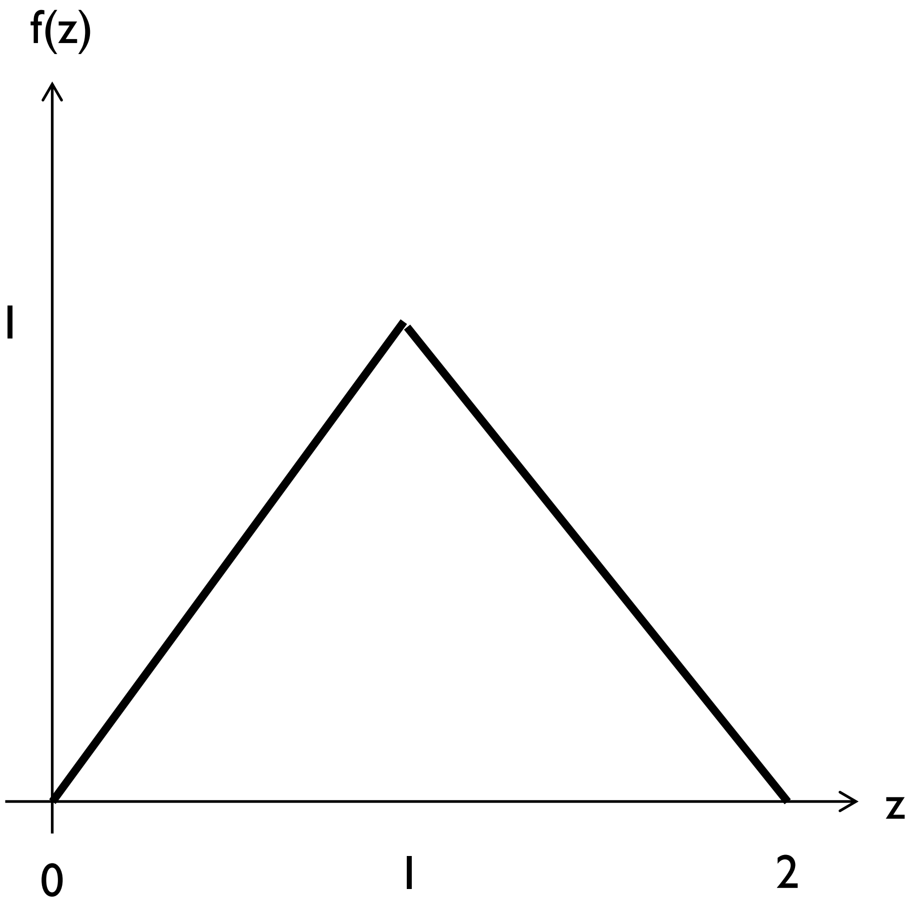
Probabilistic & Approximate Computing

Sasa Misailovic

UIUC

Distribution of sum of two uniforms



$$\begin{cases} 2 - Z & 1 \leq Z \leq 2 \\ Z & 0 \leq Z < 1 \end{cases}$$

```
X := Uniform(0,1)
Y := Uniform(0,1)

Z := X + Y

return Z
```

$ psi sum_uniform.prb

Probabilistic Programs

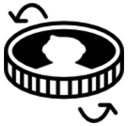
Extend Standard (Deterministic) Programs

Distribution	`X := Uniform(0, 1);`
Assertion	`assert ( X >= 0 );`
Observation	`observe ( X >= 0.5 );`
Query	`return X;`

