Representation des connaissances et raisonnement 2

« Rapport du TP °4 : Réseaux causaux bayésiens»

CHIKH Khadidja

Master 2 SII
Groupe 1

09/12/2019

Introduction

Dans ce TP, il est demandé de modéliser les réseaux bayésiens en exploitant la toolbox de Matlab « Bayes Net Toolbox », ceci en abordant les cas suivants :

- ✔ Un polyarbre
- ✔ Un graphe à connexions multiples
- ✔ Un graphe à connexions multiples de taille importante

Un réseau bayésien :

C'est un modèle graphique probabiliste représentanun ensemble de variables aléatoires sous la forme d'un graphe orienté acyclique.

Bayes Net Toolbox :

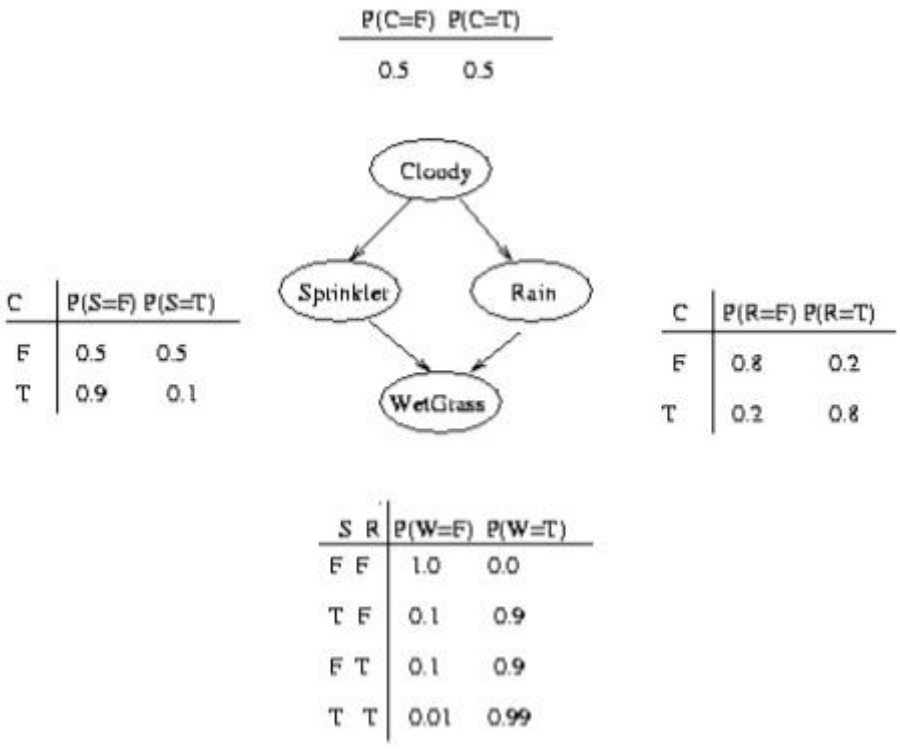
Bayes Net Toolbox (BNT) est un package Matlab *open-source* destiné aux modèles graphiques dirigés. Le BNT prend en charge de nombreux types de nœuds (distributions de probabilité), l'inférence exacte et approximative, l'apprentissage de paramètres et de structures, ainsi que les modèles statiques et dynamiques.**[1]**

1- Génération d'un polyarbre :

Pour ce cas, nous avons pris l'exemple du document explicatif donné.

1-1-Le polyarbre :

Avec T=Vrai (ex : Rain=vrai) et F=Faux(ex : Rain=faux), et les distibutions conditionnelles associées au nœud racine ainsi que les autres nœuds.



L'implémentation de cette partie est comme suit :

-Précision du nombre de nœud

-Affecter des numéros désignant les nœuds (1 pour Cloudy 2 pour Sprinkler etc)

-Représenter les arc par une matrice, en mettant 1 dans la case lorsqu'il s'agit d'une paire de nœuds connectés et 0 dans le cas contraire.

-Spécifier le nombre de valeurs(nodesizes) de chaque variable (nœud) : ici 2 ie 1 ou 2 pour chacune.

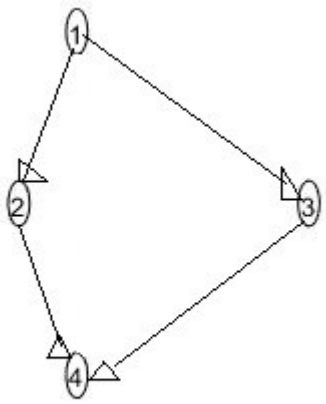
[1] https://www.researchgate.net/publication/2413249_The_Bayes_Net_Toolbox_for_Matlab

-Construire le réseaux à l'aide de la fonction prédéfinie *mk_bnet* qui prend comme paramètres les données précédentes.

