Parameter Learning in Probabilistic Databases

Bernd Gutmann

joint work with

Angelika Kimmig, Kristian Kersting, Luc De Raedt

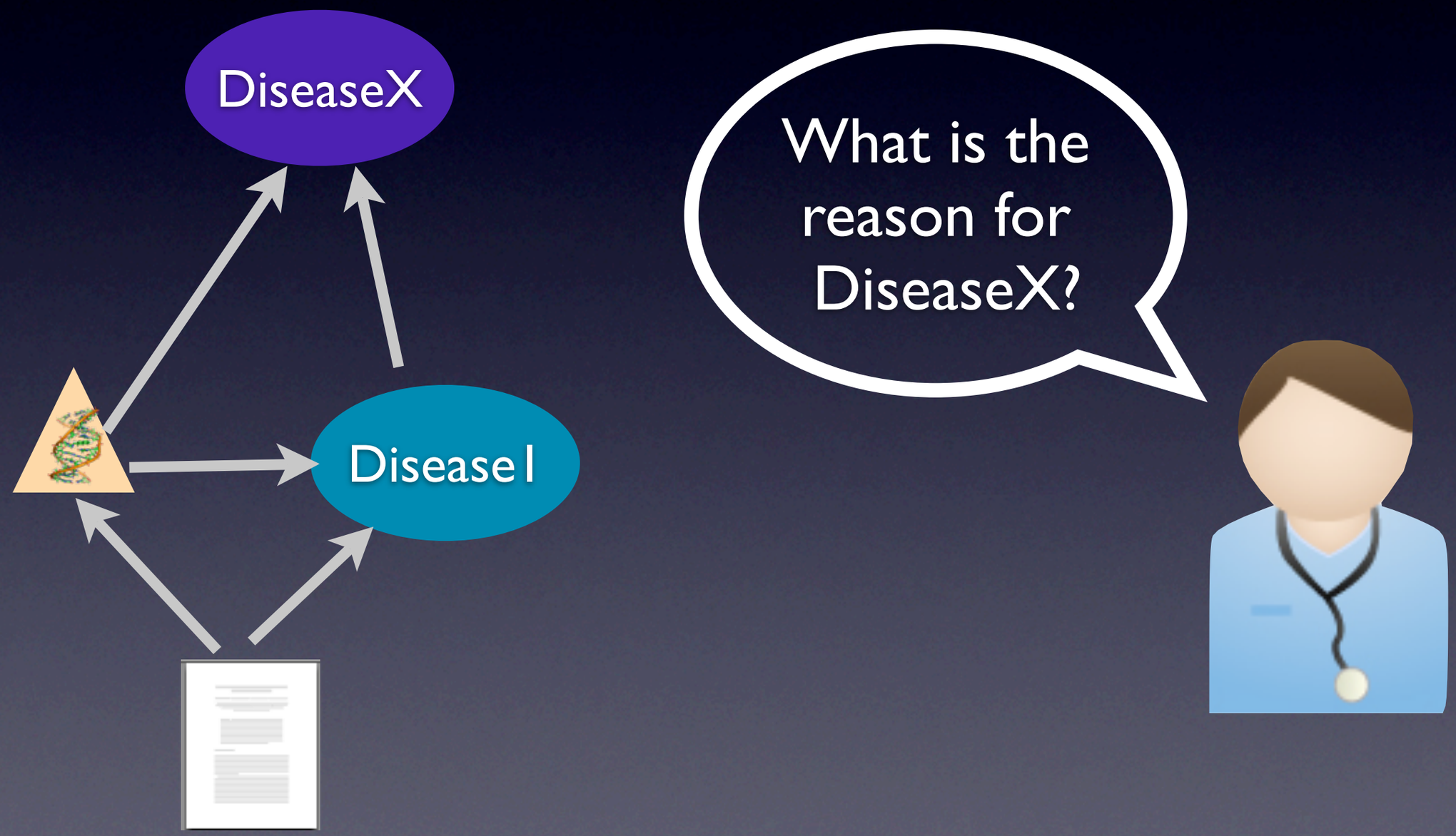
KATHOLIEKE UNIVERSITEIT LEUVEN

DTAI MACHINE LEARNING

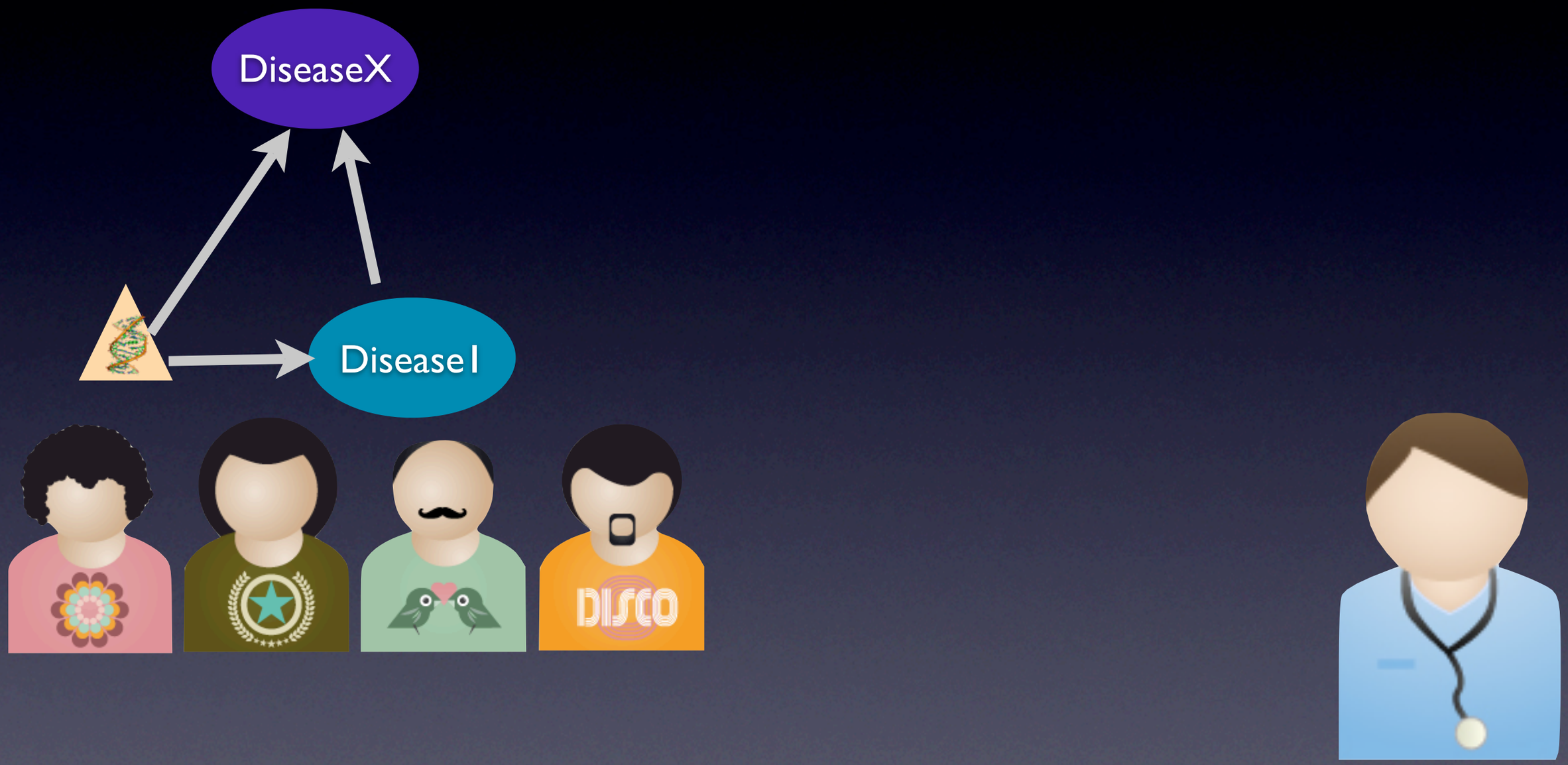
Motivation

Motivation

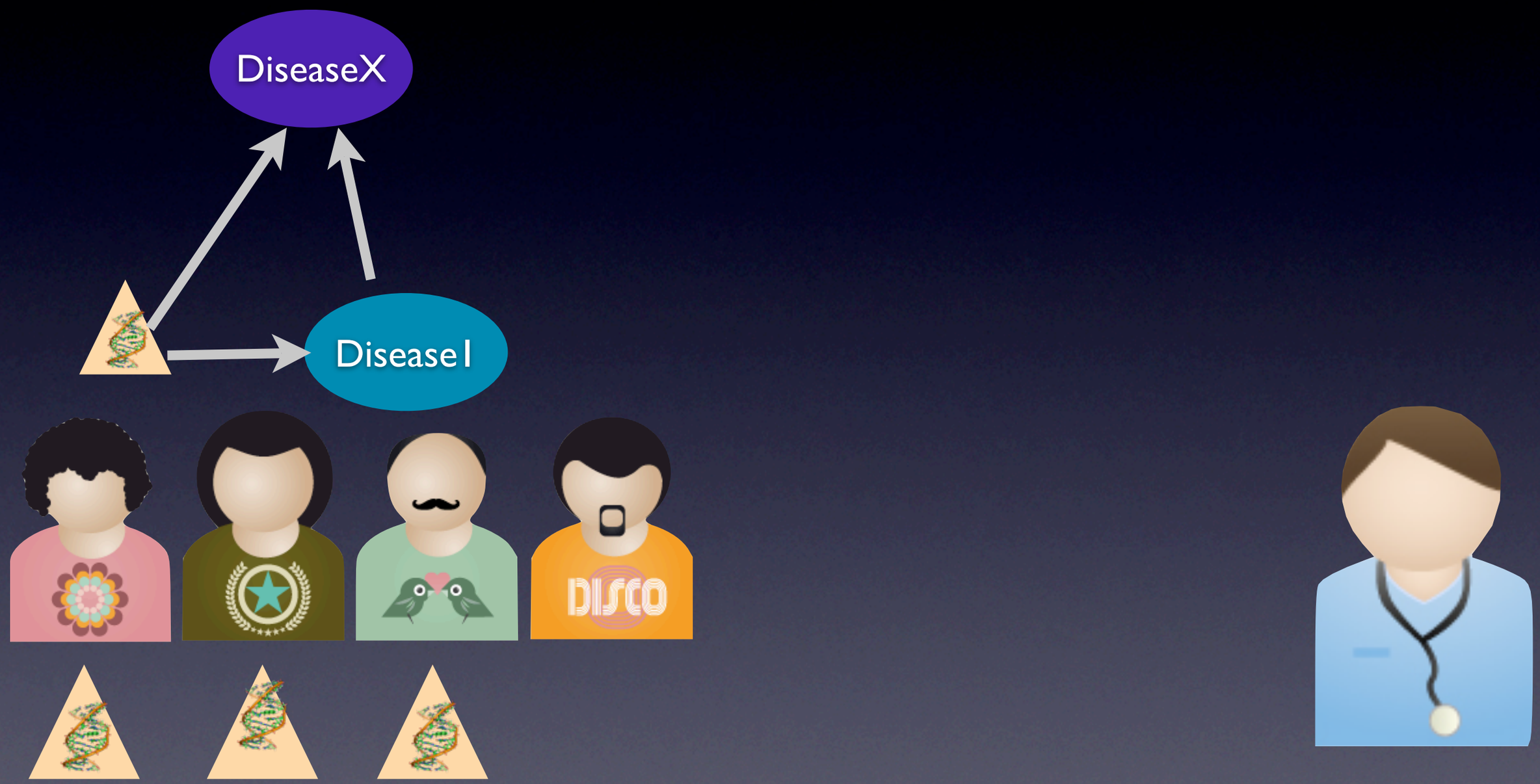
Motivation



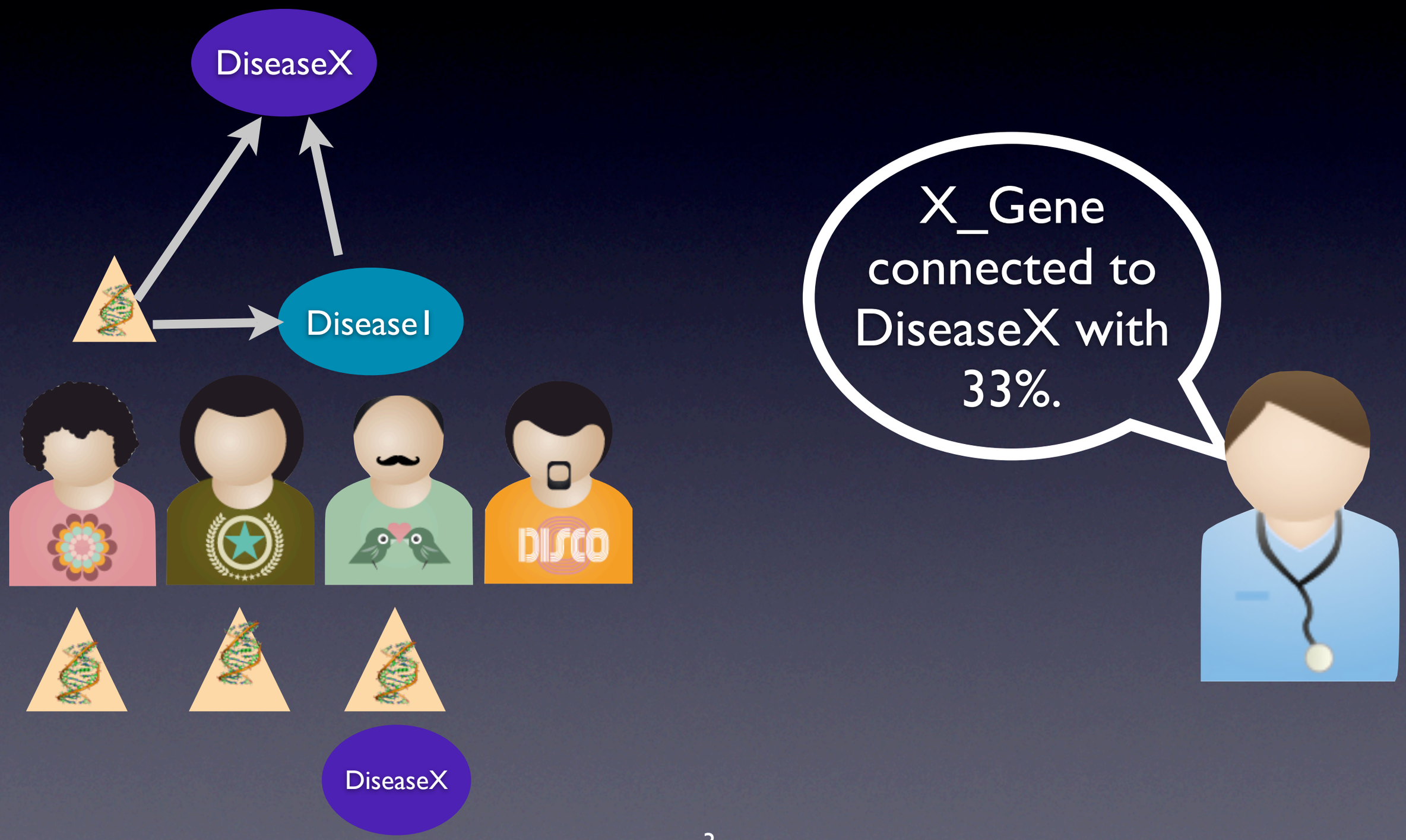
Motivation



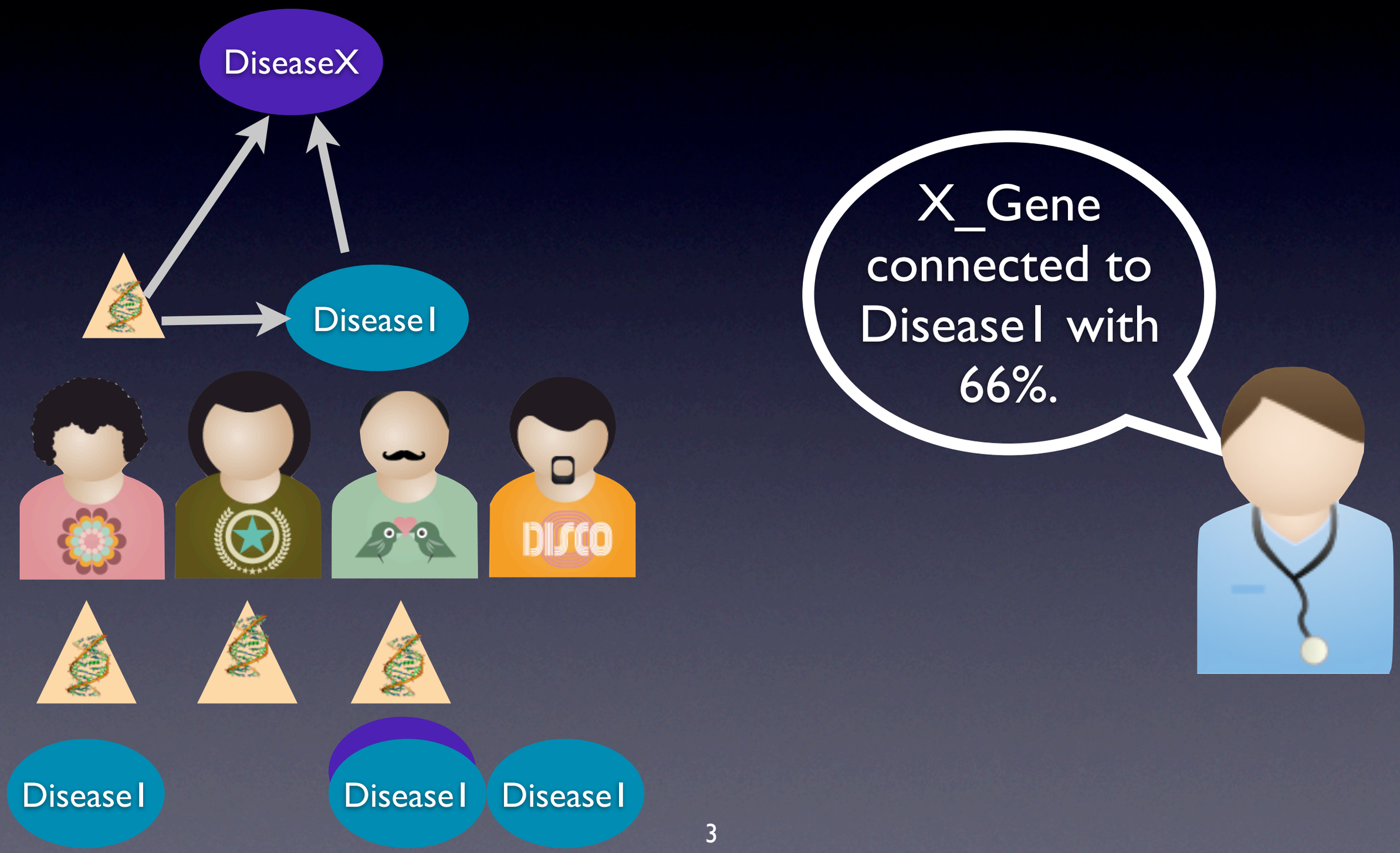
Motivation



Motivation


DiseaseX
Disease1
DiseaseX
X_Gene connected to DiseaseX with 33%.

