Imprecise Probabilities in AI

Bayesian networks

Cassio de Campos (+ F.G.Cozman + A.Antonucci)

c.decampos@tue.nl

TU Eindhoven, The Netherlands

SIPTA Summer School, Bristol, 18 August 2022

Initial setup

- Four binary variables:
  - $X_1 \in \{true, false\}$: High temperature
  - $X_2 \in \{true, false\}$: Goalkeeper's fitness
  - $X_3 \in \{true, false\}$: Attackers' fitness
  - $X_4 \in \{true, false\}$: Win the match

Initial setup

- Four binary variables:
  - $X_1 \in \{true, false\}$: High temperature
  - $X_2 \in \{true, false\}$: Goalkeeper's fitness
  - $X_3 \in \{true, false\}$: Attackers' fitness
  - $X_4 \in \{true, false\}$: Win the match
- The (joint) mass function $P(\mathbf{X}) = P(X_1, X_2, X_3, X_4)$ can be used to represent our knowledge about these variables.

Initial setup

- Four binary variables:
  - $X_1 \in \{true, false\}$: High temperature
  - $X_2 \in \{true, false\}$: Goalkeeper's fitness
  - $X_3 \in \{true, false\}$: Attackers' fitness
  - $X_4 \in \{true, false\}$: Win the match
- The (joint) mass function $P(\mathbf{X}) = P(X_1, X_2, X_3, X_4)$ can be used to represent our knowledge about these variables.
- We can build this function from data, constraints, experts, etc.
  - This is usually called *learning* in the context of prob graphical models (PGMs).

Initial setup

- Four binary variables:
  - $X_1 \in \{true, false\}$: High temperature
  - $X_2 \in \{true, false\}$: Goalkeeper's fitness
  - $X_3 \in \{true, false\}$: Attackers' fitness
  - $X_4 \in \{true, false\}$: Win the match
- The (joint) mass function $P(\mathbf{X}) = P(X_1, X_2, X_3, X_4)$ can be used to represent our knowledge about these variables.
- We can build this function from data, constraints, experts, etc.
  - This is usually called *learning* in the context of prob graphical models (PGMs).
- We can use $P(X_1, X_2, X_3, X_4)$ (somehow encoded into bits) to compute queries such as $P(X_1 = true | X_4 = true)$ or $\arg\max_{X_4} P(X_4 | X_1 = false)$.
  - This is usually called *reasoning/inferences* in the context of PGMs.

Reasoning

Using the information represented in $P(\mathbf{X})$:

Reasoning

Using the information represented in $P(\mathbf{X})$:

- Belief updating (BU): compute the probability of some event given some evidence (observed variables). E.g. $P(X_1 = true | X_4 = true)$.
  - Often used for learning, classification, and many more.

Reasoning

Using the information represented in $P(\mathbf{X})$:

- Belief updating (BU): compute the probability of some event given some evidence (observed variables). E.g. $P(X_1 = true | X_4 = true)$.
  - Often used for learning, classification, and many more.
- Most Probable Explanation (MPE): compute the most probable configuration of some variables given some evidence

Reasoning

Using the information represented in $P(\mathbf{X})$:

- Belief updating (BU): compute the probability of some event given some evidence (observed variables). E.g. $P(X_1 = true | X_4 = true)$.
  - Often used for learning, classification, and many more.
- Most Probable Explanation (MPE): compute the most probable configuration of some variables given some evidence
  - full MPE: all variables are either queried or observed. E.g. $\arg\max_{X_2, X_3} P(X_2, X_3 | X_1 = false, X_4 = true)$.
    - Often used for (multi-label) classification without missing data

Reasoning

Using the information represented in $P(\mathbf{X})$:

- Belief updating (BU): compute the probability of some event given some evidence (observed variables). E.g. $P(X_1 = true | X_4 = true)$.
  - Often used for learning, classification, and many more.
- Most Probable Explanation (MPE): compute the most probable configuration of some variables given some evidence
  - full MPE: all variables are either queried or observed. E.g. $\arg\max_{X_2, X_3} P(X_2, X_3 | X_1 = false, X_4 = true)$.
    - Often used for (multi-label) classification without missing data
  - partial MPE (a.k.a. maximum a posterior query): there might be variables that are not queried nor observed. E.g. $\arg\max_{X_4} P(X_4 | X_1 = false)$.
    - Often used for (multi-label) classification with missing data