-Attribuer les distributions conditionnelles correspondantes à chaque nœud à l'aide de la fonction prédéfinie tabular_CPD qui prend comme paramètres : le réseau, le nœud, et le tableau des distributions propre à ce nœud.

-Construire le moteur d'inférence de notre réseaux, dans notre cas c'est la fonction prédéfinie *pearl_inf_engine* qui prend comme paramètre le réseau.

Affichage du réseau :

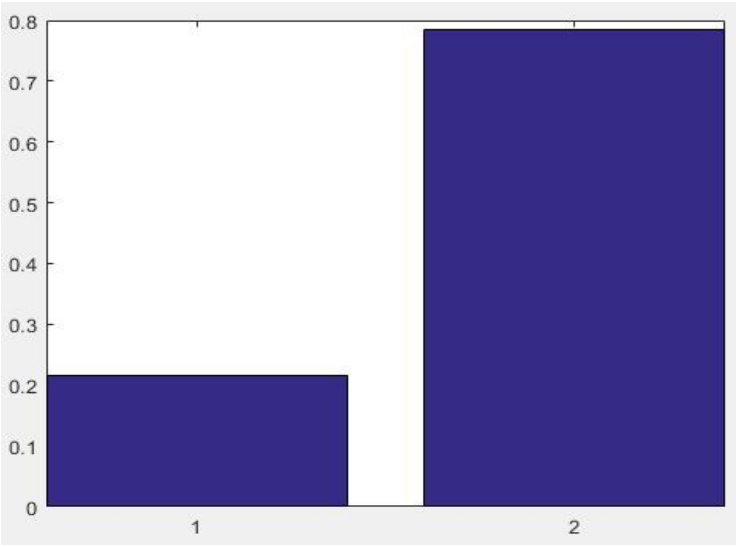


1: Cloudy **2 :** Sprinker **3 :** Rain **4 :** WetGrass

<u>1,2 Calcul de p(variable d'interêt | evidence(s)) :</u>

Prenons par exemple la variable d'interêt: **Rain**, avec l'evidence : Wet=2

Après l'injection des évidences à notre moteur d'inférence, et à l'aide de la fonction *marginal_nodes*, nous obtenons les résultats suivants:




```
>> marg.T

ans =

    0.2163
    0.7837
```

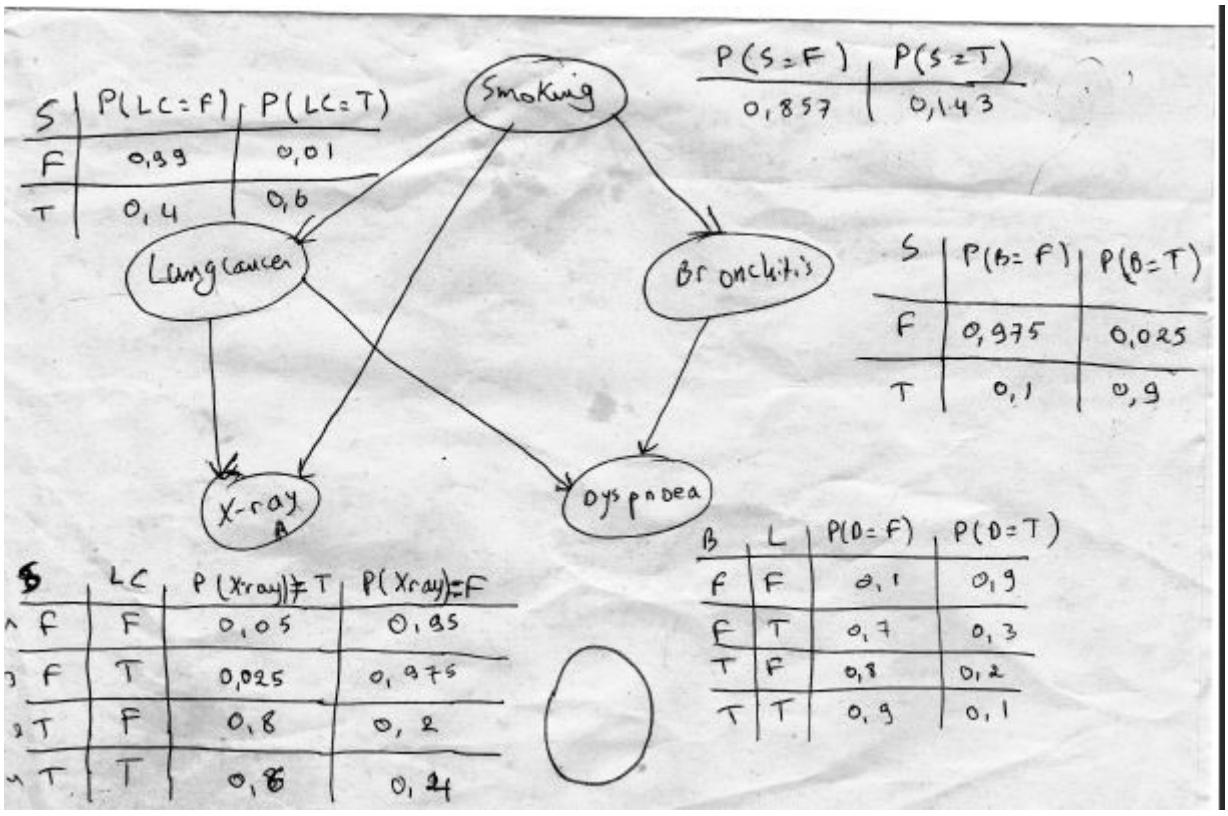
P(Rain=1|Wet=2)=0.2163 ; P(Rain=2|Wet=2)=0.7837