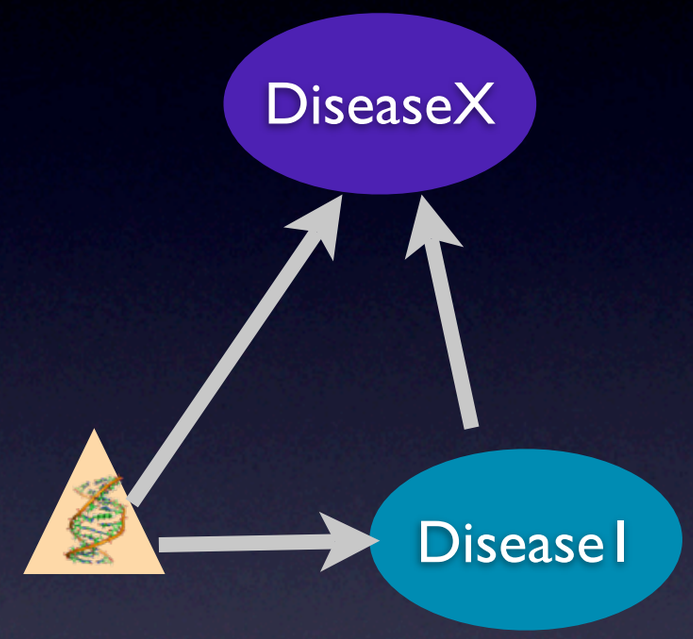
Motivation


DiseaseX
Disease1
Disease1
Disease1
Disease1
X_Gene connected to Disease1 with 66%.

The Situation

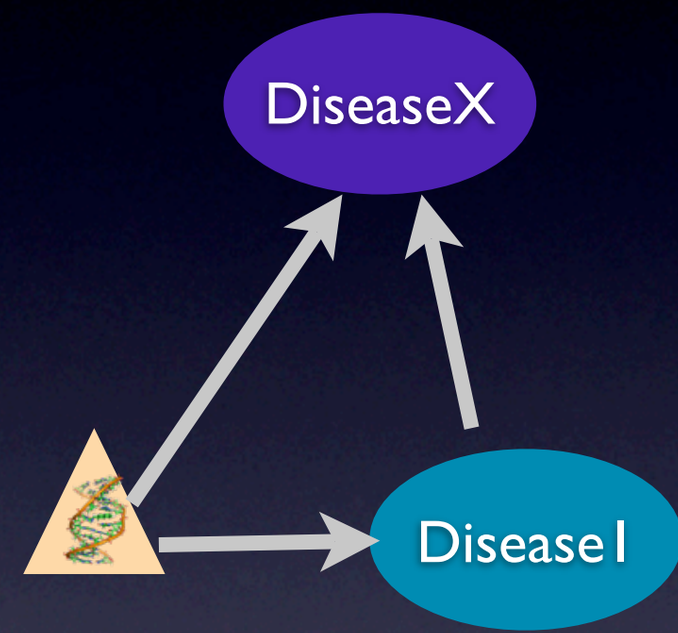
The Situation

- Domain knowledge



The Situation

- Domain knowledge



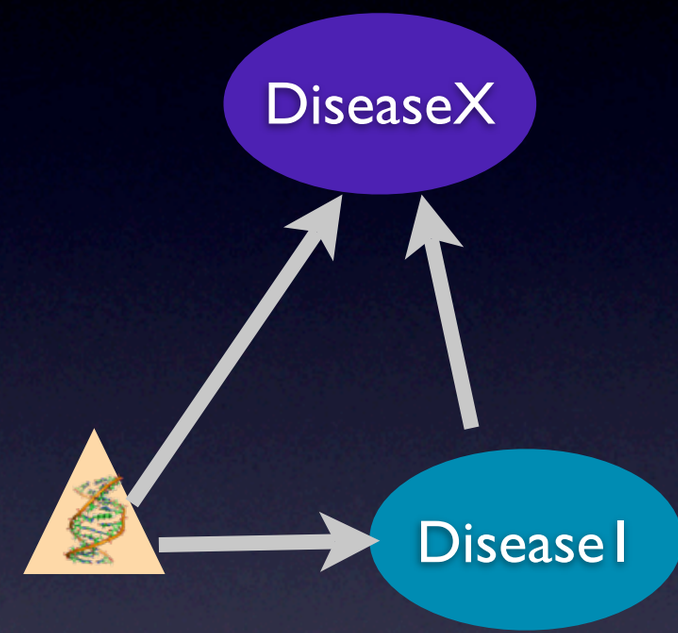
- Examples with probabilities

0.33 

0.66 

The Situation

- Domain knowledge



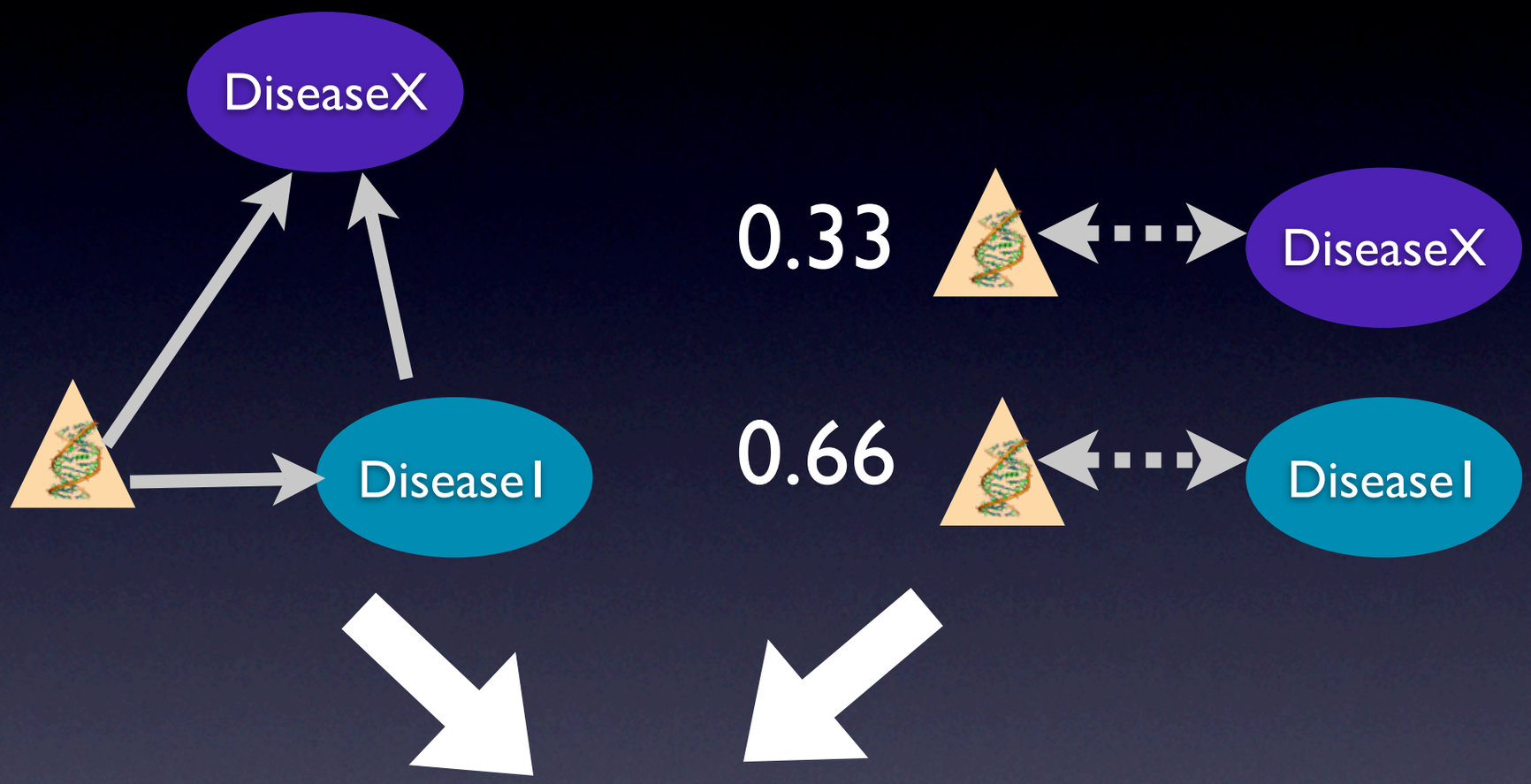
- Examples with probabilities

0.33 

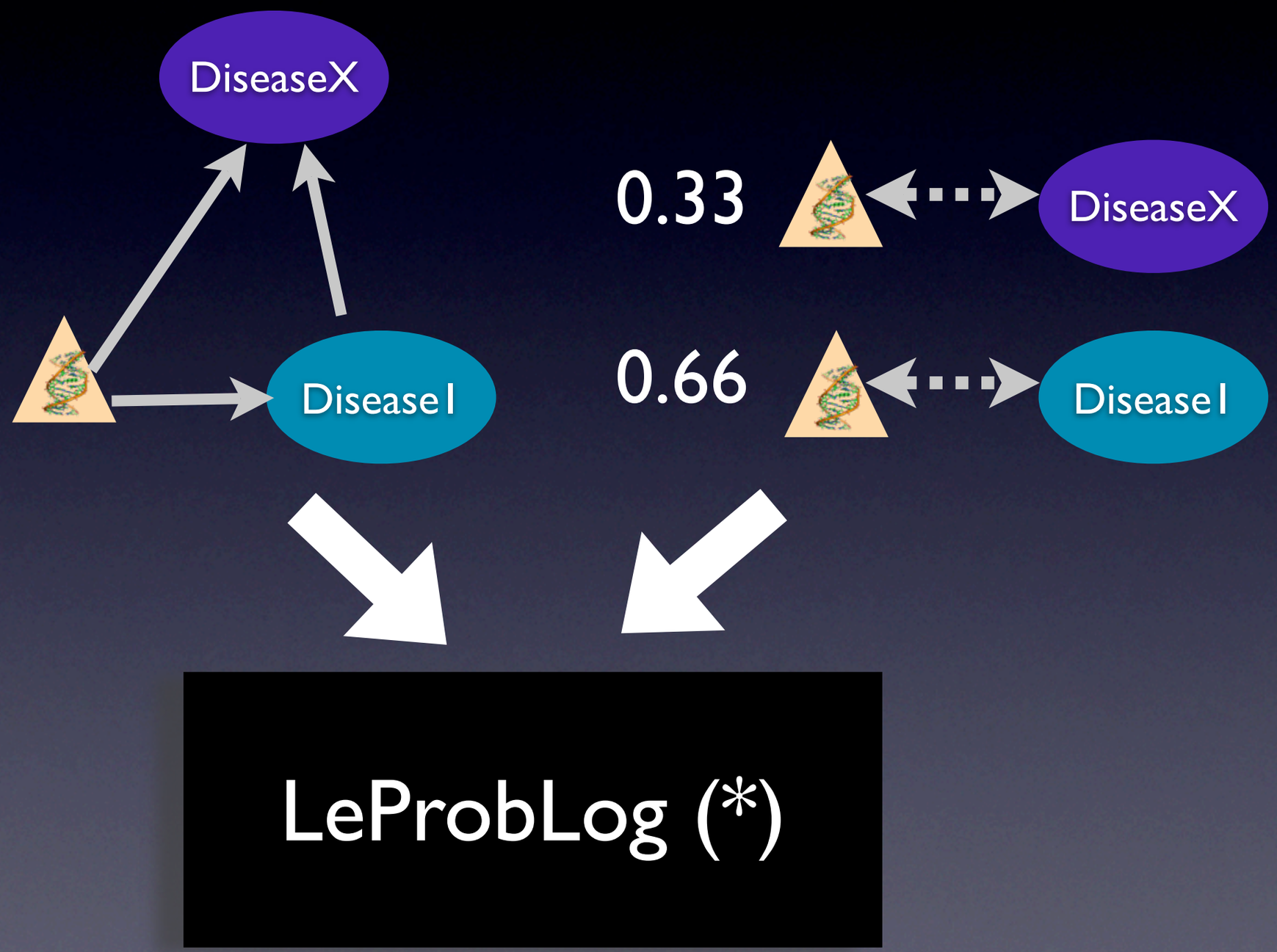
0.66 

- Uncertainty how to explain the examples

A Solution

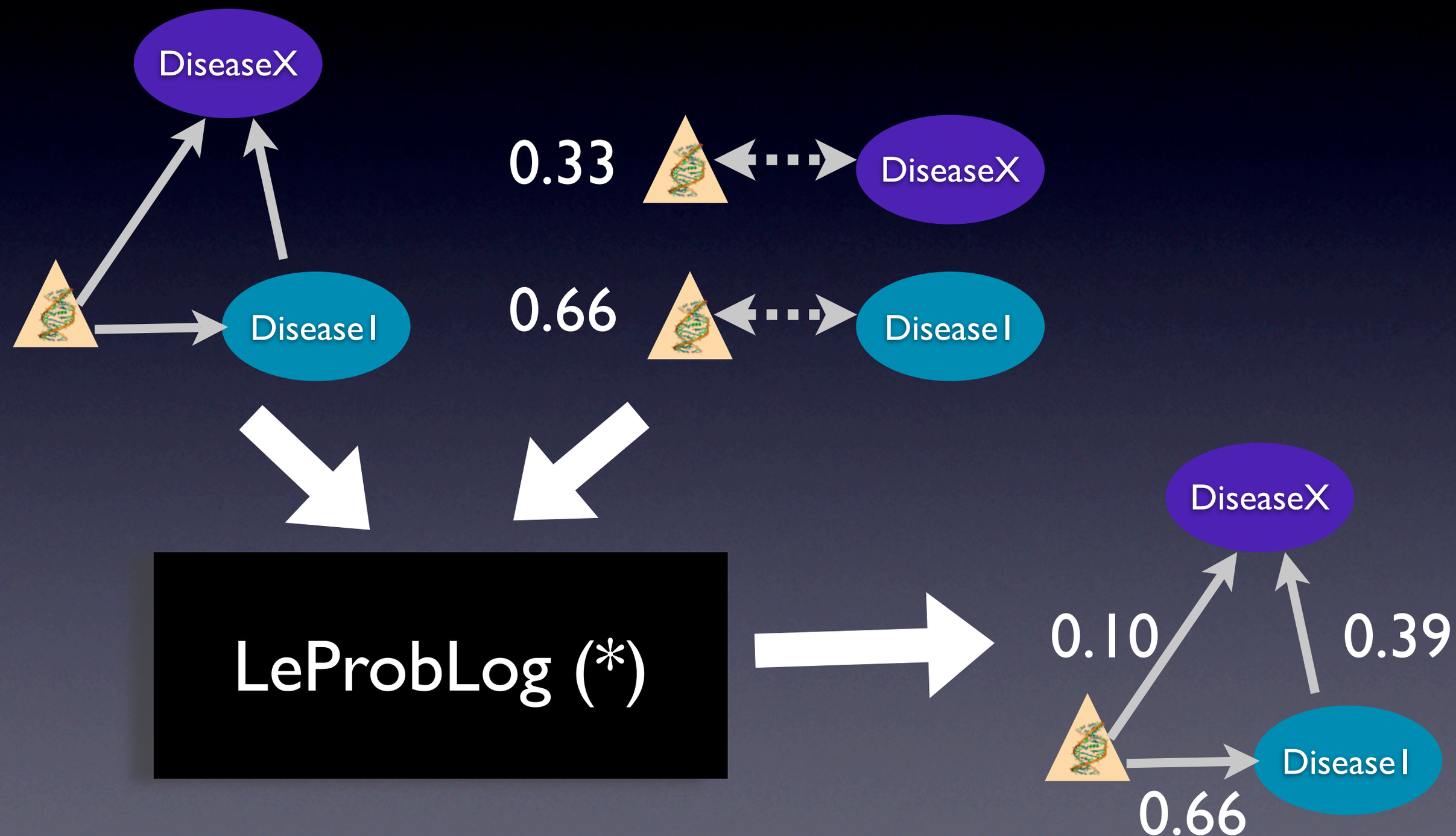


A Solution



(*) Least Square Parameter Estimation for ProbLog

A Solution

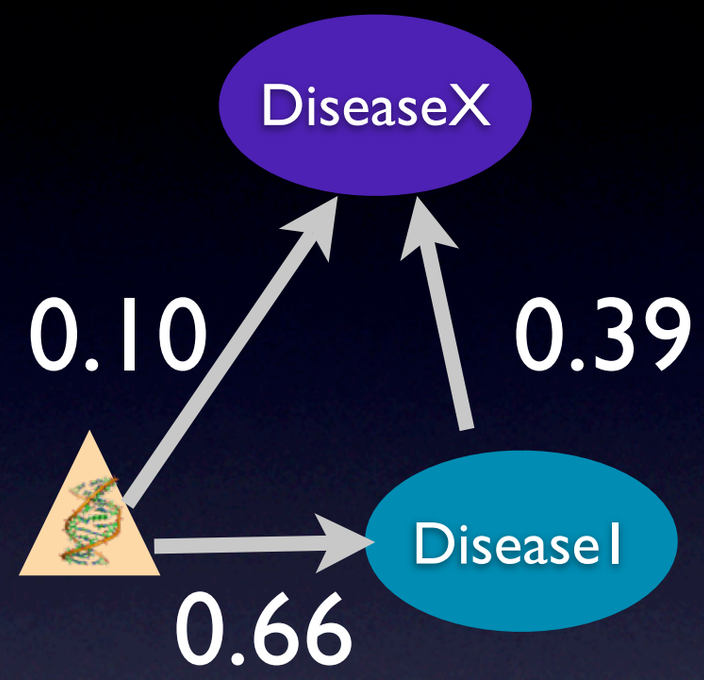


(*) Least Square Parameter Estimation for ProbLog

Outline

- Motivation
- ProbLog
- Parameter Learning for ProbLog
- Experiments

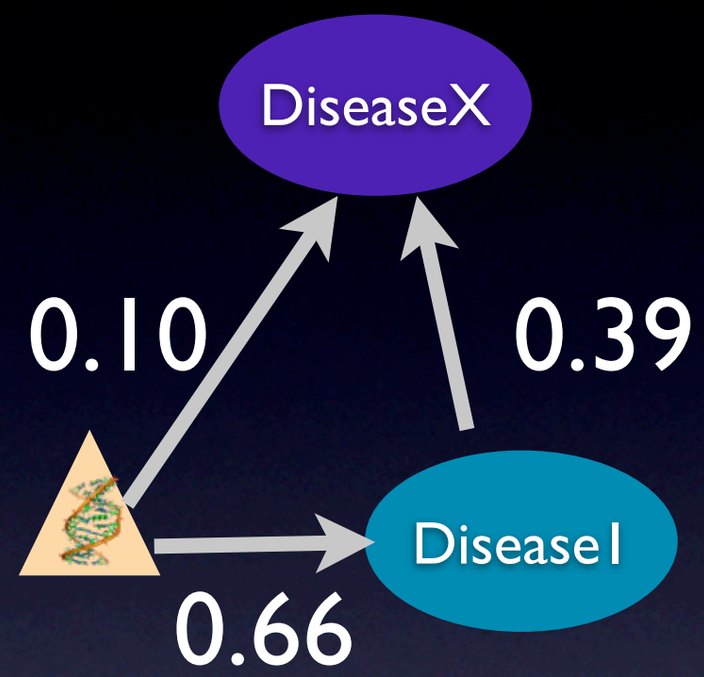
ProbLog




```
0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)
```

