Université catholique de Louvain
Faculté des Sciences Appliquées - FSA

Laboratoire de Télécommunications et Télédétection (TELE)
Département d’Eléctricité (ELEC)

Introduction to Bayesian Networks

Bayesian Networks - Dynamic Bayesian Networks
Inference - Learning
Open-Source Projects for probabilistic calculations

Kosta Gaitanis

Outline

- **Bayesian Networks**
  - What is a Bayesian Network and why use them ?
  - Probabilistic calculations in practice
  - Inference in a BN
- **Learning Bayesian Networks**
  - Why learning
  - Basic learning techniques
- **Temporal models : Dynamic Bayesian Networks**
  - The HMM
  - Inference in the HMM
- **Software Packages**
  - BNT
  - BayesNet
  - PNL

Bayesian Networks

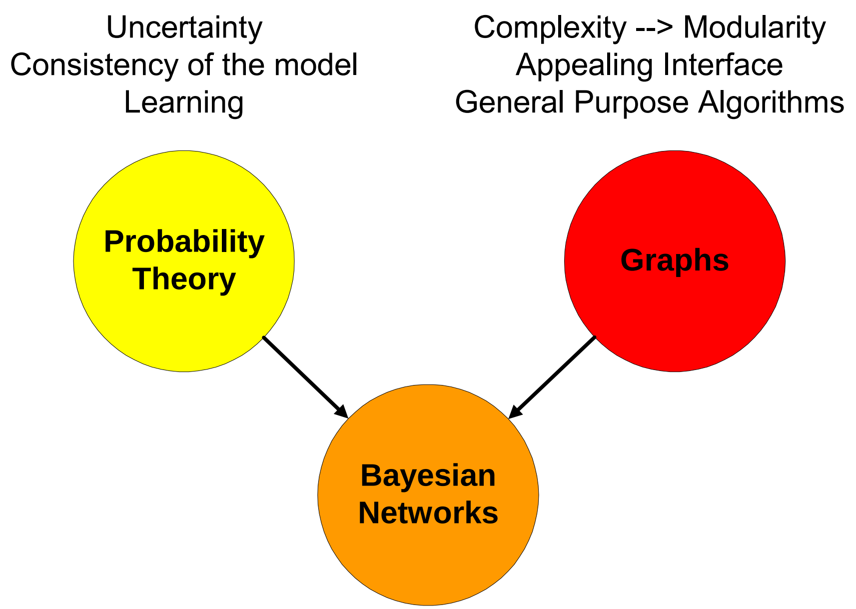
Formal Definition of BNs

Introduction to probabilistic calculations

Basic Inference in BNs

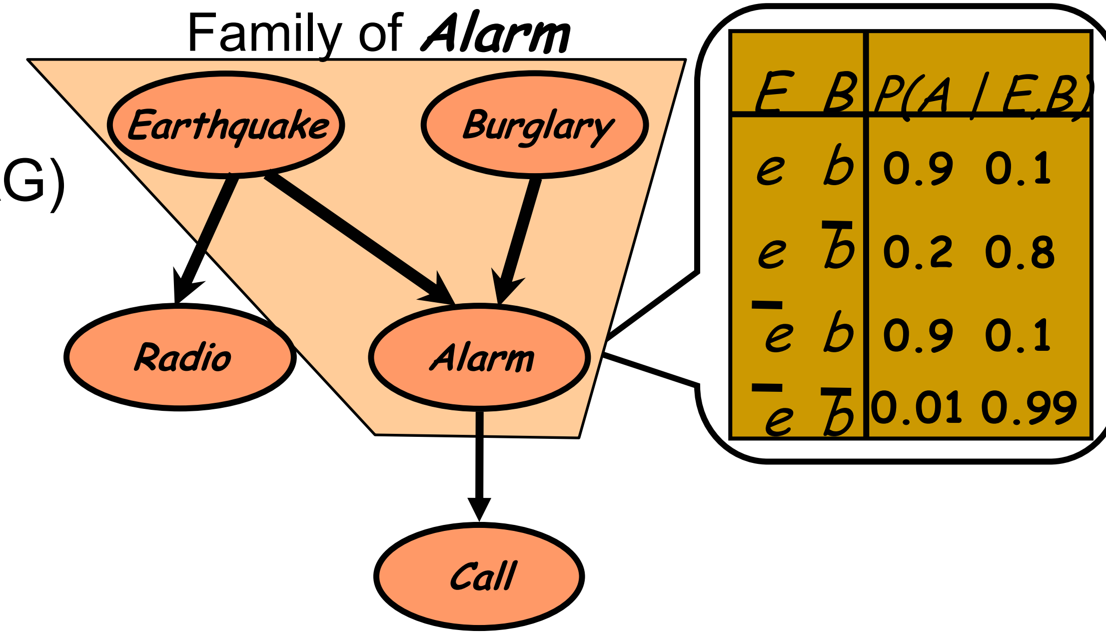
Where do Bayes Nets come from ?

- Common problems in real life :
  - Complexity
  - Uncertainty



What is a Bayes Net ?

Compact representation of joint probability distributions via conditional independence



Qualitative part:

Directed acyclic graph (DAG)

- Nodes - random vars.
- Edges - direct influence

Quantitative part:

Set of conditional probability distributions

Together:

Define a unique distribution in a factored form

Figure from N. Friedman

Why are Bayes nets useful?

- Graph structure supports
  - Modular representation of knowledge
  - Local, distributed algorithms for inference and learning
  - Intuitive (possibly causal) interpretation
- Factored representation may have exponentially fewer parameters than full joint P(X1,…,Xn) =>
  - lower sample complexity (less data for learning)
  - lower time complexity (less time for inference)

What can Bayes Nets be used for ?

- **Posterior probabilities**
  - Probability of any event given any evidence
- Most probable explanation
  - Scenario that explains evidence
- Rational decision making
  - Maximize expected utility
  - Value of Information

