Likelihood Weighting and Importance Sampling

Sargur Srihari
srihari@cedar.buffalo.edu

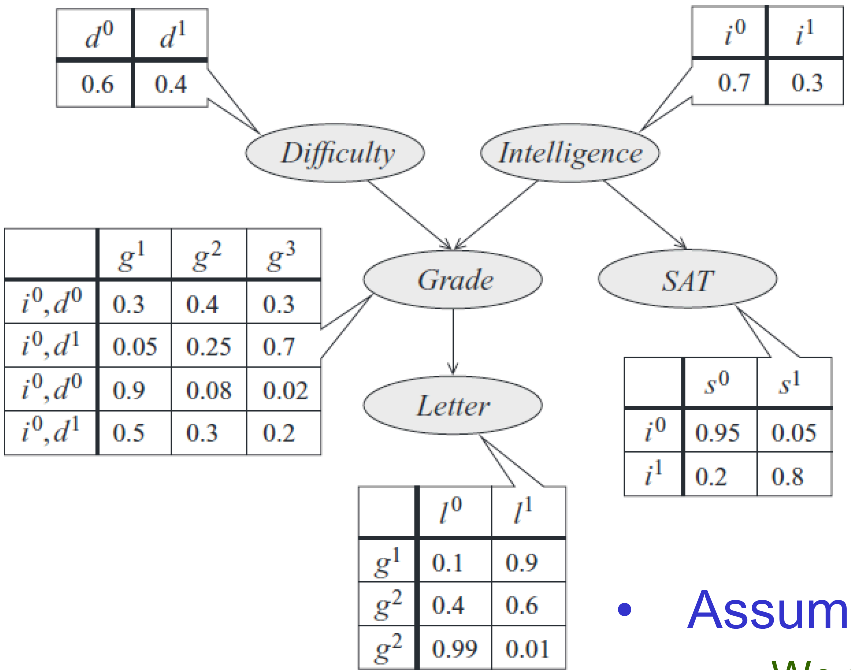
Topics

- Likelihood Weighting Intuition
- Importance Sampling
  - Unnormalized Importance Sampling
  - Normalized Importance Sampling
  - Importance Sampling for Bayesian Networks
  - Mutilated network proposal distribution
  - Computing answers to queries
  - Quality of importance sampling estimator

Inefficiency of Rejection Sampling

- Rejection sampling process is very wasteful in the way it handles evidence
  - We generate multiple samples that are inconsistent with our evidence
  - They are ultimately rejected without contributing to our estimate
- Here we consider an approach that makes our samples more relevant to our evidence

Inefficiency of Rejection Sampling



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

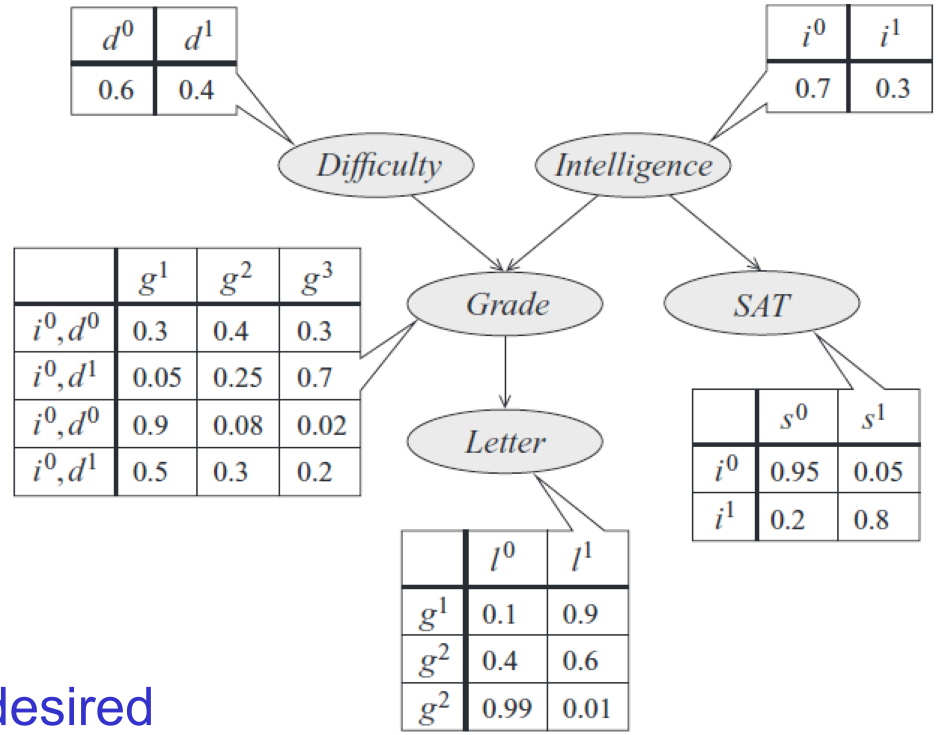
	g^1	g^2	g^3
i^0,d^0	0.3	0.4	0.3
i^0,d^1	0.05	0.25	0.7
i^0,d^0	0.9	0.08	0.02
i^0,d^1	0.5	0.3	0.2