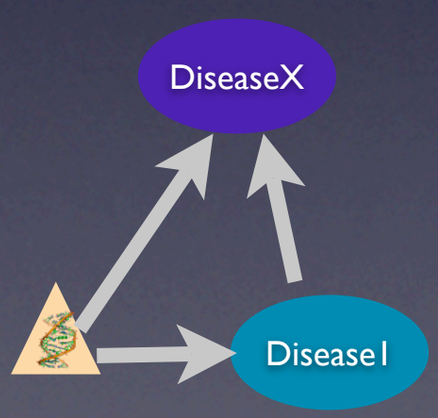
ProbLog



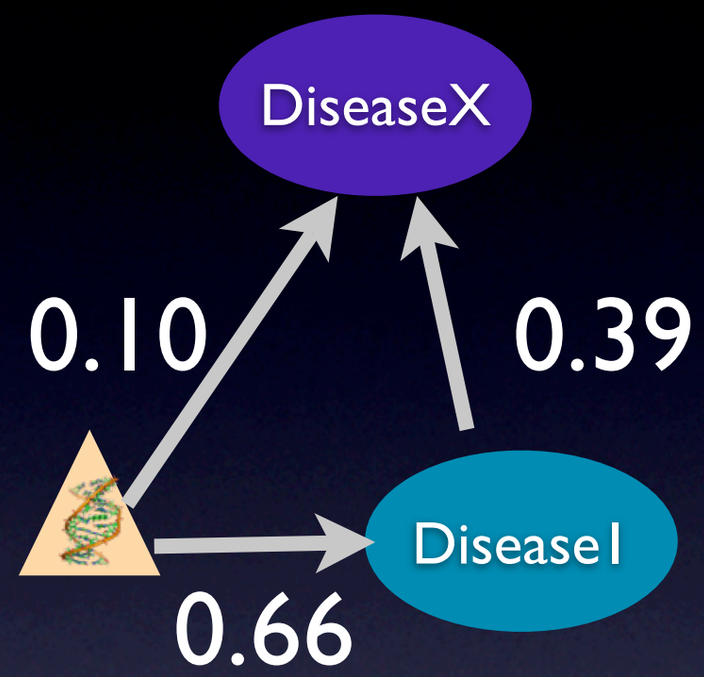

```
0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)
```

p=0.10* 0.66 * 0.39



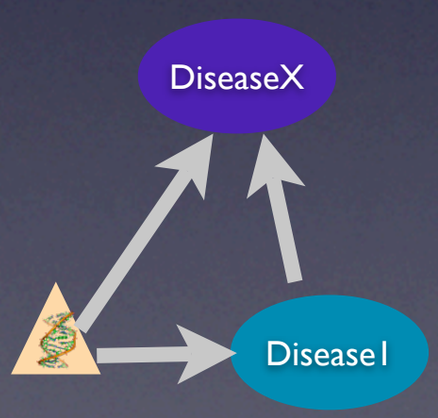
ProbLog



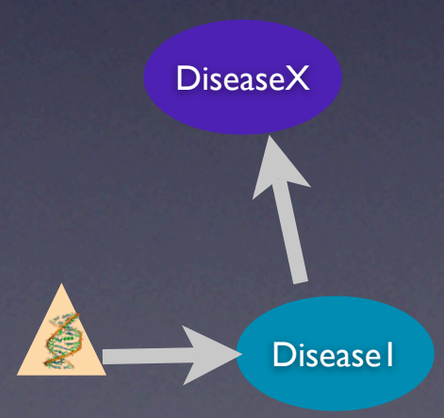

```
0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)
```

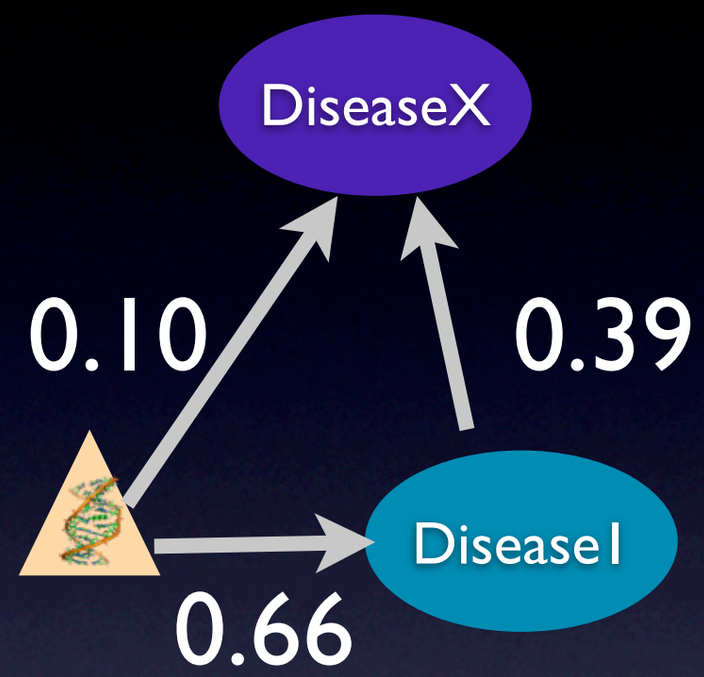
p=0.10* 0.66 * 0.39



p=(1-0.10) * 0.66 * 0.39



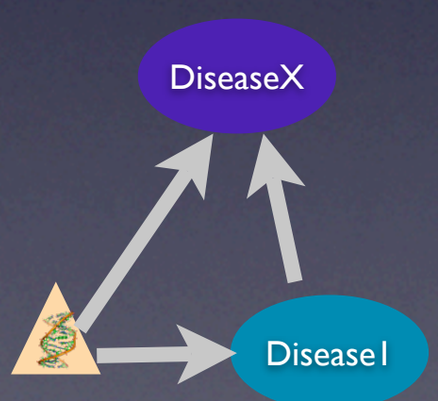
ProbLog



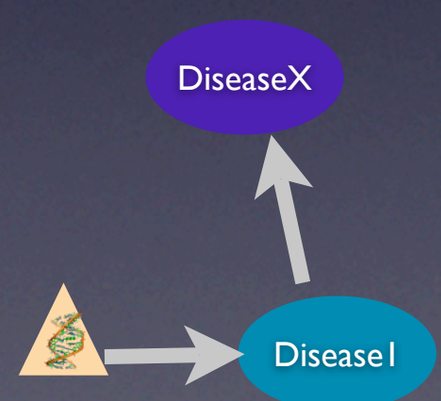

```
0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)
```

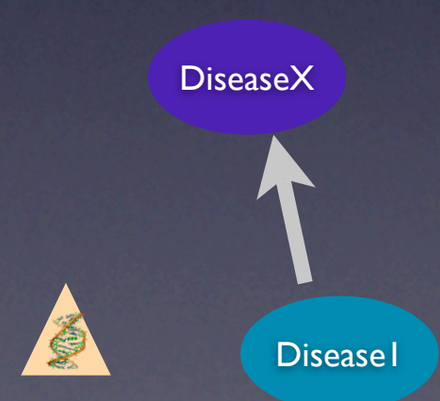
p=0.10* 0.66 * 0.39



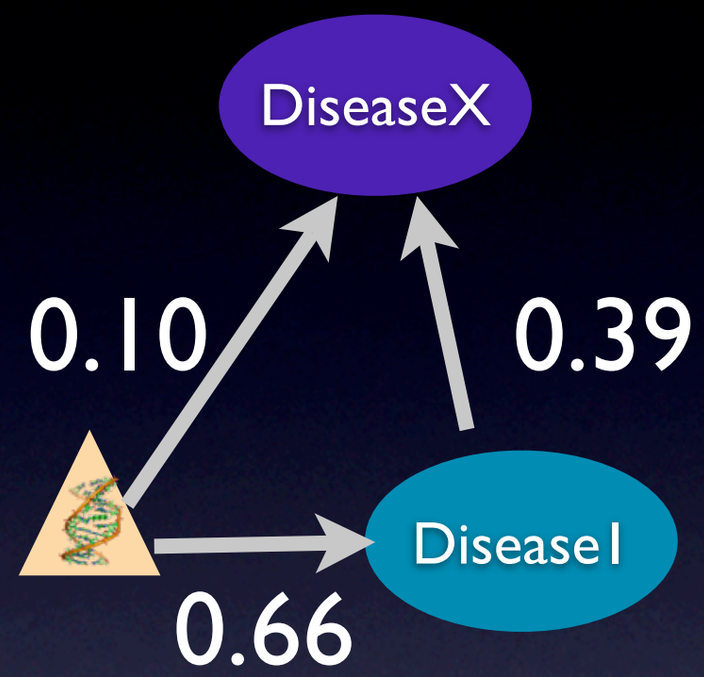
p=(1-0.10) * 0.66 * 0.39



p=(1-0.10) * (1-0.66) * 0.39



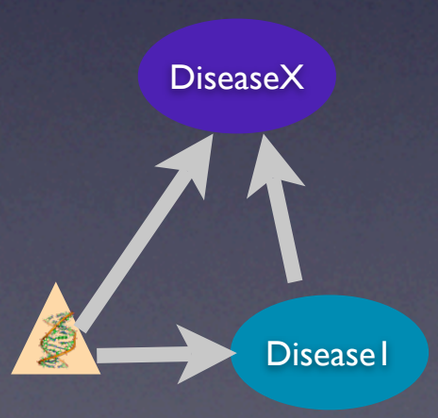
ProbLog



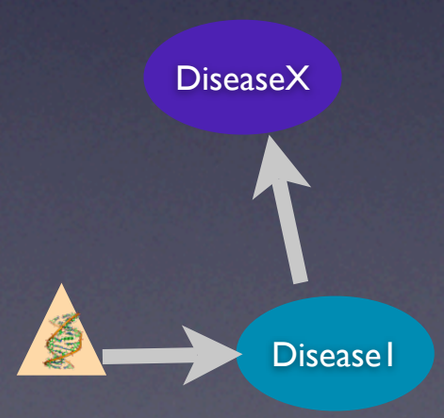

```
0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)
```

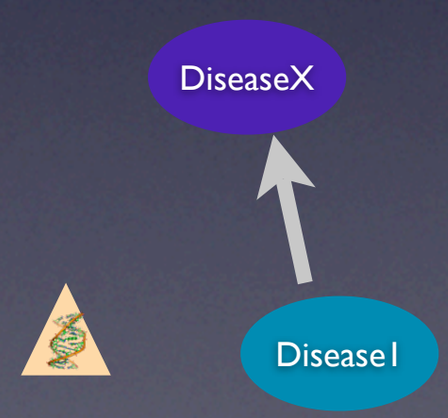
p=0.10* 0.66 * 0.39



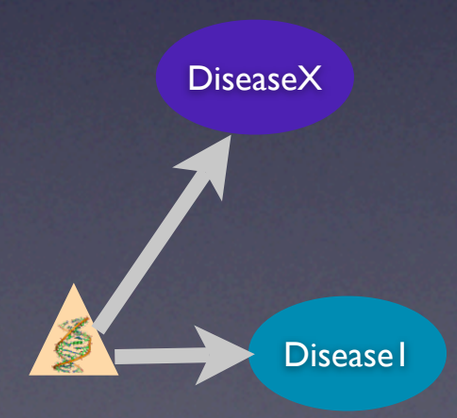
p=(1-0.10) * 0.66 * 0.39



p=(1-0.10) * (1-0.66) * 0.39

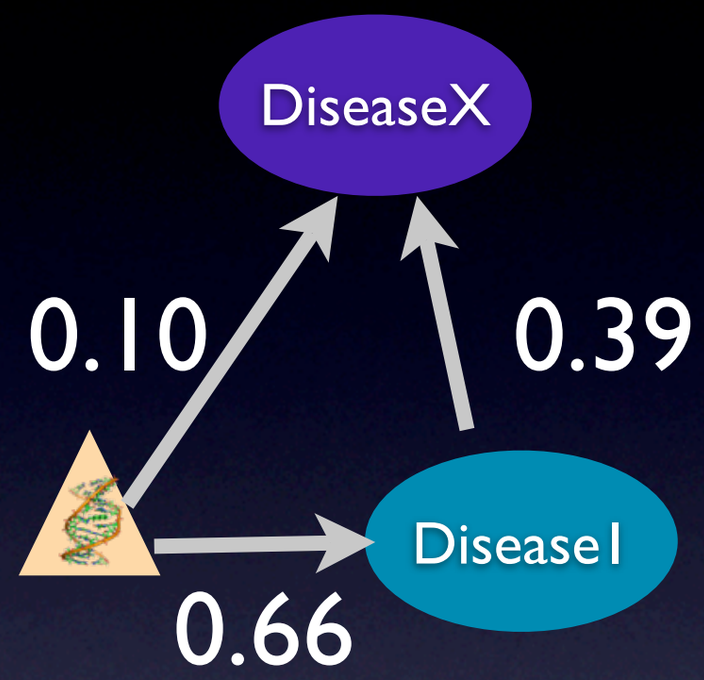


p=0.10 * 0.66 * (1-0.39)



...

ProbLog

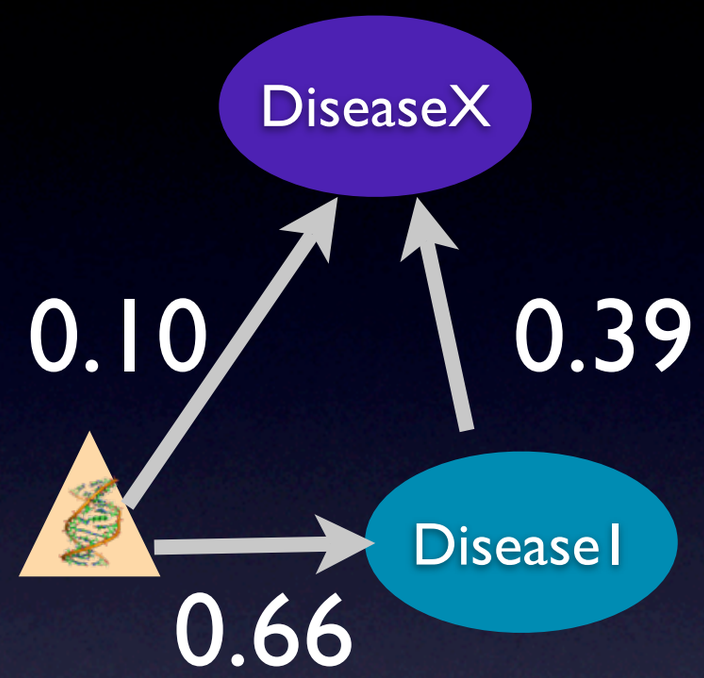


0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)

P(path('x_gene', 'DiseaseX'))

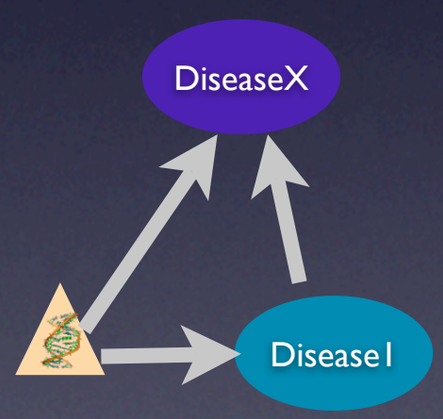
ProbLog



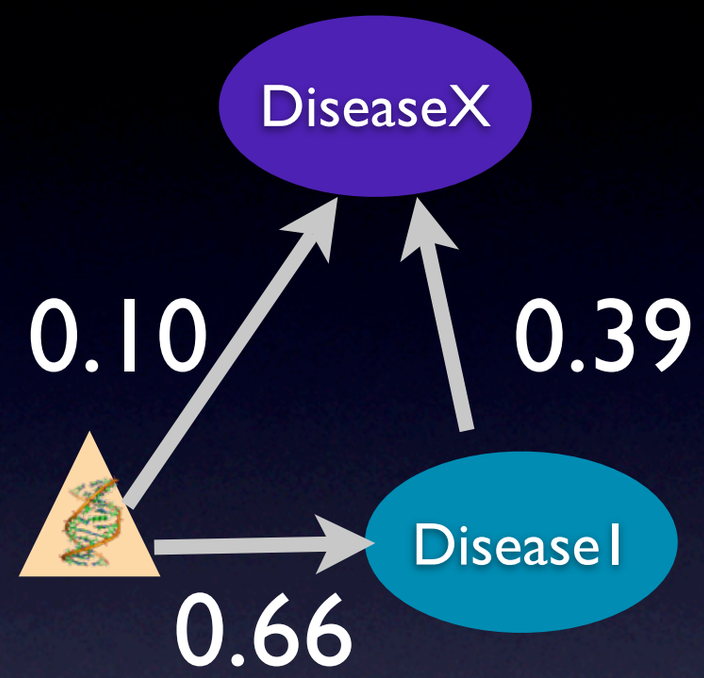
0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)

P(path('x_gene', 'DiseaseX'))



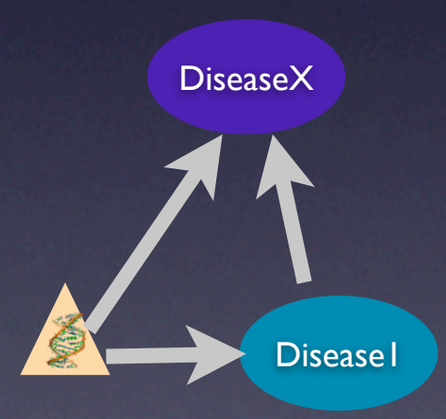
ProbLog

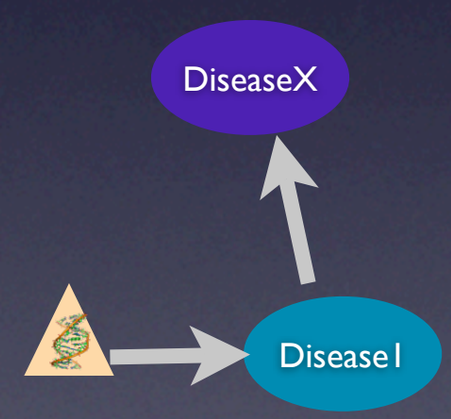


0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

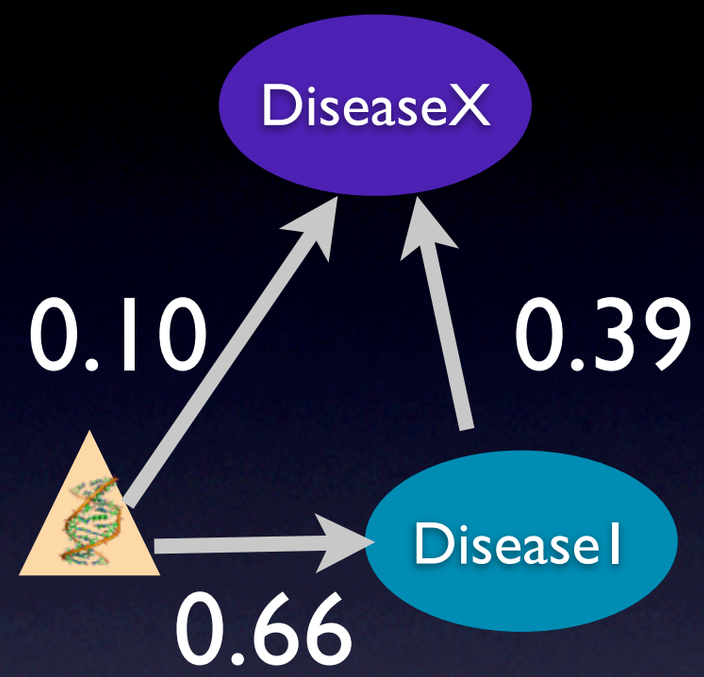
path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)

P(path('x_gene', 'DiseaseX'))