Explaining away effect

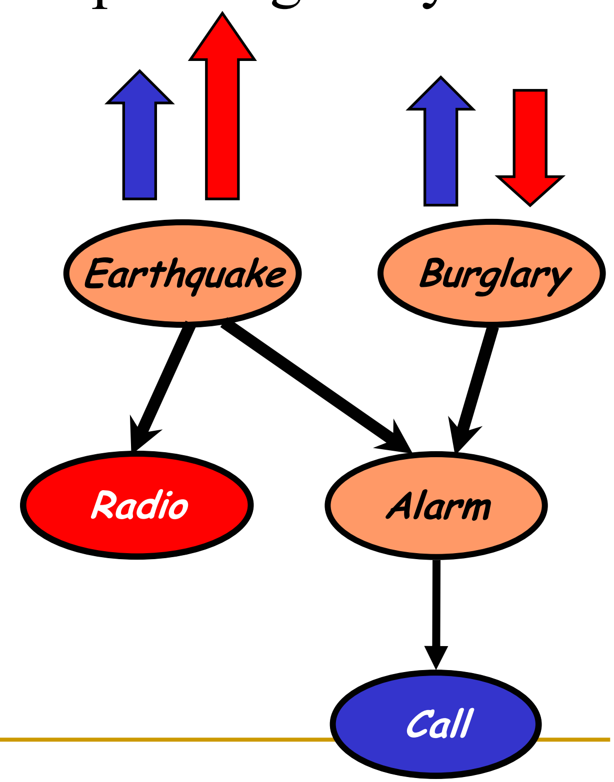


Figure from N. Friedman

A real Bayes net: Alarm

Domain: Monitoring Intensive-Care Patients

- 37 variables
- 509 parameters

…instead of 2^{37}

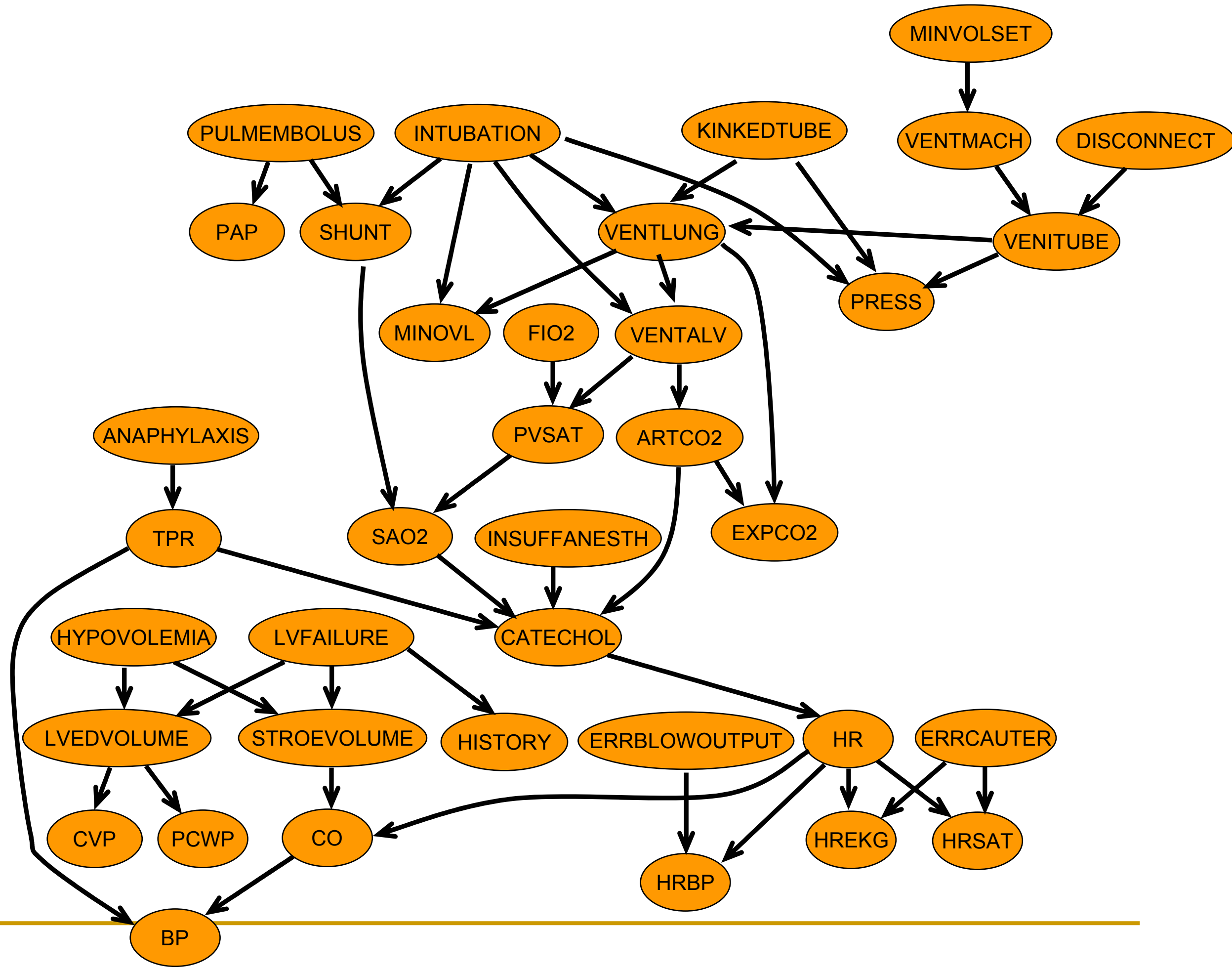
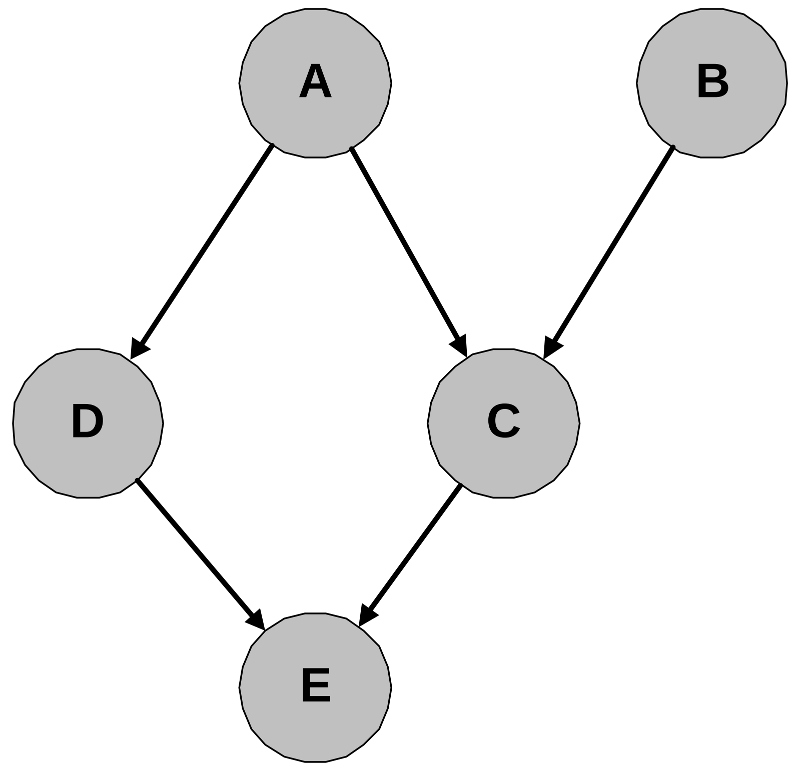


Figure from N. Friedman

Formal Definition of a BN

- **DAG :**
  Directed Acyclic Graph
- **Nodes** :
  each node is a stochastic variable
- **Edges** :
  each edge represents a direct influence between 2 variables
- **CPTs** :
  Quantifies the dependency of two variables → Pr(X|pa(X))
  Eg : Pr(C|A,B), Pr(D|A)
- **A priori distribution** :
  for each node with no parents
  Eg : Pr(A) and Pr(B)



Some Probabilities…

- Bayes Rule : $$\Pr(A,B) = \Pr(A\,|\,B)\cdot \Pr(B) = \Pr(B\,|\,A)\cdot \Pr(A)$$

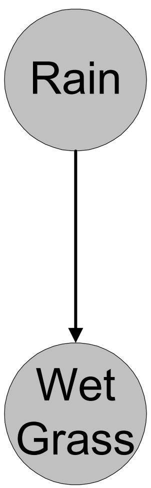
- Independence $A \perp B$ iff : $$\Pr(A\,|\,B) = \Pr(A)$$ $$\Pr(B\,|\,A) = \Pr(B)$$ $$\Rightarrow \Pr(A,B) = \Pr(A)\cdot \Pr(B)$$

- Chain Rule : $$\Pr(A,B,C,D) = \Pr(A)\cdot \Pr(B,C,D\,|\,A)$$ $$= \Pr(A)\cdot \Pr(B\,|\,A)\cdot \Pr(C,D\,|\,A,B)$$ $$= \ldots$$ $$= \Pr(A)\cdot \Pr(B\,|\,A)\cdot \Pr(C\,|\,A,B)\cdot \Pr(D\,|\,A,B,C)$$

- Marginalisation : $$\Pr(A) \propto \Pr(A,B)$$ $$\Pr(A) = \sum_b \Pr(A,B=b)$$

A small example of calculations

Rain	Pr(Rain)
T	0.5
F	0.5



Rain	Wet Grass	Pr(WetGrass|Rain)
F	F	1.0
F	T	0.0
T	F	0.1
T	T	0.9

X

$$\Pr(WG = a, R = b) = \Pr(WG = a \mid R = b) \cdot \Pr(R = b)$$

Rain	Wet Grass	Pr(WetGrass, Rain)
F	F	0.50
F	T	0.00
T	F	0.05
T	T	0.45

Marginalise

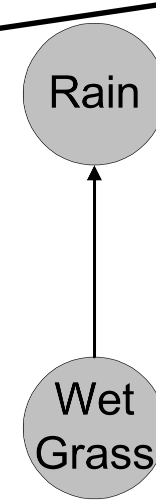
WetGrass	Pr(WetGrass)
T	0.45
F	0.55

$$\Pr(WG = a) = \sum_b \Pr(WG = a, R = b)$$

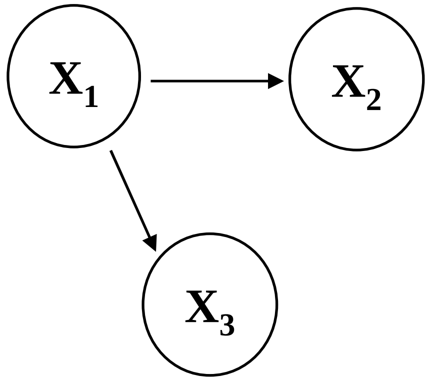
/

$$\Pr(R = a \mid WG = b) = \frac{\Pr(R = a, WG = b)}{\Pr(WG = b)}$$

Rain	Wet Grass	Pr(Rain | WetGrass)
F	F	0.91
F	T	0.0
T	F	0.09
T	T	1.0

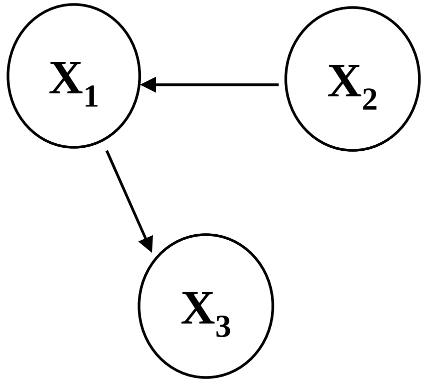


Arc Reversal - Bayes Rule



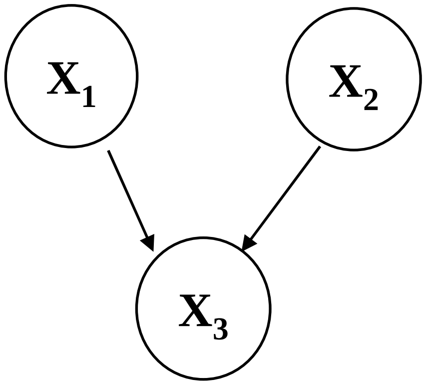
$p(x_1, x_2, x_3) = p(x_3 \mid x_1)\, p(x_2 \mid x_1)\, p(x_1)$

is equivalent to



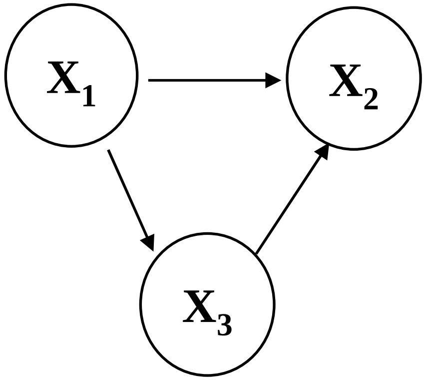
$p(x_1, x_2, x_3) = p(x_3 \mid x_1)\, p(x_2, x_1)$