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^2	0.99	0.01

- Assume that our evidence is: d^1, s^1
  - We use forward sampling
  - Say it generates: d^0 for D
  - This sample will always be rejected as being incompatible with evidence
- Better approach may be to force samples to match evidence
  - Value takes on only appropriate observed values
  - But it can generate incorrect values as shown next

Forcing Sample Value



- Example of naiive forcing process

  Evidence is $S=s^1$ (student with high SAT)

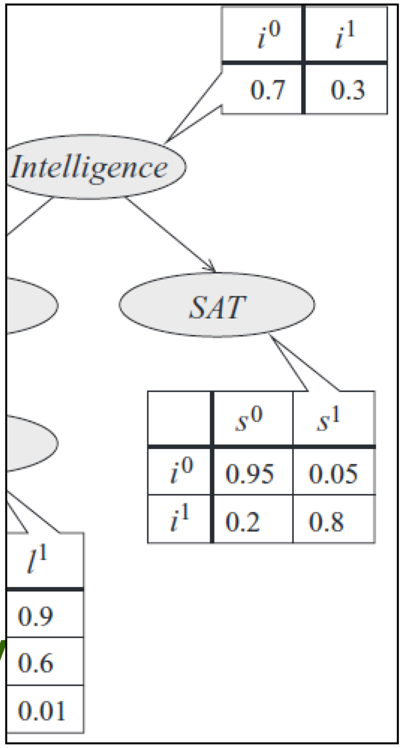
  1. Using naiive process we sample D and I
  2. Set $S=s^1$
  3. Then sample G and L appropriately

- All of our samples will have $S=s^1$ as desired
  - However expected no of samples that have i^1 (an intelligent student) will have probability 30%, same as prior
  - Thus it fails to conclude that the posterior of i^1 is higher when we observe s^1

    $P(i^1|s^1)=0.875$

Shortcoming of Forcing

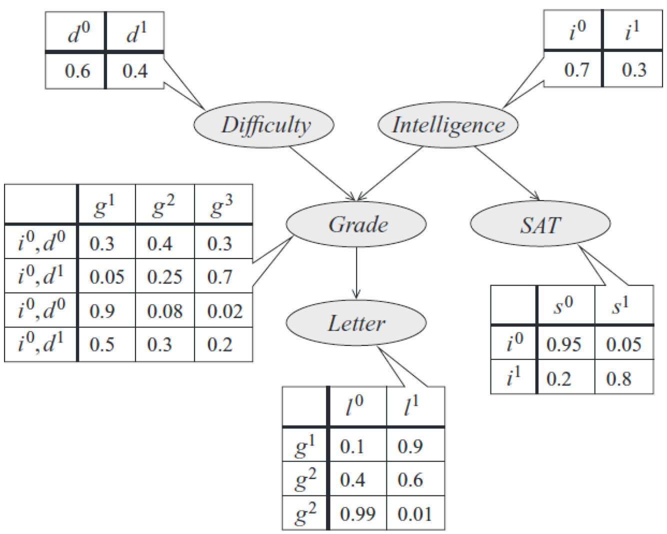
- Forcing $S=s^1$ fails to account for:
  - Node S is more likely to take value $S=s^1$ when parent $I=i^1$ than when $I=i^0$
- Contrast with rejection sampling:
  - With $I=i^1$ we would have generated $S=s^1$ 80% of the time
  - With $I=i^0$ would have generated $S=s^1$ only 5% of the time