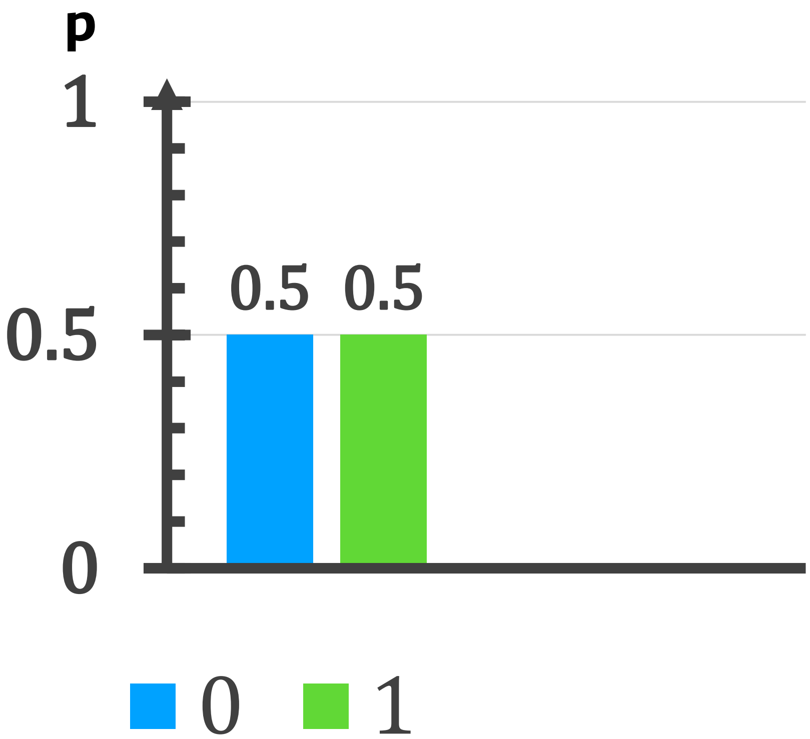
Probabilistic Model

$A \sim Bernoulli(0.5)$

$P(A = 1)$



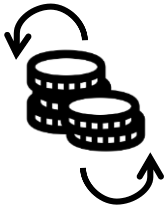
$head: 1$
$tail: \quad 0$



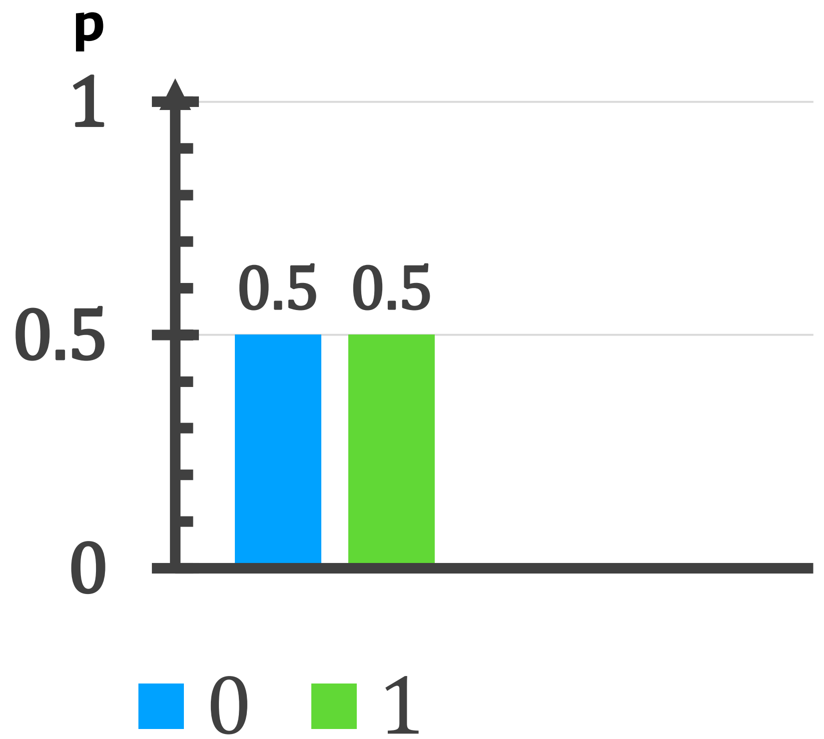
Probabilistic Model

$A \sim Bernoulli(0.5)$
$B \sim Bernoulli(0.5)$
$C \sim Bernoulli(0.5)$

$P(A = 1)$



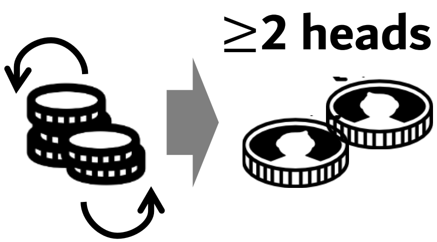
$head: 1$
$tail: \; 0$



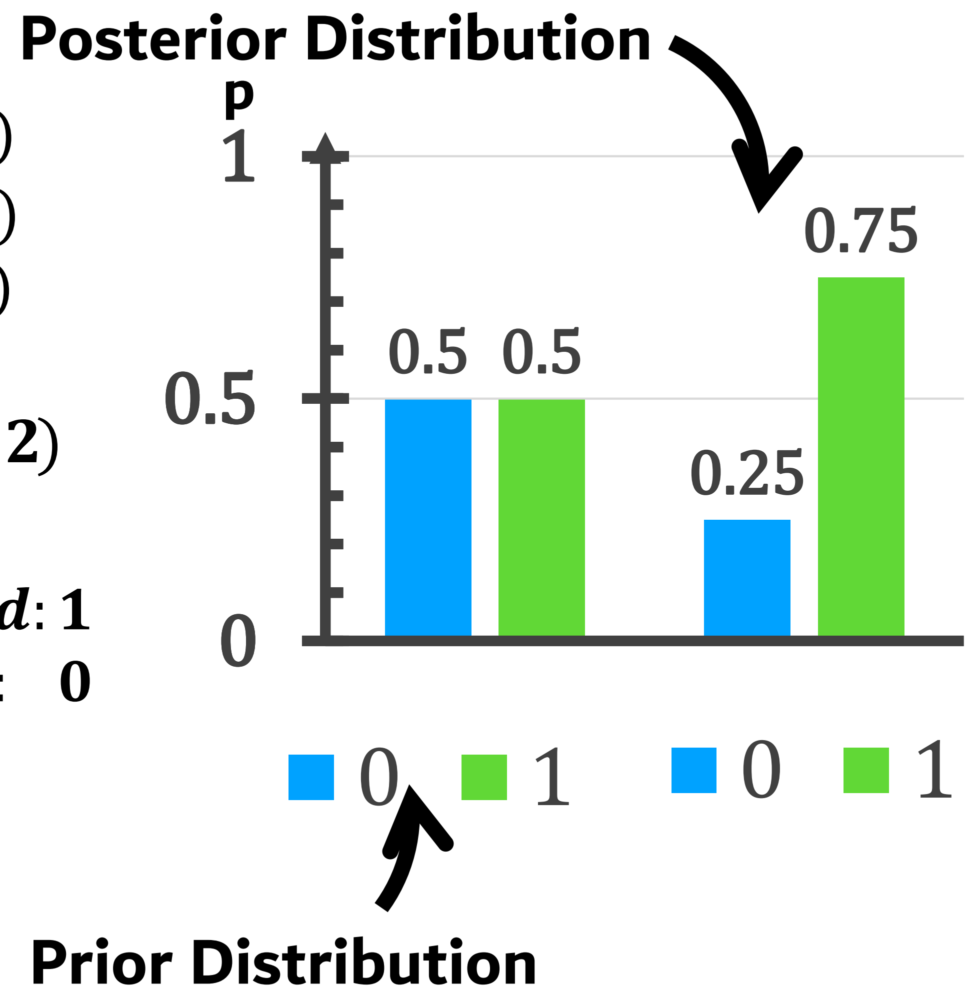
Probabilistic Model

$A \sim Bernoulli(0.5)$
$B \sim Bernoulli(0.5)$
$C \sim Bernoulli(0.5)$

$P(A = 1 | A + B + C \geq 2)$



$head: 1$