$= p(x_3 \mid x_1)\, p(x_1 \mid x_2)\, p(x_2)$



$p(x_1, x_2, x_3) = p(x_3 \mid x_2, x_1)\, p(x_2)\, p(x_1)$

is equivalent to

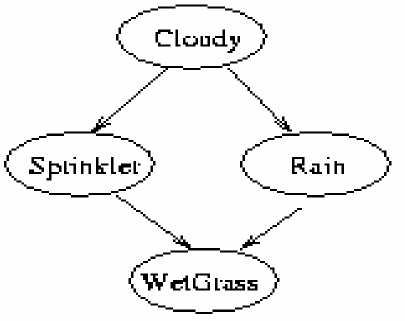


$p(x_1, x_2, x_3) = p(x_3, x_2 \mid x_1)\, p(x_1)$

$= p(x_2 \mid x_3, x_1)\, p(x_3 \mid x_1)\, p(x_1)$

Another example : Water-Sprinkler

P(C=F)	P(C=T)
0.5	0.5



C	P(S=F)	P(S=T)
F	0.5	0.5
T	0.9	0.1

C	P(R=F)	P(R=T)
F	0.8	0.2
T	0.2	0.8

S	R	P(W=F)	P(W=T)
F	F	1.0	0.0
T	F	0.1	0.9
F	T	0.1	0.9
T	T	0.01	0.99

Time needed for calculations

Using Bayes chain rule :

$$\Pr(C,R,S,W) = \Pr(C)\cdot\Pr(R\,|\,C)\cdot\Pr(S\,|\,R,C)\cdot\Pr(W\,|\,R,C,S)$$

2 x 4 x 8 x 16 = 1024

Using conditional independency properties :

$$\Pr(C,R,S,W) = \Pr(C)\cdot\Pr(R\,|\,C)\cdot\Pr(S\,|\,C)\cdot\Pr(W\,|\,R,S)$$

2 x 4 x 4 x 8 = 256

Inference in a BN

- If the grass is wet, there are 2 possible explanations : rain or sprinkler
  - Which is the more likely?

$$\Pr(S=T \mid W=T) = \frac{\Pr(S=T, W=T)}{\Pr(W=T)} = \frac{\sum_{c,r} \Pr(C, R, S=T, W=T)}{\Pr(W=T)} = \frac{0.2781}{0.6471} = 0.430$$ Sprinkler

$$\Pr(R=T \mid W=T) = \frac{\Pr(R=T, W=T)}{\Pr(W=T)} = \frac{\sum_{c,s} \Pr(C, S, R=T, W=T)}{\Pr(W=T)} = \frac{0.4581}{0.6471} = 0.708$$ Rain

The grass is more likely to be wet because of the rain

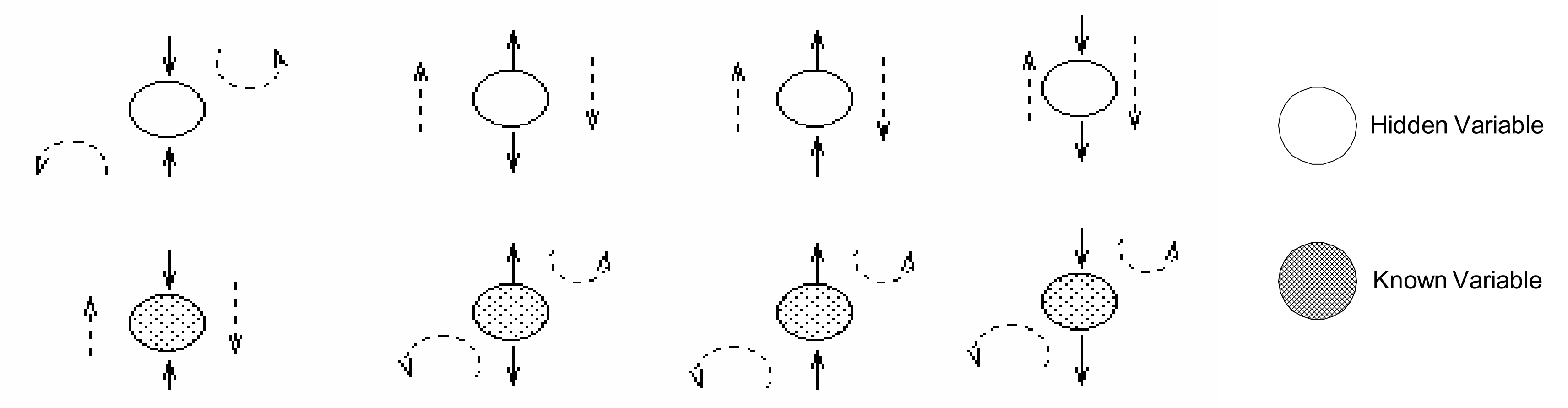
Inference in a BN (2)

- **Bottom-Up :**
  - From effects to causes → diagnostic
  - Eg. Expert systems, Pattern Recognition,…
- **Top-Down :**
  - From causes to effects → reasoning
  - Eg. Generative models, planning,…
- **Explain Away :**
  - Sprinkler and rain "compete" to explain the fact that the grass is wet
    → they are **conditionally dependent** when their common child (wet grass) is observed

Conditional Independence Properties

- **Formal Definition :**
  A node is conditionally independent *(d-separated)* of its ancestors given its parents

- **Bayes Ball Algorithm** :
  - Two variables (A and B) are conditionally independent if a ball can ***not*** go from A to B
  - Permitted movements :



Belief Propagation

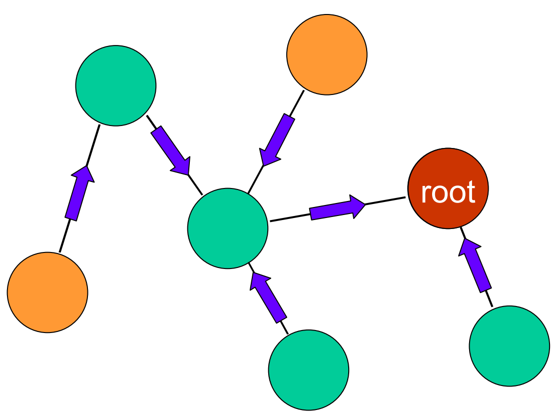
Aka Pearl's algorithm, sum-product algorithm

- 2 pass : Collect and Distribute
- Only works for Poly-trees

The algorithm's purpose is :

"… fusing and propagating the impact of new evidence and beliefs through Bayesian networks so that each proposition eventually will be assigned a certainty measure consistent with the axioms of probability theory." (Pearl, 1988, p 143)

Collect Evidence



Distribute Evidence

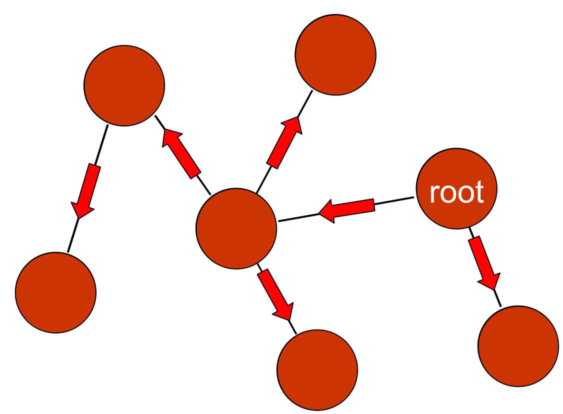
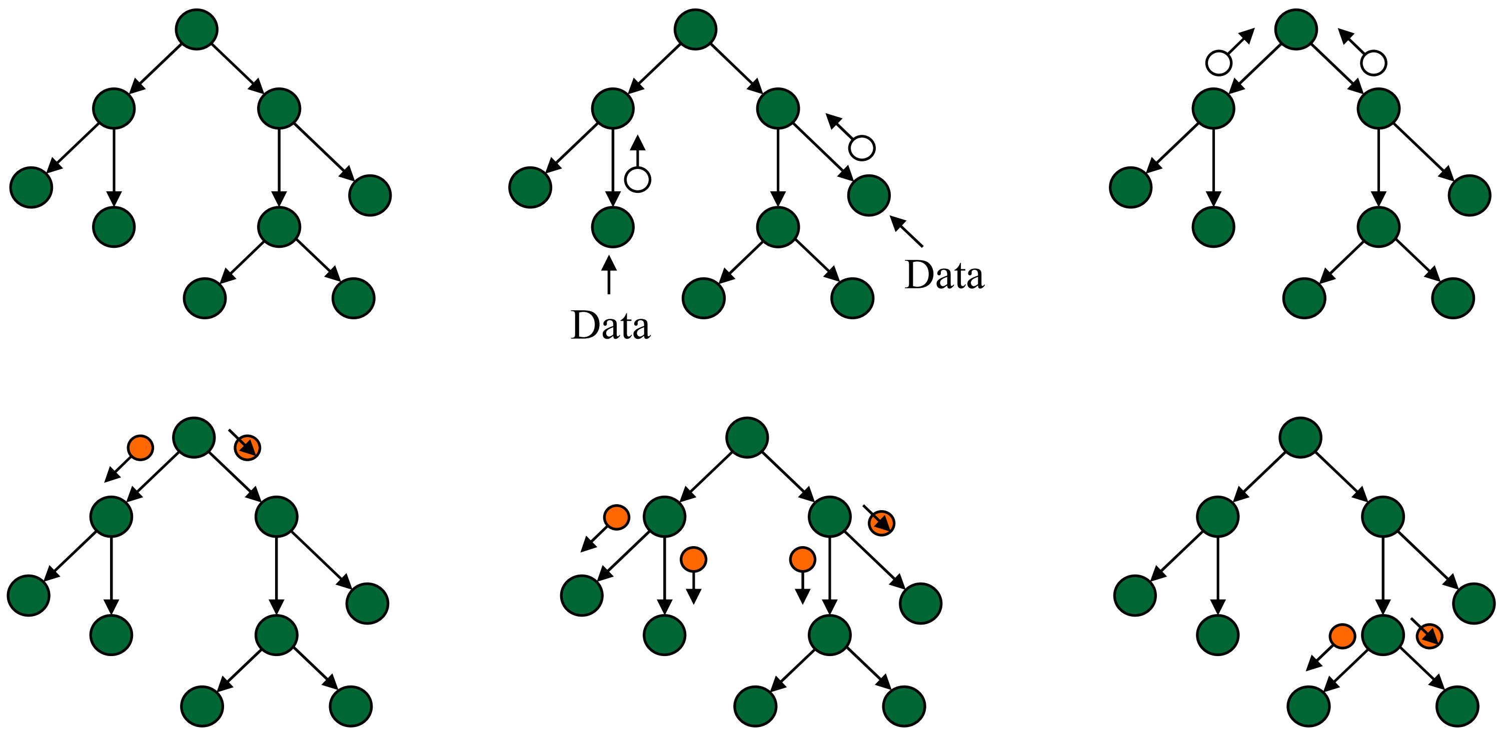