– To simulate this long-run behavior with forcing
  – Sample where we have $I=i^1$ and force $S=s^1$ should be 80% of a sample whereas $I=i^0$ and force $S=s^1$ is worth 5% of a sample

Intuition of Weighting

- Weights of samples = likelihood of evidence accumulated during sampling process



  - Evidence consists of: l^0, s^1
  - Using forward sampling, assume that we sample $D=d^1,\ I=i^0$
  - Based on evidence, Set $S=s^1$
  - Sample $G=g^2$
  - Based on evidence, Set $L=l^0$
  - Total sample is: $\{D=d^1,\ I=i^0,\ G=g^2,\ S=s^1,\ L=l^0\ \}$
  - Compensation for evidence l^0, s^1 which are not sampled
    - Given $I=i^0$, forward sampling would have generated $S=s^1$ with probability 0.05
    - Given $G=g^2$ forward sampling would have generated $L=l^0$ with probability 0.4
  - Each event is the result of an independent coin toss
    - probability that both would have occurred is their product
  - Weight required for this sample to compensate for the setting for the evidence is $0.05 \times 0.4 = 0.02$

Generalizing this intuition leads to the Likelihood Weighting (LW) algorithm

Algorithm: Likelihood Weighting (LW)

- Algorithm LW-Sample is shown next
- Here weights of samples derived from likelihood of evidence accumulated during sampling process
- This process generates a weighted particle
- It is used M times to generate a set $\mathcal{D}$ of weighted particles

Likelihood-weighted particle generation

- **Procedure** LW-Sample (
  $\mathcal{B}$, // Bayesian Network over χ
  $\boldsymbol{Z}=\boldsymbol{z}$ // Event in the network)
- Let $X_1,..,X_n$ be a topological ordering of χ
  $w \leftarrow 1$
- **for** $i=1,..,n$
  $\boldsymbol{u}_i \leftarrow \boldsymbol{x}(\mathrm{Pa}X_i)$ // $\boldsymbol{u}_i$ is assigned parents of X_i among $x_1,..,x_{i-1}$
  if $X_i \notin \boldsymbol{Z}$ **then**
  - Sample x_i from $P(X_i|\boldsymbol{u}_i)$

  else
  - $X_i \leftarrow \boldsymbol{z}(X_i)$ //Assignment to X_i in $\boldsymbol{z}$
  - $w \leftarrow w \bullet P(x_i|\boldsymbol{u}_i)$ //Multiply weight by probability of desired value
- **return** $(x_1,..,x_n), w$

Process generates a single weighted particle given event $\boldsymbol{Z}=\boldsymbol{z}$

Likelihood-Weighted (LW) Particles

- Using $\mathrm{LW\ sample}$ to estimate conditional $P(\boldsymbol{y}|\boldsymbol{e})$
  - Use it M times to generate a set $\mathcal{D}$ of weighted particles $(\xi[1],w[1]),\ldots,(\xi[M],\ w[M])$
  - We then estimate

$$\hat{P}_D(\boldsymbol{y}\mid\boldsymbol{e})=\frac{\sum_{m=1}^{M} w[m]\boldsymbol{I}\{\boldsymbol{y}[m]=\boldsymbol{y}\}}{\sum_{m=1}^{M} w[m]}$$

    - Generalizes expression for unweighted particles with forward sampling

$$\hat{P}_D(\boldsymbol{y})=\frac{1}{M}\sum_{m=1}^{M} I\{\boldsymbol{y}[m]=\boldsymbol{y}\}$$