$tail: 0$

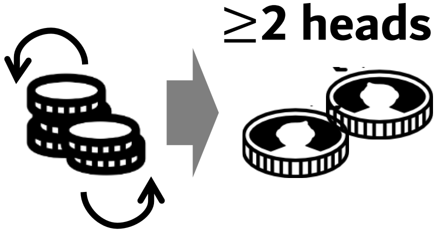


Probabilistic Programming

$$A \sim Bernoulli(0.5)$$
$$B \sim Bernoulli(0.5)$$
$$C \sim Bernoulli(0.5)$$

$$P(A = 1 | A + B + C \geq 2)$$

≥2 heads

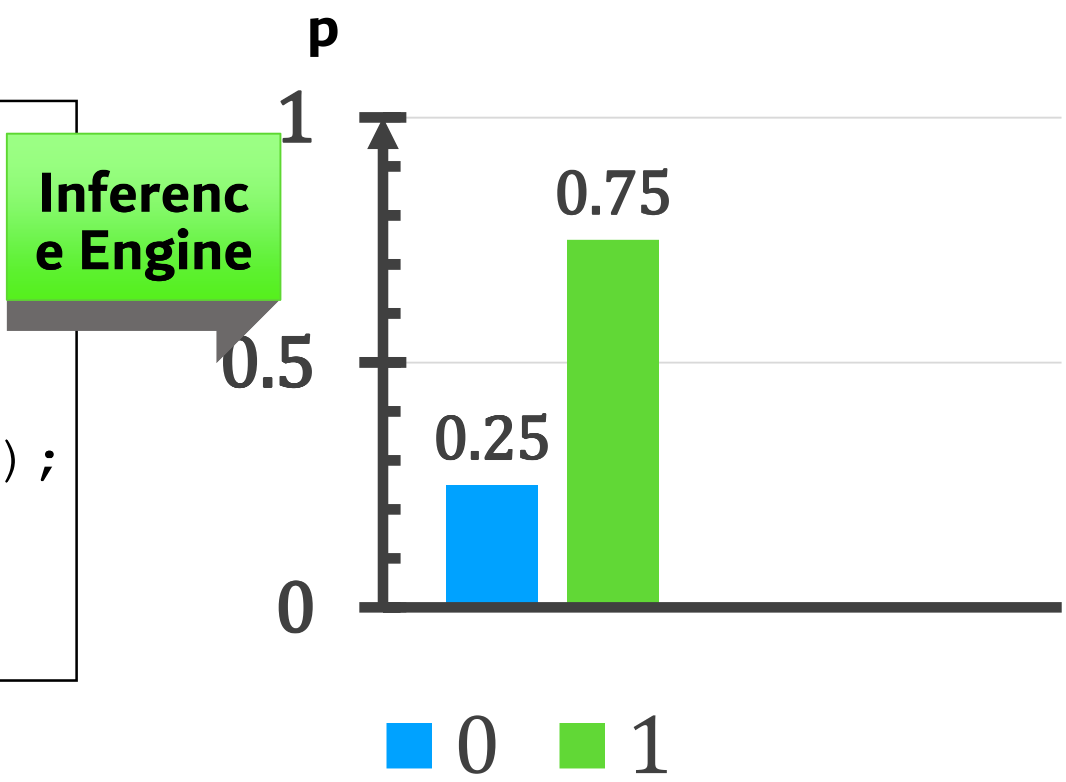


$head$: 1
$tail$: 0

```
def main(){
    A:=flip(0.5);
    B:=flip(0.5);
    C:=flip(0.5);

    observe(A+B+C>=2);
    return A;
}
```

Probabilistic Programming

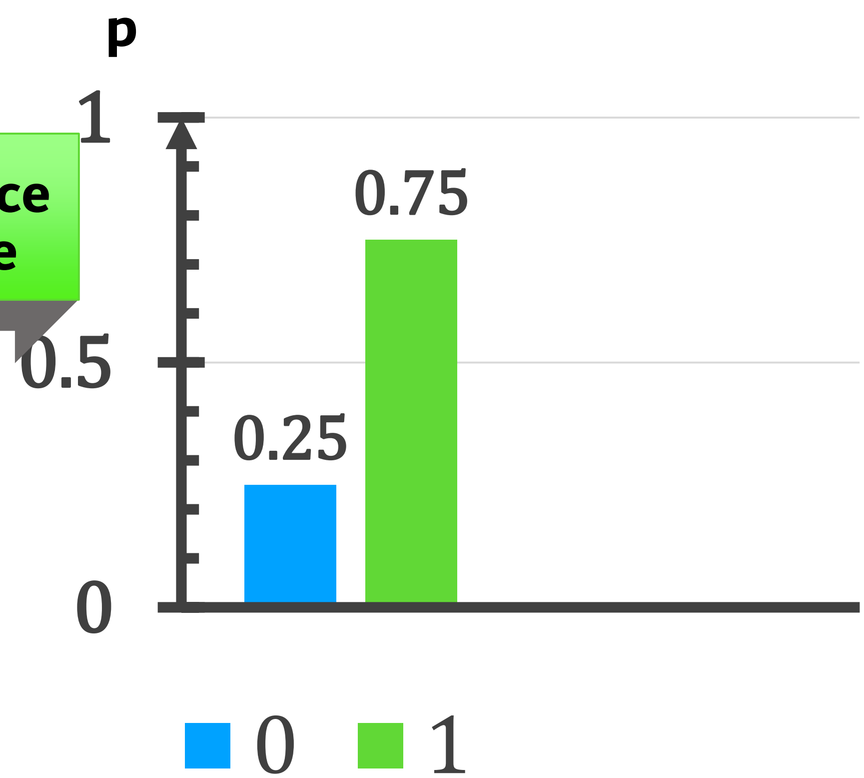
```
def main(){
    A:=flip(0.5);
    B:=flip(0.5);
    C:=flip(0.5);
    observe(A+B+C>=2);
    return A;
}
```



