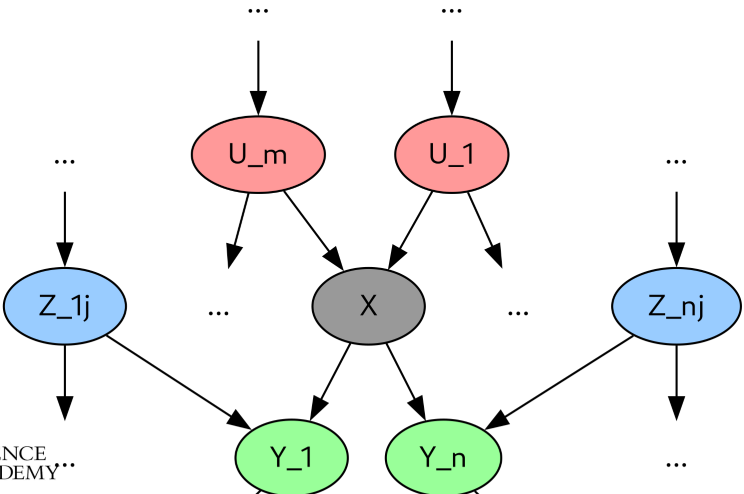
SCIENCE ACADEMY

Bayesian Network: Car Insurance Company (1/2)

- A **car insurance company**:
  - Receive an application from an individual to insure a specific vehicle
  - Analyze information about the individual and its car
  - Decide on appropriate annual premium to charge
  - Pay out a claim, based on the type of claim
- Build a Bayesian network that captures the causal structure of the domain
  - **Input information**:
    - About the applicant: *Age*, *YearsWithLicense*, *DrivingRecord*, *GoodStudent*
    - About the vehicle: *MakeModel*, *VehicleYear*, *Airbag*, *SafetyFeatures*
    - About the driving situation: *Mileage*, *HasGarage*
  - Some input informations are **important but not available**:
    - *RiskAversion*
    - *DrivingBehavior*
  - **Type of claims**:
    - *MedicalCost*: injuries sustained by the applicant
    - *LiabilityCost*: lawsuits filed by other parties against applicant
    - *PropertyCost*: vehicle damage to either party and theft of the vehicle

Bayesian Network: Car Insurance Company (2/2)

- **Blue nodes**: information provided by the applicants
- **Brown nodes**: hidden variables (not observable)
- **Violet nodes**: target variables



SCIENCE ACADEMY

- ***Exact Inference in Bayesian Networks***
- Approximate Inference in Bayesian Networks

Exact Inference in Bayesian Networks

- **Goal of exact inference**
  - Compute the posterior $P(X|\underline{\boldsymbol{E}} = \underline{\boldsymbol{e}})$ for query variable X given evidence $\underline{\boldsymbol{e}}$
- **Variables involved**
  - Query variable X
  - Evidence variables $\underline{\boldsymbol{E}} = \{E_1, \ldots, E_m\}$
  - Hidden variables $\underline{\boldsymbol{Y}} = \{Y_1, \ldots, Y_\ell\}$
- **Inference by Enumeration**
  - Use full joint distribution and sum over all hidden variables:

$$P(X|e) = \alpha \sum_{Y} P(X, e, Y)$$

- **Variable Elimination**
  - Improves efficiency by caching intermediate results
  - Eliminates variables systematically to avoid redundant sums
  - Removing irrelevant variables
    - Variables not ancestors of query or evidence can be ignored
- **Problems**
  - Exact inference is efficient $O(n)$ for trees, but intractable $O(2^n)$ in general
  - It doesn't work for continuous variables
  - Basis for approximate methods when exact is impractical

Exact Inference in Bayesian Networks: Example

- You get a call from both John and Mary, what is the probability of the burglary?

$$P(Burglary|JohnCalls = True, MaryCalls = True)$$

- A conditional probability can be computed summing terms from the full joint distribution

$$\Pr(X|\underline{\boldsymbol{e}}) = \alpha \Pr(X, \underline{\boldsymbol{e}}) = \alpha \sum_{y} \Pr(X, \underline{\boldsymbol{e}}, \underline{\boldsymbol{y}})$$

- Terms of the joint distribution can be written as products of conditional probabilities from the Bayesian network

$$\Pr(b|j, m) = \alpha \Pr(B, j, m) = \alpha \sum_{e} \sum_{a} \Pr(B, j, m, e, a)$$

- Then the joint probability is written in terms of CPTs of the Bayesian network

$$\Pr(b|j, m) = \alpha \sum_{e} \sum_{a} \Pr(b) \Pr(e) \Pr(a|b, e) \Pr(j|a) \Pr(m|a)$$

- Exact Inference in Bayesian Networks
- ***Approximate Inference in Bayesian Networks***

Monte Carlo Algorithms

- **Monte Carlo algorithms** are randomized sampling algorithms used to estimate quantities that are difficult to calculate exactly
  - E.g., samples from the posterior probability of a Bayes network
- **Pros**
  - The accuracy of the approximation depends on the number of samples generated
  - You can get arbitrarily close to the true probability distribution with enough samples
  - Is used in many branches of science
- **Cons**
  - Difficult to understand how the variables interact
  - Computationally intensive