Reasoning

Using the information represented in $P(\mathbf{X})$:

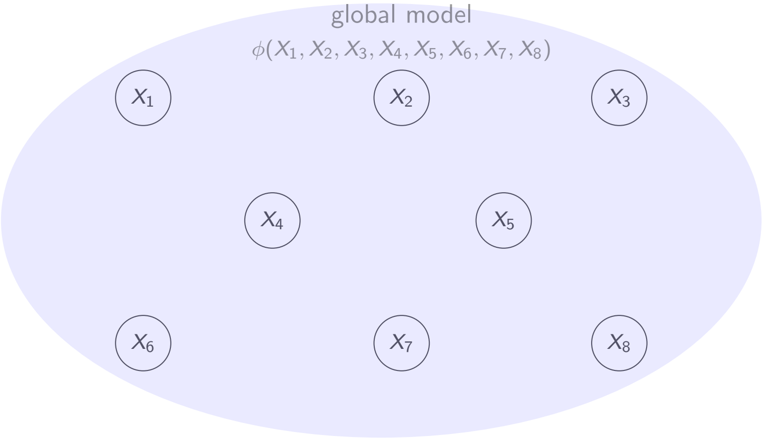
- Belief updating (BU): compute the probability of some event given some evidence (observed variables). E.g. $P(X_1 = true | X_4 = true)$.
  - Often used for learning, classification, and many more.
- Most Probable Explanation (MPE): compute the most probable configuration of some variables given some evidence
  - full MPE: all variables are either queried or observed. E.g. $\arg\max_{X_2,X_3} P(X_2, X_3 | X_1 = false, X_4 = true)$.
    - Often used for (multi-label) classification without missing data
  - partial MPE (a.k.a. maximum a posterior query): there might be variables that are not queried nor observed. E.g. $\arg\max_{X_4} P(X_4 | X_1 = false)$.
    - Often used for (multi-label) classification with missing data

- **Q:** Aren't these all "trivial" tasks? **A:** If you have a (potentially huge) table representing $P(\mathbf{X})$, then yes: just go over the table.

Probabilistic Graphical Models

aka Decomposable Multivariate Probabilistic Models

(whose decomposability is induced by independence)

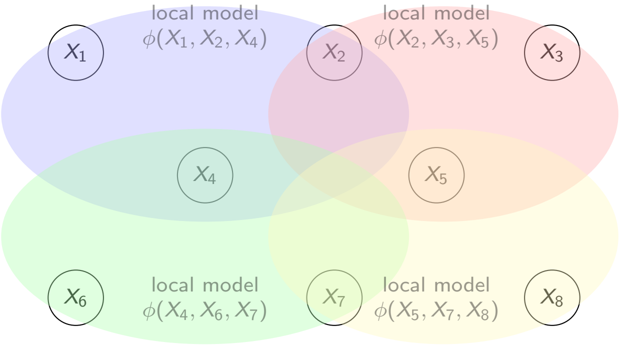


Probabilistic Graphical Models

aka Decomposable Multivariate Probabilistic Models

(whose decomposability is induced by independence)

$$\phi(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8) = \phi(X_1, X_2, X_4) \otimes \phi(X_2, X_3, X_5) \otimes \phi(X_4, X_6, X_7) \otimes \phi(X_5, X_7, X_8)$$



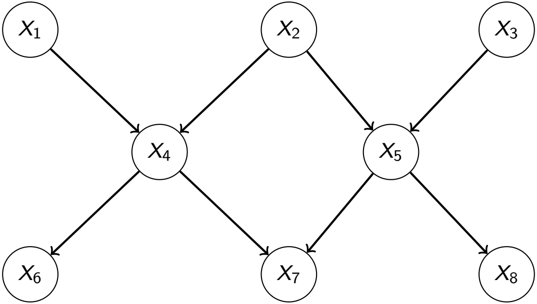
Probabilistic Graphical Models

aka Decomposable Multivariate Probabilistic Models

(whose decomposability is induced by independence)

directed graphs

Bayesian/credal networks

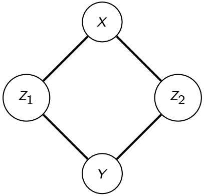


Markov Condition

- Probabilistic model over set of variables $(X_1, \ldots, X_n)$ in one-to-one correspondence with the nodes of a graph

