**University of Missouri-Columbia / College of Engineering**

**CS 8750: Artificial Intelligence II**

****

**Programming Assignment #1**

Adil A. Al-Azzawi  
Chanmann Lim  
Fernando Torre

# **Introduction**

The ability to detect persons who have been drinking alcohol has been an ongoing struggle since the inception of alcoholic drinks themselves. Despite advances in the 20th century, it is yet to be seen for an application to be able to deduce drunkenness from data alone. In this assignment, we attempt to construct Bayesian networks that can achieve this goal.

There are 5 random variables:

* Pd: drink or not. Domain {+, -}
* Xb: breathing rate. Domain {H, M, L}
* Xh: heart rate. Domain {H, M, L}
* Xt: skin temperature. Domain {H, M, L}
* Xa: ambulation status. Domain {Fast, Slow, Stationary}

Pd is our target variable, which we will attempt to derive under different combinations of the other four variables. To test the soundness of the networks; we will test 10 queries, in which we try to determine Pd from a given set of evidences.

1. Xb=H, Xh=H, Xt=H, Xa=Stationary
2. Xb=H, Xh=M, Xt=M, Xa=Fast
3. Xb=H, Xh=M, Xt=L, Xa=Slow
4. Xb=M, Xh=M, Xt=M
5. Xb=M, Xh=L, Xt=M
6. Xb=H, Xt=L, Xa=Slow
7. Xb=L, Xt=L, Xa=Fast
8. Xb=L, Xt=M
9. Xb=L, Xt=H
10. Xb=M

# **BN#1**

We start our exploration with a naïve Bayesian network, flowing from Pd as the root cause, to Xb, Xh, and Xt as the effects.

## **Formula derivations**

## **Queries 1-5**

To answer the first five queries with this network, we must calculate for given evidences Xb, Xh, Xt with known values which we’ll call A, B, and C respectively

Notice that Xb, Xh, Xt are independent given Pd, but are not when Pd is unknown. Therefore we can expand into

Finally,

## **Queries 6-9**

In the next four queries, two nodes of the Bayesian network are known, For this derivation, well use A and B to denote the respective variable holding the query value. Thus, for query 6, shall represent, while for query 8 it shall represent. We apply the same steps as in the previous derivation.

## **Query 10**

The last query is comparatively straightforward:

## **Pseudo code**

For the implementation of this network, we will store all the conditional probabilities in a series of arrays. Thenceforth, the main focus of the program becomes accessing the correct values to plug into the corresponding formula, and thus, answer the query.

**Step.1:** Setup the arrays values // using Bn1 function

p\_pd = [0.13, 0.87];

% the prob. of the breathing rate. Domain

p\_xb\_pb\_plus = [ 0.64 , 0.22 ,0.14];

p\_xb\_pb\_neg = [ 0.09 , 0.42 ,0.49];

% the Prob. of the plus heart rate. Domain

p\_hx\_pd\_plus = [0.54 ,0.31,0.15];

p\_hx\_pd\_neg = [0.12 ,0.42, 0.46];

% the Prob. of the skin temperature. Domain

p\_xt\_pd\_plus = [0.73 ,0.18, 0.09];

p\_xt\_pd\_neg = [0.03 ,0.76,0.21]

**Step 2:** **For** i=1 **to** 10 // find the query from 1 to 10

**Step 2.1:** Get the query features // five random variable values (Xb,Xh,Xt=’High or …,Pd=’+/-‘

**Step 2.2:** For each variable // (Xa,Xh,Xt,Pd) and pd=’+’

**Step 2.2.1:** Find the specific symbol for each variable // X\_Variable=’H’ or ’M’or ’L’

**Step 2.2.1.1:** Enter the random variable text features. // using feature function

**Step 2.2.1.2**: Check the input text if it valid for the feature or not.

**IF** input in ‘High ‘or ‘Medium ‘or ‘Low’ …then Flag=’TRUE’

E**lse** Flag=’False’

**Step 2.2.1.3:** **IF** the Flag=’True’ then go to **step.2.2**

**Else** go to **Step.2.2.1.1**

**Step 2.2.2:** **IF** pd=’+’ **then**

pb= pb\_plus(1,1) //P(pd=’+’))

**Else**  pb= pb\_plus(1,2) //P(pd=’-’))

**Step 2.2.3:** Find the prob. Value for each variable using bn1 function but with pd=’+‘.

**Step 2.2.3.1:** **For** each variable **=**1 to 4 by using each variable symbol

**Step 2.2.3.2: IF** (XB ==’-‘) // the variable is not given

Xb=1; go to **Step 2.2.3.4**

**Else** go to **Step.2.2.3.3**

**Step 2.2.3.3:** Depending on the specific letter of each variable go to **Step.1**

**Step 2.2.3.4: IF** (Xh ==’-‘) // the variable is not given

Xh=1; go to **Step 2.2.3.6**

**Else** go to **step.2.2.3.5**

**Step 2.2.3.5:** Depending on the specific letter of each variable go to **Step.1**

**Step 2.2.3.6: IF** (Xt ==’-‘) // the variable is not given

Xt=1; go to **Step 2.2.3.8**

**Else** go to **Step.2.2.3.7**

**Step 2.2.3.7:** Depending on the specific letter of each variable go to **Step.1**

**Step 2.2.3:** Find the prob. Value for each variable using bn1 function but with **pd=’-‘**.

**Step.2.3:** Find the first and the second part of the equation using Eqs in **sec.2.1**.

**Step.2.4:** Do the printing of the result.

**Step 3:** Next loop.

**Step 4:** End.

## **Matlab code**

What follows is the actual code put into Matlab to run this network. The code is divided into two parts: the main program, containing the user interface (inputting queries, outputting results), and the encoding of the Bayesian network.