Code source de l'implémentation :

```
n=4;
dag=zeros(n,n);
c=1;s=2;r=3;w=4;
dag(c,[r s])=1;
dag(r,w)=1;
dag(s,w)=1;
discnodes=1:n;
nodesizes=2*ones(1:n);
onodes=[];
bnet=mk_bnet(dag,nodesizes,'discrete',discnodes,'observed',onodes);
bnet.CPD{w}=tabular_CPD(bnet,w,[1 0.1 0.1 0.01 0 0.9 0.9 0.99]);
bnet.CPD{r}=tabular_CPD(bnet,r,[0.8 0.2 0.2 0.8]);
bnet.CPD{s}=tabular_CPD(bnet,s,[0.5 0.9 0.5 0.1]);
bnet.CPD{c}=tabular_CPD(bnet,c,[0.5 0.5]);
engine=pearl_inf_engine(bnet);
draw_graph(bnet.dag);
evidence=cell(1,n);
evidence{w}=2;
[engine,liglik]=enter_evidence(engine,evidence);
marg=marginal_nodes(engine,r);
marg.T
bar(marg.T)
```

2- Génération d'un graphe à connexions multiples :

Nous avons pris l'exemple suivant :

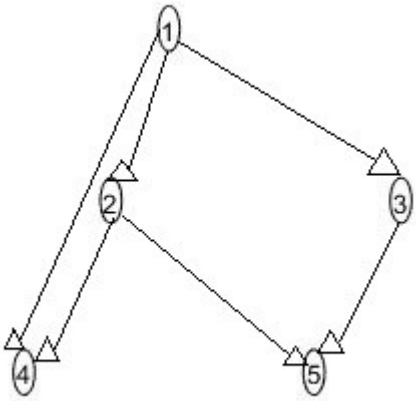


Tel que: **X-ray** signifie l'apparence d'une anomalie sur la radiographie thoracique.

2-2- Le graphe à connexions multiples:

Construit et généré de la meme manière que dans le cas précédent, sauf pour la construction du moteur d'inférence, nous utilisons la fonction prédéfinie *jtree_inf_engine* qui est dédiée à ce type de réseaux.

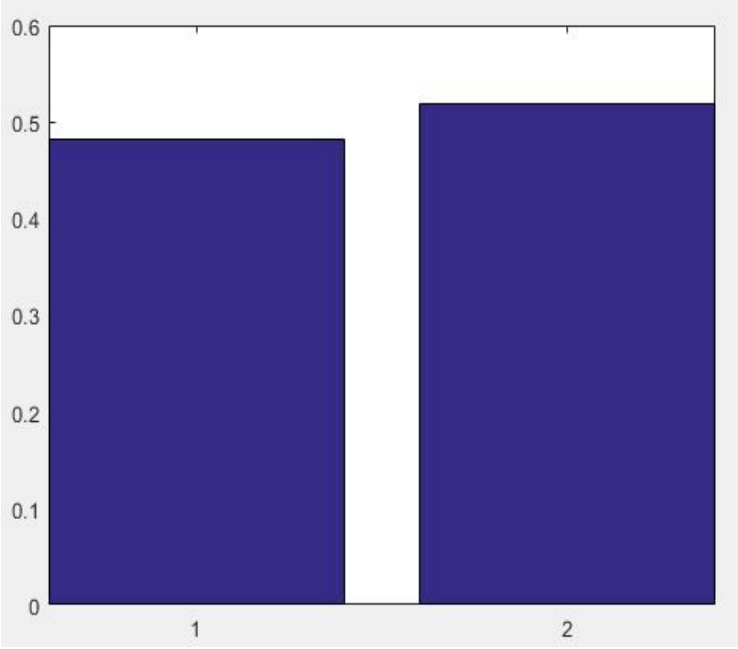
Affichage du réseau :