      - Each particle had weight 1; hence terms of numerator unweighted; denominator is sum of all particle weights which is M
  - As in forward sampling same set of samples can be used to estimate $P(\boldsymbol{y})$

LW: a case of Importance Sampling

- We have not yet provided a formal justification for the correctness of LW
- It turns out that LW is a special case of a very general approach: *Importance Sampling*
- It also provided the basis for analysis
- We first give general description and analysis of importance sampling
- Then reformulate LW as a special case of the framework

Importance Sampling

- Let $\boldsymbol{X}$ be a set of variables that take on values in some space $Val(\boldsymbol{X})$
- Importance sampling is a general approach for estimating the expectation of a function $f(\boldsymbol{x})$ relative to some distribution $P(\boldsymbol{X})$ typically called the *target distribution*
- We can estimate this expectation by generating samples $\boldsymbol{x}[1],..,\boldsymbol{x}[M]$ from P

Sampling from a Different Distribution

- Given samples $\boldsymbol{x}[1],..,\boldsymbol{x}[M]$ we can estimate expectation of f relative to P by

$$\boxed{E_P[f] = \frac{1}{M}\sum_{i=1}^{M} f(\boldsymbol{x}[m])}$$

- We might prefer to generate samples from a different distribution Q known as the
  - *proposal distribution* or *sampling distribution*
- Reasons to sample from different distribution:
  - It may be impossible or Computationally very expensive to sample from P
    - Ex: Posterior distribution of a MN or Prior of a MN

Requirements of Proposal Distribution

- We discuss how we might obtain estimates of an expectation relative to P by generating samples from a different distribution Q
- In general proposal distribution can be arbitrary
- We require only that $Q(\boldsymbol{x})>0$ whenever $P(\boldsymbol{x})>0$
  - So that Q does not ignore states that have nonzero probability
    - Support of Q contains the support of P
    - Computational performance depends on the extent to which Q is similar to P

Unnormalized Importance Sampling

- If we generate samples from Q instead of P we cannot simply average the f-value of the samples generated
- We need to adjust our estimator to compensate for the incorrect sampling distribution
- An obvious way of adjusting our estimator is:

$$\boxed{E_{P(\boldsymbol{X})}[f(\boldsymbol{X})] = E_{Q(\boldsymbol{X})}\left[f(\boldsymbol{X})\frac{P(\boldsymbol{X})}{Q(\boldsymbol{X})}\right]}$$

Proof:

$$\begin{aligned} E_{Q(\boldsymbol{X})}\left[f(\boldsymbol{X})\frac{P(\boldsymbol{X})}{Q(\boldsymbol{X})}\right] &= \sum_x Q(\boldsymbol{x})f(\boldsymbol{x})\frac{P(\boldsymbol{x})}{Q(\boldsymbol{x})} \\ &= \sum_x f(\boldsymbol{x})P(\boldsymbol{x}) \\ &= E_{P(\boldsymbol{X})}\left[f(\boldsymbol{X})\right] \end{aligned}$$

Unnormalized Importance Sampling

- Based on the above observation,
  - We generate samples $\mathcal{D}=\{\boldsymbol{x}[1],..,\boldsymbol{x}[M]\}$ from Q and estimate

$$\boxed{\hat{E}_D[f]=\frac{1}{M}\sum_{i=1}^{M} f(\boldsymbol{x}[m])\frac{P(x[m])}{Q(x[m])}}$$

- We call this estimator the unnormalized importance sampling estimator
  - Also called unweighted importance sampling
- The factor $P(\boldsymbol{x}[m])/Q(\boldsymbol{x}[m])$ is a correction weight to the term $f(\boldsymbol{x}[m])$
- We use $w(\boldsymbol{x})$ to denote $P(\boldsymbol{x})/Q(\boldsymbol{x})$