## **Main program code:**

%==========================================================================

% CS 8750 - Artificial Intelligence II...

% Programming Assignment #1 ...

% Adil Al-Azzawi ... ECE

% Chanmann Lim ... CS

% Fernando Torre ... CS

%==========================================================================

%%

close all;clc ; clear ;

t=0;

while (t ~= 1)

clc;

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

display(' Programming Assignment #1 ');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

display(' 1: For Bayesian Networks No.1 ');

display(' 2: For Bayesian Networks No.2 ');

display(' 3: For Exit ');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

x = input(' Select the Baysian Network that you want to implement : ');

display(' ');

if (x ==1)

clc;

BayNet\_1(x)

t=0;

elseif (x==2)

clc;

BayNet\_2(x)

t=0;

else

t=1;

end

end

## **BN#1 function code:**

function [ ] = BayNet\_1(ch)

%% Using Bayesian Network No.1...

%% --------------------- Compute Queries No.1 to 10 ------------------------

for i=1:10

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

display(' Bayesian Network No.1 ');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

fprintf('Compute Query No: %d\n', i);

display(' ');

%%----------------------- Bayesia Network #1--------------------

% Find the query features (five random variables)..

[xb,xh,xt,xa,pd] = features(ch );

% From Text to Prob. symbol...

[c1,~] = pdf2(xb);

[c2,~] = pdf2(xh);

[c3,~] = pdf2(xt);

%[c4,~] = pdf2(xa);

[c5\_post,c5\_neg] = pdf2(pd);

% Find the prob. of given Pd...

if (c1~='-')

[p\_xb] = bn1(c1,' ',' ',' ',c5\_post)

else

p\_xb=1;

end

if (c1~='-')

[p\_xh] = bn1(' ',c2,' ',' ',c5\_post)

else

p\_xh=1;

end

if (c1~='-')

[p\_xt] = bn1(' ',' ',c3,' ',c5\_post)

else

p\_xt=1;

end

%[p\_xa] = bn1(' ',' ',' ',c4,c5\_post)

[pd\_plus] = bn1(' ',' ',' ',' ',c5\_post)

% Find the prob. of not given Pd...

p\_xb\_not\_pd= bn1(c1,' ',' ',' ',c5\_neg)

p\_xh\_not\_pd= bn1(' ',c2,' ',' ',c5\_neg)

p\_xt\_not\_pd= bn1(' ',' ',c3,' ',c5\_neg)

pd\_neg\_pd= bn1(' ',' ',' ',' ',c5\_neg)

%

%-------------------- P(xb,xh,xt/pd(+))P(pd(+) -------------

p\_xbxhxt\_pd=p\_xb\*p\_xh\*p\_xt\*pd\_plus

%

%---------------------- p(xb,xh,xt) ------------------------

p\_xbxhxt=p\_xbxhxt\_pd+(p\_xb\_not\_pd\*p\_xh\_not\_pd\*p\_xt\_not\_pd\*pd\_neg\_pd)

%

%------------------Final Result P(Pd/Xb,Xh,Xt) -------------

result=p\_xbxhxt\_pd/ p\_xbxhxt

Print(i,c1,c2,c3,'-',result,p\_xb,p\_xh,p\_xt,0)

input('Press enter to continue...','s');

close all;clc;

end

end

function [ xb,xh,xt,xa,pd] = features(ch )

fla=0;clc;

while (fla ~= 1)

clc;

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

display(' Breathing Rate Domain ');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' 1: H for High...');

display(' 2: M for Medium...');

display(' 3: L for Low...');

display(' 4: X for Non...');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

xb = input(' Enter the breathing rate domain (xb):','s');

display(' ');

[fla] = check('b',xb);

if (fla == 1)

break;

else

msgbox('Invalid Value', 'Error','error');

end

end

fla=0;clc;

while (fla ~=1)

clc;

fprintf(' Baysian Network No.(%d%s\n', ch,')');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

display(' Heart Rate Domain ');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' 1: H for High...');

display(' 2: M for Medium...');

display(' 3: L for Low...');

display(' 4: X for Non...');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

xh = input(' Enter Heart rate domain (xh) : ','s');

display(' ');

[fla] = check('h',xh);

if (fla == 1)

break;

else

msgbox('Invalid Value', 'Error','error');

fla=0;

end

end

fla=0;clc;

while (fla ~= 1)

clc;

fprintf(' Baysian Network No.(%d%s\n', ch,')');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

display(' Skin Temperature Domain ');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' 1: H for High... ');

display(' 2: M for Medium... ');

display(' 3: L for Low... ');

display(' 4: X for Non... ');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

xt = input(' Enter skin temperature domain (xt): ','s');

display(' ');

[fla] = check('t',xt);

if (fla == 1)

break;

else

fla=0;

msgbox('Invalid Value', 'Error','error');

end

end

fla=0;clc;

while fla ~=1

if (ch==1)

xa='-';

break;

else

clc;

fprintf(' Baysian Network No.(%d%s\n', ch,')');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

display(' Ambulation Status Domain ');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' 1: F for Fast...');

display(' 2: S for Slow...');

display(' 3: St for Stationart...');

display(' 4: X for Non...');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

xa = input(' Enter ambulation status domain (xa) : ','s');

display(' ');