Figure from P. Green

Propagation Example

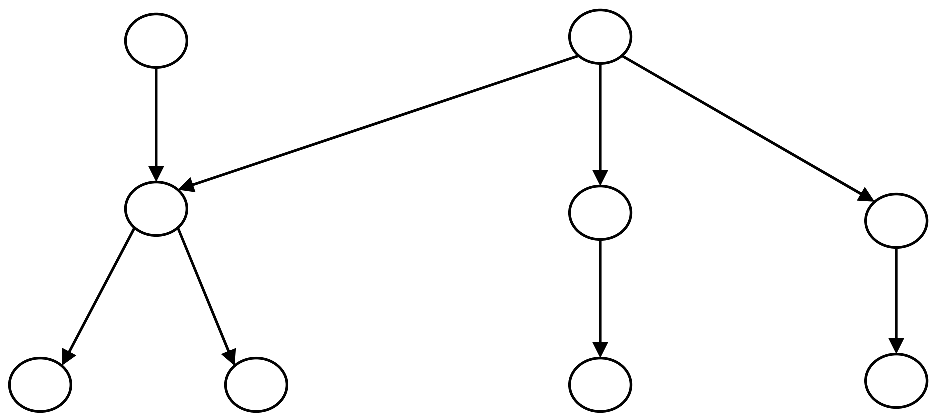
"The impact of each new piece of evidence is viewed as a perturbation that propagates through the network via message-passing between neighboring variables . . ." (Pearl, 1988, p 143`



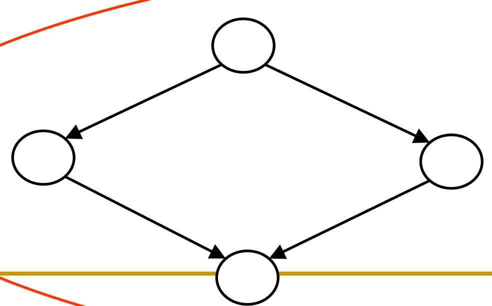
- The example above requires five time periods to reach equilibrium after the introduction of data (Pearl, 1988, p 174)

Singly Connected Networks (or Polytrees)

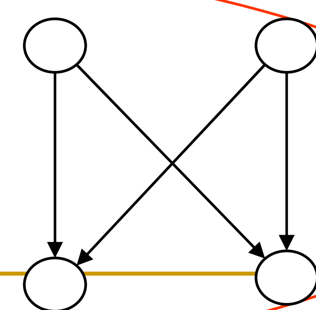
Definition : **A directed acyclic graph (DAG) in which only one semipath (sequence of connected nodes ignoring direction of the arcs) exists between any two nodes.**



Multiple parents and/or multiple children



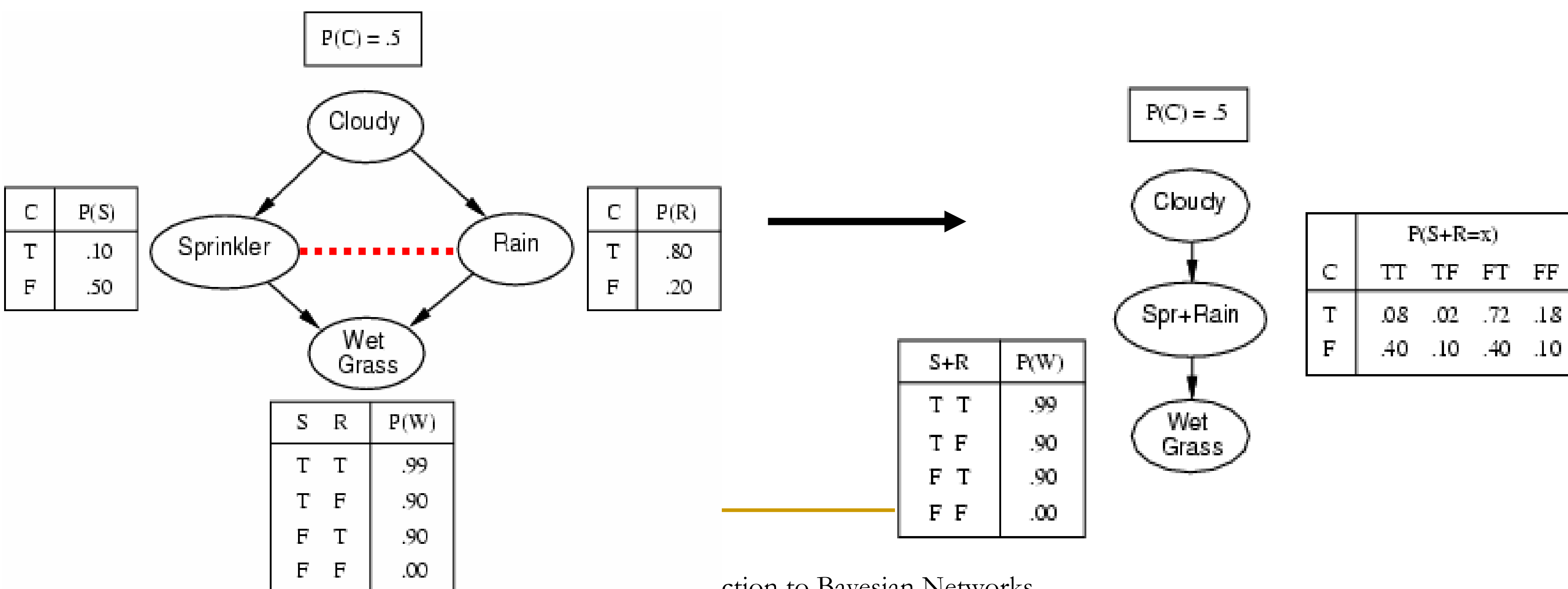
Do not satisfy definition



Inference in general graphs

- BP is only guaranteed to be correct for trees
- A general graph should be converted to a junction tree, by clustering nodes
- Computational complexity is exponential in size of the resulting clusters

→ Problem : Find an optimal Junction Tree (NP-hard)



Approximate inference

- Why?
  - to avoid exponential complexity of exact inference in discrete loopy graphs
  - Because cannot compute messages in closed form (even for trees) in the non-linear/non-Gaussian case

- How?
  - Deterministic approximations: loopy BP, mean field, structured variational, etc
  - Stochastic approximations: MCMC (Gibbs sampling), likelihood weighting, particle filtering, etc

- Algorithms make different speed/accuracy tradeoffs

- Should provide the user with a choice of algorithms

Continuous and discrete nodes

- Discrete stochastic variables are quantified using CPTs
- Continuous stochastic variables (eg. Gaussian) are quantified using σ and μ
- In most cases we use discrete variables because anyway data is discretised when digitalized.
- Any combination of discrete and continuous variables can be used in the same BN

Summary : why are graphical models useful?

- Factored representation may have exponentially fewer parameters than full joint
  - lower time complexity (less time for inference)
  - lower sample complexity (less data for learning)
- Graph structure supports
  - Modular representation of knowledge
  - Local, distributed algorithms for inference and learning
  - Intuitive (possibly causal) interpretation

More real-world BN applications

- "Microsoft's competitive advantage lies in its expertise in Bayesian networks"
  -- Bill Gates, quoted in LA Times, 1996
- MS Answer Wizards, (printer) troubleshooters
- Medical diagnosis
- Genetic pedigree analysis
- Speech recognition (HMMs)
- Gene sequence/expression analysis
- Turbocodes (channel coding)

References

- **A Brief Introduction to Graphical Models and Bayesian Networks** (Kevin Murph, 1998)
  - http://www.cs.ubc.ca/~murphyk/Bayes/bnintro.html
- **Artificial Intelligence I** (Dr. Dennis Bahler)
  - http://www.csc.ncsu.edu/faculty/bahler/courses/csc520f02/bayes1.html
- **Nir Friedman**
  - http://www.cs.huji.ac.il/~nir/
- **Judea Pearl, Causality** (on-line book)
  - http://bayes.cs.ucla.edu/BOOK-2K/index.html
- **Introduction to Bayesian Networks**
  - A tutorial for the 66th MORS symposium
  - Dennis M. Buede, Joseph A. Tatmam, Terry A. Bresnick

Learning Bayesian Networks

Why Learning ?

Basic Learning techniques

Learning Bayesian Networks

- **Process** :
  - **Input:** dataset and prior information
  - **Output:** Bayesian Network

- **Prior Information** :
  - A Bayesian Network (or fragments of it…)
  - Dependency between variables
  - Prior probabilities

Why learn a Bayesian Network?