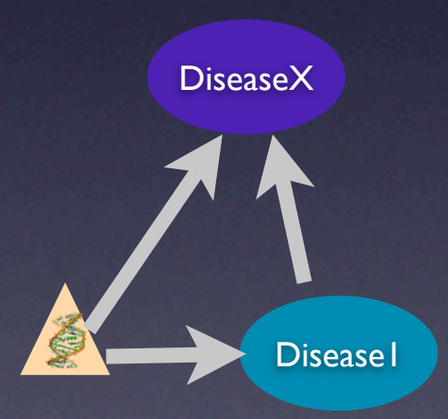
ProbLog

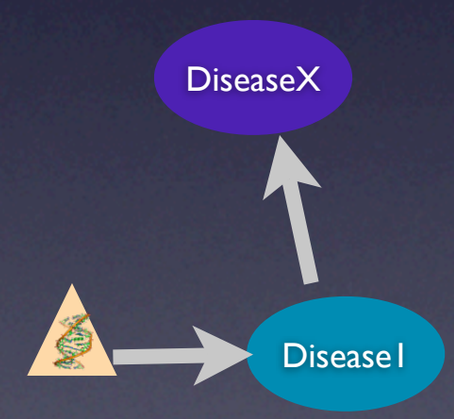


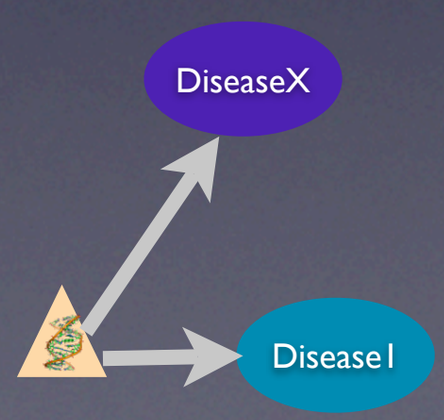
0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)

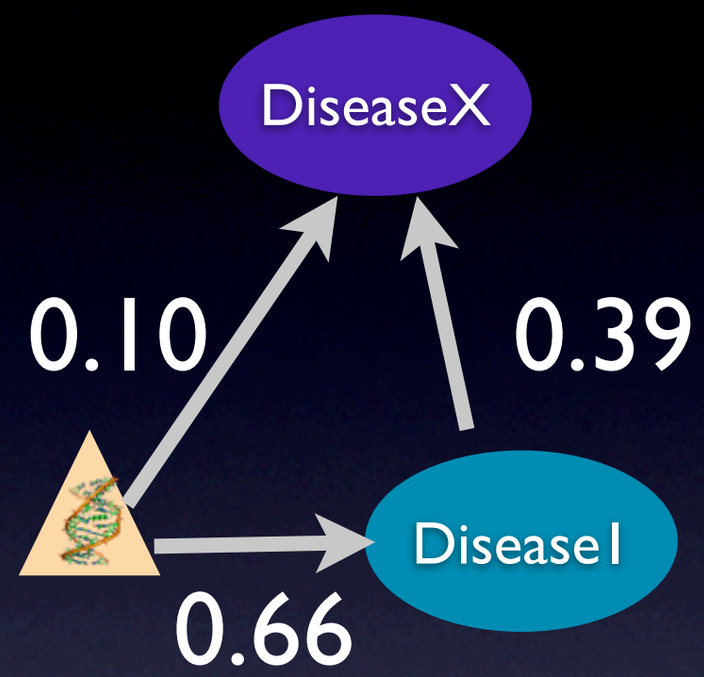
P(path('x_gene', 'DiseaseX'))







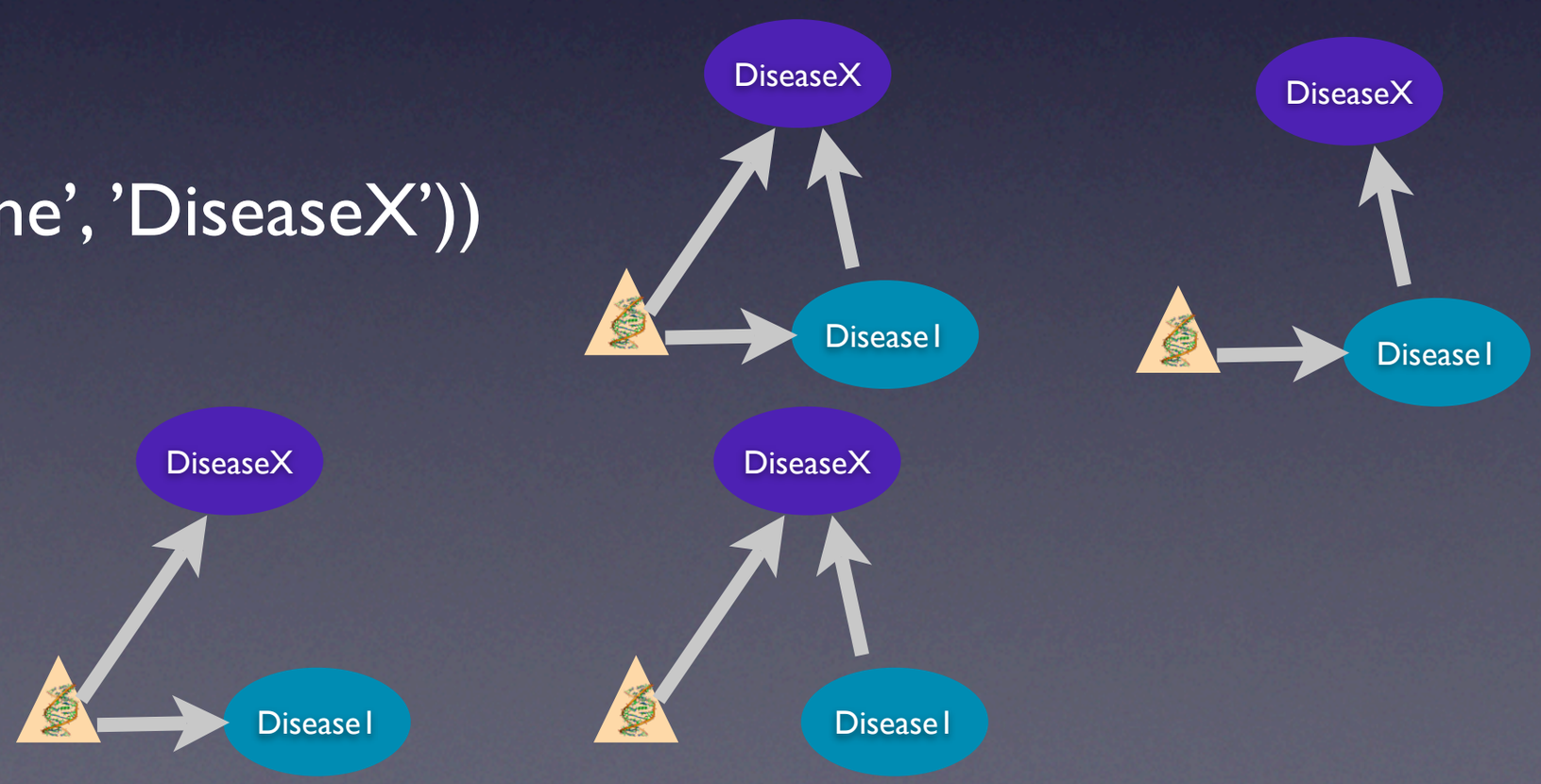
ProbLog



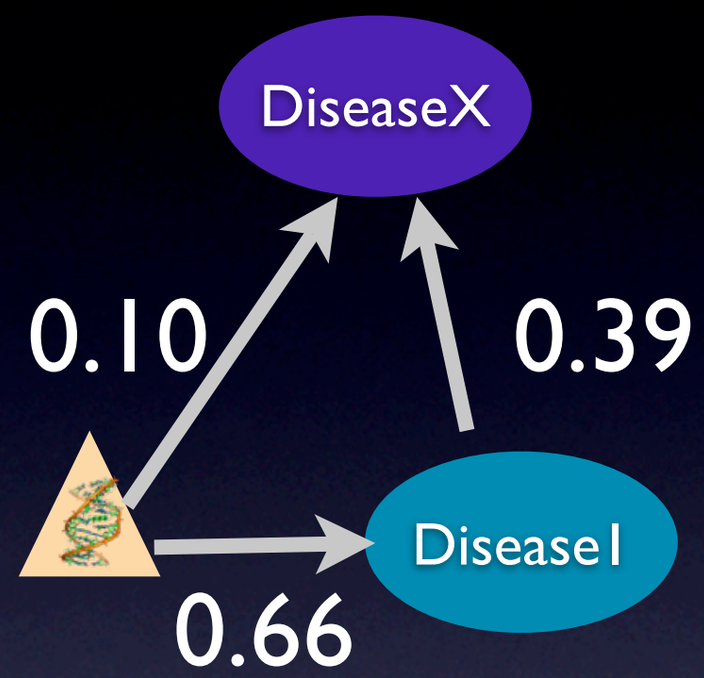
0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)

P(path('x_gene', 'DiseaseX'))



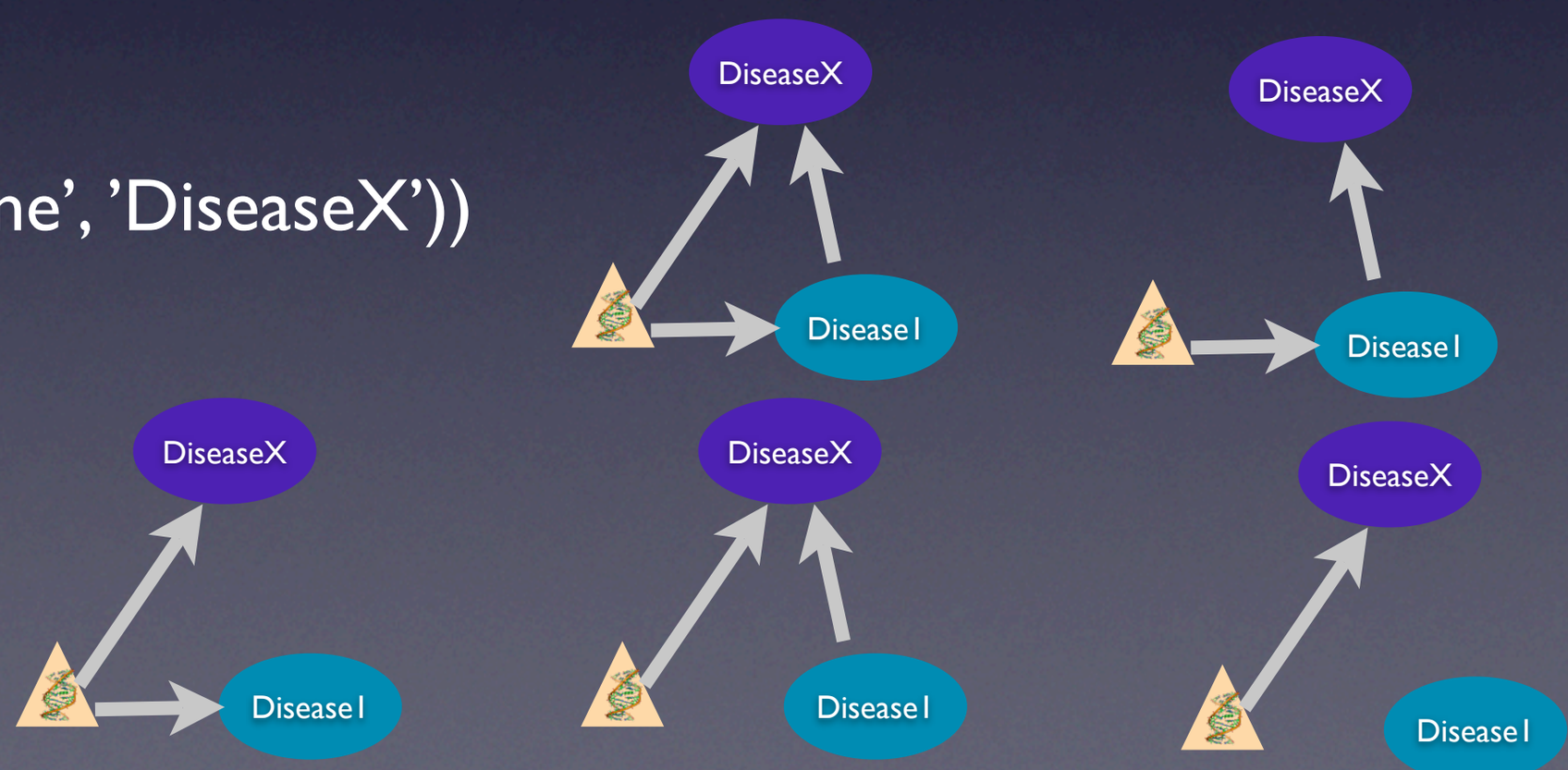
ProbLog



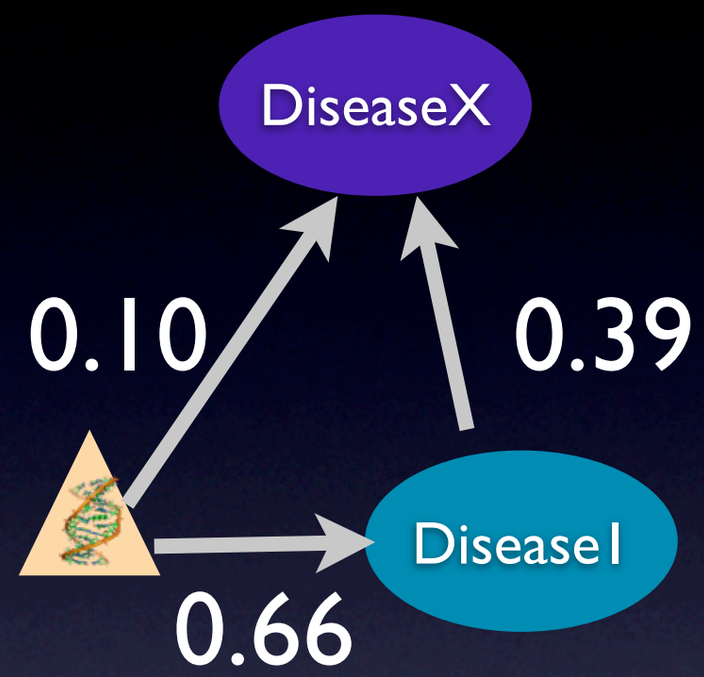
0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)

P(path('x_gene', 'DiseaseX'))

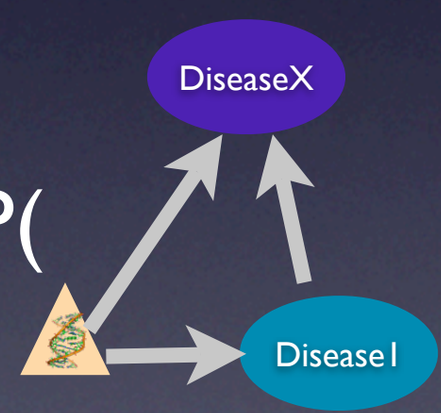
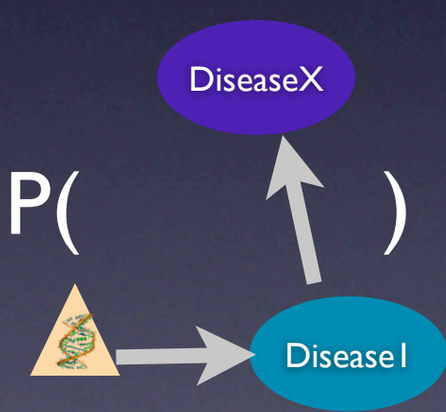


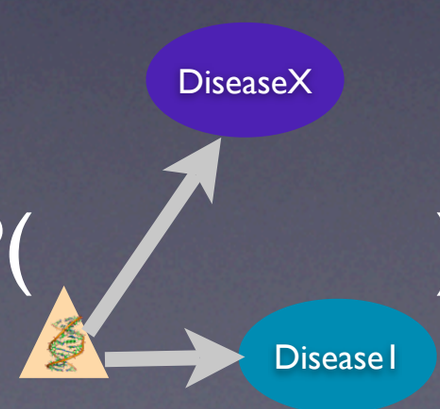
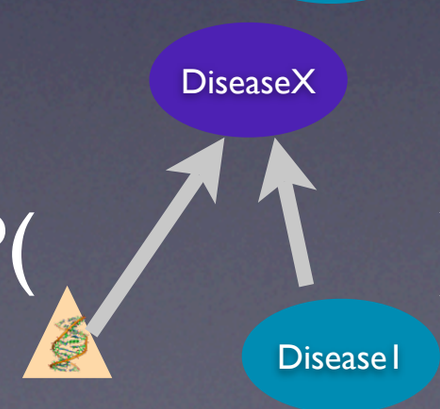
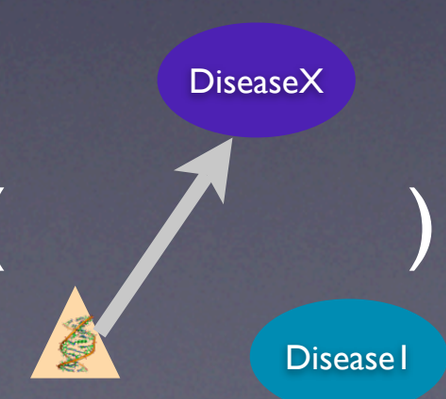
ProbLog



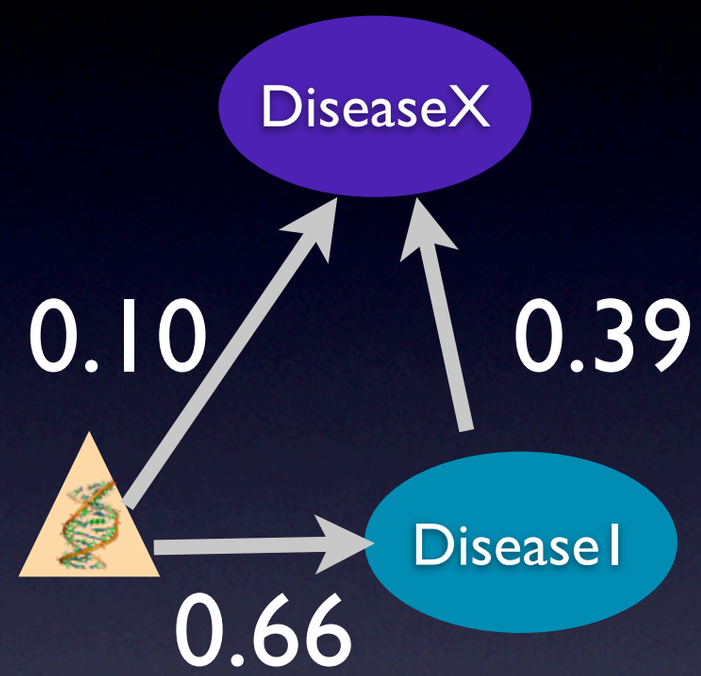
0.10 :: edge('x_gene', 'DiseaseX')
0.66 :: edge('x_gene', 'Disease1')
0.39 :: edge('Disease1', 'x_gene')

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)