Probabilistic Programming

```
def main(){
    A:=flip(0.5);
    B:=flip(0.5);
    C:=flip(0.5);
    observe(A+B+C>=2);
    return A;
}
```

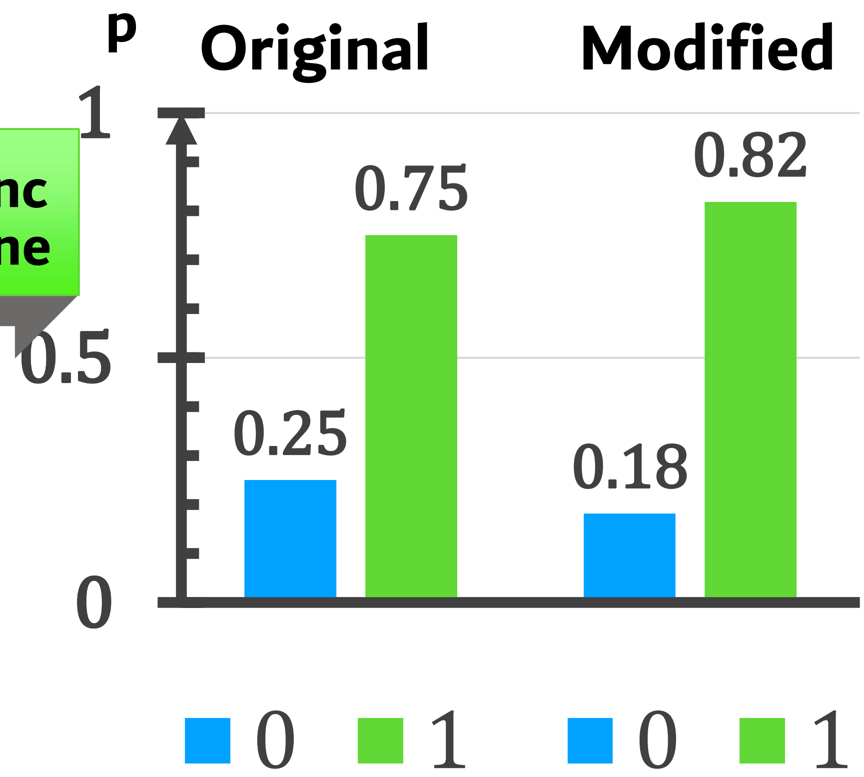
Inference Engine



Probabilistic Programming

```
def main(){
   A:=flip(0.5+0.1);
   B:=flip(0.5);
   C:=flip(0.5);
   observe(A+B+C>=2);
   return A;
}
```

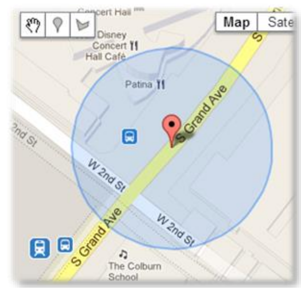
Inference Engine



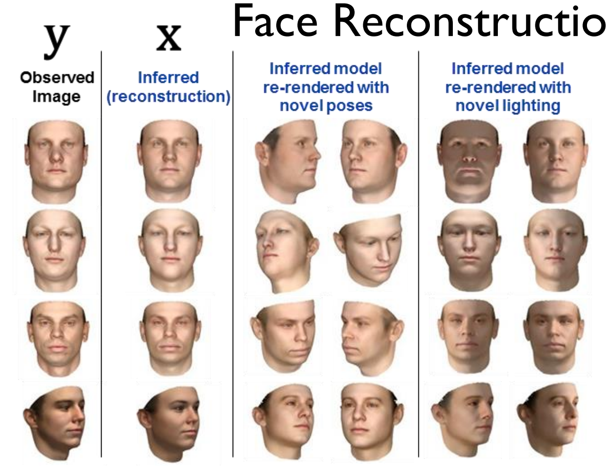
Probabilistic Applications

Modeling of
Complex Systems

GPS & Navigation



Face Reconstruction



Scene labeling

Spam
Filter

Microsoft

XBOX

facebook

UBER

Pyro

Google

Tensorflow

Example Language:

WWW.WEBPPL.ORG

Probability Refresher

2.1. Basic definition.

We define a *probability triple* or *(probability) measure space* or *probability space* to be a triple $(\Omega, \mathcal{F}, \mathbf{P})$, where:

- the *sample space* Ω is any non-empty set (e.g. $\Omega = [0, 1]$ for the uniform distribution considered above);
- the *σ-algebra* (read "sigma-algebra") or *σ-field* (read "sigma-field") $\mathcal{F}$ is a collection of subsets of Ω, containing Ω itself and the empty set $\emptyset$, and closed under the formation of complements* and countable unions and countable intersections (e.g. for the uniform distribution considered above, $\mathcal{F}$ would certainly contain all the intervals $[a, b]$, but would contain many more subsets besides);
- the *probability measure* $\mathbf{P}$ is a mapping from $\mathcal{F}$ to $[0, 1]$, with $\mathbf{P}(\emptyset) = 0$ and $\mathbf{P}(\Omega) = 1$, such that $\mathbf{P}$ is countably additive as in (1.2.3).

Rosenthal J.; A First Look at Rigorous Probability Theory 2 ed.