Unnormalized Sampling needs P

- Preceding discussion assumed that P is known
- One of the reasons why we must resort to sampling from a different distribution Q is that P is known only up to a normalizing constant Z
  - Specifically we have access to $\tilde{P}(\boldsymbol{X})$ such that $\tilde{P}(\boldsymbol{X})$ is not a normalized distribution but $\tilde{P}(\boldsymbol{X}) = ZP(\boldsymbol{X})$
    - For example, in a BN $\mathcal{B}$ we might have $P(\boldsymbol{X})$ is our posterior distribution $P_{\mathcal{B}}(\boldsymbol{X}|\mathrm{e})$ and $\tilde{P}(\boldsymbol{X})$ be the unnormalized distribution $P_{\mathcal{B}}(\boldsymbol{X},\mathrm{e})$
    - In a MN $P(\boldsymbol{X})$ might be $P_{\mathcal{H}}(\boldsymbol{X})$ and $\tilde{P}(\boldsymbol{X})$ might be product of clique potentials without normalization

Importance Sampling (Normalized)

- Since we do not know P we define $w(\boldsymbol{X}) = \dfrac{\tilde{P}(\boldsymbol{X})}{Q(\boldsymbol{X})}$
  - But we cannot use $\hat{E}_D[f] = \dfrac{1}{M}\sum_{i=1}^{M} f(\boldsymbol{x}[m])\dfrac{P(x[m])}{Q(x[m])}$
  - We can use a slightly different estimator based on

$$E_{Q(X)}[w(X)] = \sum_x Q(x)\frac{\tilde{P}(x)}{Q(x)} = \sum_x \tilde{P}(x) = Z$$

Its expectation is simply Z

- We can rewrite $E_{P(\boldsymbol{X})}[f(\boldsymbol{X})] = E_{Q(\boldsymbol{X})}\left[f(\boldsymbol{X})\dfrac{P(\boldsymbol{X})}{Q(\boldsymbol{X})}\right]$ as:

$$\begin{aligned} E_{P(\boldsymbol{X})}[f(\boldsymbol{X})] &= \sum_x P(\boldsymbol{x})f(\boldsymbol{x}) = \sum_{\boldsymbol{x}} Q(\boldsymbol{x})f(\boldsymbol{x})\frac{P(\boldsymbol{x})}{Q(\boldsymbol{x})} \\ &= \frac{1}{Z}\sum_{\boldsymbol{x}} Q(\boldsymbol{x})f(\boldsymbol{x})\frac{\hat{P}(\boldsymbol{x})}{Q(\boldsymbol{x})} = \frac{1}{Z}E_{Q(X)}[f(\boldsymbol{x})w(\boldsymbol{X})] = \frac{E_{Q(X)}[f(\boldsymbol{x})w(\boldsymbol{X})]}{E_{Q(X)}[w(\boldsymbol{X})]} \end{aligned}$$

- Can use empirical estimator for denominator and num.
- Given M samples $\mathcal{D}=\{\boldsymbol{x}[1],..,\boldsymbol{x}[M]\}$ from Q

$$\hat{E}_D(f) = \frac{\sum_{m=1}^{M} f(\boldsymbol{x} \mid m)w(\boldsymbol{x}[m])}{\sum_{m=1}^{M} w(\boldsymbol{x}[m])}$$

This is called the normalized importance sampling estimator

Importance Sampling for BNs

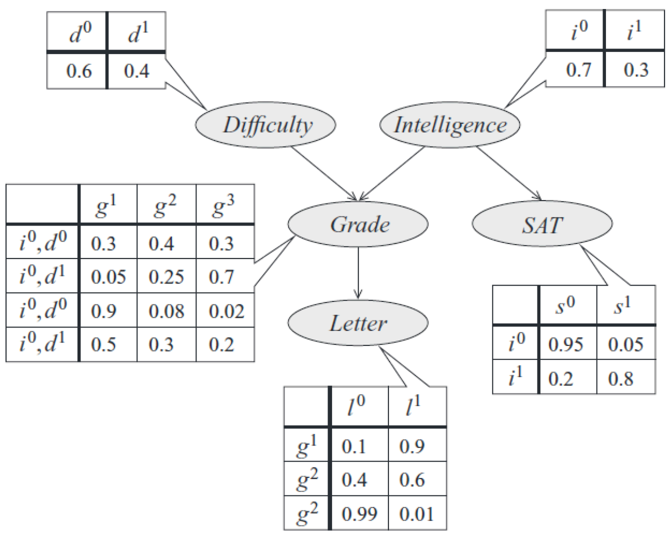
- With the theoretical foundation we can describe application of importance sampling to BNs
- Distribution Q uses network structure/CPDs to focus on part of the joint distribution– the one consistent with a particular event $Z=z$
- Several ways this construction can be applied to BN inference, dealing with various types of probability queries
- Finally discuss several proposal distributions
  – more complex to implement but perform better

Defining the Proposal Distribution

- Assume we are interested in a particular event $Z = z$ either because
  - We wish to estimate its probability, or
  - We have observed it as evidence
- We wish to focus our sampling process on the parts of the joint that are consistent with this event
- We define a process that achieves this goal

Goal of Proposal Distribution

- We are interested in the student's grade $G=g^2$
- We wish to bias our sampling toward parts of the space where this event holds



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0, d^0	0.3	0.4	0.3
i^0, d^1	0.05	0.25	0.7
i^0, d^0	0.9	0.08	0.02
i^0, d^1	0.5	0.3	0.2

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^2	0.99	0.01

Easy to take this into account while sampling L
We simply sample from $P(L|g^2)$

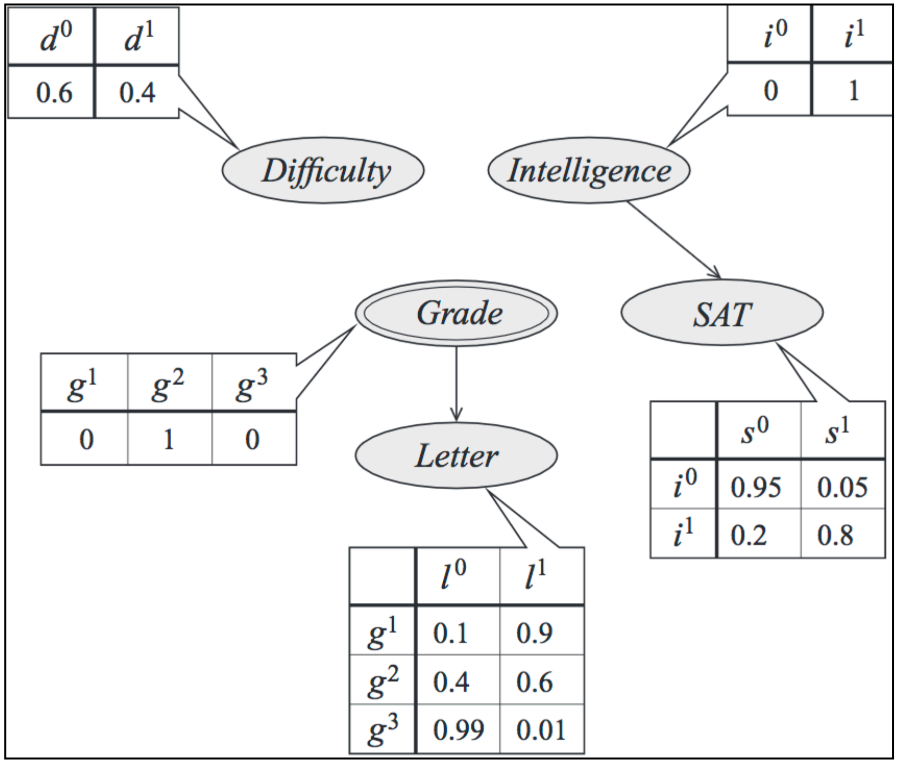
However it is difficult to account for G's influence on D, L and S without doing inference in the network.

- Goal is to define a simple proposal distribution that allows the efficient generation of particles

Mutilated Bayesian Network, $\mathcal{B}_{Z=z}$

- We define a proposal distribution that sets $\boldsymbol{Z}$ to take pre-specified value in a way that influences the sampling process for its descendants but not the other nodes in the network
- Proposal distribution is described as a BN
  - Each node $Z_i \varepsilon \boldsymbol{Z}$ has no parents in $\mathcal{B}_{Z=z}$
  - Parents and CPDs of all other nodes $X \not\varepsilon \boldsymbol{Z}$ are unchanged

Ex: Mutilated Net for $\mathcal{B}_{I=i1,G=g2}$



- Node G is decoupled from its parents, eliminating its dependence on them
- I has no parents in original network
- Both I and G are deterministic ascribing probability 1 to their observed values

Importance sampling with this proposal distribution is precisely equivalent to the *Likelihood Weighting algorithm* seen earlier

If ξ is a sample generated by the LW algorithm and w is its weight. Then

$$w(\xi) = \frac{P_B(\xi)}{P_{B_{Z=z}}(\xi)}$$

Use of the proposal distribution

- Mutilated net can be used for estimating a variety of BN queries

1. Unconditional probability of an event $\boldsymbol{Z}=\boldsymbol{z}$
2. Conditional probability of an event $P(\boldsymbol{y}|\boldsymbol{e})$ for a specific event $\boldsymbol{y}$
3. An entire joint distribution $P(\boldsymbol{Y}|\boldsymbol{e})$ for a subset of variables $\boldsymbol{Y}$

Computing Unconditional of Event

- Simple problem of computing the unconditional probability of an event $Z=z$
  - We can clearly use forward sampling
  - We can also use unnormalized importance sampling where P is defined by $P_{\mathcal{B}}(X)$ and Q is defined by the mutilated network $\mathcal{B}_{Z=z}$
- Our goal is to estimate expectation of a simple function f which is the indicator function of a query z: $f(\xi)=I\{\xi(Z)=z\}$
  - The estimator for this case is simply

$$P_D(z)=\frac{1}{M}\sum_{m=1}^{M} I\{\xi[m](Z)=z\}w(\xi[m])=\frac{1}{M}\sum_{m=1}^{M} w[m]$$

Computing the Conditional $P(\boldsymbol{y}|\boldsymbol{e})$

- Compute it as $P(\boldsymbol{y},\boldsymbol{e})/P(\boldsymbol{e})$
  - Called *Ratio Likelihood Weighting* (*Ratio LW*)
- We use unnormalized importance sampling for both numerator and denominator
  - Estimate conditional probability in two phases
  - Use LW algorithm twice,
    - first M times with $\boldsymbol{Y}=\boldsymbol{y},\boldsymbol{E}=\boldsymbol{e}$ to generate $\mathcal{D}$ weighted samples $\{\xi[1],w[1],..,\xi[\mathrm{M}],w[\mathrm{M}]\}$
    - and then M' times with argument $\boldsymbol{E}=\boldsymbol{e}$ to generate $\mathcal{D}'$ samples with weights w'
    - We can then estimate

$$\hat{P}_{D}(\boldsymbol{y}\,|\,\boldsymbol{e}) = \frac{\hat{P}_{D}(\boldsymbol{y},\boldsymbol{e})}{\hat{P}_{D'}(\boldsymbol{e})} = \frac{1/M\sum_{m=1}^{M} w[m]}{1/M'\sum_{m=1}^{M'} w'[m]}$$