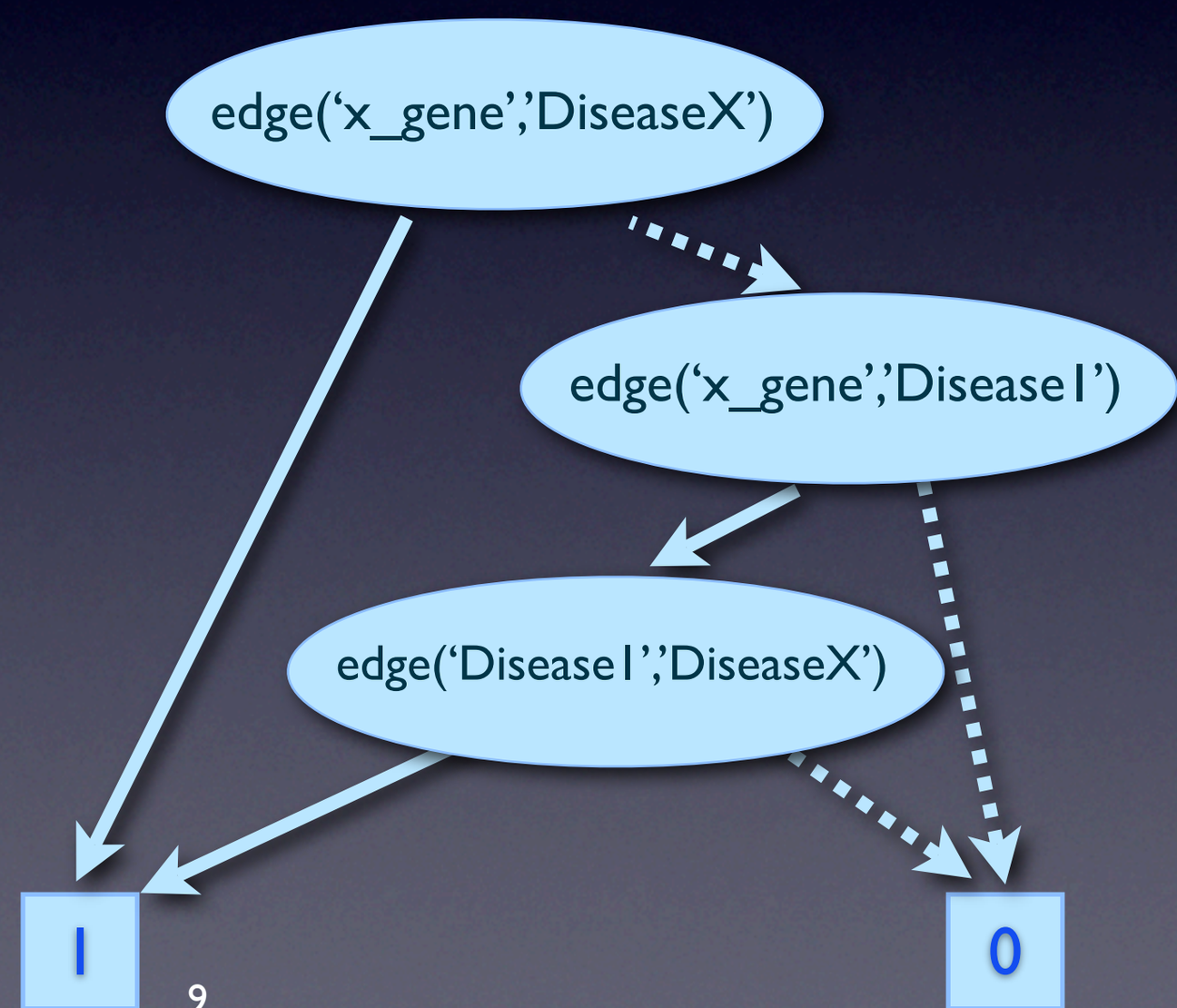
P(path('x_gene', 'DiseaseX'))=P() + P() +

P() + P() + P()

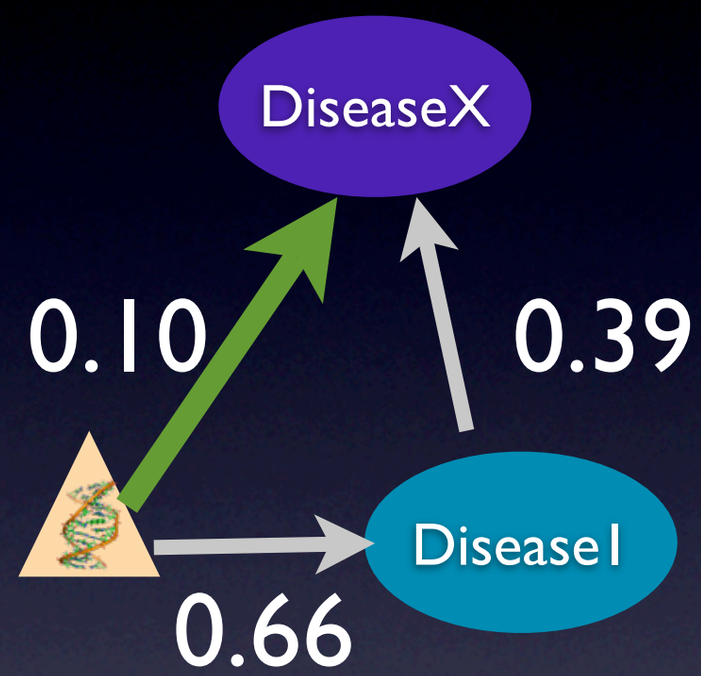
Calculating Probabilities



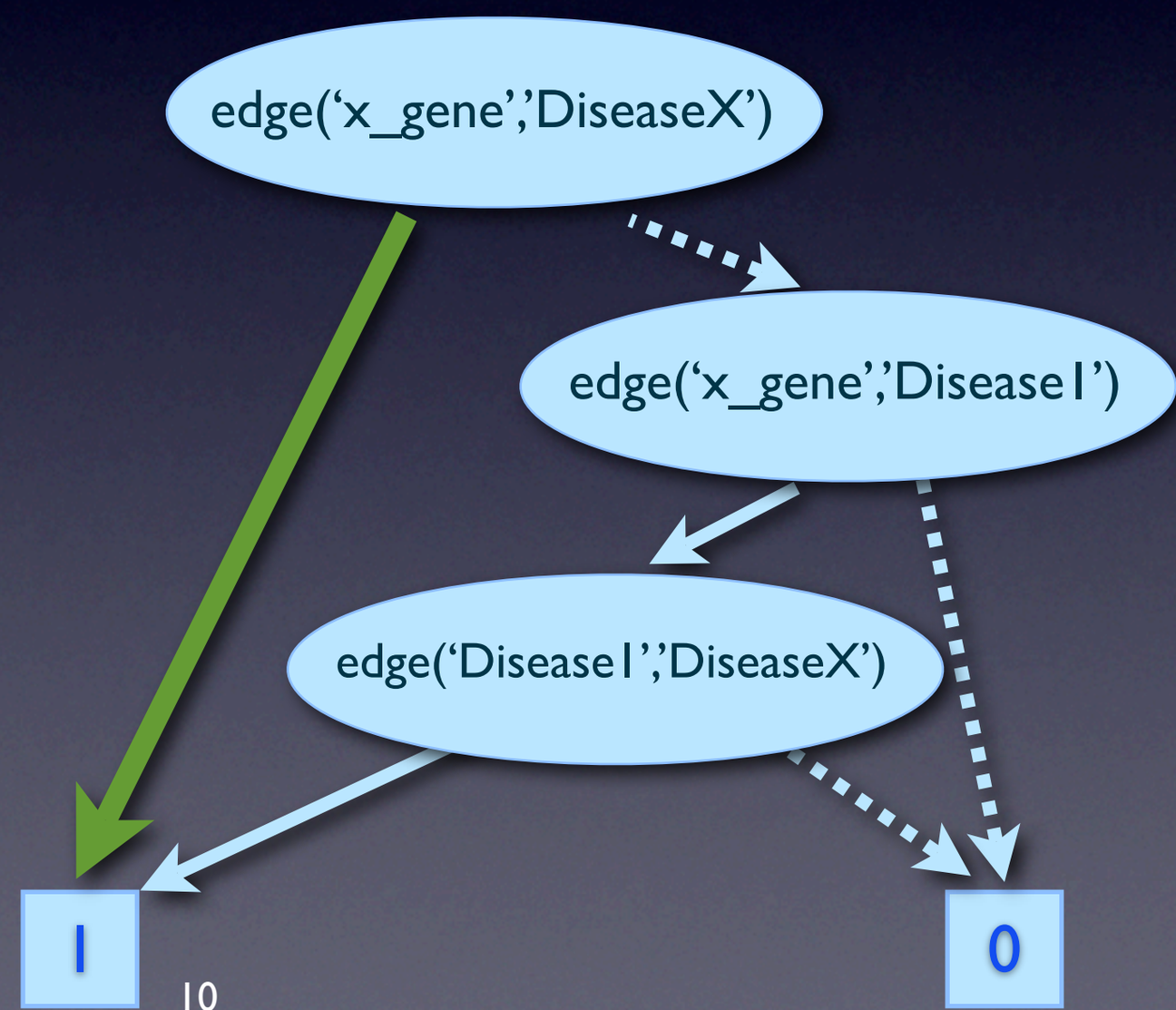
P_s('x_gene', 'DiseaseX'|L)=?



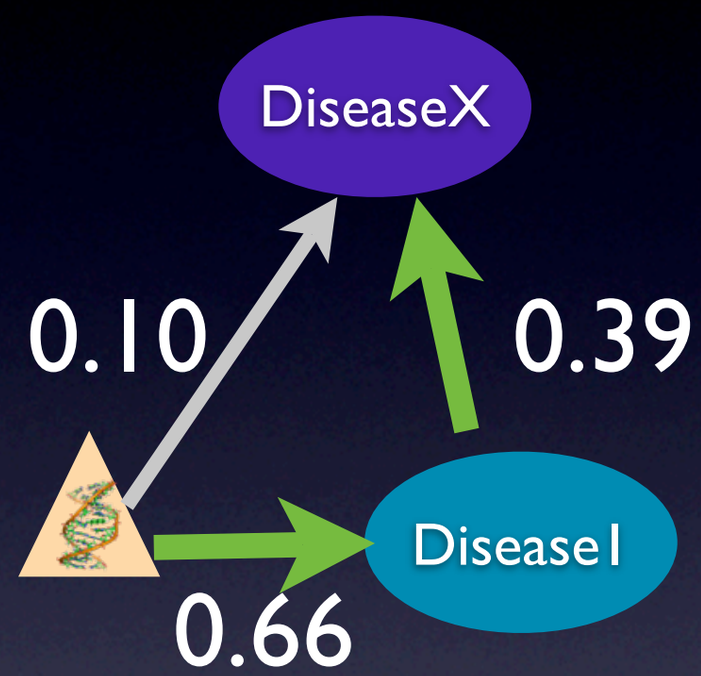
Calculating Probabilities



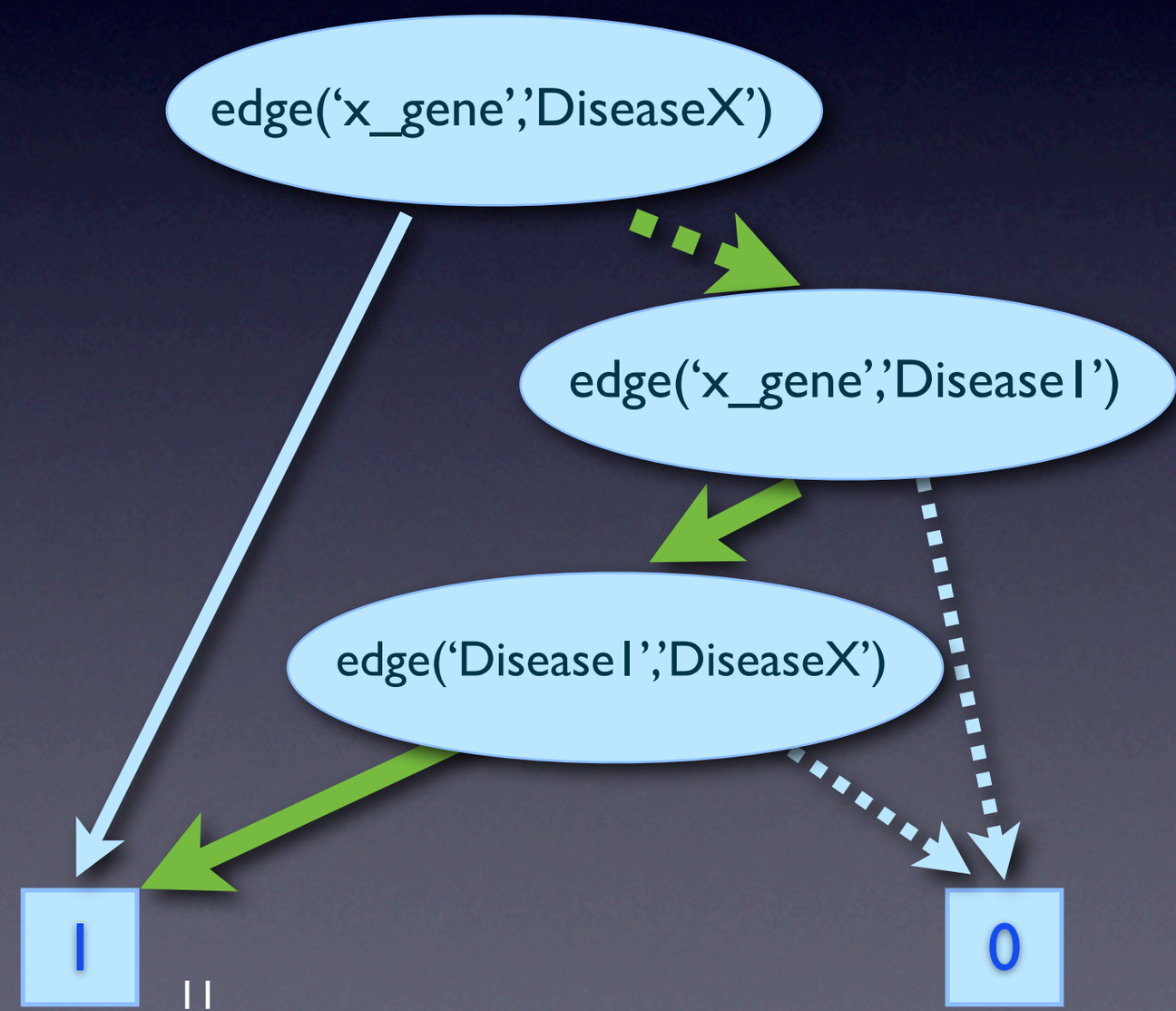
P_s('x_gene', 'DiseaseX'|L)=?



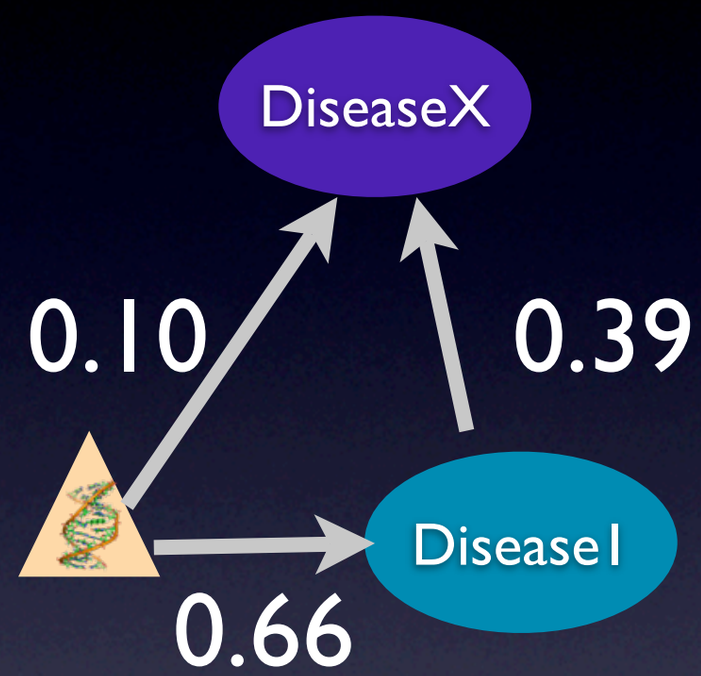
Calculating Probabilities



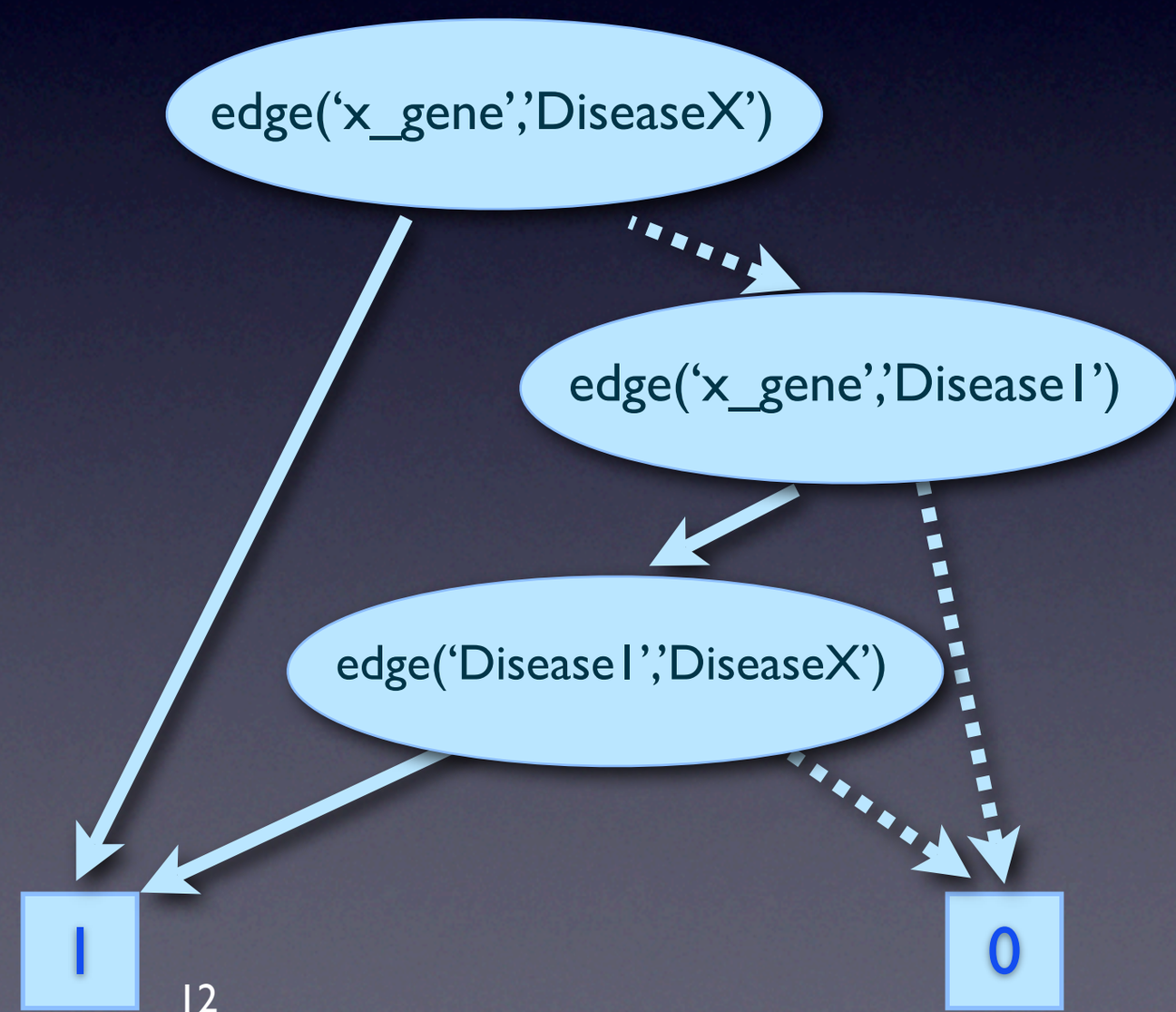
P_s('x_gene', 'DiseaseX'|L)=?