Probability Refresher

Probability Distribution

- Discrete Distributions
- Continuous Distributions
- Hybrid Joint Distributions

Distribution Function

Probability Distribution Function

Probability Mass Function

Probability Density Function

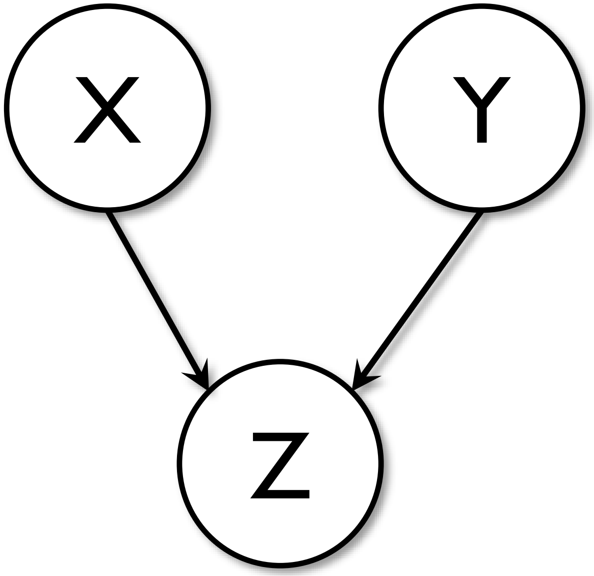
Expectation

Expected value: measure of central tendency

Variance: measure of spread

Probabilistic Programs and Graphical Models

```
X := Uniform(0,1)
Y := Uniform(0,1)

Z := X + Y

return Z
```



Dependency Graph

Bayes' Rule

Belief Revision

Thomas Bayes
1701 –1761

Bayes' Rule

Belief Revision

Hypothesis

$$\Pr(\theta \mid x) = \frac{\Pr(x \mid \theta) \cdot \Pr(\theta)}{\Pr(x)}$$

Data

Bayes' Rule

Belief Revision

Prior Distribution

Likelihood

Posterior Distribution

$$\Pr(\theta \mid x) = \frac{\Pr(x \mid \theta) \cdot \Pr(\theta)}{\Pr(x)}$$

Normalization Constant

Is Our Brain Statistical?*

Probability of sickness is 1%

If a patient is sick, the probability that medical test returns positive is 80% (true positive)

If a patient is not sick, the probability that medical test returns positive is 9.6% (false positive)

For a given patient, the test returned positive.

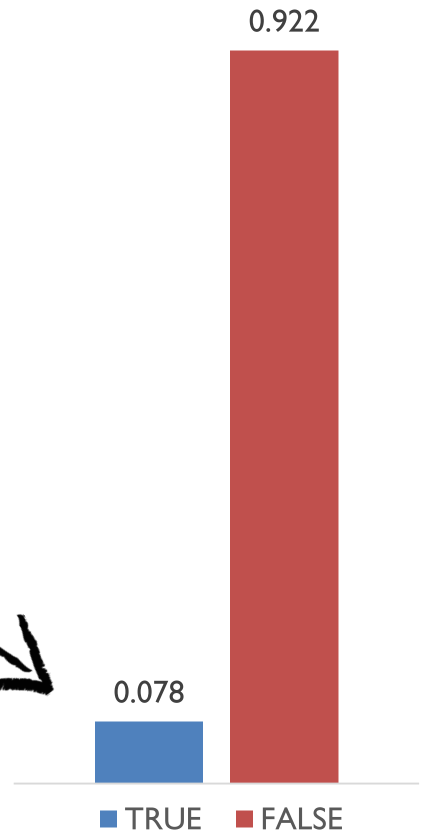
What is the probability that the patient is sick?

* Kahneman and Tversky (1974)

Is Our Brain Statistical?

```
var test_effective = function() {
 var PatientSick = flip(0.01);

 var PositiveTest =
   PatientSick? flip(0.8): flip(0.096);

 condition (PositiveTest == true);

 return PatientSick;
}

Infer ({method: 'enumerate'},
        test_effective)
```

Fallacy:
Base rate
neglect



For discussion: Goodman & Tenenbaum,
Probabilistic Models of Cognition (Ch. 3)

Bayesian Nets

Alternative representation of probabilistic models

Graphical representation of dependencies among random variables:

- Nodes are variables
- Links from parent to child nodes are direct dependencies between variables
- Instead of full joint distribution, now terms $\Pr(X | parents(X))$.

The graph has no cycles! DAG

Queries

Posterior distribution – what we got

Expected value – $$\mathbb{E}(\boldsymbol{X}) = \sum_{\boldsymbol{x} \in \boldsymbol{Dom}(\boldsymbol{X})} \boldsymbol{x} \cdot \Pr(\boldsymbol{x})$$

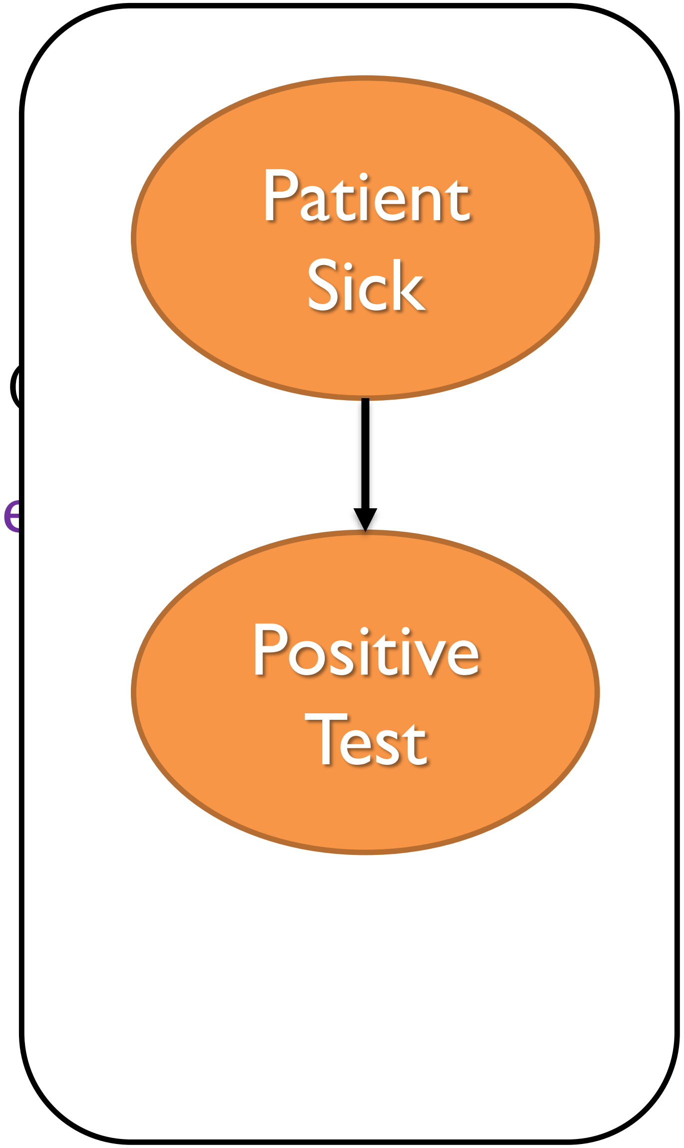
Most likely value – Mode of the distribution

