymbol{x}$
  1. Select $\boldsymbol{x}'$ uniformly at random from $\boldsymbol{x}^G$
  2. Move from $\boldsymbol{x}$ to $\boldsymbol{x}'$ with probability $\min\left\{\frac{Pr(\boldsymbol{x})}{Pr(\boldsymbol{x}')}, 1\right\}$
  3. Otherwise: stay in $\boldsymbol{x}$ (reject)
  4. Repeat

Colour passing

and an ordinary (base) Markov chain B

- With prob. α follow B
- With $(1-\alpha)$ follow A

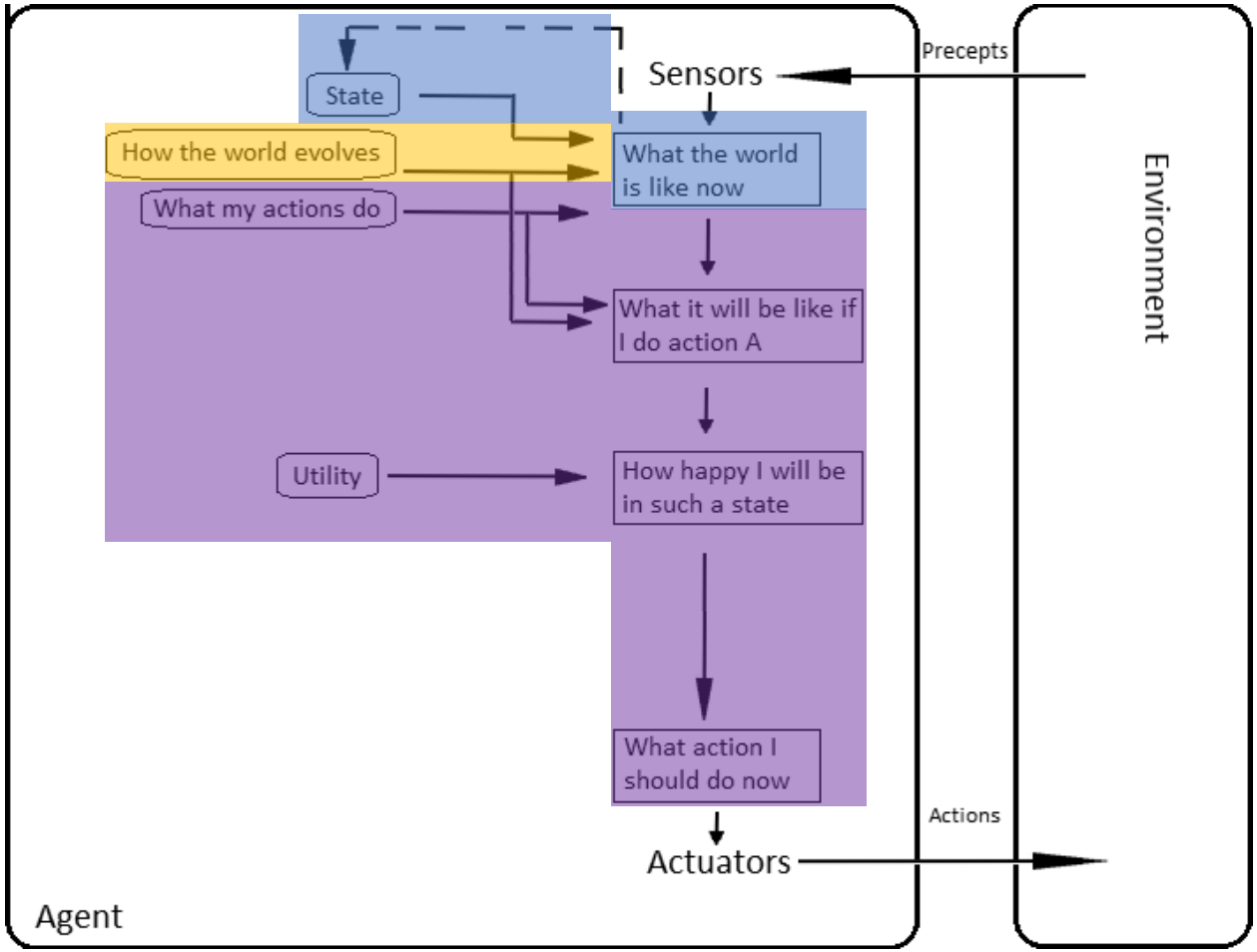
Account for evidence that may break symmetries, using e.g. approx. symmetries

Guy Van den Broeck and Mathias Niepert: Lifted Probabilistic Inference for Asymmetric Graphical Models . In *AAAI-15 Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 2015.

Interim Summary

- Approximate inference based on sampling and counting helps to overcome complexity of exact inference
- Goodness of approximation depends on the number of samples generated
- Importance sampling
  - Use proposal distribution for sampling, weight samples to correct the difference between proposal distribution and target distribution
  - Use domain knowledge about groups of indistinguishable instances to reduce variance
- MCMC sampling
  - Build a Markov chain and sample a new state based on the previous state
  - Find orbits for faster convergence

Setting: Agent with Utilities



Agents: Monte Carlo vs. Las Vegas

- Agent has to work with available resources, requires an answer in a given time T
- Monte Carlo → Approximate inference (sampling)
  - The best possible but not necessarily correct result that could be generated in the given time
- Las Vegas → Exact inference
  - Either get the correct result in the given time or bust!
- Combine Monte Carlo & Las Vegas
  - While current time $t < T$
    - One thread works on exact inference
    - One thread works on approximate inference
  - If exact inference produces a result before t reaches T, break and return result
  - Otherwise: use result of approximate inference at T

Outline: 5. Approximate Inference

A. (Lifted) sampling

- Importance sampling: Likelihood weighting, importance sampling, lifted importance sampling (LIS)
- Markov Chain Monte Carlo (MCMC) sampling: Gibbs Sampling, Metropolis Hastings, Lifted MCMC

⇒ Next: Sequential Models & Inference