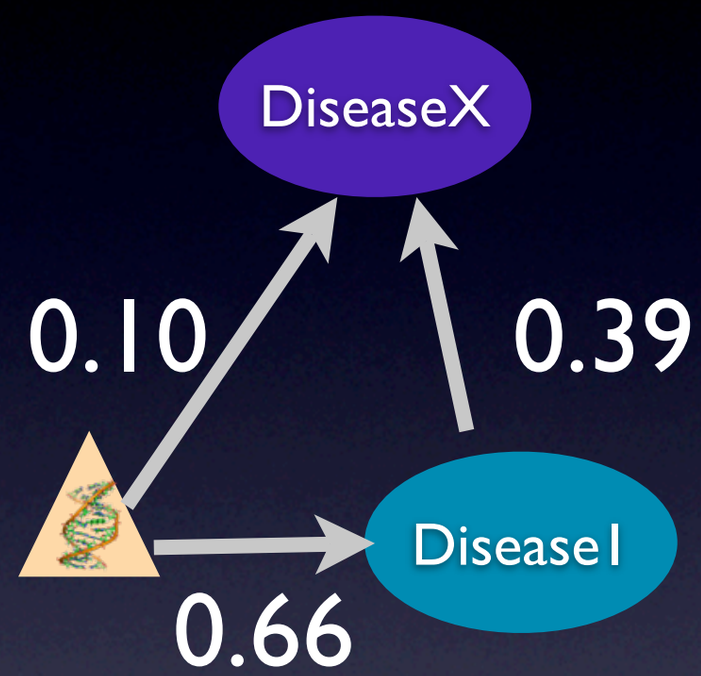
Calculating Probabilities



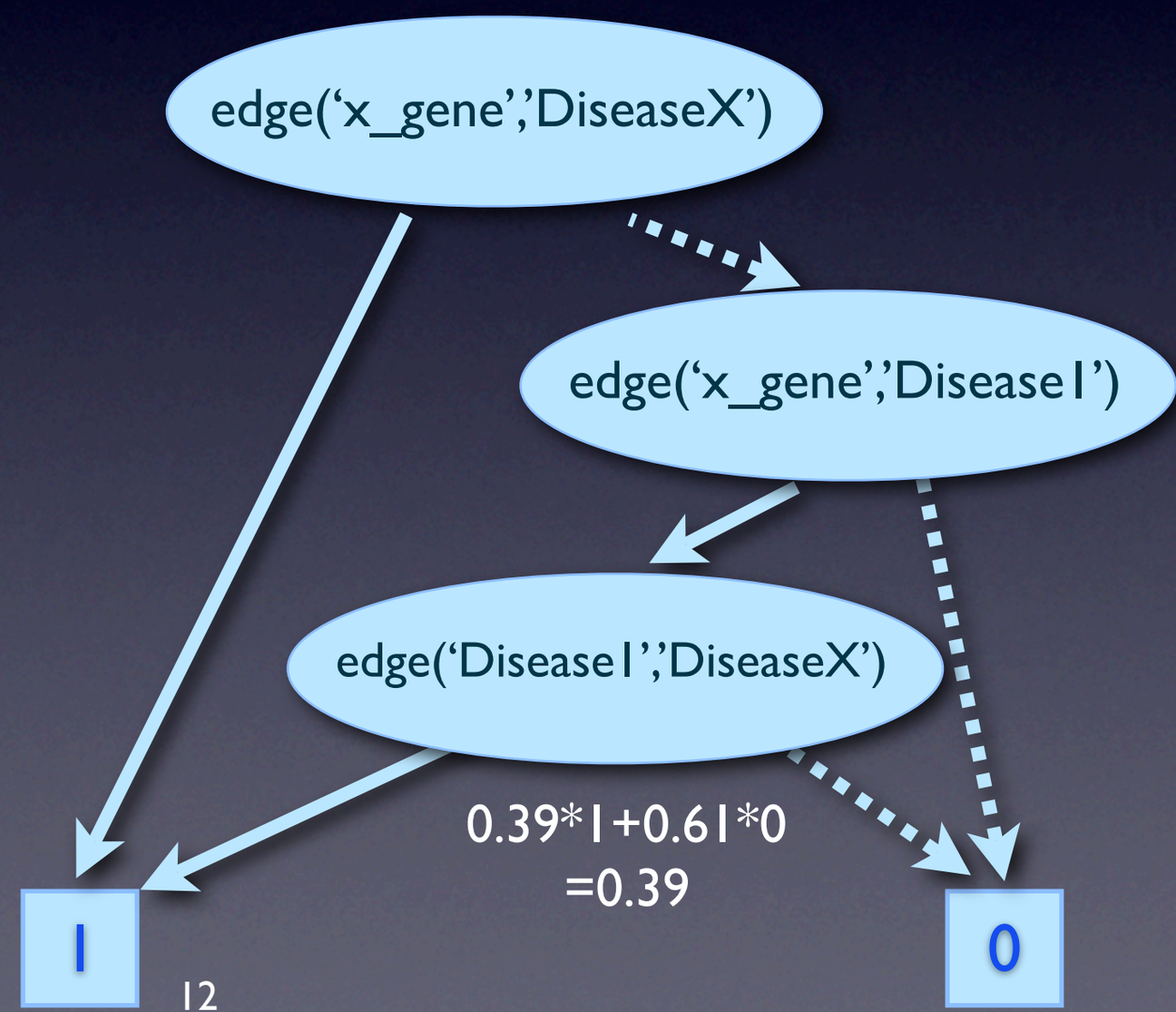
P_s('x_gene', 'DiseaseX'|L)=?



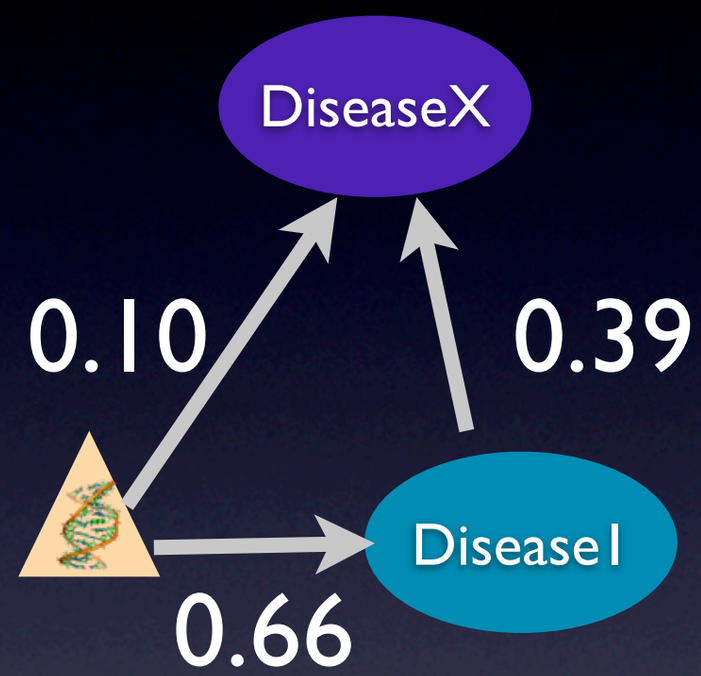
Calculating Probabilities



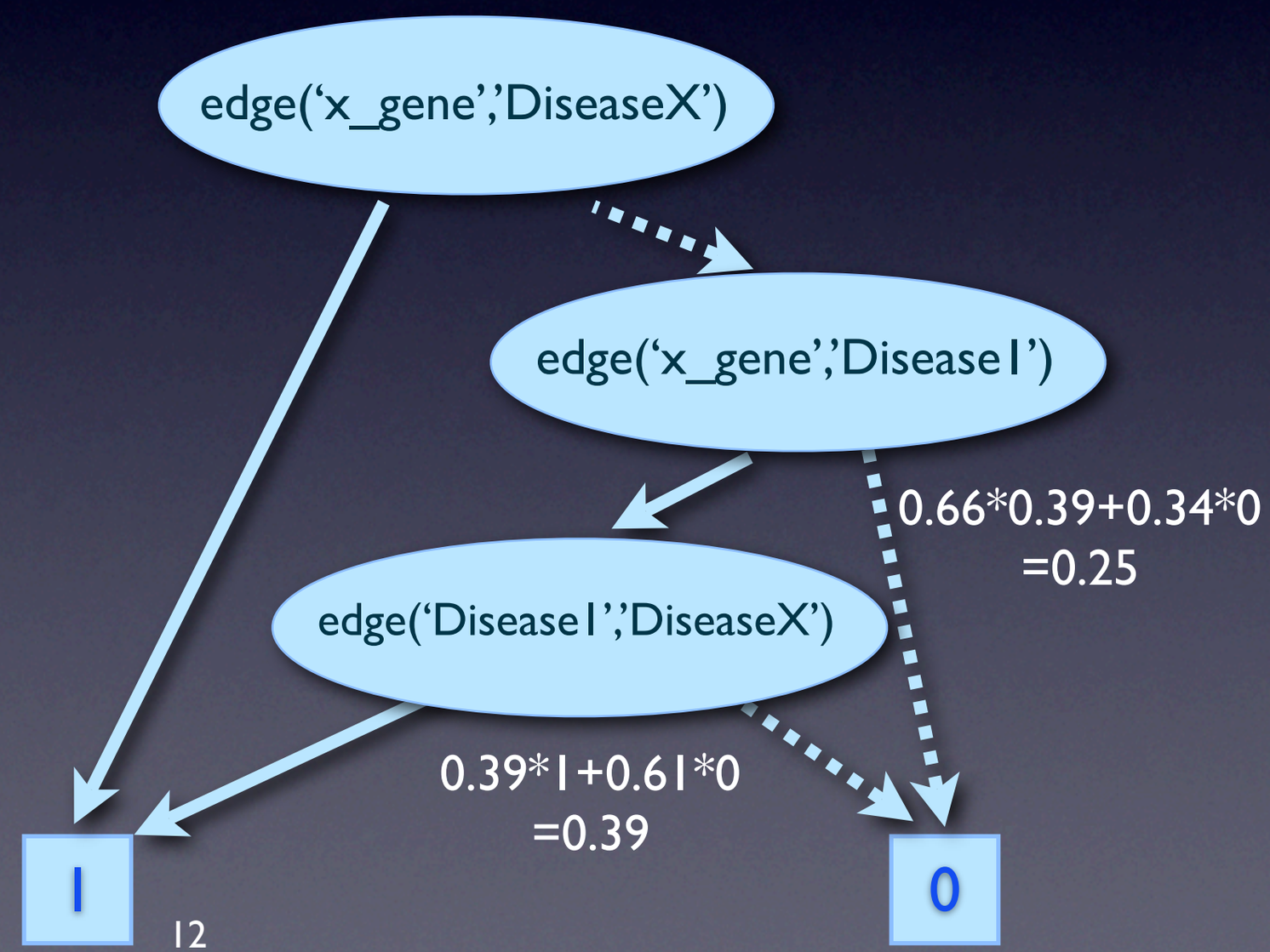
P_s('x_gene', 'DiseaseX'|L)=?



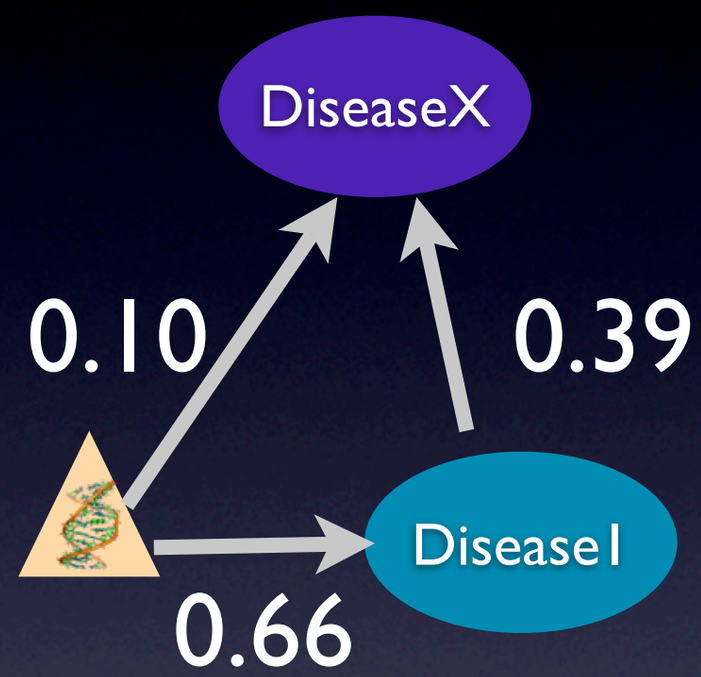
Calculating Probabilities



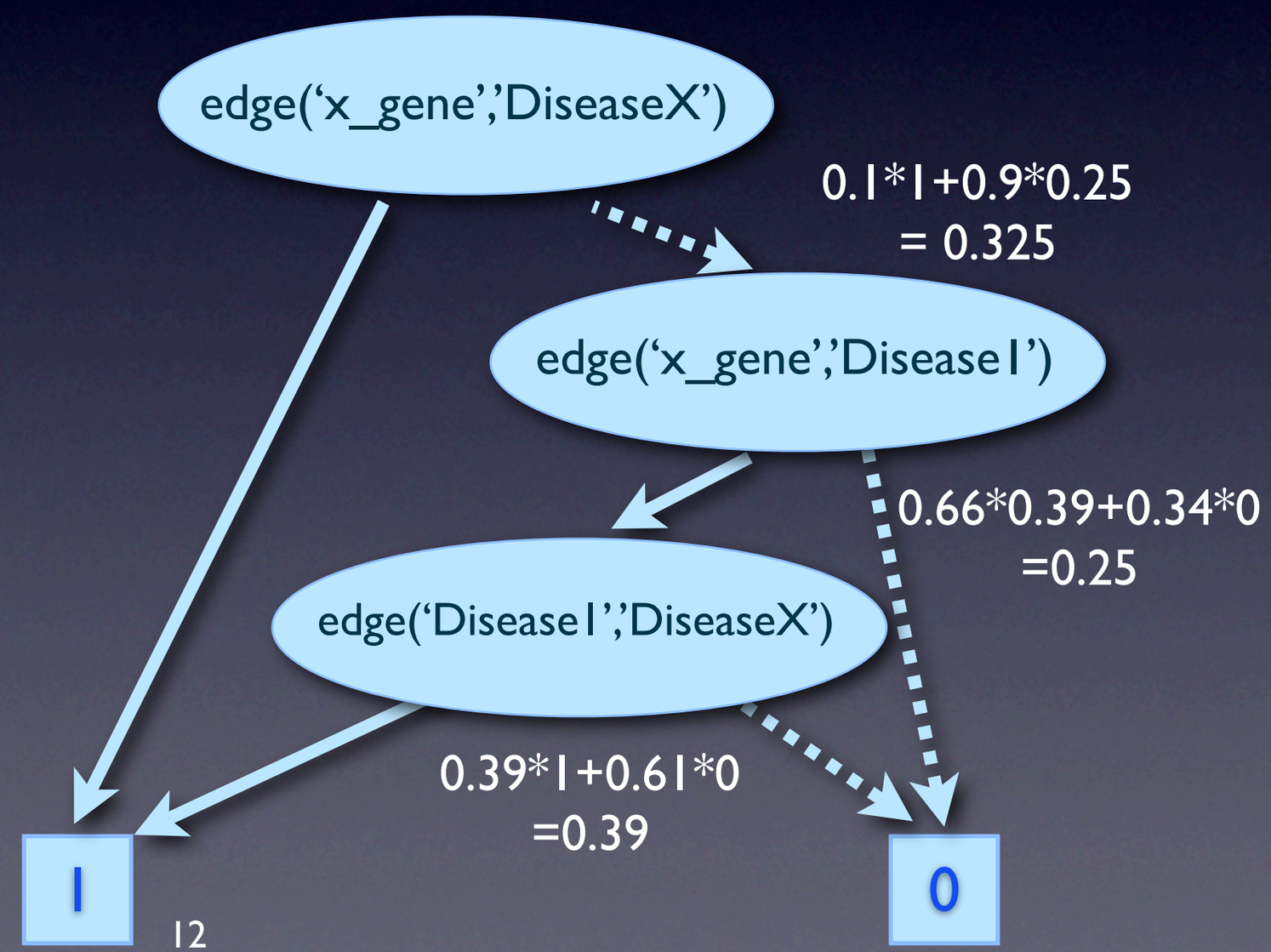
P_s('x_gene', 'DiseaseX'|L)=?



Calculating Probabilities



P_s('x_gene', 'DiseaseX'|L)=?



Parameter Learning

?? :: edge(‘x_gene’, ’DiseaseX’)
?? :: edge(‘x_gene’, ’Disease1’)
?? :: edge(‘Disease1’, ’x_gene’)

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)

Parameter Learning

?? :: edge(‘x_gene’, ’DiseaseX’)
?? :: edge(‘x_gene’, ’Disease1’)
?? :: edge(‘Disease1’, ’x_gene’)

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)

0.33 :: path(‘x_gene’, ’DiseaseX’)
0.66 :: path(‘x_gene’, ’Disease1’)

Parameter Learning

?? :: edge(‘x_gene’, ’DiseaseX’)
?? :: edge(‘x_gene’, ’Disease1’)
?? :: edge(‘Disease1’, ’x_gene’)

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)

P_s(path(‘x_gene’, ’DiseaseX’) |L) <=> 0.33 :: path(‘x_gene’, ’DiseaseX’)
P_s(path(‘x_gene’, ’Disease1’) |L) <=> 0.66 :: path(‘x_gene’, ’Disease1’)

Parameter Learning

?? :: edge(‘x_gene’, ’DiseaseX’)
?? :: edge(‘x_gene’, ’Disease1’)
?? :: edge(‘Disease1’, ’x_gene’)

path(X,Y) :- edge(X,Y)
path(X,Y) :- edge(X,Z), path(Z,Y)

P_s(path(‘x_gene’, ’DiseaseX’) |L) <=> 0.33 :: path(‘x_gene’, ’DiseaseX’)
P_s(path(‘x_gene’, ’Disease1’) |L) <=> 0.66 :: path(‘x_gene’, ’Disease1’)

$$MSE(T) = \frac{1}{K} \sum_{1 \leq i \leq K} \left(P_s(q_i|T) - \tilde{p}_i \right)^2$$

Gradient Descent

Gradient Descent

- Init probabilities randomly

Gradient Descent

- Init probabilities randomly
- while MSE is too high

Gradient Descent

- Init probabilities randomly
- while MSE is too high
  - find proofs for queries and build BDDs

Gradient Descent

- Init probabilities randomly
- while MSE is too high
  - find proofs for queries and build BDDs
  - calculate gradient for MSE

Gradient Descent

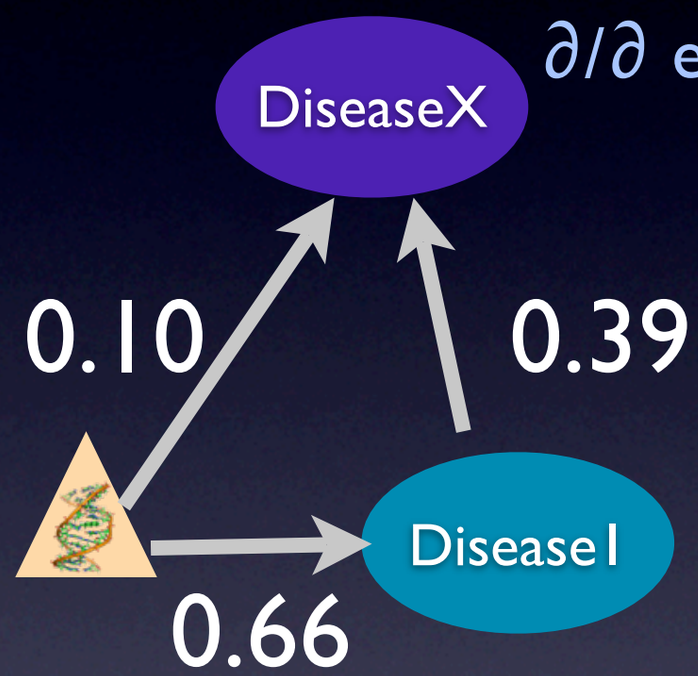
- Init probabilities randomly
- while MSE is too high
  - find proofs for queries and build BDDs
  - calculate gradient for MSE
  - add gradient

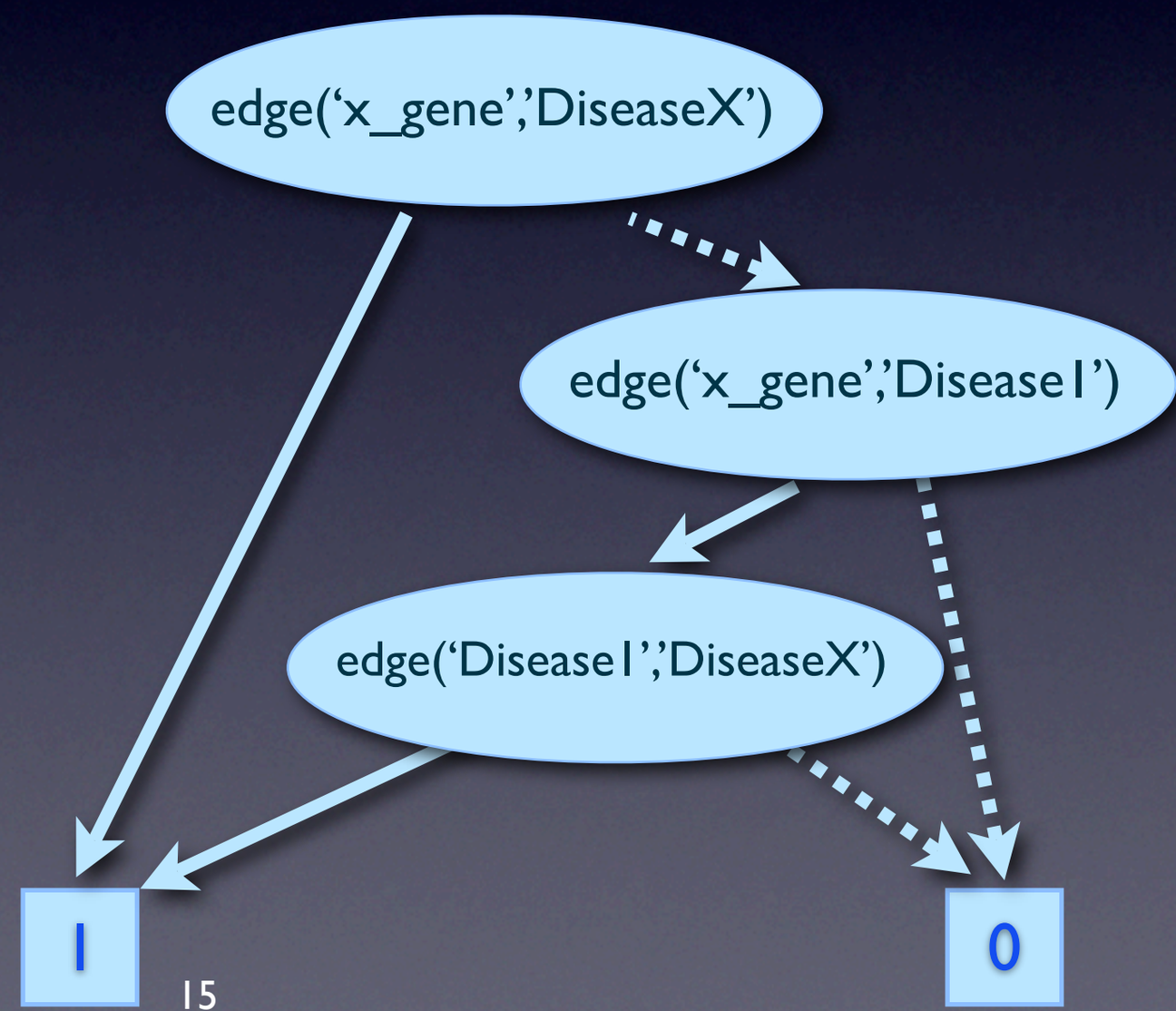
Gradient Descent

- Init probabilities randomly
- while MSE is too high
    - find proofs for queries and build BDDs
    - calculate gradient for MSE
    - add gradient
- return probabilities

How to calculate the Gradient of the MSE?

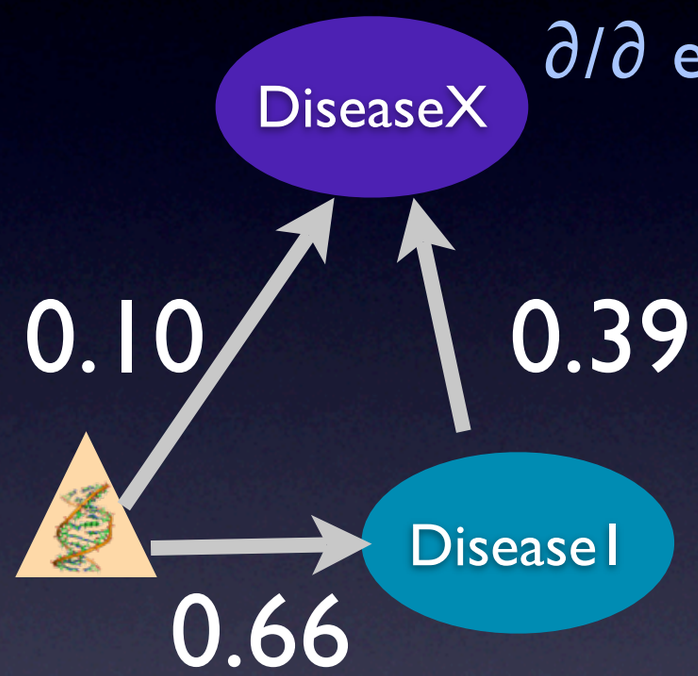
∂/∂ edge('x_gene','Disease1') P_s(path('x_gene', 'DiseaseX') |L)

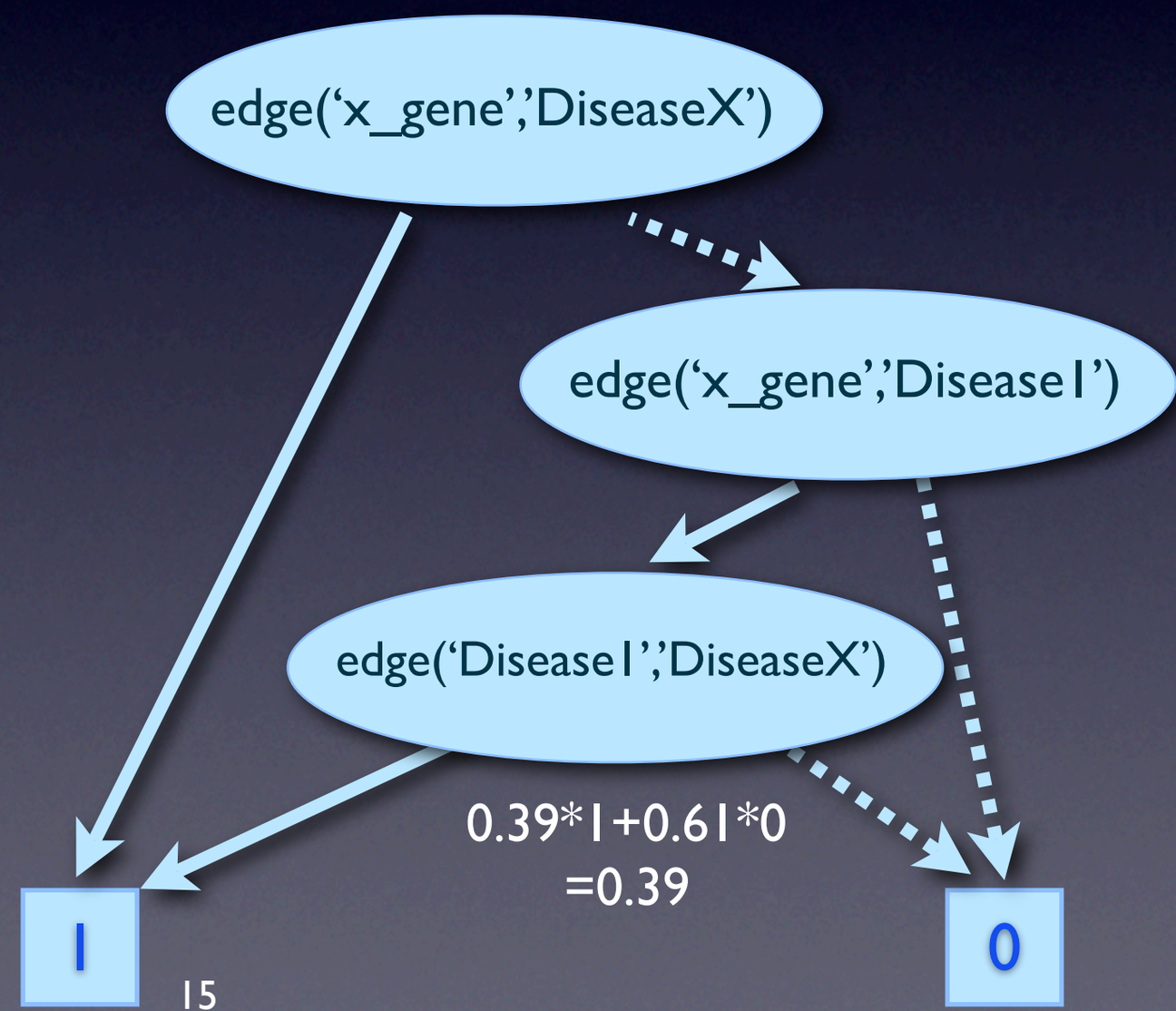




How to calculate the Gradient of the MSE?

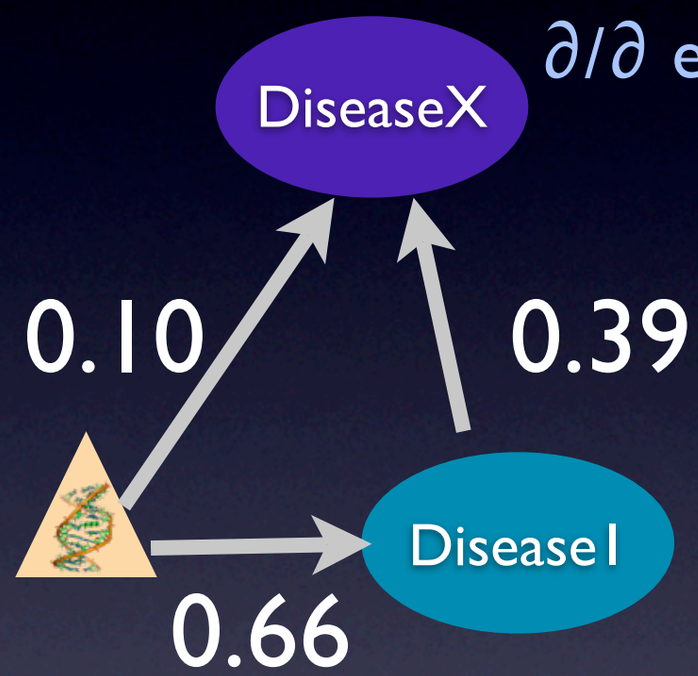
∂/∂ edge('x_gene','Disease1') P_s(path('x_gene', 'DiseaseX') |L)

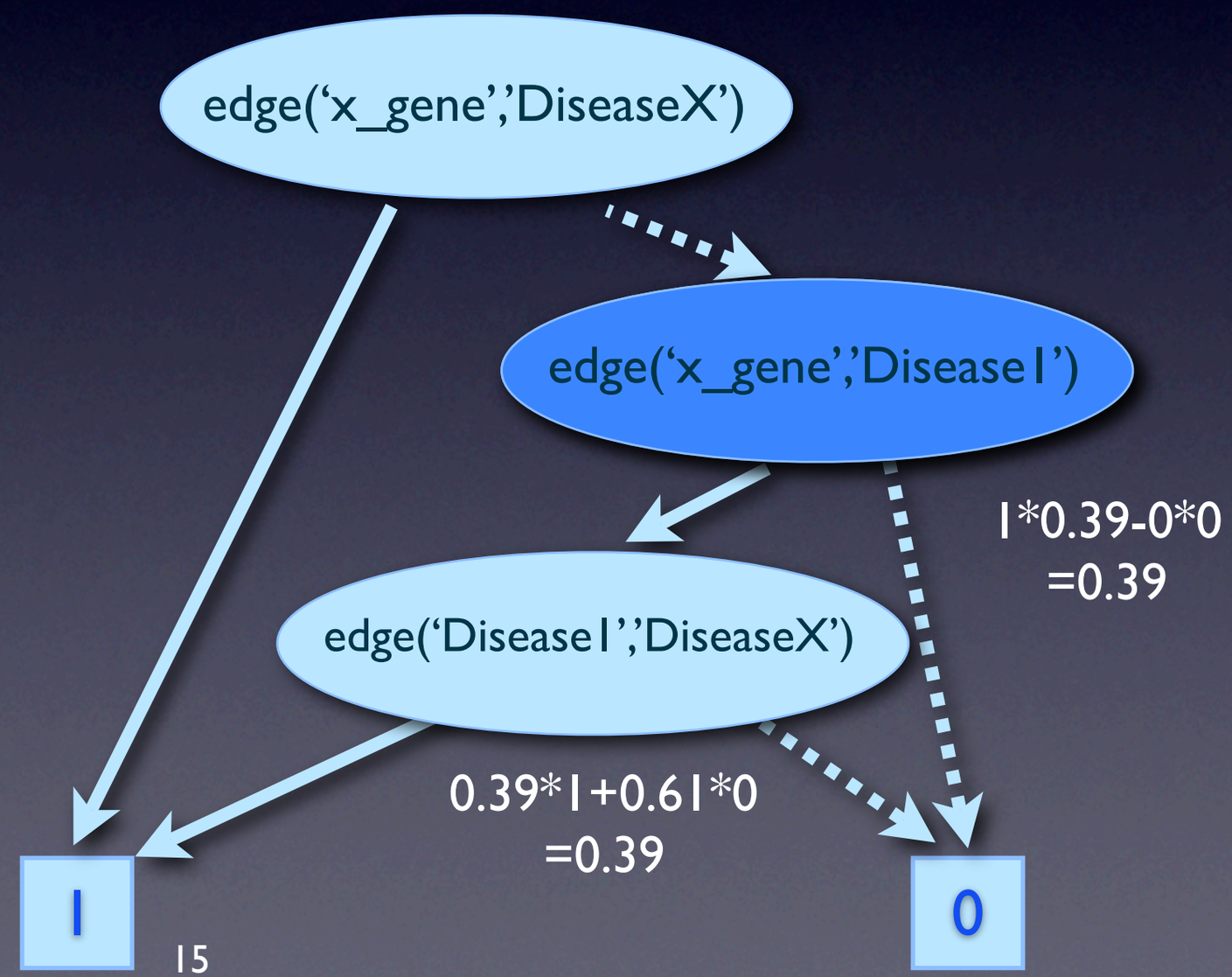




How to calculate the Gradient of the MSE?

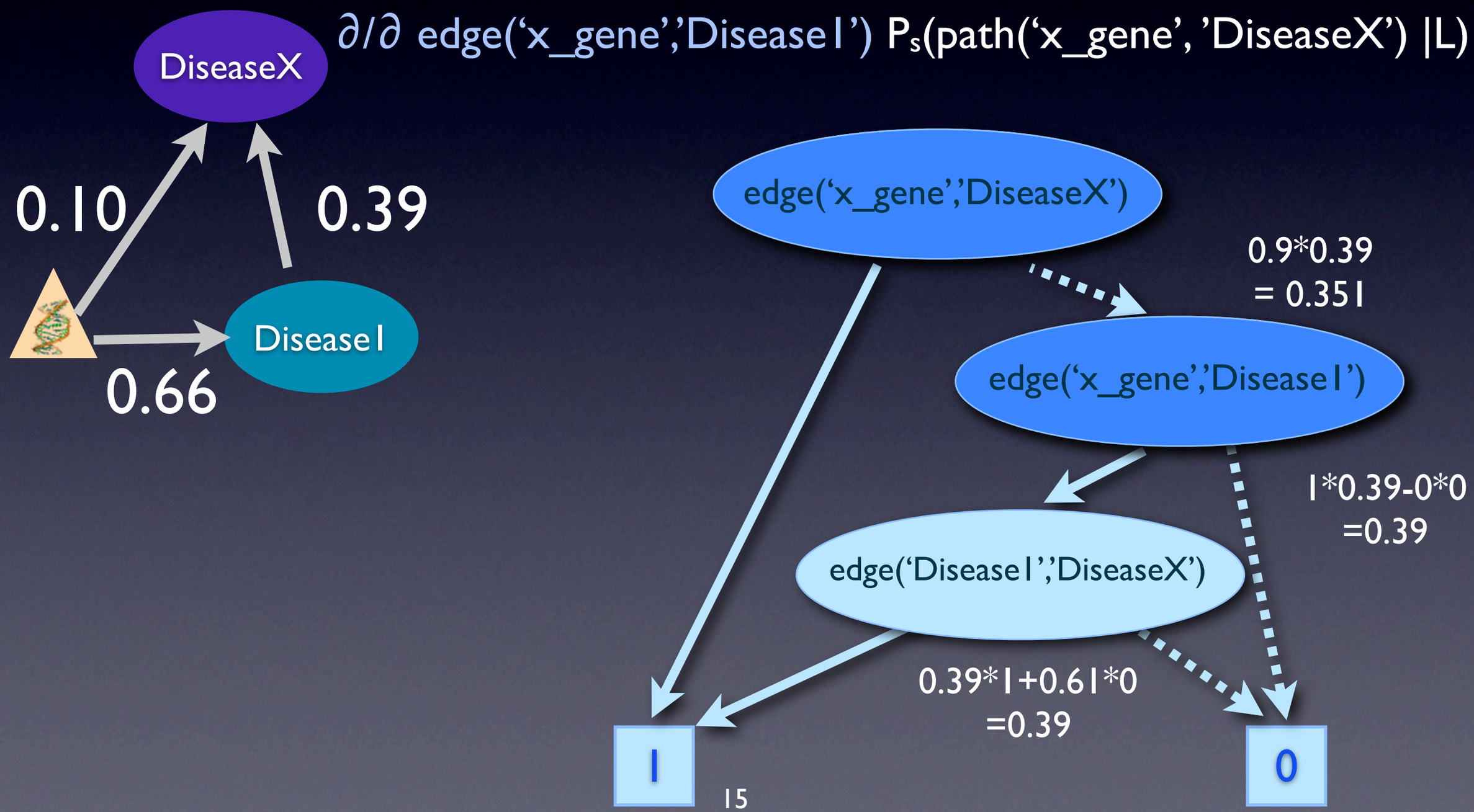
∂/∂ edge('x_gene','Disease1') P_s(path('x_gene', 'DiseaseX') |L)





How to calculate the Gradient of the MSE?

∂/∂ edge('x_gene','Disease1') P_s(path('x_gene', 'DiseaseX') |L)



Proofs are Queries

- queries explain that something is true, but they don't carry the concrete explanation

Proofs are Queries

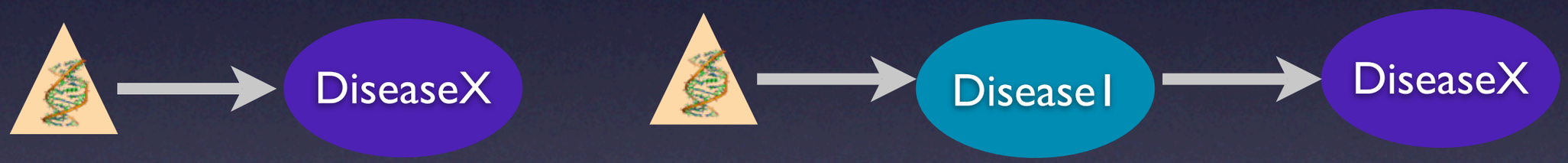
- queries explain that something is true, but they don't carry the concrete explanation

  path('x_gene', 'DiseaseX')

Proofs are Queries

- queries explain that something is true, but they don't carry the concrete explanation

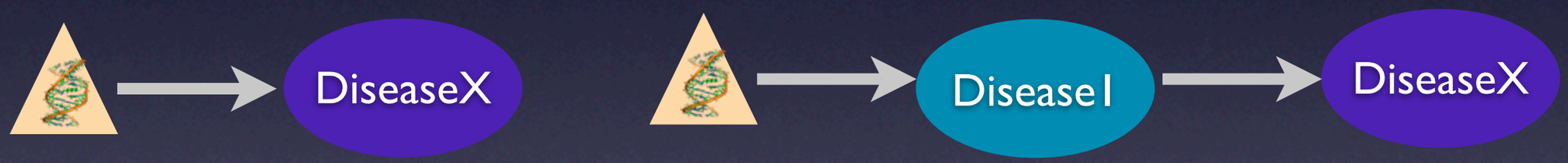
path('x_gene', 'DiseaseX')



Proofs are Queries

- queries explain that something is true, but they don't carry the concrete explanation

  path('x_gene', 'DiseaseX')

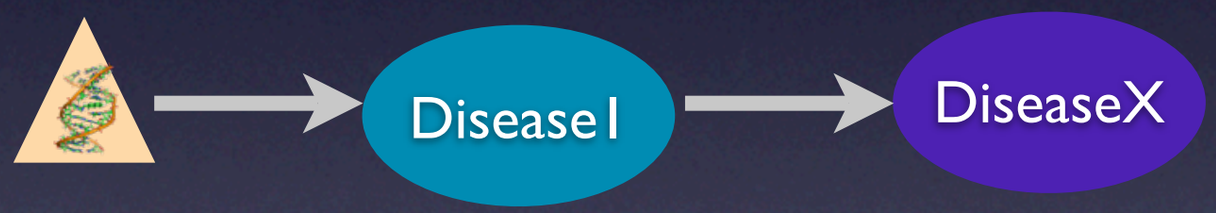


- a proof states, how something happens

Proofs are Queries

- queries explain that something is true, but they don't carry the concrete explanation

path('x_gene', 'DiseaseX')



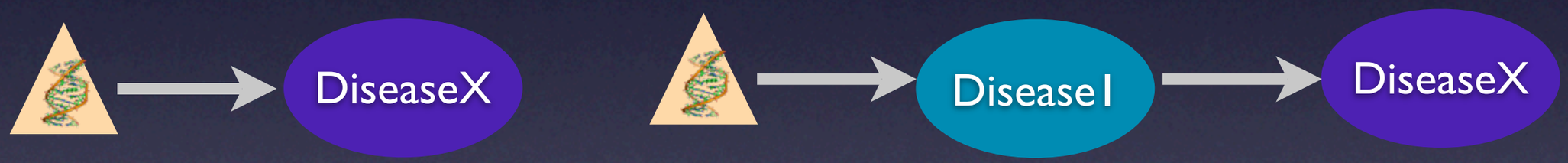
- a proof states, how something happens

edge('x_gene', 'Disease1'), edge('Disease1', 'DiseaseX')

Proofs are Queries

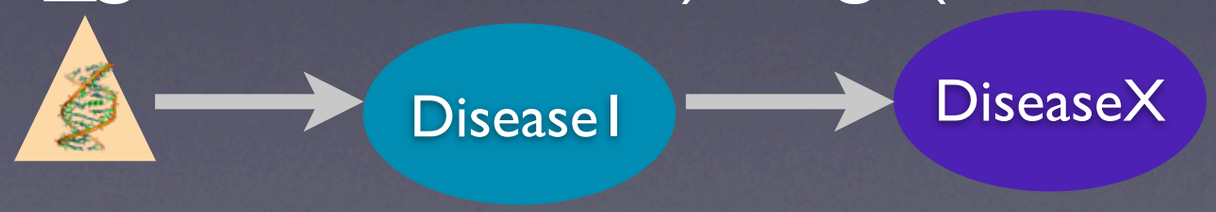
- queries explain that something is true, but they don't carry the concrete explanation

path('x_gene', 'DiseaseX')



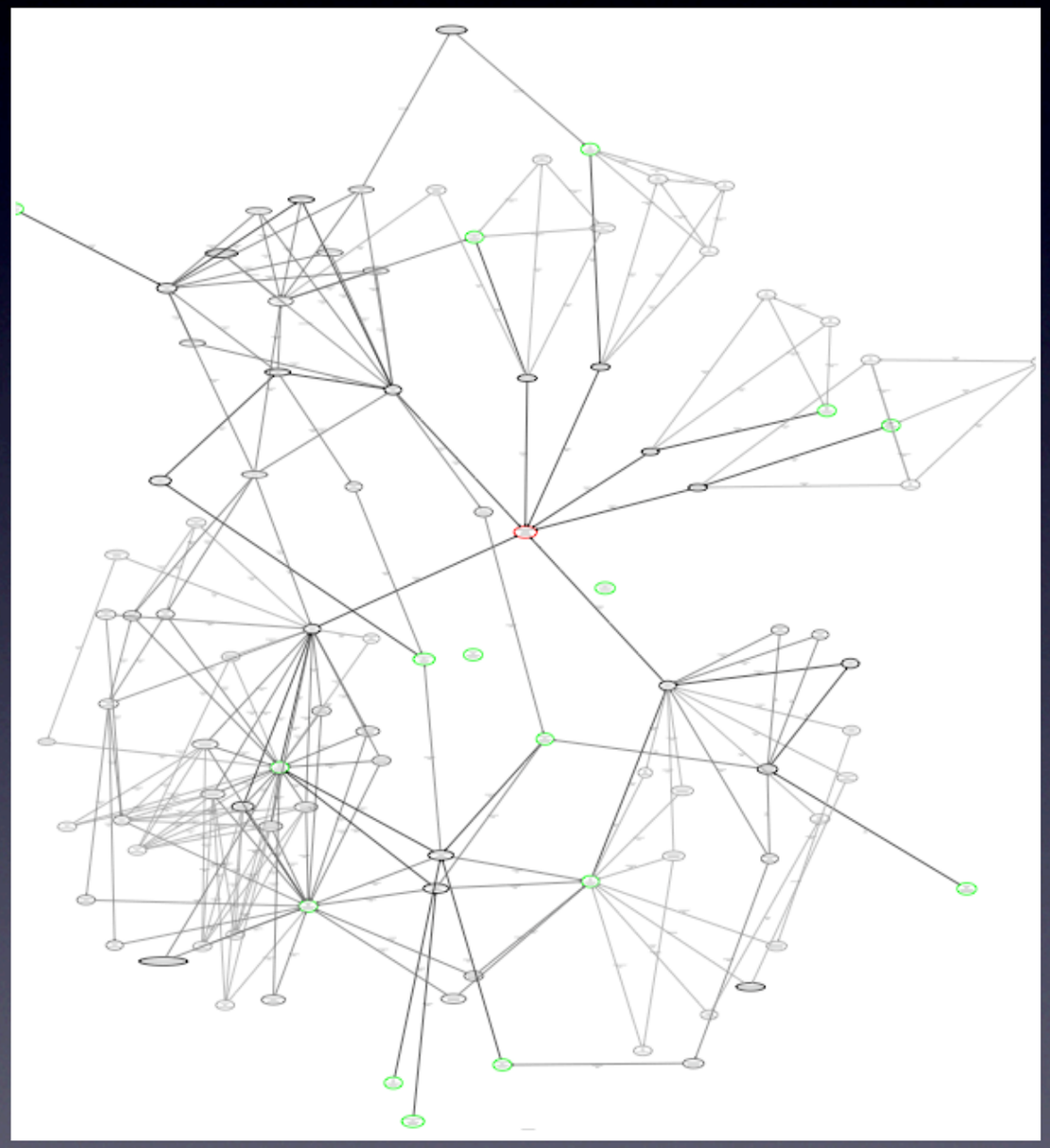
- a proof states, how something happens

edge('x_gene', 'Disease1'), edge('Disease1', 'DiseaseX')

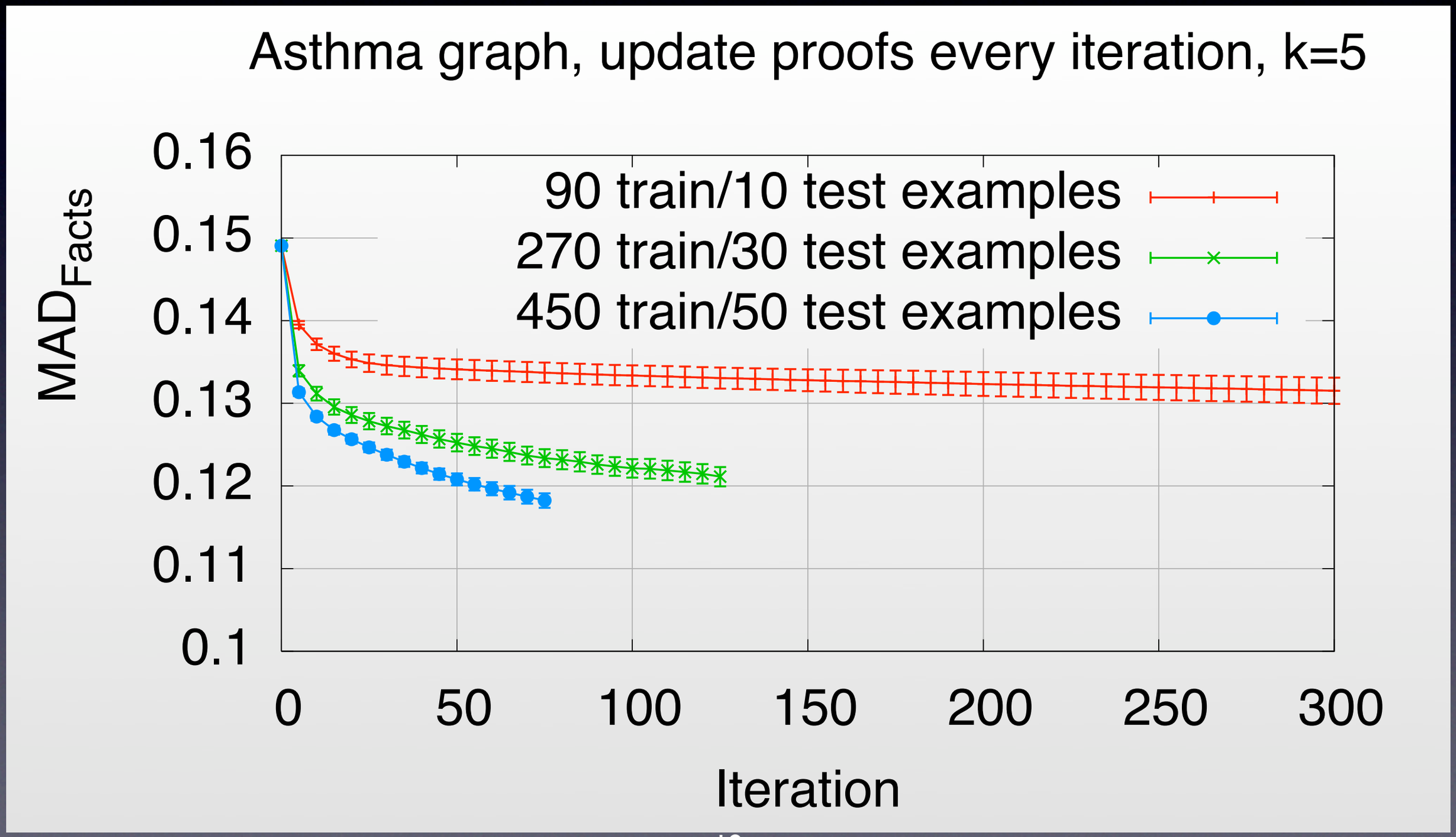


Experiments

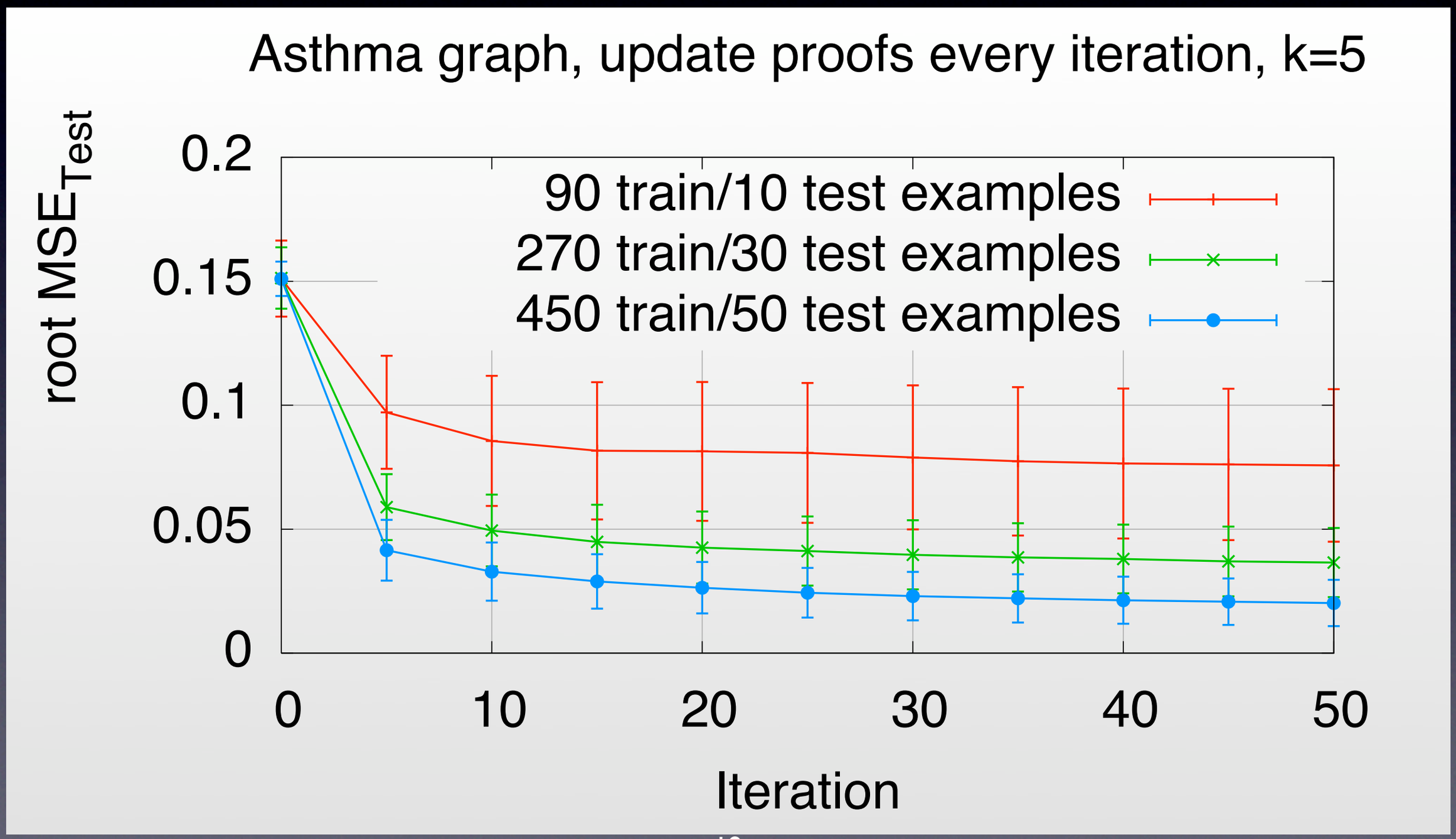
- subgraph from the BIOMINE database around asthma genes
- 127 nodes, 241 edges
- sampled 500 random node pairs
- calculated P(path(X,Y))
- 10-fold cv



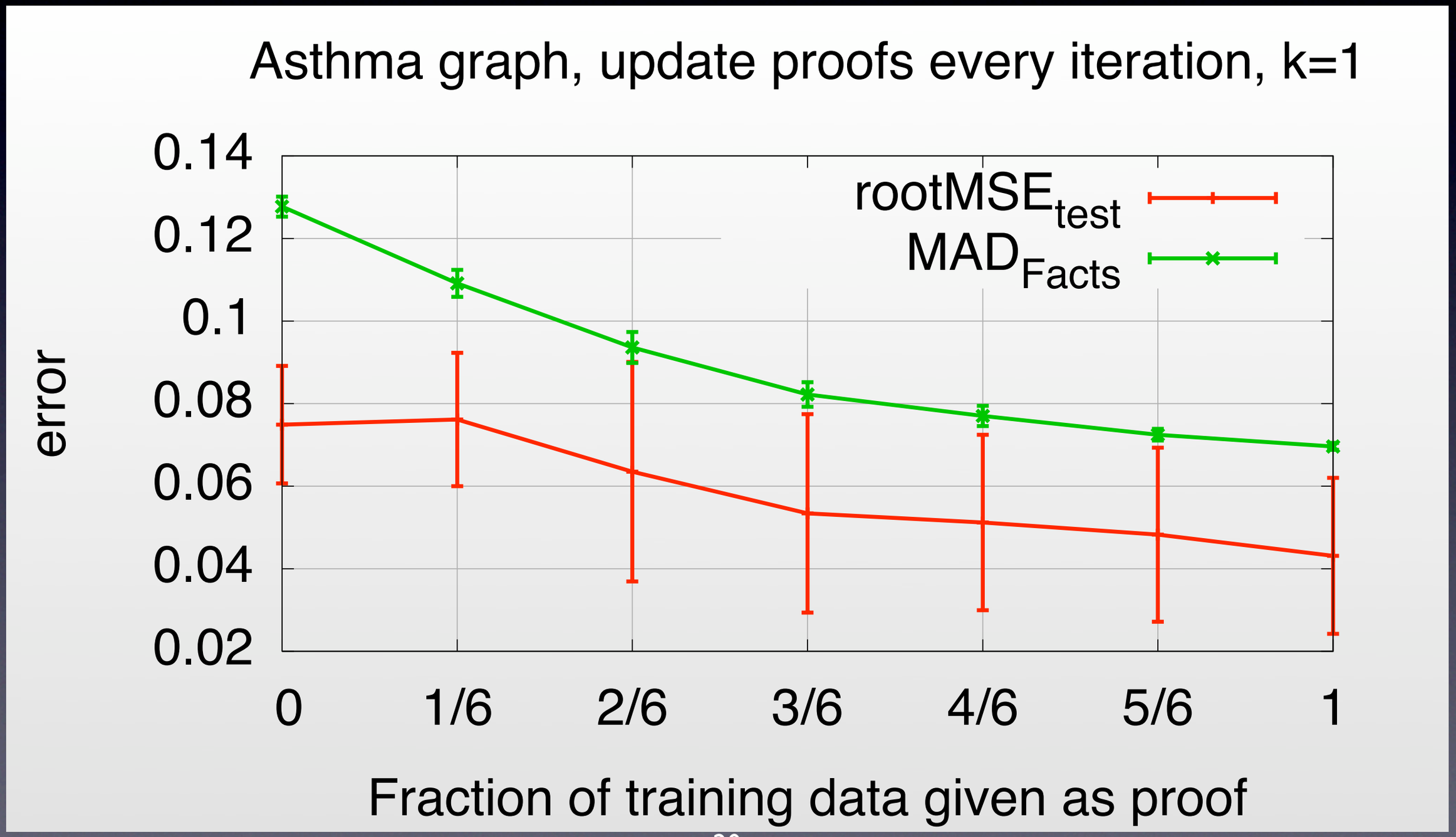
Recovering Probabilities



Reducing Test Error



Using Proofs



Summary

- it works
- it can naturally deal with proofs
- it is efficient due to the BDDs
- applicable for any prob. database where gradient of queries can be calculated

Questions?

More Details: See our ECML 2008 paper