PRISM and implementation of Bayesian Networks

Henning Christiansen
KIIS course
Program for today

• Introduction to PRISM
• How to do Bayesian networks in PRISM – learn and apply probs.
• Exercise: Implement and play with your BN in PRISM
• Hidden Markov Models and their implementation in PRISM
• Exercise: ....

NB: Only very little theory :(
Conditional prop’s ≈ logical rules

• $P(\text{effect}|\text{cause}) \approx \text{effect} :- \text{cause}$.
• easier to measure than $P(\text{cause}|\text{effect})$
• Bayesian network: Graph (DAG) of cause-effect relationships
  ≈ a logic program
  – with limited structure and no arguments
  – but with probabilities
• Here: Discrete BNs
  – examples even binary = boolean
  – but any finite no. of possible outcomes of each random variables
Example (Charniak)

```
abducibles fo/0, bp/0.
lo:- fo.
do:- fo.
do:- bp.
hb:- do.

?- hb.
...
?- hb, lo.
```

(This and another figure copied from slides by J.-C. Latombe; found at http://www.cs.ualberta.ca/~lindek/366/)
Purpose of BN

• Probabilistically based, *abductive reasoning*, i.e., reasoning from “observed effect” to “(hidden) causes” with probabilities

• Based on conditional probabilities and Bayes’ theorem ≈ a way of “reasoning backwards” in conditional probs.
Assumption of independence

\[ P(a|b,c) = P(a|b,c,d,e) \]

Intuitively:
a depends on actual values of \( b \) and \( c \), but not on why \( b \) and \( c \)

You may try to read def. of “d-connected”, but you’re not expected to be able to reproduce it ;-)
Adding conditional probabilities

**Notice:**

- $P(lo|fo)$ stands for $P(lo=true|fo=true)$
- $P(lo|fo)=0.6$ indicates implicitly $P(not\ lo|fo)=P(lo=false|fo=true)=0.6$
A little exercise

Given $fo=true$ and $bp=false$, calculate probability for $P(hb=true)$
Another little exercise

Given $P(hb=true)$, calculate probabilities for $fo$, $bp$

A trivial but cumbersome manipulation using Bayes’ formula. You are welcome to try, but let us wait a bit and use the computer.
You exercise:

Exercise 4.1 in the note for today
   – design a Bayesian network for the familiar power supply example
Exercise 4.2 (discussion; if time)
   – on “intelligent” but annoying systems
PRISM: Logical-Statistical models and parameter learning

• Developed by T. Sato and colleagues from around 1995 and onwards
• Combines Prolog with random variables and machine learning.
• Subsumes discrete Bayesian networks, HMMs, SCFG, etc.
• Has a declarative semantics: a probabilistic version of least Herbrand model semantics,
  – i.e., given probabilities of all random variables in program, we have a probability for each atom.
• High-level tool with exact semantics; approx. evaluation strategies does not fit in
PRISM’s multi-valued switch

values(coin, [head,tail]).
set_sw(coin,[0.501, 0.499]).

throw(X):- msw(coin,X).

?- throw(X).

or

?- sample(throw(X)).
PRISM can learn probabilities from observations (smart, eh?)

values(coin, [head,tail]). % no set_sw!!

throw(X):- msw(coin,X).

target(throw,1).
data(fileWithThrows).

?- learn.
PRISM has parameterized msw’s

values(rainfall(_), [yes,no]).
values(sky,[cloudy,clear]).
....
msw(rainfall(cloudy), X), ...
....
msw(sky,S), msw(rainfall(S), X), ....

This way we can simulate conditional probabilities – so we have what is needed for Bayesian networks.
Example: Charniak’s network

values(fo,[foTrue, foFalse]).
values(bp,[bpTrue, bpFalse]).
values(lo(_),[loTrue, loFalse]).
  % lo({foTrue, foFalse})
values(do(_,_),[doTrue, doFalse]).
  % do({foTrue, foFalse}, {bpTrue, bpFalse})
values(hb(_),[hbTrue, hbFalse]).
  % hb({doTrue, doFalse})
The network

world(LO, HB):-
    world(_, _, LO, _, HB).

world(FO, BP, LO, DO, HB):-
    msw(fo, FO),
    msw(bp, BP),
    msw(lo(FO), LO),
    msw(do(FO,BP), DO),
    msw(hb(DO), HB).

?- world(LO,HB).
LO = loFalse
HB = hbFalse
Application 1: Probabilities given

set_params:-
    set_sw(fo, [0.15, 0.85]),
    set_sw(bp, [0.01, 0.99]),

    set_sw(lo(foTrue), [0.6, 0.4]),
    set_sw(lo(foFalse), [0.05, 0.95]),

    set_sw(do(foTrue,bpTrue), [0.99, 0.01]),
    set_sw(do(foTrue,bpFalse), [0.9, 0.1]),
    set_sw(do(foFalse,bpTrue), [0.97, 0.03]),
    set_sw(do(foFalse,bpFalse), [0.3, 0.7]),

    set_sw(hb(doTrue), [0.7, 0.3]),
    set_sw(hb(doFalse), [0.01, 0.99]).

______________________________

?- set_params.
Application 2: Learning probabilities from observations

target(world,5).
data(famOutData).

?- learn.

% file famOutData.dat

.....

?- show_sw.

world(foFalse,bpFalse,loFalse,doFalse,hbFalse).
world(foFalse,bpFalse,loFalse,doFalse,hbFalse).
world(foFalse,bpFalse,loTrue,doFalse,hbTrue).

...

world(foFalse,bpFalse,loFalse,doFalse,hbFalse).
world(foFalse,bpFalse,loFalse,doFalse,hbFalse).
world(foFalse,bpFalse,loFalse,doFalse,hbFalse).
Evaluating hidden prob’s from obs.

?- chindsight( world(loTrue, hbFalse), world(_,_,_,_,_)).

conditional hindsight probabilities:
  world(foFalse,bpFalse,loTrue,doFalse,hbFalse): 0.440219169184873
  world(foFalse,bpFalse,loTrue,doTrue,hbFalse): 0.057171320673360
  world(foFalse,bpTrue,loTrue,doFalse,hbFalse): 0.000190571068911
  world(foFalse,bpTrue,loTrue,doTrue,hbFalse): 0.001867211483271
  world(foTrue,bpFalse,loTrue,doFalse,hbFalse): 0.133175546980298
  world(foTrue,bpFalse,loTrue,doTrue,hbFalse): 0.363206037218994
  world(foTrue,bpTrue,loTrue,doFalse,hbFalse): 0.000134520754526
  world(foTrue,bpTrue,loTrue,doTrue,hbFalse): 0.004035622635767

... or in a more useful way:
?- chindsight_agg( world(loTrue, hbFalse),
                   world(query,_,_,_,_,_)).

conditional hindsight probabilities:
  world(foFalse,*,*,*,*): 0.499448272410416
  world(foTrue,*,*,*,*): 0.500551727589584
Your exercise

Exercise 4.3:
Implement in PRISM the Bayesian network that you design in exercise 4.1 for the power supply network.
Test it.