Variable Dependencies

```
var test_effective = function() {
 var PatientSick = flip(0.01);

 var PositiveTest =
   PatientSick? flip(0.8): flip(0.096);

 condition (PositiveTest == true);

 return PatientSick;
}

Infer ({method: 'enumerate'},
        test_effective)
```

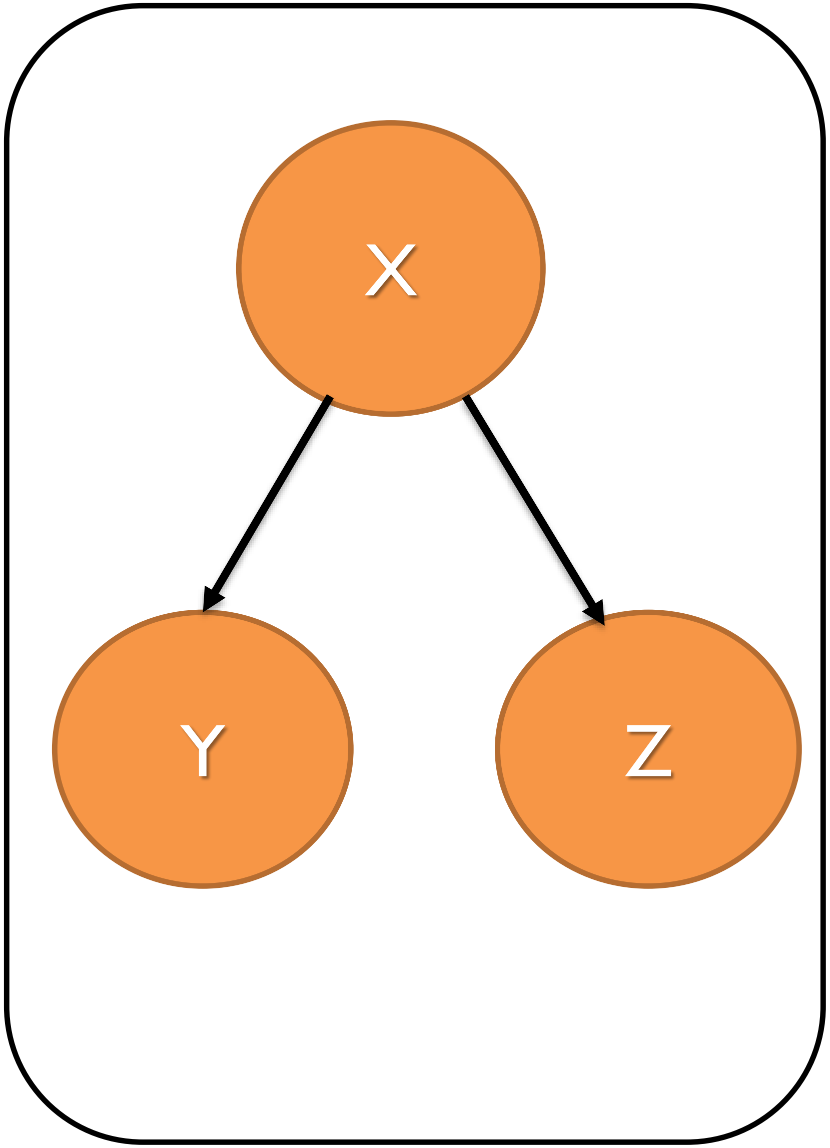
Variable Dependencies

```
var test_effective = function()
 var PatientSick = flip(0.01);

 var PositiveTest =
   PatientSick? flip(0.8): flip(

 condition (PositiveTest == true

 return PatientSick;
}

Infer ({method: 'enumerate'},
        test_effective)
```