[fla] = check('a',xa);

if (fla == 1)

break;

else

fla=0;

msgbox('Invalid Value', 'Error','error');

end

end

end

fla=0;clc;

while (fla ~=1)

clc;

fprintf(' Bayesian Network No.(%d%s\n', ch,')');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

display(' Personal Information ');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' 1: + for Positive...');

display(' 2: - for Negative...');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

pd = input(' Does the person drink or not (pd): ','s');

display(' ');

[fla] = check('d',pd);

if (fla == 1)

break;

else

fla=0;

msgbox('Invalid Value', 'Error','error');

end

end

end

function [net] = bn1(xb,xh,xt,xa,pd)

%%

% the prob. of the drink or not. Domain {+, -}

p\_pd = [0.13, 0.87];

% the prob. of the breathing rate. Domain

p\_xb\_pb\_plus = [ 0.64 , 0.22 ,0.14];

p\_xb\_pb\_neg = [ 0.09 , 0.42 ,0.49];

% the Prob. of the plus heart rate. Domain

p\_hx\_pd\_plus = [0.54 ,0.31,0.15];

p\_hx\_pd\_neg = [0.12 ,0.42, 0.46];

% the Prob. of the skin temperature. Domain

p\_xt\_pd\_plus = [0.73 ,0.18, 0.09];

p\_xt\_pd\_neg = [0.03 ,0.76,0.21];

% the Prob. Xa: ambulation status. Domain {Fast, Slow, Stationary}

p\_xa = [0.21 ,0.22, 0.57];

%% Using BN#1....

% Pd prob...

if pd == 'p'

c = p\_pd(1);

else

c = p\_pd(2);

end

% prob. of Xb: breathing rate. Domain {H, M, L}

if (pd == 'p')&& (xb == 'h')

c = p\_xb\_pb\_plus(1);

end

if (pd == 'p')&&(xb == 'm')

c = p\_xb\_pb\_plus(2);

end

if (pd == 'p')&&(xb == 'l')

c = p\_xb\_pb\_plus(3);

end

if (pd == 'n')&& (xb == 'h')

c = p\_xb\_pb\_neg(1);

end

if (pd == 'n')&&(xb == 'm')

c = p\_xb\_pb\_neg(2);

end

if (pd == 'n')&&(xb == 'l')

c = p\_xb\_pb\_neg(3);

end

% prob. of Xh: heart rate. Domain {H, M, L}

if (pd == 'p')&&(xh == 'h')

c = p\_hx\_pd\_plus(1);

end

if (pd == 'p')&&(xh == 'm')

c =p\_hx\_pd\_plus(2);

end

if (pd == 'p')&&(xh == 'l')

c = p\_hx\_pd\_plus(3);

end

if (pd == 'n')&&(xh == 'h')

c = p\_hx\_pd\_neg(1);

end

if (pd == 'n')&&(xh == 'm')

c = p\_hx\_pd\_neg(2);

end

if (pd == 'n')&&(xh == 'l')

c = p\_hx\_pd\_neg(3);

end

% prob. of Xt: skin temperature. Domain {H, M, L}

if (pd == 'p')&&(xt == 'h')

c = p\_xt\_pd\_plus(1);

end

if (pd == 'p')&&(xt == 'm')

c =p\_xt\_pd\_plus(2);

end

if (pd == 'p')&&(xt == 'l')

c = p\_xt\_pd\_plus(3);

end

if (pd == 'n')&&(xt == 'h')

c = p\_xt\_pd\_neg(1);

end

if (pd == 'n')&&(xt == 'm')

c = p\_xt\_pd\_neg(2);

end

if (pd == 'n')&&(xt == 'l')

c = p\_xt\_pd\_neg(3);

end

% Prob. of Xa: ambulation status. Domain {Fast, Slow, Stationary}

if xa == 'x'

c=0;

else

if xa == 'f'

c = p\_xa(1);

end

if xa == 'w'

c =p\_xa(2);

end

if xa == 't'

c =p\_xa(3);

end

end

net = c;

end

function [O1,O2] = pdf2( I )

%% Pd: drink or not. Domain {+, -}

if strcmp(I,'+')

O1='p';O2='n';

elseif strcmp(I,'-')

O1='p';O2='p';

end

%% Xb: breathing rate. Domain {H, M, L}

% Xh: heart rate. Domain {H, M, L}

% Xt: skin temperature. Domain {H, M, L}

if strcmp(I,'H') || strcmp(I,'h')

O1='h';O2=' ';

elseif strcmp(I,'M') || strcmp(I,'m')

O1='m';O2=' ';

elseif strcmp(I,'L') || strcmp(I,'l')

O1='l';O2=' ';

elseif strcmp(I,'X')|| strcmp(I,'x')

O1='-';O2=' ';

end

%% Xa: ambulation status. Domain { Fast, Slow, Stationary}

if strcmp(I,'F') || strcmp(I,'f')||strcmp(I,'Fast') || strcmp(I,'fast')||strcmp(I,'FAST')

O1='f';O2=' ';

elseif strcmp(I,'s')|| strcmp(I,'S')||strcmp(I,'SLOW') || strcmp(I,'slow')|| strcmp(I,'Slow')

O1='w';O2=' ';

elseif strcmp(I,'St') || strcmp(I,'st')|| strcmp(I,'ST')

O1='t';O2=' ';

elseif strcmp(I,'n')|| strcmp(I,'N')

O1='-';O2=' ';

end

end

function [flag] = check(x,I)

%UNTITLED Summary of this function goes here

% Detailed explanation goes here

% check the Xb vaild values...