- **Combine knowledge, engineering and statistical induction**
  - Covers the whole spectrum from knowledge-intensive model construction to data-intensive model induction
- **More than a learning black-box**
  - Explanation of outputs
  - Interpretability and modifiability
  - Algorithms for decision making, value of information, diagnosis and repair
- **Causal representation, reasoning and discovery**
  - Does smoking cause cancer?

The Learning Problem

	Known Structure	Unknown Structure
Complete Data	Statistical parametric estimation (closed-form eq.)	Discrete optimization over structures (discrete search)
Incomplete Data	Parametric optimization (EM, gradient descent,…)	Combined (Structural EM, mixture models,…)

Example : Binomial Experiment

- When tossed, it can land in one of two positions: *Head* or *Tail*
- We denote θ the (unknown) probability *P(H)*

Estimation Task:

Given a sequence of toss samples D=x[1],x[2],…,x[M], we want to estimate the probabilities P(H)= θ and P(T)=1- θ

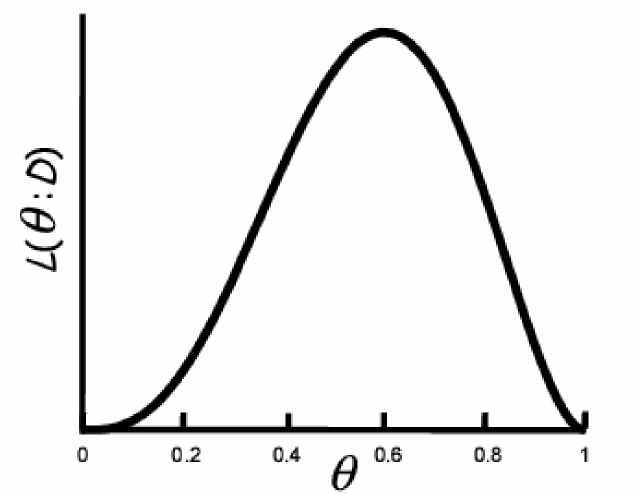
The Likelihood Function

- How good is a particular θ?

  It depends on how likely it is to generate the observed data

$$L(\theta : D) = P(D \mid \theta) = \prod_m P(x[m] \mid \theta)$$

- Thus, the likelihood for the sequence H,T,T,H,H is :



$$L(\theta : D) = \theta \cdot (1-\theta) \cdot (1-\theta) \cdot \theta \cdot \theta$$

Sufficient Statistics

- To compute the likelihood in the thumbtack example, we only require N_H and N_T

$$L(\theta : D) = \theta^{N_H} \cdot (1-\theta)^{N_T}$$

N_H and N_T are **sufficient statistics** for the binomial distribution

- A sufficient statistic is a function that summarizes, from the data, the relevant information for the likelihood :
  - If $s(D)=s(D')$, then $L(\theta|D)=L(\theta|D')$

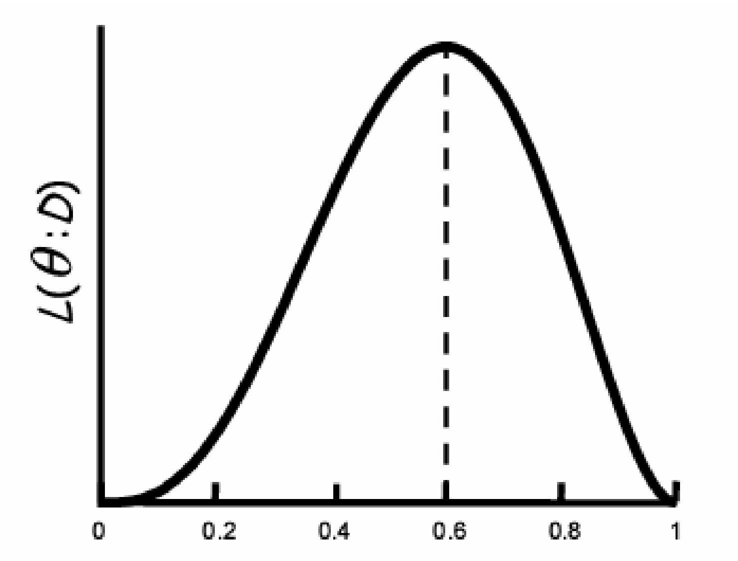
Maximum Likelihood Estimation

- MLE principle :

<u>Learn parameters that maximize the likelihood function</u>

- In our example we get :

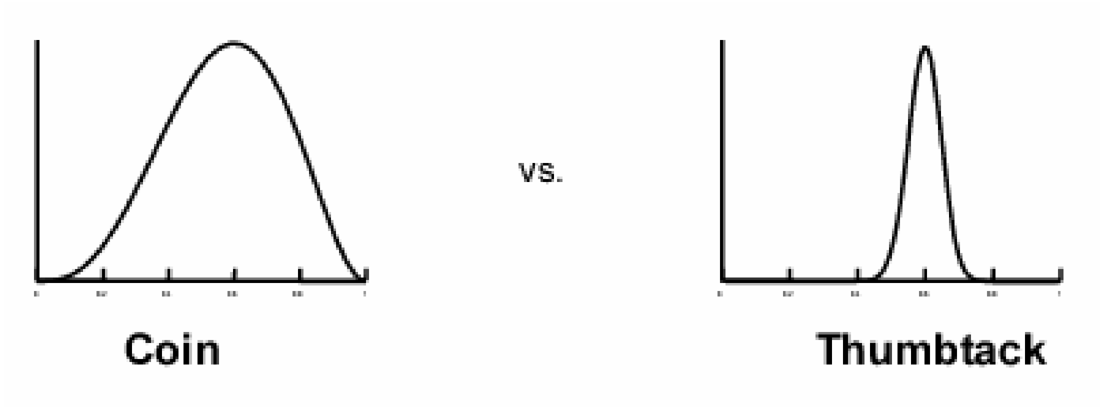
$$\hat{\theta} = \frac{N_H}{N_H + N_T}$$



which is what would one except…

More on Learning

- More than 2 possible values :
  - Same principle but more complex equations, multiple maxima, θ_i ,…
- **Dirichlet Priors :**
  - Add our knowledge of the system to the training data in form of "imaginary" counts
  - Avoid never observed distributions and augment confidence because we have a bigger sample size
- **Confidence :**
  - Confidence in prediction can clearly vary on the data



An alternative exists based on a Bayesian Network to estimate θ

and confidence

More on Learning (2)

- Missing Data :
  - Estimate missing data using bayesian inference
  - Multiple maxima in likelihood function → gradient descent

  - Complicative issue :
    - The fact that a value is missing, might be indicative of its value

    The patient did not undergo X-Ray since she complained about fever and not about broken bones…

References

- **Learning Bayesian Networks from Data** (Nir Friedman, Moises Goldszmidt)
  - http://www.cs.berkeley.edu/~nir/Tutorial
- **A Tutorial on Learning With Bayesian Networks** (David Heckerman, November 1996)
  - Technical Report, MSR-TR-95-06

Temporal Models

Dealing with time

Hidden Markov Models

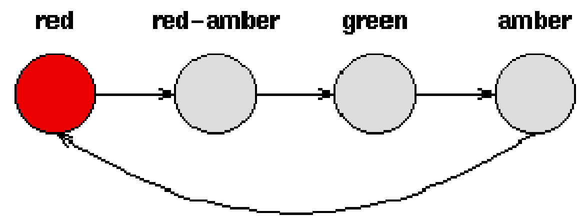
Inference in the HMM

Dealing with time

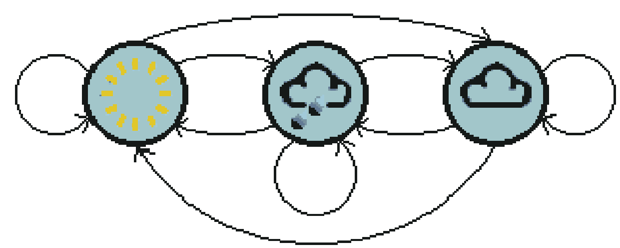
- In many systems, data arrives sequentially
- We want to recognize patterns that appear over time
- Dynamic Bayes nets (DBNs) can be used to model such time-series (sequence) data
- Special cases of DBNs include
  - State-space models
  - Hidden Markov models (HMMs)

Deterministic, non-deterministic patterns

- **Deterministic patterns** :
  - Traffic light
  - FSMs
  - …



- **Non-Deterministic patterns** :
  - Weather
  - Speech
  - Tracking
  - …

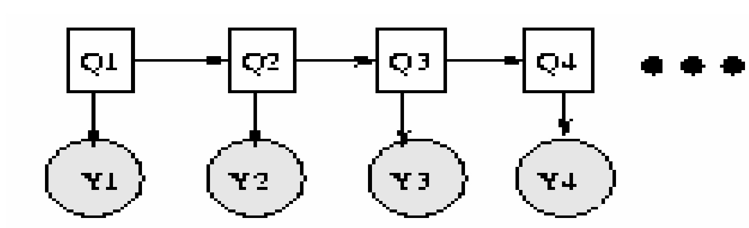


Hidden Markov Models

(Andrei Andreyevich Markov)

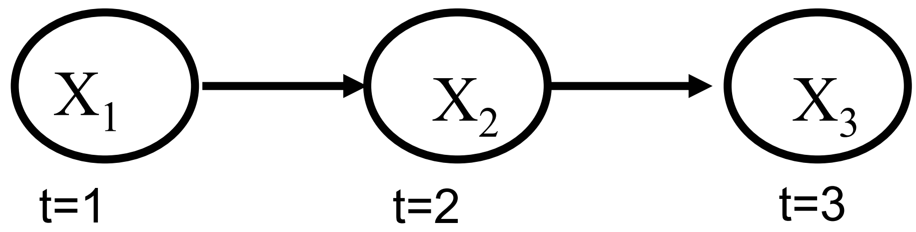
- **Markov Assumption** :
  - Each state at time t only depends on the state at time t-1
  - Eg. The weather today only depends on the weather of yesterday