1: Smoking **2 :** Lung cancer **3 :** Bronchitis **4 :** X-ray **5 :** Dyspnoea

2,2 Calcul de p(variable d'interêt | evidence(s)) :

Prenons par exemple la variable d'interêt: **Smoking**, avec les evidences : X-ray=2 et Lung cancer=1, de la même manière que dans le cas précédent, nous obtenons les résultats suivants :




```
>> marg.T

ans =

    0.4811
    0.5189
```

P(Smoking=1 | Lung cancer=1,X-ray=2)=0.4811 ;
P(Smoking=2 | Lung cancer=1,X-ray=2)=0.5189

Code source de l'implémentation :

```
n=5;
dag=zeros(n,n);
s=1;lc=2;b=3;x=4;d=5;
dag(s,[lc b x])=1;
dag(lc,[x d])=1; dag(b,d)=1;
discnodes=1:n;
nodesizes=2*ones(1:n);
onodes=[];
bnet=mk_bnet(dag,nodesizes,'discrete',discnodes,'observed',onodes);
bnet.CPD{d}=tabular_CPD(bnet,d,[0.1 0.7 0.8 0.9 0.9 0.3 0.2 0.1]);
bnet.CPD{b}=tabular_CPD(bnet,b,[0.975 0.1 0.025 0.9]);
bnet.CPD{s}=tabular_CPD(bnet,s,[0.857 0.143]);
bnet.CPD{lc}=tabular_CPD(bnet,lc,[0.99 0.4 0.01 0.6]);
bnet.CPD{x}=tabular_CPD(bnet,x,[0.95 0.2 0.975 0.4 0.05 0.8 0.025 0.6]);
engine=jtree_inf_engine(bnet);
draw_graph(bnet.dag)
evidence=cell(1,n);
evidence{x}=2;
evidence{lc}=1;
[engine,liglik]=enter_evidence(engine,evidence);
marg=marginal_nodes(engine,s);
marg.T
bar(marg.T)
```

3- Génération d'un graphe à connexions multiples de taille grande:

Pour les observations, nous avons pris au delà de 20 (100 nœuds), et le résultat obtenu est le suivant :

```
Error using ones
Maximum variable size allowed by the program is exceeded.
```

le maximum de nombre (de nœuds) supporté est dépassé ce qui implique l'impossibilité de la construction de l'arbre associé.
Pour moins de 20 (12 par exemple) le programme beugue.

Code source :

```
n=12;
dag=zeros(n,n);
for noeud=1:n
  max=randi([1 5]);
  for j=1:max
    if i~=j
      parent=randperm(max,1);
      dag(j,parent)=1;
    end
  end
end
discnodes=1:n;
nodesizes=2*ones(1:n);
onodes=[];
cpt=0;
while graphisdag(sparse(dag))==0
  i=randi(n);
  j=randi(n);
  dag(i,j)=1-dag(i,j);
  for k=1:n
    for l=1:n
      if dag(k,l)==0
         cpt=cpt+1;
      end
    end
    if cpt==n
      m=randi([1 n]);
      while(m==k)
        m=randi([1 n]);
      end
      dag(k,m)=1;
    end
  end
end
bnet=mk_bnet(dag,nodesizes,'discrete',discnodes,'observed',onodes);
```

```
for noeud=1:n
 cpt=0;
 for parent=1:n
   if dag(noeud,parent)==1
     cpt=cpt+1;
   end
 end
 tabproba=rand(2^(cpt+1),1);
 bnet.CPD{noeud}=tabular_CPD(bnet,noeud,tabproba');
end
engine=jtree_inf_engine(bnet);
draw_graph(bnet.dag);
```

Conclusion :

La *Bayesian Network s Toolbox* nous a permit, à travers ce TP, d'implémenter des moteurs d'inférences basés sur des réseaux bayesiens de différents types et sous différentes conditions d'une manière facile et rapide, et de principalement voir que la complexité de ces réseaux est relative à leur developpement.