if (x =='b')||(x =='h')||(x =='t')

if strcmp(I,'H') || strcmp(I,'h')

flag=1;

elseif strcmp(I,'M') || strcmp(I,'m')

flag=1;

elseif strcmp(I,'L') || strcmp(I,'l')

flag=1;

elseif strcmp(I,'x')

flag=1;

else

flag=0;

end

end

% check the Pd vaild values...

if (x =='d')

if strcmp(I,'+')

flag=1;

elseif strcmp(I,'-')

flag=1;

else

flag=0;

end

end

% check the Xa vaild values...

if (x =='a')

if strcmp(I,'F') || strcmp(I,'f')||strcmp(I,'Fast') || strcmp(I,'fast')||strcmp(I,'FAST')

flag=1;

elseif strcmp(I,'s')|| strcmp(I,'S')||strcmp(I,'SLOW') || strcmp(I,'slow')|| strcmp(I,'Slow')

flag=1;

elseif strcmp(I,'St') || strcmp(I,'st')|| strcmp(I,'ST')

flag=1;

elseif strcmp(I,'X') || strcmp(I,'x')

flag=1;

else

flag=0;

end

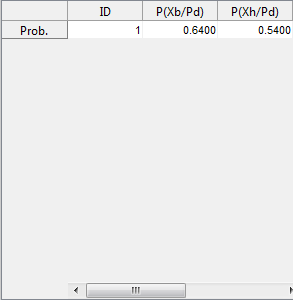
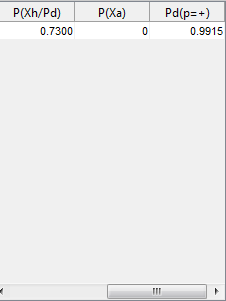
end

end

## **Query execution results**

The Bayesian Network 1 queries results are shown in the next tables

* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:



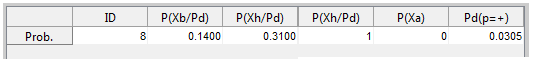
* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:

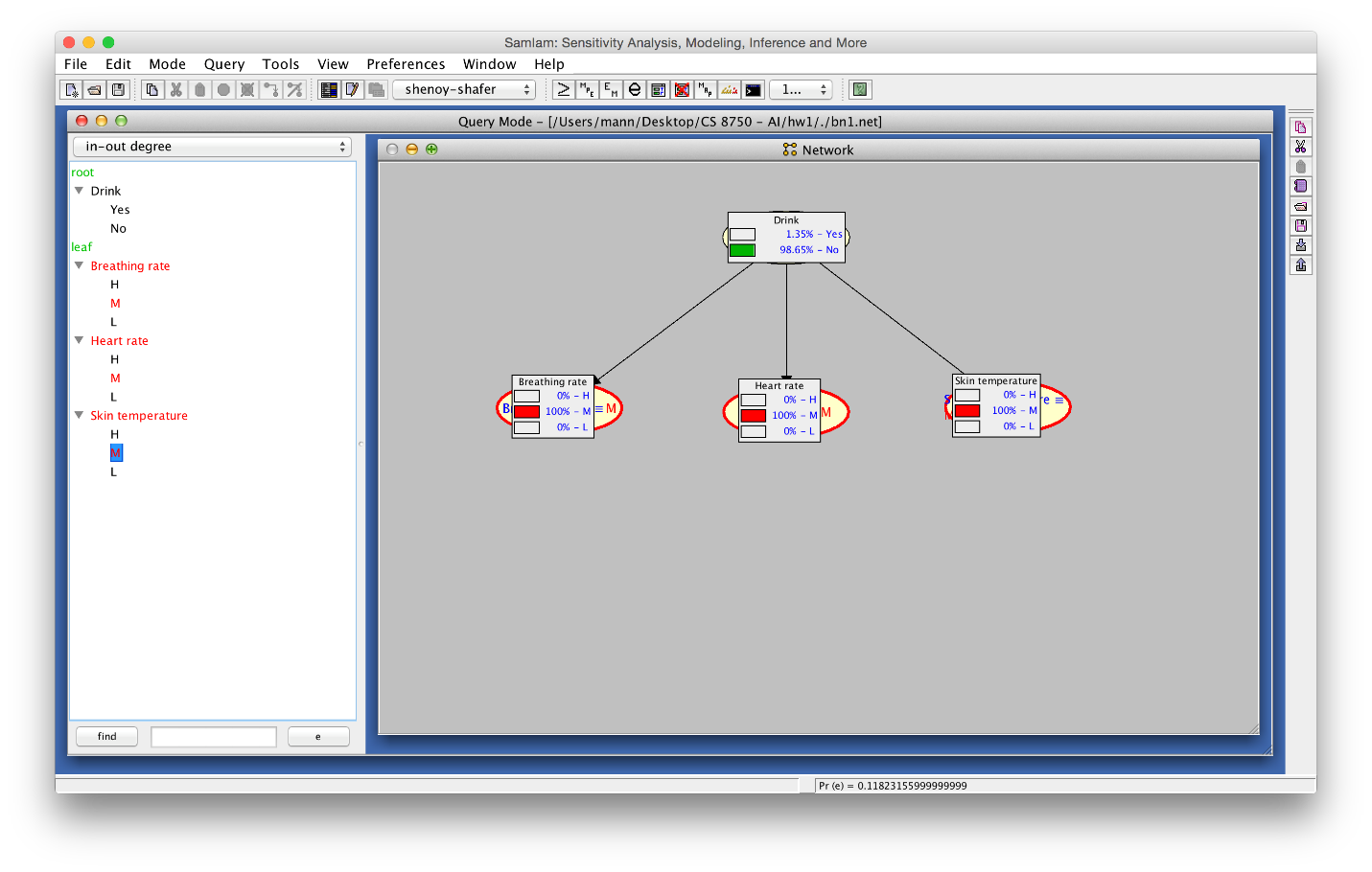


* 1. Query number 1:



## **SamIam implementation**

The naïve Bayesian network was implemented explicitly in SamIam. The results were verified against the Matlab program and matched. The graph is shown below. As an example, we’ve selected the evidence to answer query number 4:



The results of running the query using junction tree algorithm Shenoy-Shafer:

# **BN#2**

We build a second Bayesian network that considers Xa. Xa is posited as an alternative cause for Xb, Xh, and Xt. It is hoped that if Xa can explain away Xb, Xh and Xt better than Pd, Pd will be use as the explanation less often.

## **Formula derivations**

## **Queries 1-3**

To answer the first three queries with this network, we must calculate for Xa, Xb, Xh, X with known values. In the interest of legibility, we will denote as A, as , (Where is the desired value for the query) as , as , as , and as . Thus, in each of the three queries, we are seeking

## **Queries 4-5**

We use the same procedure and notation as to derive the previous expression, however, we add the following: to denote , to denote , and to denote . We therefore have:

We derive separately and and later plug them into the expression

Finally,

## **Queries 6-7**

We use the same procedure and notation as before.

## **Queries 8-9**

Once again we use the same notation

Let us expand separately

We note that

Therefore

## **Query 10**

Once more, the same notation is used.

We calculate separately

Therefore:

## **Pseudocode**

For the implementation of this network, we will use

**Step.1:** Setup the arrays values for BN#1 // using Bn1 function

p\_pd = [0.13, 0.87];

% the prob. of the breathing rate. Domain

p\_xb\_pb\_plus = [ 0.64 , 0.22 ,0.14];

p\_xb\_pb\_neg = [ 0.09 , 0.42 ,0.49];

% the Prob. of the plus heart rate. Domain

p\_hx\_pd\_plus = [0.54 ,0.31,0.15];

p\_hx\_pd\_neg = [0.12 ,0.42, 0.46];

% the Prob. of the skin temperature. Domain

p\_xt\_pd\_plus = [0.73 ,0.18, 0.09];

p\_xt\_pd\_neg = [0.03 ,0.76,0.21]

**Step.2:** Setup the arrays values for BN#2 // using Bn1 function

%P(Xh|Pd, Xa)

p\_xb\_pdxa\_plus=[0.95,0.03,0.02 ;0.77,0.19,0.04; 0.71,0.2,0.09];

p\_xb\_pdxa\_neg=[0.87,0.11,0.02 ;0.14,0.74,0.12; 0.03,0.16,0.81];

%P(Xh|Pd, Xa)

p\_xh\_pdxa\_plus=[0.97,0.02,0.01 ;0.76,0.2,0.04; 0.63,0.23,0.14];

p\_xh\_pdxa\_neg=[0.92,0.07,0.01 ;0.11,0.82,0.07; 0.07,0.08,0.85];

%P(Xt|Pd, Xa)

p\_xt\_pdxa\_plus=[0.91,0.06,0.03 ;0.54,0.36,0.1; 0.49,0.38,0.13];

p\_xt\_pdxa\_neg=[0.74,0.18,0.08 ;0.21,0.47,0.32; 0.11,0.62,0.27];

**Step 3:** **For** i=1 **to** 10 // find the query from 1 to 10

**Step 3.1:** Get the query features // five random variable values (Xb,Xh,Xt=’High or …,Pd=’+/-‘

**Step 3.2:** For each variable // (Xa,Xh,Xt,Pd) and pd=’+’

**Step 3.2.1:** Find the specific symbol for each variable // X\_Variable=’H’ or ’M’or ’L’

**Step 3.2.1.1:** Enter the random variable text features. // using feature function

**Step 3.2.1.2**: Check the input text if it valid for the feature or not.

**IF** input in ‘High ‘or ‘Medium ‘or ‘Low’ …then Flag=’TRUE’

E**lse** Flag=’False’

**Step 3.2.1.3:** **IF** the Flag=’True’ then go to **step.2.2**

**Else** go to **Step.3.2.1.1**

**Step 3.3:** Find the prob. Value for each variable using bn1 function but with **pd=’-‘**and **pd=’-‘**.

**Step 3.4:** **For** i=1 **to** 10 computes the queries

**Step 3.4.1:** **IF** pd=’+’ **then**

pb= pb\_plus(1,1) //P(pd=’+’))

**Else**  pb= pb\_plus(1,2) //P(pd=’-’))

**Step 3.4.2:** Find the prob. of given Pd=’+’

[p\_xb1],[p\_xh1], [p\_xt1], [pd\_plus],[p\_xa]

**Step 3.4.3:** P(variables/P(pd(+))^P(xa)

**Step 3.4.4:** Find the prob. of not given Pd=’-’

[p\_xb2],[p\_xh2], [p\_xt2], [pd\_neg]

**Step 3.4.5:** P(variables/P(~pd(+))^P(xa)

**Step 3.4.6:** Find the queries from 1 to 4

**Step.3.5:** Do the printing of the result.

**Step 3:** Next loop.

**Step 4:** End.

## **Matlab code**

What follows is the actual code put into matlab to run this network. The main program code is the same as for the first Bayesian network.

function [ ] = BayNet\_2(ch)

%% using Baysian Netwrok No.2 ...

% --------------------- Compute Queries No.1 to 3 ------------------------

for i=1:10

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ');

display(' Baysian Network No.2 ');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_');

display(' ')

fprintf('Compute Query No: %d\n', i);

display(' ');

% Find the query#1 ,#2 & #3 results...

[xb,xh,xt,xa,pd ] = features(ch);

% From Text to Prob. values...

[c1,~] = pdf2(xb)

[c2,~] = pdf2(xh)

[c3,~] = pdf2(xt)

[c4,~] = pdf2(xa)

[c5\_post,c5\_neg] = pdf2(pd)

[pd\_plus] = bn1(' ',' ',' ',' ',c5\_post)

[pd\_neg] = bn1(' ',' ',' ',' ',c5\_neg)

%% Find the query #4&5 results...

if (c4 =='-')&&(c3~='-')&&(c2~='-')

% Find the prob. of given Pd...

% Postive P(Pd(+))...

c4='f';

[p\_xa\_p1] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_p1] = bn2(c1,' ',' ',c4,c5\_post)

[p\_xh\_p1] = bn2(' ',c2,' ',c4,c5\_post)

[p\_xt\_p1] = bn2(' ',' ',c3,c4,c5\_post)

c4='w';

[p\_xa\_p2] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_p2] = bn2(c1,' ',' ',c4,c5\_post)

[p\_xh\_p2] = bn2(' ',c2,' ',c4,c5\_post)

[p\_xt\_p2] = bn2(' ',' ',c3,c4,c5\_post)

c4='t';

[p\_xa\_p3] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_p3] = bn2(c1,' ',' ',c4,c5\_post)

[p\_xh\_p3] = bn2(' ',c2,' ',c4,c5\_post)

[p\_xt\_p3] = bn2(' ',' ',c3,c4,c5\_post)

% Negative P(Pd(+))...

c4='f';

[p\_xa\_n1] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_n1] = bn2(c1,' ',' ',c4,c5\_neg)

[p\_xh\_n1] = bn2(' ',c2,' ',c4,c5\_neg)

[p\_xt\_n1] = bn2(' ',' ',c3,c4,c5\_neg)

c4='w';

[p\_xa\_n2] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_n2] = bn2(c1,' ',' ',c4,c5\_neg)

[p\_xh\_n2] = bn2(' ',c2,' ',c4,c5\_neg)

[p\_xt\_n2] = bn2(' ',' ',c3,c4,c5\_neg)

c4='t';

[p\_xa\_n3] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_n3] = bn2(c1,' ',' ',c4,c5\_neg)

[p\_xh\_n3] = bn2(' ',c2,' ',c4,c5\_neg)

[p\_xt\_n3] = bn2(' ',' ',c3,c4,c5\_neg)

% --------------------- P(xhxhxt/P(pd(+))^P(xa) -----------------------

p\_A = (pd\_plus\*p\_xb\_p1\*p\_xh\_p1\*p\_xt\_p1\*p\_xa\_p1) + (p\_xb\_p2\*p\_xh\_p2\*p\_xt\_p2\*p\_xa\_p2\*pd\_plus) + (p\_xb\_p3\*p\_xh\_p3\*p\_xt\_p3\*p\_xa\_p3\*pd\_plus)

% --------------------- P(xhxhxt/P(pd(+))^~P(xa) -----------------------

p\_B = p\_A + (pd\_neg\*p\_xb\_n1\*p\_xh\_n1\*p\_xt\_n1\*p\_xa\_n1) + (p\_xb\_n2\*p\_xh\_n2\*p\_xt\_n2\*p\_xa\_n2\*pd\_neg) + (p\_xb\_n3\*p\_xh\_n3\*p\_xt\_n3\*p\_xa\_n3\*pd\_neg)

result=p\_A/ p\_B

% for printing...

p\_xb1=p\_xb\_p1\*p\_xb\_p2\*p\_xb\_p3\*pd\_plus

p\_xh1=p\_xh\_p1\*p\_xh\_p2\*p\_xh\_p3\*pd\_plus

p\_xt1=p\_xt\_p1\*p\_xt\_p2\*p\_xt\_p3\*pd\_plus

p\_xa1=p\_xa\_p1\*p\_xa\_p2\*p\_xa\_p3\*pd\_plus

Print(i,c1,c2,c3,'-',result,p\_xb1,p\_xh1,p\_xt1,p\_xa1)

input('Press enter to continue...','s');

close all;clc;

%% Find the query #6&7 results...

elseif (c2 =='-')&&(c3~='-')&&(c4~='-')

% Find the prob. of given Pd...

[p\_xa] = bn1(' ',' ',' ',c4,' ')

[p\_xb1] = bn2(c1,' ',' ',c4,c5\_post)

[p\_xt1] = bn2(' ',' ',c3,c4,c5\_post)

p\_A=pd\_plus\*p\_xa\*p\_xb1\*p\_xt1

% Find the prob. of not given Pd...

[p\_xb2] = bn2(c1,' ',' ',c4,c5\_neg)

[p\_xt2] = bn2(' ',' ',c3,c4,c5\_neg)

p\_B=p\_A+(pd\_neg\*p\_xa\*p\_xb2\*p\_xt2)

% Final Result...

result=p\_A/ p\_B

% For printing...

p\_xb=p\_xb1\*pd\_plus

p\_xt=p\_xt1\*pd\_plus

Print(i,c1,'-',c3,c4,result,p\_xb,0,p\_xt,p\_xa)

input('Press enter to continue...','s');

close all;clc;

%% Find the query #8&9 results...

elseif (c2~='-')&&(c3 =='-')&&(c4=='-')

% Find the prob. of given Pd...

c4='f';

[p\_xa\_p1] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_p1] = bn2(c1,' ',' ',c4,c5\_post)

[p\_xh\_p1] = bn2(' ',c2,' ',c4,c5\_post)

c4='w';

[p\_xa\_p2] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_p2] = bn2(c1,' ',' ',c4,c5\_post)

[p\_xh\_p2] = bn2(' ',c2,' ',c4,c5\_post)

c4='t';

[p\_xa\_p3] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_p3] = bn2(c1,' ',' ',c4,c5\_post)

[p\_xh\_p3] = bn2(' ',c2,' ',c4,c5\_post)

% Negative P(Pd(+))...

c4='f';

[p\_xa\_n1] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_n1] = bn2(c1,' ',' ',c4,c5\_neg)

[p\_xh\_n1] = bn2(' ',c2,' ',c4,c5\_neg)

c4='w';

[p\_xa\_n2] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_n2] = bn2(c1,' ',' ',c4,c5\_neg)

[p\_xh\_n2] = bn2(' ',c2,' ',c4,c5\_neg)

c4='t';

[p\_xa\_n3] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_n3] = bn2(c1,' ',' ',c4,c5\_neg)

[p\_xh\_n3] = bn2(' ',c2,' ',c4,c5\_neg)

% --------------------- P(xhxhxt/P(pd(+))^P(xa) -----------------------

p\_A = (pd\_plus\*p\_xb\_p1\*p\_xh\_p1\*p\_xa\_p1) + (p\_xb\_p2\*p\_xh\_p2\*p\_xa\_p2\*pd\_plus) + (p\_xb\_p3\*p\_xh\_p3\*p\_xa\_p3\*pd\_plus)

% --------------------- P(xhxhxt/P(pd(+))^P(xa) -----------------------

p\_B = p\_A + (pd\_neg\*p\_xb\_n1\*p\_xh\_n1\*p\_xa\_n1) + (p\_xb\_n2\*p\_xh\_n2\*p\_xa\_n2\*pd\_neg) + (p\_xb\_n3\*p\_xh\_n3\*p\_xa\_n3\*pd\_neg)

result = p\_A / p\_B

% For printing...

p\_xb1=p\_xb\_p1\*p\_xb\_p2\*p\_xb\_p3\*pd\_plus

p\_xh1=p\_xh\_p1\*p\_xh\_p2\*p\_xh\_p3\*pd\_plus

p\_xa1=p\_xa\_p1\*p\_xa\_p2\*p\_xa\_p3\*pd\_plus

Print(i,c1,c2,'-','-',result,p\_xb1,p\_xh1,0,p\_xa1)

%% Find the query #10 result...

elseif (c2=='-')&&(c3 =='-')&&(c4=='-')&&(c1~='-')

c4='f';

[p\_xa\_p1] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_p1] = bn2(c1,' ',' ',c4,c5\_post)

c4='w';

[p\_xa\_p2] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_p2] = bn2(c1,' ',' ',c4,c5\_post)

c4='t';

[p\_xa\_p3] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_p3] = bn2(c1,' ',' ',c4,c5\_post)

% Negative P(Pd(+))...

c4='f';

[p\_xa\_n1] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_n1] = bn2(c1,' ',' ',c4,c5\_neg)

c4='w';

[p\_xa\_n2] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_n2] = bn2(c1,' ',' ',c4,c5\_neg)

c4='t';

[p\_xa\_n3] = bn1(' ',' ',' ',c4,' ')

[p\_xb\_n3] = bn2(c1,' ',' ',c4,c5\_neg)

% --------------------- P(xhxhxt/P(pd(+))^P(xa) -----------------------

p\_A=(pd\_plus\*p\_xb\_p1\*p\_xa\_p1)+(p\_xb\_p2\*p\_xa\_p2\*pd\_plus)+(p\_xb\_p3\*p\_xa\_p3\*pd\_plus)

% --------------------- P(xhxhxt/P(pd(+))^P(xa) -----------------------

p\_B=p\_A+(pd\_neg\*p\_xb\_n1\*p\_xa\_n1)+(p\_xb\_n2\*p\_xa\_n2\*pd\_neg)+(p\_xb\_n3\*p\_xa\_n3\*pd\_neg)

result=p\_A/ p\_B

% For printing...

p\_xb1=p\_xb\_p1\*p\_xb\_p2\*p\_xb\_p3\*pd\_plus

p\_xa1=p\_xa\_p1\*p\_xa\_p2\*p\_xa\_p3\*pd\_plus

Print(i,c1,'-','-','-',result,p\_xb1,0,0,0)

else

% Find the prob. of given Pd...

[p\_xb1] = bn2(c1,' ',' ',c4,c5\_post)

[p\_xh1] = bn2(' ',c2,' ',c4,c5\_post)

[p\_xt1] = bn2(' ',' ',c3,c4,c5\_post)

[pd\_plus] = bn1(' ',' ',' ',' ',c5\_post)

[p\_xa] = bn1(' ',' ',' ',c4,' ')

% --------------------- P(xhxhxt/P(pd(+))^P(xa) -----------------------

p\_A=p\_xb1\*p\_xh1\*p\_xt1\*pd\_plus

% Find the prob. of not given Pd...

[p\_xb2] = bn2(c1,' ',' ',c4,c5\_neg)

[p\_xh2] = bn2(' ',c2,' ',c4,c5\_neg)

[p\_xt2] = bn2(' ',' ',c3,c4,c5\_neg)

[pd\_neg] = bn1(' ',' ',' ',' ',c5\_neg)

% --------------------- P(xhxhxt/P(~pd(+))^P(xa) -----------------------

p\_B=p\_A+(p\_xb2\*p\_xh2\*p\_xt2\*pd\_neg)

result=p\_A/ p\_B

Print(i,c1,c2,c3,c4,result,p\_xb1,p\_xh1,p\_xt1,p\_xa)

input('Press enter to continue...','s');

close all;clc;

end

end

## **Query execution results**

* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:



* 1. Query number 1:

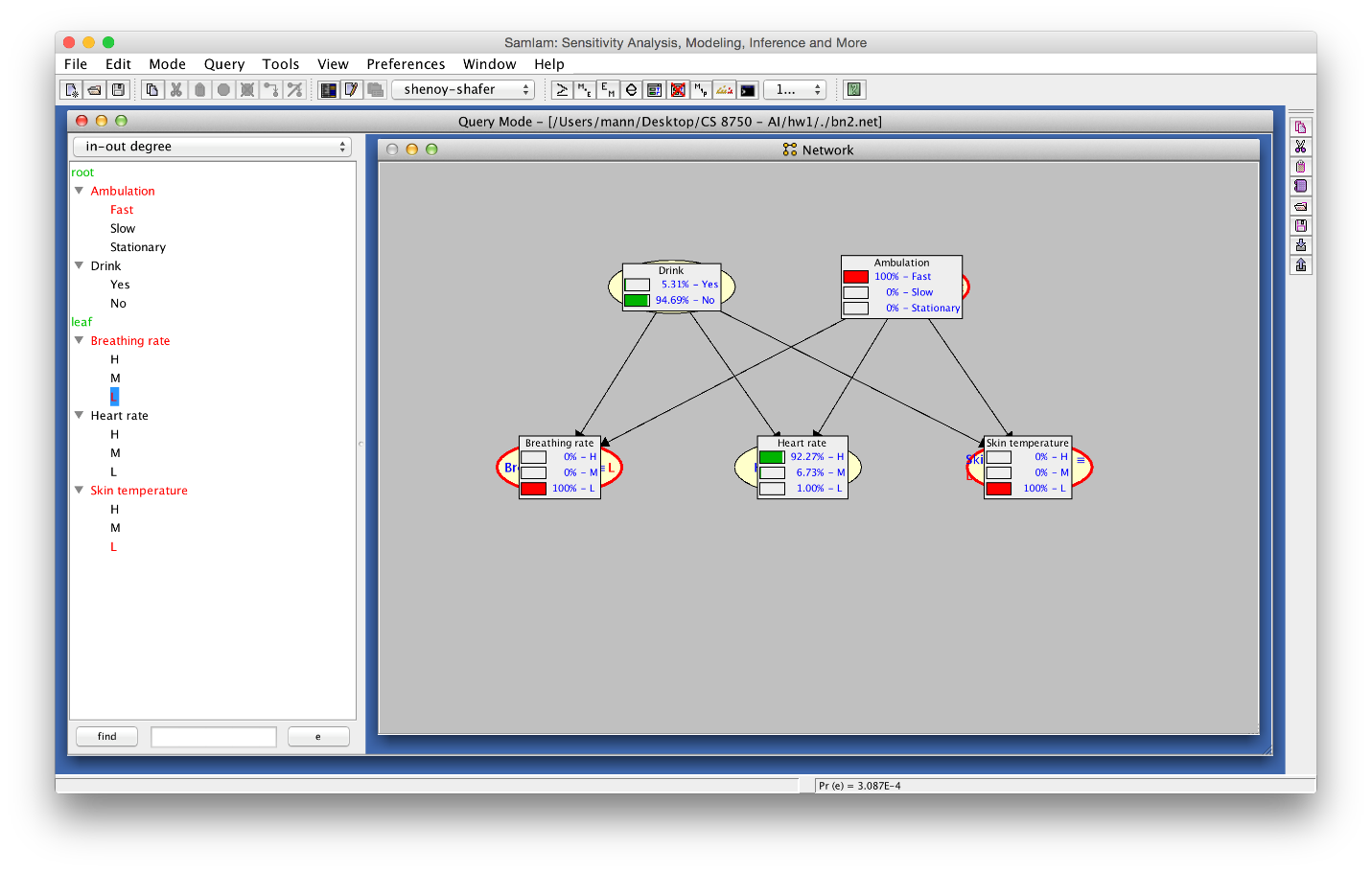


* 1. Query number 1:



## **SamIam implementation**

The complex Bayesian network was implemented explicitly in SamIam. The results were verified against the Matlab program and matched. The graph is shown below. As an example, we’ve selected the evidence to The Bayesian Network 2 graph constructed using SamIam to answer query number 7 :



The results of running the query using junction tree algorithm Shenoy-Shafer:

# **Conclusions**

We implemented successfully two Bayesian networks: one naïve and one complex. Using the equations derived in sections 2.2 and 3.2, these implementations cover all possible cases for missing values, as tested by the 10 queries set out in the introduction of this assignment. We also modeled the Bayesian netoworks in SamIam and were able to get results that matched the output of our implementations.

When comparing the same queries in both networks, we observe that Pr decreases in the second network. This is expected to be because of the introduction of an alternate cause to explain away Xb Xh and Xt. The new probabilities are more in line with the prior probabilities of Pd, suggesting a more accurate model.