- **Hidden State** :
  - The state of the system is not directly observable (hidden)
  - Eg. We can not know the exact state of the weather, we only have access to some parameters (wind, temperature,…)



Simple Example : Weather prediction

- Only 3 possible weather states :
  - Sunny, Cloudy, Rainy



- Transition Matrix :
  - A=Pr(today | yesterday)

Weather Today

Weather Yesterday	Sunny	Cloudy	Rainy	
Sunny	0.5	0.25	0.25	=1
Cloudy	0.375	0.125	0.375	=1
Rainy	0.125	0.625	0.375	=1

- Initialisation vector :
  - Π : Pr(weather on first day)

Sunny	Cloudy	Rainy
1.0	0.0	0.0

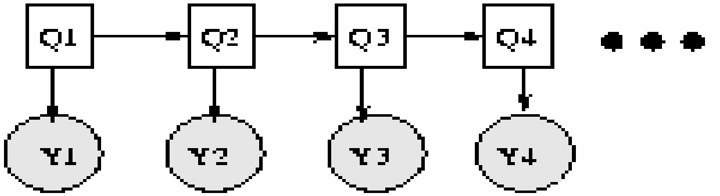
Weather Prediction (2)

- Suppose we only know the state of grass
  - 4 possible states : Dry, Dryish, Wet or Soggy

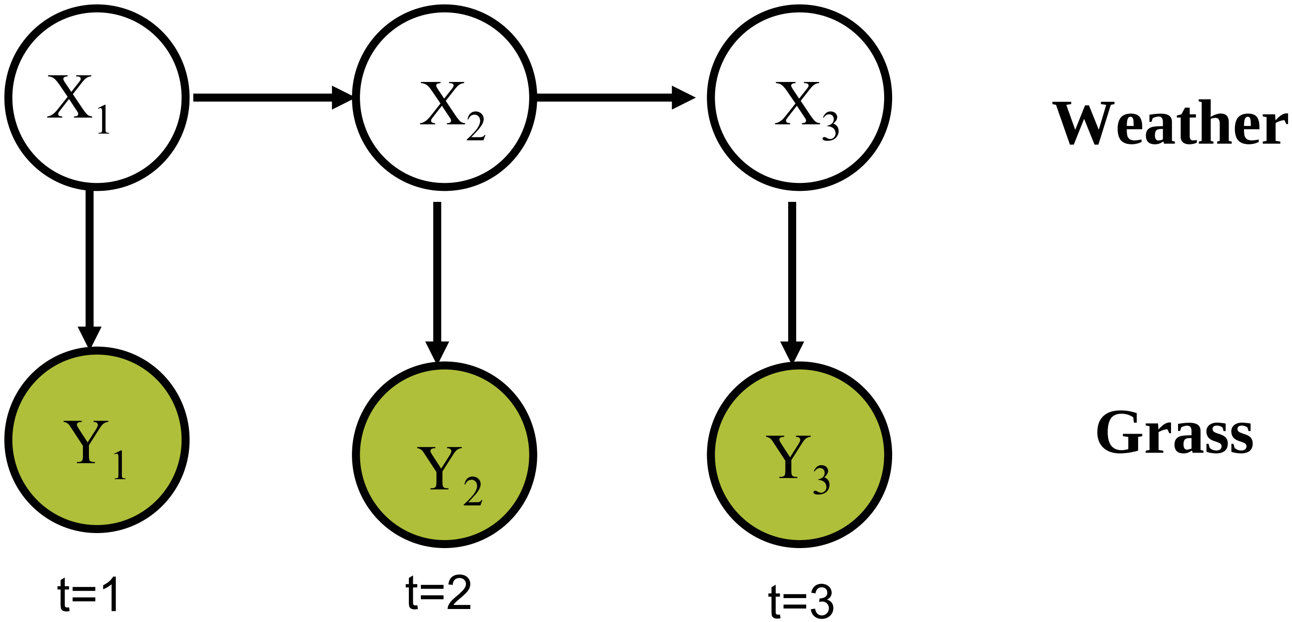
- Observation Matrix
  - B = Pr(Grass|weather)

Grass

Weather	Dry	Dryish	Wet	Soggy	
Sunny	0.60	0.20	0.15	0.05	=1
Cloudy	0.25	0.25	0.25	0.25	=1
Rainy	0.05	0.10	0.35	0.50	=1



HMM : Formal Definition



$$\Pr(X_t = j \mid X_{t-1} = i) = A(i, j)$$ transition matrix

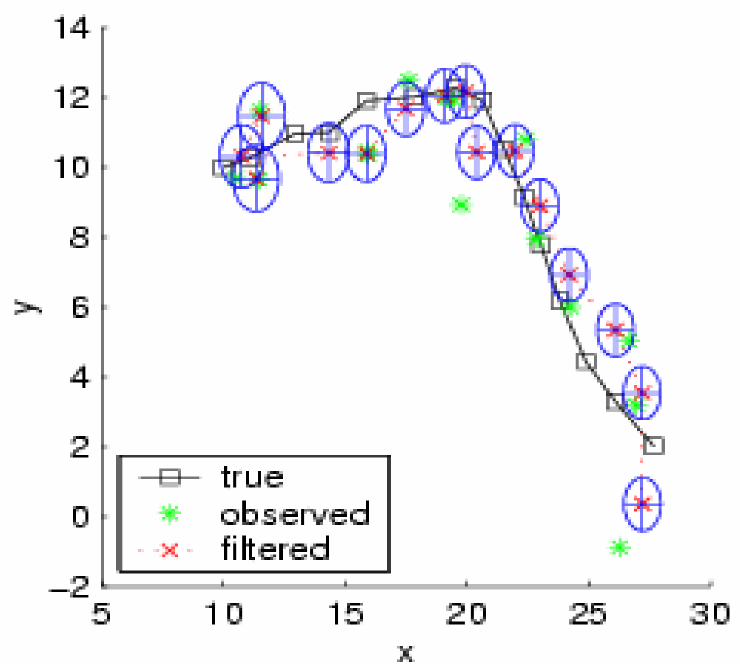
$$\Pr(Y_t = j \mid X_t = i) = B(i, j)$$ Observation/Confusion Matrix

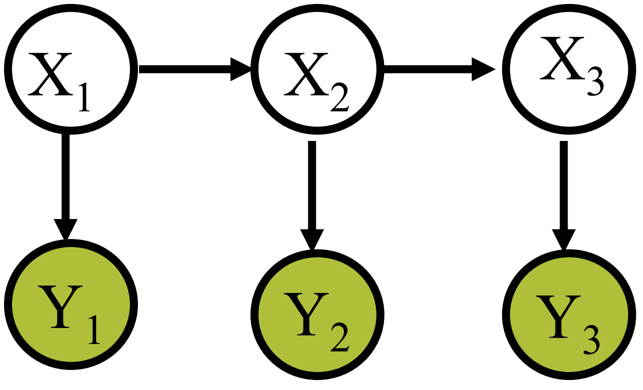
$$\Pi = \Pr(X_1)$$ Initialization Vector

Common Tasks for HMMs

- **Evaluation** (Forward algorithm)
  - Calculate the likelihood of a sequence of observations
  - Choose between many HMMs the one that best explains the observation (speech recognition)
- **Decoding** (Viterbi algorithm)
  - Find the most probable sequence of states given some observations
  - Same as Forward Algorithm, replace + by max
- **Learning** (Forward-Backward Algorithm)
  - Find the parameters of the HMMs that best suit the observations (transition and observation matrices)

Kalman filtering (Forward algorithm)





Estimate $P(X_t|y_{1:t})$ from $P(X_{t-1}|y_{1:t-1})$ and y_t

- **Predict**
- **Update**

Forwards algorithm for HMMs

Predict:

$$P(X_t|y_{1:t-1}) = \sum_{x_{t-1}} P(X_t|x_{t-1})P(X_{t-1}|y_{1:t-1})$$

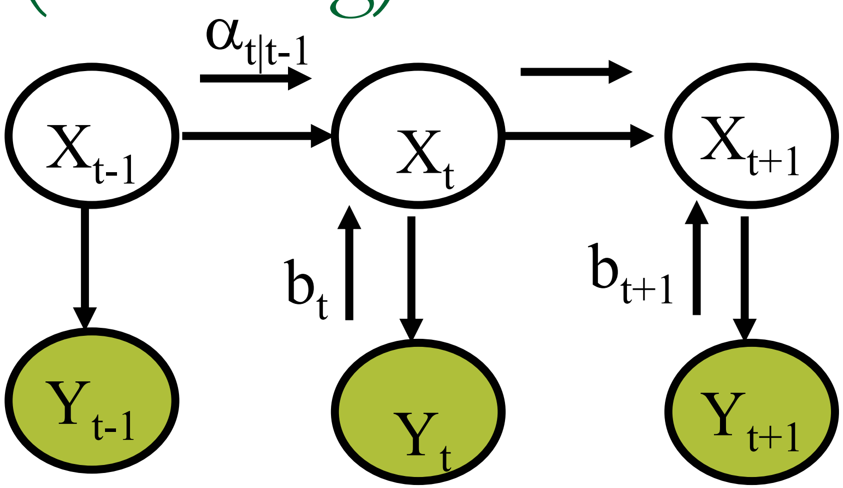
Update:

$$P(X_t = i|y_{1:t}) \propto P(X_t = i|y_{1:t-1})p(y_t|X_t = i)$$

Discrete-state analog of Kalman filter

$O(T\ S^2)$ time using dynamic programming

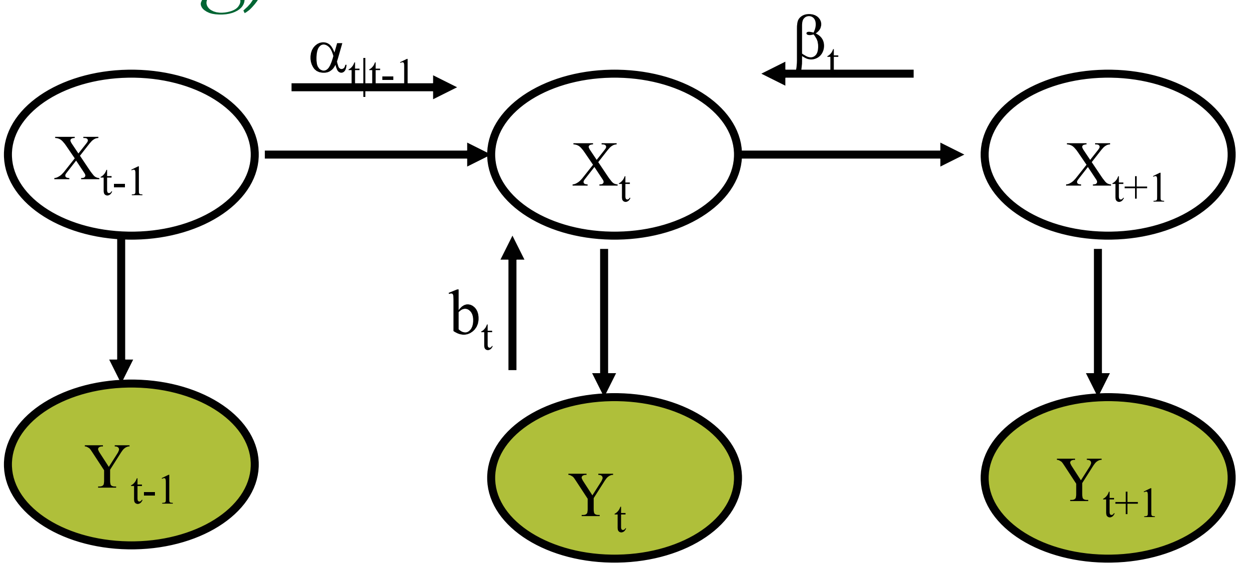
Message passing view of forwards algorithm (Filtering)



$$\alpha_{t|t-1} = A^T \alpha_{t-1}$$

$$\alpha_t \propto \alpha_{t|t-1} .* b_t$$

Forwards-backwards algorithm (Smoothing)



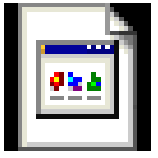
Discrete analog of RTS smoother

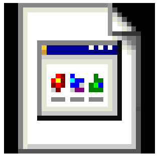
$$P(X_t|y_{1:T}) \propto P(X_t|y_{1:t-1})P(y_t|X_t)P(y_{t+1:T}|X_t)$$

$$\gamma_t(i) \propto \alpha_{t|t-1}(i) b_t(i) \beta_t(i)$$

Example : Viterbi algorithm

- Java Applet


Forward.html


Viterbi.html

References

- **HMMs – Summary** (*L R Rabiner and B H Juang*)
  - http://www.comp.leeds.ac.uk/roger/HiddenMarkovModels/html_dev/summary/s1_pg2.html
- **Nir Friedman**
  - http://www.cs.huji.ac.il/~nir/
- **Inference in belief Networks** : A procedural gruide (Cecile Huang)
- **Tutorial on graphical models and BNT**
  - presented to the Mathworks, May 2003
- **Java Applets** : Prof R.D. Boyle (roger@comp.leeds.ac.uk)

Software Packages

BNT for MatLAB

PNL for C++

BayesNet for Python

BNT for MatLAB

http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html

What is BNT?

- BNT is an open-source collection of matlab functions for inference and learning of (directed) graphical models
- Started in Summer 1997 (DEC CRL), development continued while at UCB
- Over 100,000 hits and about 30,000 downloads since May 2000
- About 43,000 lines of code (of which 8,000 are comments)

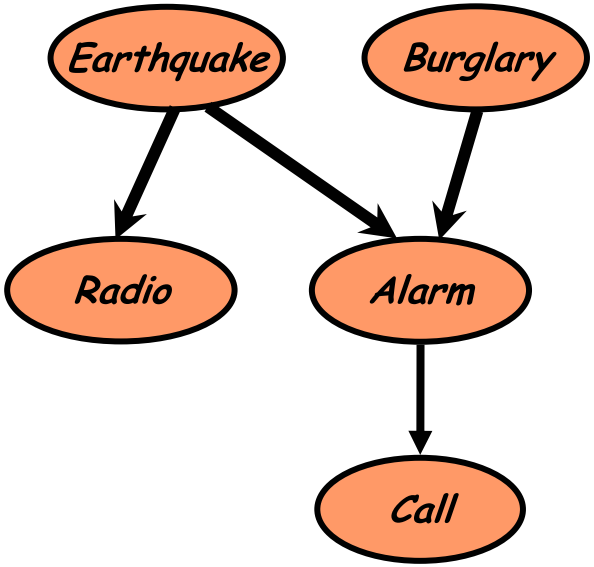
The Burglar-Alarm example

```
% Burglar alarm example

N = 5;
dag = zeros(N,N);
E = 1; B = 2; R = 3; A = 4; C = 5;
dag(E,[R A]) = 1;
dag(B,A) = 1;
dag(A,C)=1;

% true = state 1, false = state 2
ns = 2*ones(1,N); % binary nodes
bnet = mk_bnet(dag, ns);

bnet.CPD{E} = tabular_CPD(bnet, E, [0.1 0.9]);
bnet.CPD{B} = tabular_CPD(bnet, B, [0.01 0.99]);
bnet.CPD{R} = tabular_CPD(bnet, R, [0.65 0.01 0.35 0.99]);
bnet.CPD{A} = tabular_CPD(bnet, A, [0.95 0.8 0.3 0.001 0.05 0.2 0.7 0.999]);
bnet.CPD{C} = tabular_CPD(bnet, C, [0.7 0.05 0.3 0.95]);
```



Inference using BNT

```
engine = jtree_inf_engine(bnet);
ev  = cell(1,N);
ev{C} = 1;
engine = enter_evidence(engine, ev);
mE = marginal_nodes(engine, E);
mB = marginal_nodes(engine, B);
fprintf('P(E|c)=%5.3f, P(B|c)=%5.3f\n', mE.T(1), mB.T(1))
```

Summary of BNT

- Provides many different kinds of models/ CPDs – lego brick philosophy
- Provides many inference algorithms, with different speed/ accuracy/ generality tradeoffs (to be chosen by user)
- Provides several learning algorithms (parameters and structure)
- Source code is easy to read and extend

What is wrong with BNT?

- It is soooo slowwwwwwwwww
- It has little support for undirected models
- Learning engines are not objects
- It does not support online inference/learning
- It has no GUI
- It has no file parser
- It is more complex than necessary

BayesNet for Python

http://cheeseshop.python.org/bayesnet/0.1

BayesNet for Python

- OpenSource project for performing inference on static Bayes Nets using Python

- Python is a high-level programming language
  - Easy to learn
  - Easy to use
  - Fast to write programs
  - Not as fast as C (about 5 times slower), but C routines can be called very easily

The Water-Sprinkler Example

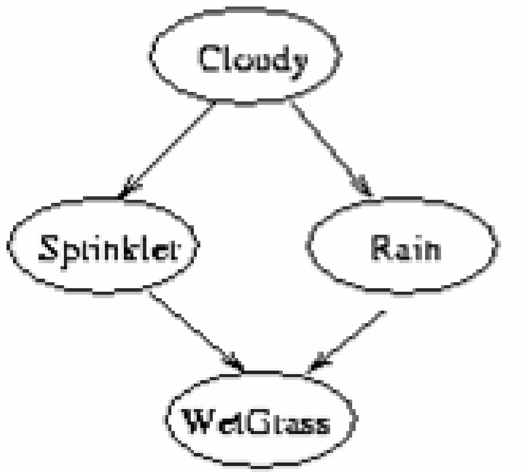
```
# Create empty network
G = BNet('Water Sprinkler Bayesian Network')

# Create boolean nodes: (C)loudy, (S)prinkler, (R)ain, (W)etGrass
c,s,r,w = [ G.add_v(BVertex(nm,2,True))
           for nm in 'c s r w'.split()]

# Add edges : c->r, c->s, r->w, s->w
for ep in [(c,r), (c,s), (r,w), (s,w)]:
        G.add_e(graph.DirEdge(len(G.e), *ep))

# Initialise CPTs
G.InitCPTs()

# Put values into the CPTs
c.setCPT([0.5, 0.5])
s.setCPT([0.5, 0.9, 0.5, 0.1])
r.setCPT([0.8, 0.2, 0.2, 0.8])
w.setCPT([1, 0.1, 0.1, 0.01, 0.0, 0.9, 0.9, 0.99])
```



Inference Using BayesNet

```
JT = JoinTree(G)

JT.SetObs(['w','r'],[1,1])
JT.MargAll()
```