Undirected Graphs

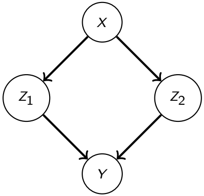
X and Y are independent given Z
if any path between X and Y
containts an element of Z



Directed Graphs

Given its parents, every node is independent of its (non-parent) non-descendants

X and Y are d-separated by Z if, along every path between X and Y there is a W such that either W has converging arrows and is not in Z and none of its descendants are in Z, or W has no converging arrows and is in Z



Directed Graphs

We choose to discuss further on directed graphs here. Some reasons:

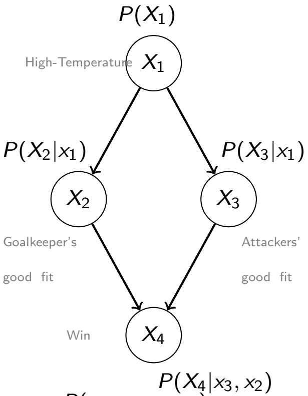
- Normalised distributions help with interpretability of local models.
- Directed edges help with sampling.
- Directed edges can be used for causal inference.
- Representation power (a.k.a. expressivity) is similar to undirected models.
- Computational costs are similar to undirected models.

(The Markov condition is arguably harder to describe/interpret than with its undirected counterpart.)

Bayesian networks

- Set of categorical variables $X_1, \ldots, X_n$
- Directed acyclic graph
  - conditional (stochastic) independencies according to the Markov condition:

    "any node is conditionally independent of its non-descendents given its parents"
- A conditional mass function for each node and each possible value of the parents
  - $\{P(X_i|\mathrm{pa}(X_i)),\ \forall i = 1, \ldots, n,\ \forall \mathrm{pa}(X_i)\}$
- Defines a joint probability mass function
  - $P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i|\mathrm{pa}(X_i))$



$P(x_1, x_2, x_3, x_4) =$
$P(x_1)P(x_2|x_1)P(x_3|x_1)P(x_4|x_3, x_2)$

E.g., given temperature, fitnesses independent

Bayesian network - simple example - bn1.txt

```
library(bnlearn)
source('my.bn.inference.r')
net = model2network("[x1][x2|x1][x3|x1][x4|x2:x3]")

cpt1.x1 = matrix(c(0.7, 0.3), ncol = 2,
    dimnames = list(NULL, c('true', 'false')))
cpt1.x2 = c(0.1, 0.9, 0.3, 0.7)
dim(cpt1.x2) = c(2, 2)
dimnames(cpt1.x2) = list("x2" = c("true", "false"),
            "x1" =  c("true", "false"))
cpt1.x3 = c(0.5, 0.5, 0.2, 0.8)
dim(cpt1.x3) = c(2, 2)
dimnames(cpt1.x3) = list("x3" = c("true", "false"),
            "x1" =  c("true", "false"))
cpt1.x4 = c(0.9, 0.1, 0.5, 0.5, 0.4, 0.6, 0.1, 0.9)
dim(cpt1.x4) = c(2, 2, 2)
dimnames(cpt1.x4) = list("x4" = c("true", "false"),
            "x2" =  c("true", "false"),
            "x3" = c("true", "false"))
```

Bayesian network - simple example - bn2.txt

```
net.1 = custom.fit(net, dist = list(x1=cpt1.x1,
                             x2=cpt1.x2, x3=cpt1.x3, x4=cpt1.x4))

query=rep(NA,length(net.1))
names(query) <- names(net.1)
query[2]='false'
my.bn.inference(net.1,query)
## $logp.evi
## [1] 0
## $logp.query
## [1] -0.1743534
## $p
## [1] 0.84
## $evidence
## [1] ""
## $query
## [1] "x2=false"
```