Sampling from Arbitrary Distributions

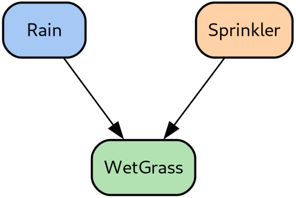
- **Goal**: Sample from a discrete or continuous probability distribution
- **Solution**
  - Start with uniform random numbers $r \in [0, 1]$
  - Construct CDF (cumulative distribution function) $F(x)$
    - $F(x) = \Pr(X \leq x)$
  - For discrete distributions:
    - Create table of outcomes and cumulative probabilities
    - Find smallest outcome where $F(x) > r$
  - For continuous distributions:
    - Use inverse transform: $x = F^{-1}(r)$, e.g.,

$$F(x) = 1 - e^{-\lambda x} \rightarrow x = F^{-1}(r) = -\frac{1}{\lambda} \ln(1 - r)$$

    - If F^{-1} has no closed form, use numerical methods

Sampling Bayesian Network Without Evidence

- **Goal**: Generate events from a Bayesian network without evidence (prior sampling)
- **Solution**
  - Sample variables in topological order (to ensure parents have values)
  - Source nodes have known unconditional probability distribution
    - E.g., $\Pr(Rain) = 0.5$
  - Conditional variable's probability distribution depends on parent's values
    - E.g., $\Pr(WetGrass|Rain = T) = 0.1$
  - Implement Bayesian network semantics, representing joint probability:



$$f_{PS}(x_1, ..., x_n) = \prod_{i=1}^{n} \Pr(x_i|parents(X_i))$$

where PS means "Prior Sampling"

Consistency of Sampling

- **Consistency of estimation**: distribution from prior sampling converges to true probability as $N \to \infty$
- If N_{PS} is the number of times event $x_1, ..., x_n$ occurs:

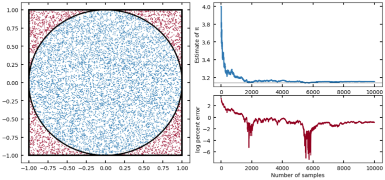
$$\lim_{N\to\infty} \frac{N_{PS}(x_1, ..., x_n)}{N} = \Pr(x_1, ..., x_n)$$

- Estimate probability using:

$$\Pr(x_1, ..., x_n) \approx \frac{N_{PS}(x_1, ..., x_n)}{N}$$

  - Converges with rate $\frac{1}{\sqrt{N}}$

Rejection Sampling



- **Rejection sampling** is a method for sampling from a hard-to-sample distribution
- **Goal**: Compute $\Pr(X = x|E = e)$ when evidence e is rare
  1. Generate samples from the prior distribution
     - Estimate $\Pr(x, e)$
  2. Reject samples not matching evidence, i.e., $X \wedge E \neq e$
     - Remaining samples $X \wedge E = e$ estimate $\Pr(X, E = e)$
  3. Count occurrences of $X = x$ in remaining samples $X \wedge E = e$
     - Estimate $\Pr(X = x|E = e)$
- **Example**:
  - You want to estimate $\Pr(Rain|Sprinkler = T)$
  - Sample 100 times
    - You get 73 samples with $\neg Sprinkler$ and they are rejected
    - You are left with 27 samples with *Sprinkler*
    - Out of them only 8 have *Rain* and 19 have $\neg Rain$

SCIENCE ACADEMY

Rejection Sampling: Pros and Cons

- **Pros**
  - Consistent estimate
    - Converges to true value as number of samples increases
- **Cons**
  - Many samples are rejected, depending on rarity of $\Pr(E = e)$
  - Fraction of samples matching evidence e decreases exponentially with more evidence variables
    - Curse of dimensionality
    - Not suitable for complex systems
  - Difficult with continuous variables
    - E.g., $\Pr(E = e)$ is theoretically 0 due to limited floating-point precision

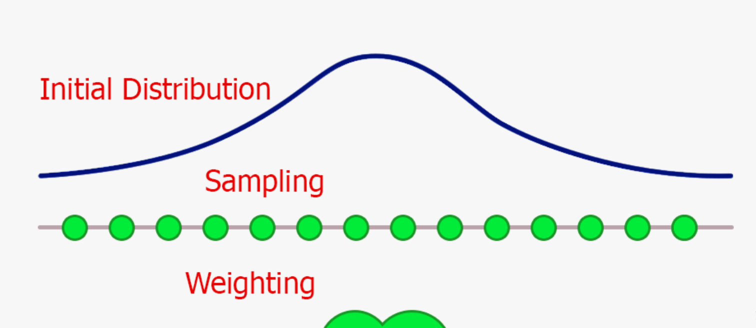
Importance Sampling

::: columns :::: {.column width=60%}

- **Importance sampling**
  - Draw samples from "easier" distribution $Q(X)$
  - Weight each sample by importance weight $w = \frac{\Pr(X)}{Q(X)}$
  - Estimate probability by averaging weighted samples:

$$E[f(X)] \approx \frac{1}{N}\sum_{i=1}^{N} w_i f(X_i)$$

:::: :::: {.column width=35%}



Markov Chain Monte Carlo

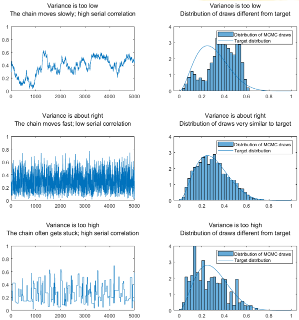
- This is one of the top 10 most mind blowing algorithms in history
  - Euclide's GCD
  - Fundamental theorem of calculus
  - Quicksort
  - Fast Fourier Transform
  - Viterbi algorithm
  - MCMC sampling
  - Kalman filter
  - RSA Algorithm
  - ...
- Invented by Ulam, Von Neumann, Metropolis and others during the Manhattan Project (1940)
  - Used to solve high-dimensional integrals, Bayesian inference, ...
- **Purpose**: Approximate inference for Bayesian networks when exact inference is hard
  - MCMC differs from rejection and importance sampling
  - Make random changes to preceding sample instead of generating each sample independently
  - **Magic**: two very different objects (Markov Chains) and Bayesian Networks are connected

Markov Chain Construction

- A **Markov chain** is a "random walk" through states, where the future depends only on the present
  - *Sequence of states*: $\underline{\boldsymbol{x}}^{(0)}, \underline{\boldsymbol{x}}^{(1)}, \underline{\boldsymbol{x}}^{(2)}, \ldots$
  - *Initial state*: starting configuration
  - *Transition probabilities*: $\Pr(\underline{\boldsymbol{x}} \rightarrow \underline{\boldsymbol{x}}')$
  - After t steps, distribution is $\pi_t(\underline{\boldsymbol{x}})$
  - When $\pi_t(\underline{\boldsymbol{x}}) = \pi_{t+1}(\underline{\boldsymbol{x}})$, chain reaches stationary distribution
- **Transition operator**: moves from one state to another:
  - Gibbs sampling: resample one variable given its Markov blanket
  - Metropolis–Hastings: propose a new state, then accept/reject based on a probability ratio
- There are algorithms generate a Markov Chain from a Bayesian network
- Under certain conditions:
  - Ergodicity: chain can reach any state
  - Aperiodicity: chain does not get stuck in cycles stationary distribution equals the **posterior distribution** over non-evidence variables given evidence

Markov Chain Monte Carlo: Mixing

- **Mixing** describes how quickly a Markov chain forgets its starting point and explores the whole state space efficiently
  - A well-mixed chain:
    - Moves between different high-probability regions often
    - Has low correlation between successive samples
  - Poor mixing:
    - Chain gets stuck in one mode for a long time
    - Leads to biased estimates and high variance
- In practice:
  - Discard initial samples as a burn-in period (before convergence)
  - After convergence, collected samples approximate the true posterior
- **Example**
  - If sampling from a bimodal distribution:
    - "Poor mixing" means the chain stays in one peak
    - "Good mixing" jumps between both



SCIENCE ACADEMY

Gibbs Sampling in Bayesian Networks

- Special case of Markov Chain Monte Carlo (MCMC) method that samples one variable at a time
- **Algorithm**:
  - Start with an initial complete assignment to all non-evidence variables
  - Keep evidence variables fixed at observed values
  - For each non-evidence variable X_i:
    - Sample X_i from $P(X_i|\mathrm{MB}(X_i))$ where $\mathrm{MB}(X_i)$ is the Markov blanket, i.e., parents, children, spouse of a node
- **Example**:
  - Weather network: $P(Cloudy|Sprinkler, Rain, WetGrass)$
  - Fix $WetGrass = true$, $Sprinkler = true$
  - Sample *Cloudy* and *Rain* iteratively
- **Pros**
  - Simple to implement for any Bayesian network
  - Handles large, complex graphs with local updates
- **Cons**
  - Can mix slowly if variables are highly correlated

Metropolis–Hastings Sampling

- More general Markov Chain Monte Carlo method than Gibbs sampling
- **Algorithm**
  - Start at a current state $\underline{\boldsymbol{x}}$
  - Propose a new state $\underline{\boldsymbol{x}}'$ from a proposal distribution $q(x'|x)$, e.g.,
    - With 95% probability Gibbs sampling
    - Otherwise use importance sampling
  - Compute the acceptance probability:
    - $A(x, x') = \min(1, \frac{\pi(x')q(x|x')}{\pi(x)q(x'|x)})$
  - Move to x' with probability $A(x, x')$, otherwise stay at x
- **Intuition**
  - Propose local moves
  - Accept if they lead to higher probability, or sometimes accept lower-probability states to explore
  - Balances exploration and exploitation to avoid getting stuck in local modes
- **Pros**
  - Very flexible: works with any proposal distribution
  - Can handle high-dimensional spaces

SCIENCE ACADEMY