Pros and Cons

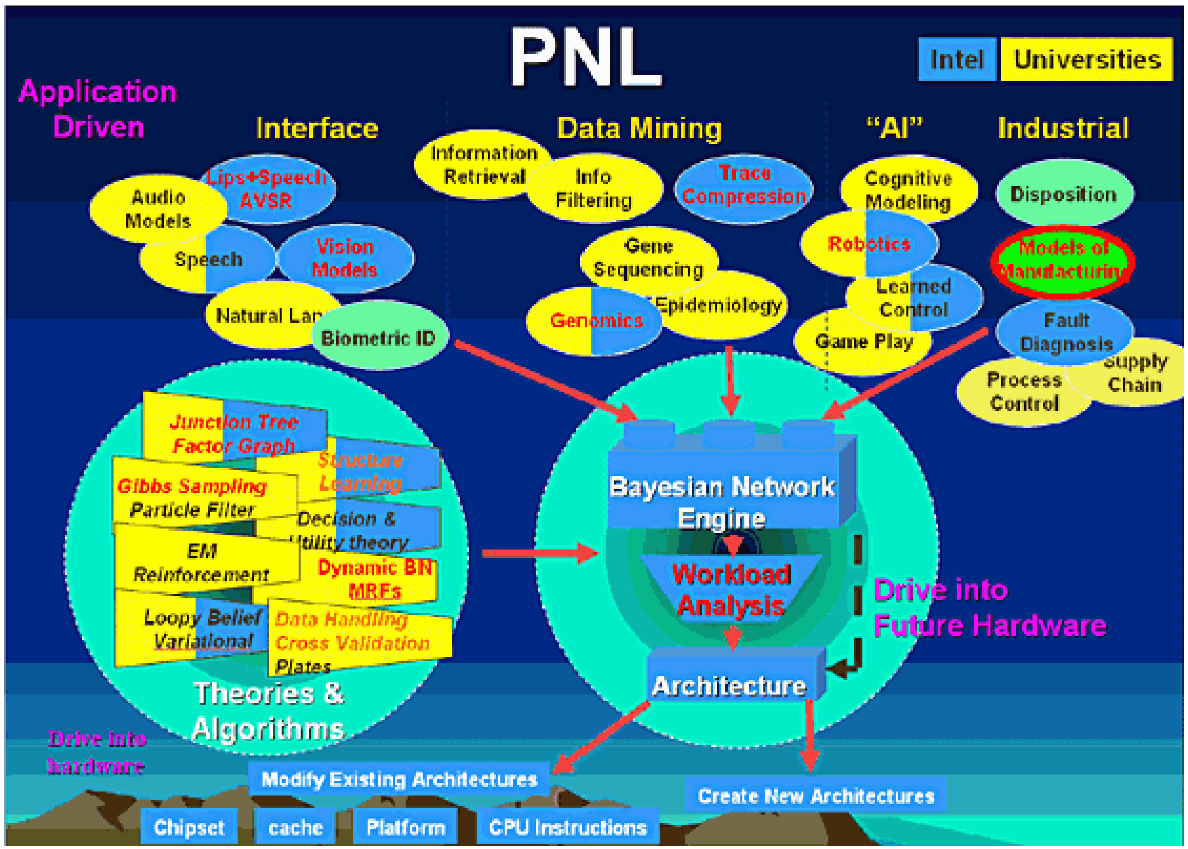
- A little bit easier than BNT
- Much faster than BNT (10 times faster) and can do better
- Only supports :
  - Discrete variables
  - Static BNs
  - Junction-Tree inference
- No learning…(for the moment)

PNL for C++

http://www.intel.com/research/mrl/pnl/

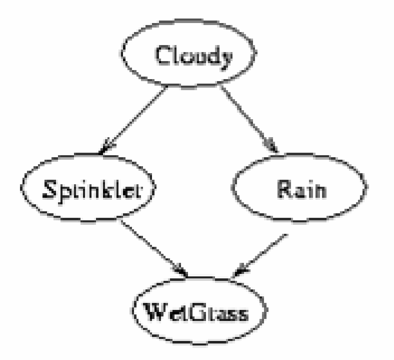
PNL for C++

- Intel and various universities OpenSource Project



An example using PNL

```
PNL_API CBNet* pnlExCreateWaterSprinklerBNet()
{
    // Creation Water-Sprinkler Bayesian network
    const int numOfNds = 4;
    // 1 STEP:
    // need to specify the graph structure of the model;
    // there are two way to do it
    CGraph *pGraph;
        // Graph creation using adjacency matrix
        int numAdjMatDims  = 2;
        int ranges[] = { numOfNds, numOfNds };
        intVector matrixData( numOfNds*numOfNds, 0 );
        CDenseMatrix<int>* adjMat = CDenseMatrix<int>::Create( numAdjMatDims,
            ranges, &matrixData.front() );
        int indices[] = { 0, 1 };
        adjMat->SetElementByIndexes( 1, indices );
        indices[1] = 2;
        adjMat->SetElementByIndexes( 1, indices );
        indices[0] = 1;
        indices[1] = 3;
        adjMat->SetElementByIndexes( 1, indices );
        indices[0] = 2;
        adjMat->SetElementByIndexes( 1, indices );
        // this is a creation of directed graph for the BNet model based on adjacency matrix
        pGraph = CGraph::Create(adjMat);
```




```
    //2 STEP:
    // Creation NodeType objects and specify node types for all nodes of the model.
    nodeTypeVector  nodeTypes;
    // number of node types is 1, because all nodes are of the same type
    // all four are discrete and binary
    CNodeType nt(1,2);
    nodeTypes.push_back(nt);
    intVector nodeAssociation;
    // reflects association between node numbers and node types
    // nodeAssociation[k] is a number of node type object in the
    // node types array for the k-th node
    nodeAssociation.assign(numOfNds, 0);

    // 2 STEP:
    // Creation base for BNet using Graph, types of nodes and nodes association
    CBNet* pBNet = CBNet::Create( numOfNds, nodeTypes, nodeAssociation, pGraph );

    // 3 STEP:
    // Allocation space for all factors of the model
    pBNet->AllocFactors();

    // 4 STEP:
    // Creation factors and attach their to model
    //create raw data tables for CPDs
    float table0[] = { 0.5f, 0.5f };
    float table1[] = { 0.5f, 0.5f, 0.9f, 0.1f };
    float table2[] = { 0.8f, 0.2f, 0.2f, 0.8f };
    float table3[] = { 1.0f, 0.0f, 0.1f, 0.9f, 0.1f, 0.9f, 0.01f, 0.99f };
    float* table[] = { table0, table1, table2, table3 };

    int i;
    for( i = 0; i < numOfNds; ++i )
    {
        pBNet->AllocFactor(i);

        CFactor* pFactor = pBNet->GetFactor(i);

        pFactor->AllocMatrix( table[i], matTable );
    }

    return pBNet;
}
```

Inference using PNL

```
//create Water - Sprinkler BNet
  CBNet* pWSBnet = pnlExCreateWaterSprinklerBNet();

  //display content of Graph
  pWSBnet->GetGraph()->Dump();

  //create simple evidence for node 0 from BNet
  CEvidence* pEvidForWS = CreateEvidenceForWSBNet(pWSBnet);

  //create Naive inference for BNet
  CNaiveInfEngine* pNaiveInf = CNaiveInfEngine::Create( pWSBnet );

  //enter evidence created before
  pNaiveInf->EnterEvidence( pEvidForWS );

  //get a marginal for query set of nodes
  int numQueryNds = 2;
  int queryNds[] = { 1, 3 };

  pNaiveInf->MarginalNodes( queryNds, numQueryNds );
  const CPotential* pMarg = pNaiveInf->GetQueryJPD();

  intVector obsNds;
  pConstValueVector obsVls;
  pEvidForWS->GetObsNodesWithValues(&obsNds, &obsVls);

  int i;
  for( i = 0; i < obsNds.size(); i++ )
  {
      std::cout<<" observed value for node "<<obsNds[i];
      std::cout<<" is "<<obsVls[i]->GetInt()<<std::endl;
  }
```

```
int nnodes;
  const int* domain;
  pMarg->GetDomain( &nnodes, &domain );
  std::cout<<" inference results: \n";

  std::cout<<" probability distribution for nodes [ ";

  for( i = 0; i < nnodes; i++ )
  {
      std::cout<<domain[i]<<" ";
  }

  std::cout<<"]"<<std::endl;

  CMatrix<float>* pMat = pMarg->GetMatrix(matTable);

  // graphical model hase been created using dense matrix
  // so, the marginal is also dense
  EMatrixClass type = pMat->GetMatrixClass();
  if( ! ( type == mcDense || type == mcNumericDense || type == mc2DNumericDense ) )
  {
      assert(0);
  }

  int nEl;
  const float* data;
  static_cast<CNumericDenseMatrix<float>*>(pMat)->GetRawData(&nEl, &data);
  for( i = 0; i < nEl; i++ )
  {
      std::cout<<" "<<data[i];
  }
  std::cout<<std::endl;
```

PNL Pros and Cons

- Very complete
- Very fast (C++)
- Few documentation
- Difficult to use
- Very difficult to create Inference Engines that don't already exist
- Very big project → slow evolution
- Only beta releases (bugs…)

Other Software Packages

By Kevin Murphy

(Commercial and free software)

http://www.cs.ubc.ca/~murphyk/Software/BNT/bnsoft.html

References

- **Bayes Net Toolbox (BNT),** Kevin Murphy
  http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html
- **BayesNet for Python (ver 0.1),** Kosta Gaitanis
  http://cheeseshop.python.org/bayesnet/0.1
  Thanks to Robert Dick dickrp@ece.northwestern.edu
- **Probabilistic Network Library (PNL)**
  Intel and various universities
  http://www.intel.com/research/mrl/pnl/