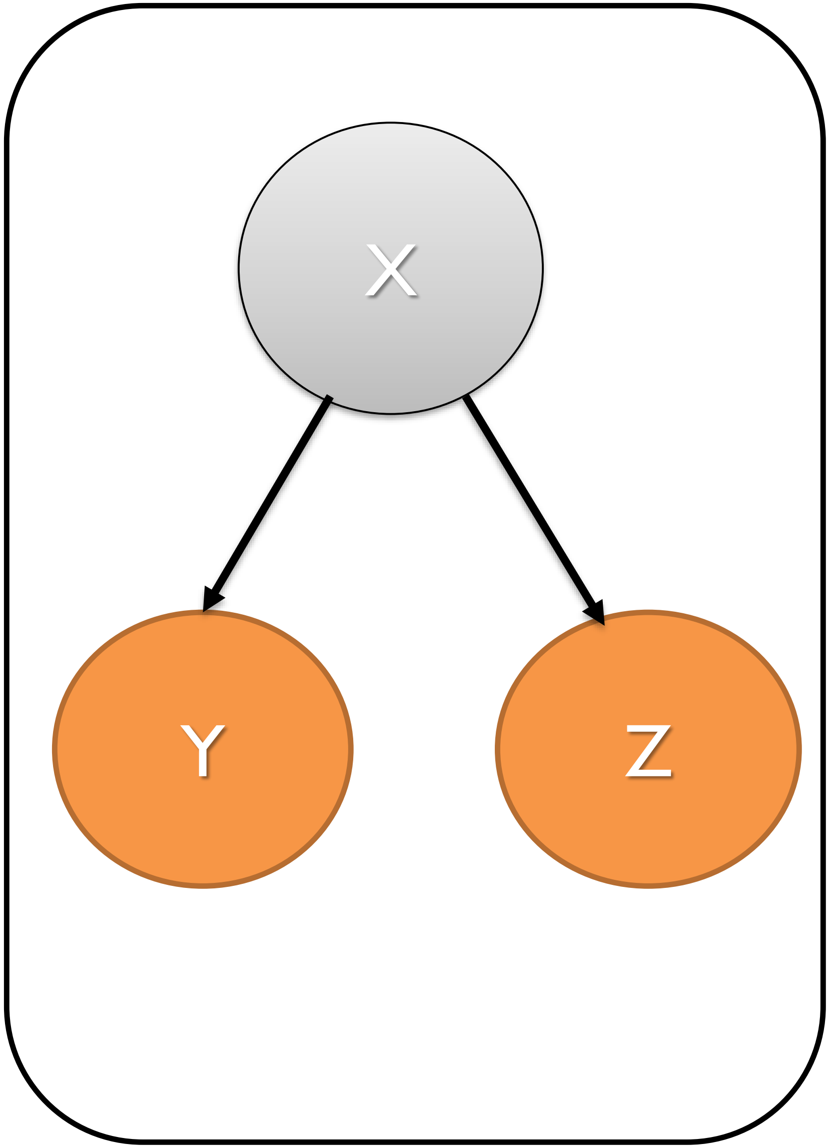
Variable Dependencies

```
var test_x = function() {
 var x = flip(0.50);

 var y = x?
   flip(0.1): flip(0.2);

 var z = x?
    flip(0.3): flip(0.4);

 condition(x == 1)

 return [y, z]
}
```



Variable Dependencies

```
var test_x = function() {
 var x = flip(0.50);

 var y = x?
   flip(0.1): flip(0.2);

 var z = x?
    flip(0.3): flip(0.4);

 condition(x == 1)

 return [y, z]
}
```



Reminder: Independence

Definition:

$$Pr(X, Y) = Pr(X) \cdot Pr(Y)$$

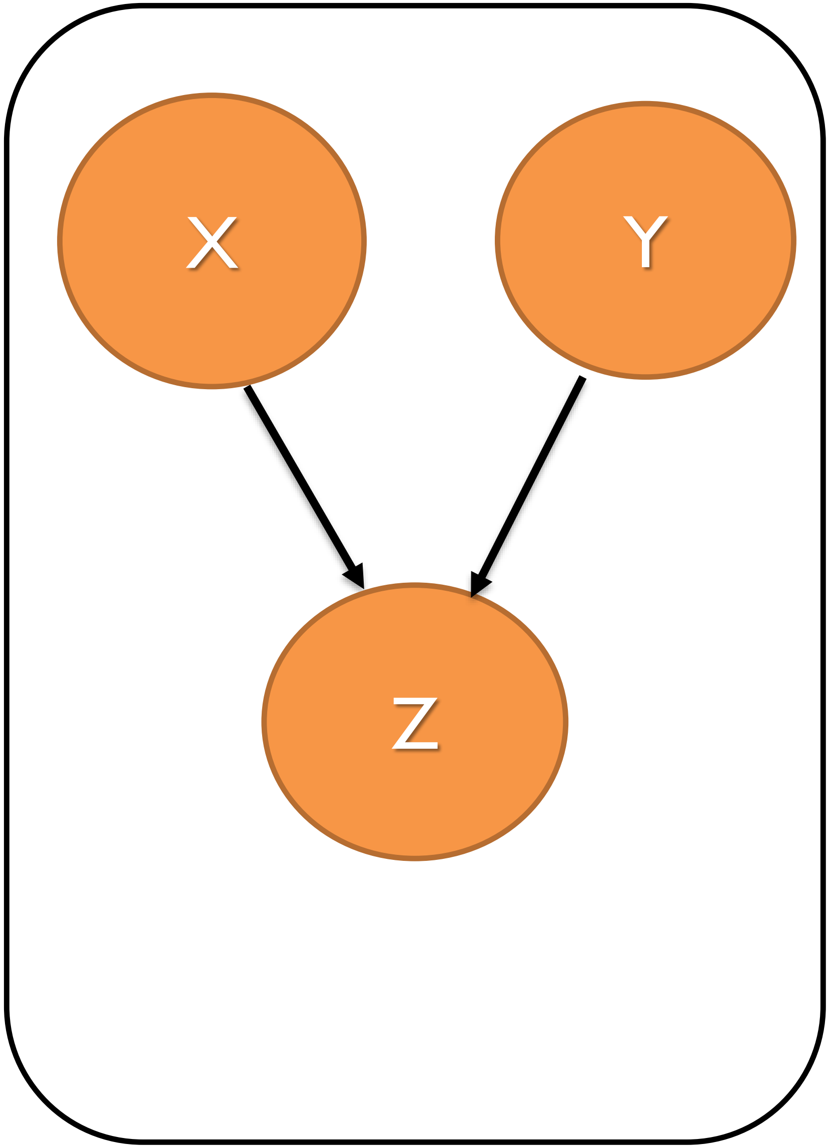
But also*:

$$Pr(X \mid Y) = Pr(X)$$
$$Pr(Y \mid X) = Pr(Y)$$

*Using the fact that for any two variables $Pr(X, Y) = Pr(X|Y) \cdot Pr(Y)$

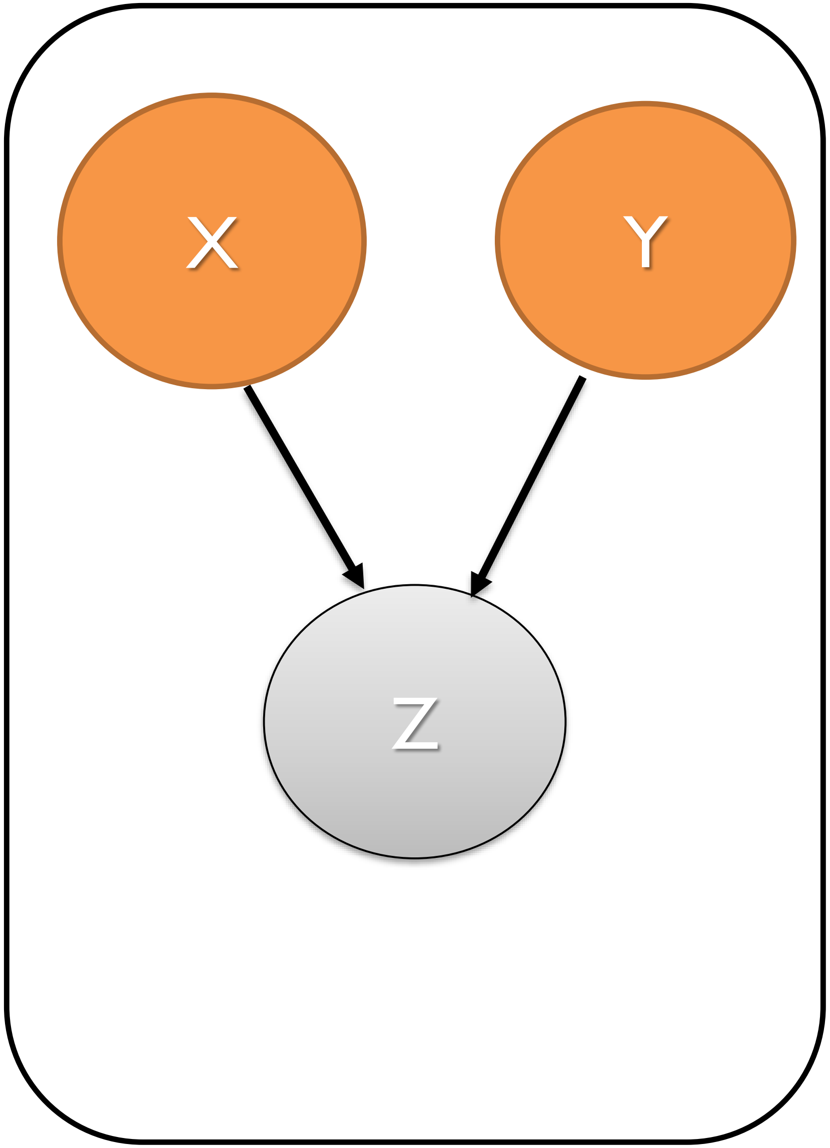
Variable Dependencies

```
var test_z = function(){
 var x = flip(0.50);

 var y = flip(0.1);

 var z = x+y;

 condition(z == 1);

 return x;
}
```



Variable Dependencies

```
var test_z = function(){
 var x = flip(0.50);

 var y = flip(0.1);

 var z = x+y;

 condition(z == 1);

 return x;
}
```



Bayes' Rule

Belief Revision

Prior

Likelihood — Distribution

Posterior

Distribution

$$\Pr(\theta \mid x) = \frac{\Pr(x \mid \theta) \cdot \Pr(\theta)}{\Pr(x)}$$

Normalization

Constant

Bayes' Rule

Belief Revision

Posterior Distribution

Likelihood

Prior Distribution

$$\Pr(\theta \mid x) \sim \Pr(x \mid \theta) \cdot \Pr(\theta)$$

Enough to order different interpretations and select the most likely one

Bayes' Rule

Belief Revision

Posterior Distribution

Likelihood

Prior Distribution

Equvi-probable

$$\Pr(\theta \mid x) \sim \Pr(x \mid \theta) \cdot \Pr(\theta)$$

Enough to order different interpretations and select the most likely one

Bayes' Rule

Belief Revision

Likelihood

Posterior Distribution

$$\Pr(\theta \mid x) \sim \Pr(x \mid \theta)$$

Enough to order different interpretations and select the most likely one

Beyond Bayesian Net Models

Geometric Distribution: Probability of the number of Bernoulli trials to get one success

```
var geometric = function() {
  return flip(.5) ? 0 : geometric() + 1;
}

var dist = Infer({method: 'enumerate', maxExecutions: 10},
                 geometric);

viz.auto(dist);
```

Exact Inference

Naïve approach: Compute $P(x_1, x_2, \dots, x_n)$

Better approach:

Take advantage of (conditional) independencies

- Whenever we can expose conditional independence, e.g., $\mathrm{P}(x_1, \mathrm{x}_2 | \mathrm{x}_3) = \mathrm{P}(\mathrm{x}_1 | \mathrm{x}_3) \cdot \mathrm{P}(\mathrm{x}_2 | \mathrm{x}_3)$ the computation is more efficient

Compute distributions from parents to children

Complexity of Exact Inference

Number of variables: $\boldsymbol{n}$

Naïve enumeration: complexity is $O(2^n)$

Variable Elimination: if the maximum number of parents of the nodes is $k \in \{1, \dots, n\}$, then the complexity is $n \cdot O(2^k)$.

For many models this is a good improvement, but always possible to construct pathological models.