Computing the Conditional $P(\boldsymbol{Y}|\boldsymbol{e})$

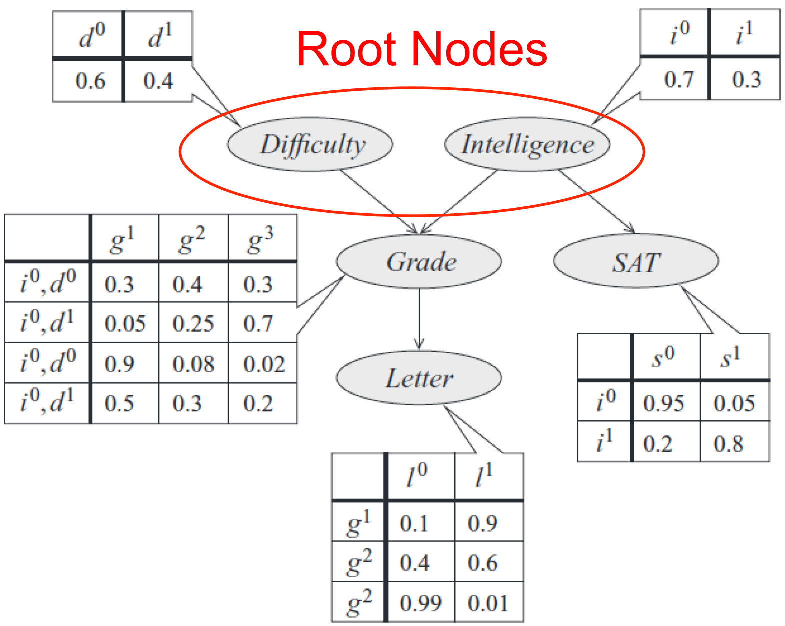
- For a subset of variables Y
- Running Ratio LW for each $\boldsymbol{y} \, \varepsilon \, Val(\boldsymbol{Y})$ is computationally impractical
- An alternative approach is to use is to use normalized likelihood weighting

Quality of Importance Sampling

- Depends on how close the proposal distribution Q is to the target distribution P.
- Extreme cases:
  - All evidence at root nodes
  - All evidence is at leaf nodes
- Discussed next

All evidence at root nodes

- Proposal distribution is precisely the posterior
- No evidence encountered along the way
- All samples will have the same weight $P(\boldsymbol{e})$



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0,d^0	0.3	0.4	0.3
i^0,d^1	0.05	0.25	0.7
i^0,d^0	0.9	0.08	0.02
i^0,d^1	0.5	0.3	0.2

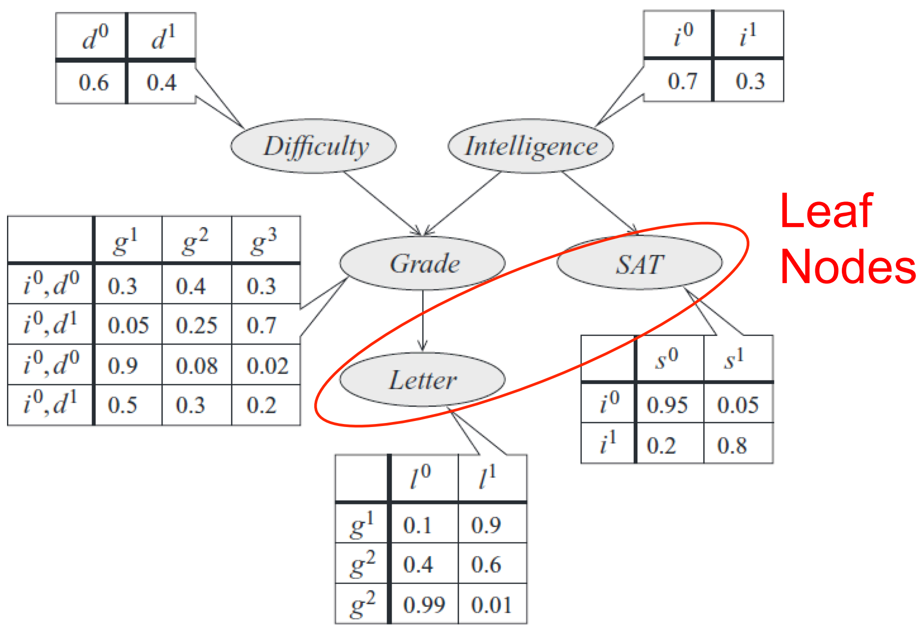
	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^2	0.99	0.01

Q: Posterior: $P(G,S,L|D=d^1,I=i^1)$

All evidence is at leaf nodes

- Proposal distribution $Q(\chi)$ is the Prior distribution $P_{\mathcal{B}}(\chi)$, leaving the correction purely to the weights
- Will work well only if prior is similar to posterior
- Otherwise most samples will be irrelevant



Q: Prior: $P(G,S,L,D,I)$