Bayesian network - simple example - bn3.txt

```
query[1]='true'
my.bn.inference(net.1,query)
## $logp.evi
## x[1] 0
## $logp.query
## [1] -0.4620355
## $p
## [1] 0.63
## $evidence
## [1] ""
## $query
## [1] "x1=true,x2=false"
```

Bayesian network - simple example - bn4.txt

```
evidence=rep(NA,length(net.1))
names(evidence) <- names(net.1)
evidence[4]='true'
res <- my.bn.inference(net.1,query,evidence)
##res not shown
evidence[4]='false'
my.bn.inference(net.1,query,evidence)
## $logp.evi
## [1] -0.3816998
## $logp.query
## [1] -0.8187104
## $p
## [1] 0.6459646
## $evidence
## [1] "x4=false"
## $query
## [1] "x1=true,x2=false"
```

Bayesian network - simple example - bn5.txt

```
my.bn.inference(net.1,NA,NA,map.query=TRUE)
## $logp.evi
## [1] -1.260543
## $p
## [1] 0.2835
## $map
##      x1   x2    x3    x4
## 16 true false false false
## $evidence
## [1] ""
evidence
##     x1      x2      x3      x4
##     NA      NA      NA "false"
my.bn.inference(net.1,NA,evidence,map.query=TRUE)
evidence[4]='true'
my.bn.inference(net.1,NA,evidence,map.query=TRUE)
```

Reasoning with Bayesian networks

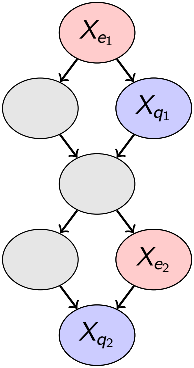
- Conditional probs for a variable of interest X_q given observations $X_E = x_E$

$$P(x_q|x_E) = \frac{P(x_q, x_E)}{P(x_E)} = \frac{\sum_{\mathbf{x}\setminus\{x_q,x_E\}} \prod_{i=1}^n P(x_i|\pi_i)}{\sum_{\mathbf{x}\setminus\{x_E\}} \prod_{i=1}^n P(x_i|\pi_i)}$$

- MAP/MPE queries can be even harder

$$\max_{\mathbf{x}_Q} P(\mathbf{x}_Q, \mathbf{x}_E) = \max_{\mathbf{x}_Q} \sum_{\mathbf{x}\setminus\{\mathbf{x}_Q,\mathbf{x}_E\}} \prod_{i=1}^n P(x_i|\pi_i)$$

- Updating Bayesian nets is NP-hard
(fast algorithms for belief updating and full MPE in polytree/bounded treewidth nets)



Bayesian network structures

Graph structures induce different factorisations. Examples:

- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2)P(X_3)P(X_4)$

Bayesian network structures

Graph structures induce different factorisations. Examples:

- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2)P(X_3)P(X_4)$
- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)P(X_4|X_1, X_2, X_3)$

Bayesian network structures

Graph structures induce different factorisations. Examples:

- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2)P(X_3)P(X_4)$
- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)P(X_4|X_1, X_2, X_3)$
- $P(X_1, X_2, X_3, X_4) = P(X_1|X_2, X_3, X_4)P(X_2|X_3, X_4)P(X_3|X_4)P(X_4)$

Bayesian network structures

Graph structures induce different factorisations. Examples:

- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2)P(X_3)P(X_4)$
- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)P(X_4|X_1, X_2, X_3)$
- $P(X_1, X_2, X_3, X_4) = P(X_1|X_2, X_3, X_4)P(X_2|X_3, X_4)P(X_3|X_4)P(X_4)$
- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2|X_1)P(X_3|X_1)P(X_4|X_2, X_3)$

Bayesian network structures

Graph structures induce different factorisations. Examples:

- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2)P(X_3)P(X_4)$
- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)P(X_4|X_1, X_2, X_3)$
- $P(X_1, X_2, X_3, X_4) = P(X_1|X_2, X_3, X_4)P(X_2|X_3, X_4)P(X_3|X_4)P(X_4)$
- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2|X_1)P(X_3|X_1)P(X_4|X_2, X_3)$
- ~~$P(X_1, X_2, X_3, X_4) = P(X_1|X_2)P(X_2|X_3)P(X_3|X_4)P(X_4|X_1)$~~

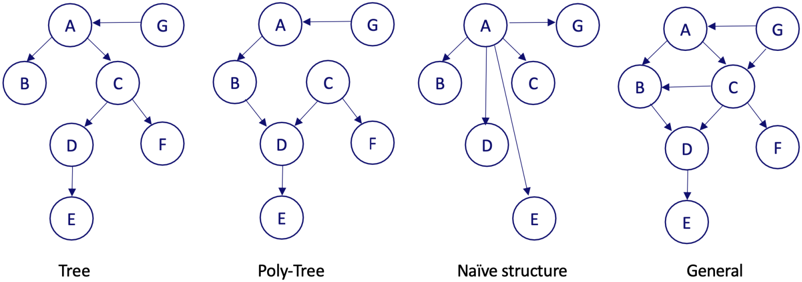
Bayesian network structures

Graph structures induce different factorisations. Examples:

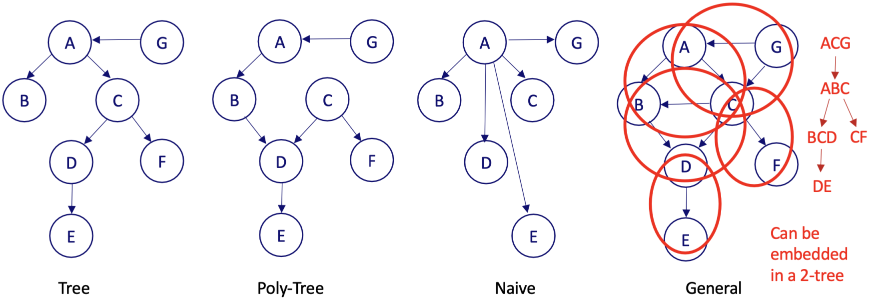
- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2)P(X_3)P(X_4)$
- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)P(X_4|X_1, X_2, X_3)$
- $P(X_1, X_2, X_3, X_4) = P(X_1|X_2, X_3, X_4)P(X_2|X_3, X_4)P(X_3|X_4)P(X_4)$
- $P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2|X_1)P(X_3|X_1)P(X_4|X_2, X_3)$
- ~~$P(X_1, X_2, X_3, X_4) = P(X_1|X_2)P(X_2|X_3)P(X_3|X_4)P(X_4|X_1)$~~
- ...

Key fact: model size grows linearly in the number of variables if local conditional functions are bounded (i.e. not so many parents per node).

Bayesian network structure complexity



Bayesian network tree-width



Minimum treewidth: how small is the k to which we can embed the moralised graph of a Bayesian net in a k-tree graph (roughly a hyper-tree where each node is contained in a (k+1)-clique).

Learning Bayesian networks

Two main approaches to structure learning:

- Find the graph structure with highest score. Possible scores:
  - Max. Likelihood (break ties towards simplicity)
  - Bayesian Dirichlet (Equivalent Uniform)
  - Bayesian Information Criterion
  - Minimum Description Length
  - Akaike Information Criterion
- Use statistical testing to find a collection of (conditional) (in)dependences and use a graph compatible with the results.

Learning Bayesian networks

Two main approaches to structure learning:

- Find the graph structure with highest score. Possible scores:
  - Max. Likelihood (break ties towards simplicity)
  - Bayesian Dirichlet (Equivalent Uniform)
  - Bayesian Information Criterion
  - Minimum Description Length
  - Akaike Information Criterion
- Use statistical testing to find a collection of (conditional) (in)dependences and use a graph compatible with the results.

Parameter learning usually done with standard estimators:

- (Penalised) Maximum likelihood
- Bayesian updating (multinomial Dirichlet)

Some references

- Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models*. MIT Press.
- Darwiche, A. (2009). *Modeling and Reasoning with Bayesian Networks* Cambridge Press.
- Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann.