Credal networks

Cassio P. de Campos[1] and Fabio G. Cozman[2]

[1]Rensselaer Polytechnic Institute
Troy, NY, USA
[2]Engineering School, University of São Paulo
São Paulo, SP, Brazil

SIPTA School, July, 2008

Overview

- Graphical models: basic definitions and applications.
- Credal networks: basic definitions.
- Inference and learning in credal networks.
- Quick break: software packages.
- Formal definitions and computational complexity.
- Applications.

Later

Exact and approximate algorithms for reasoning.

Agenda

Some motivation

Graph-theoretical statistical models, directed and undirected
- Bayesian networks
- Markov random fields and other graph-theoretical models

Credal networks
- Other models related to credal networks
- Learning and Reasoning
- Software packages

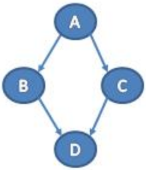
Computational Complexity

Final remarks

Applications
- Computer vision problems
- Military planning

What is this? Why spend time on such a topic?

- A credal network is a *compact* representation for a set of probability distributions.
  - It is a based on *graphs*, so it is easy to draw and to understand.
  - It offers a compact and easy *language* in which to express complicated situations.
- It is closely related to very popular statistical models such as Markov chains, Bayesian networks, Markov random fields, etc.

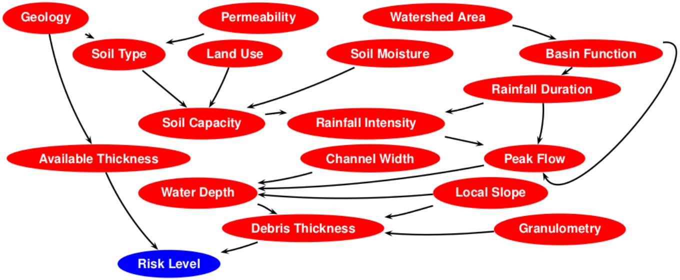


The appeal of graph-theoretical models

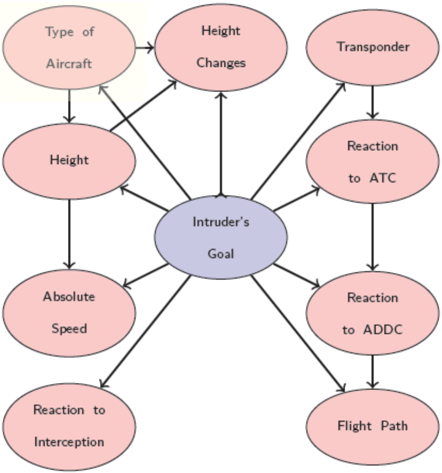
- Compact and easy language to represent uncertainty.

Expert system: debris flows hazard assessment

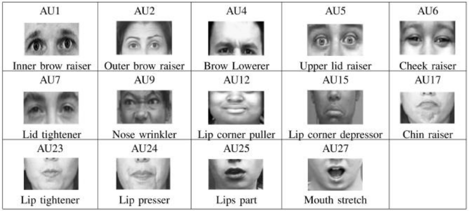
- Network developed by the IDSIA team.
- Goal: to support decision making regarding such flows.

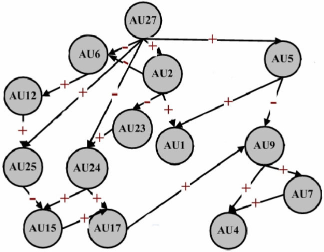


Another expert system: detecting intruders



Another application: Facial expression recognition





Yet another application (details later): image segmentation

(c) Segmentation produced by Bayesian network

(d) Segmentation produced by credal network

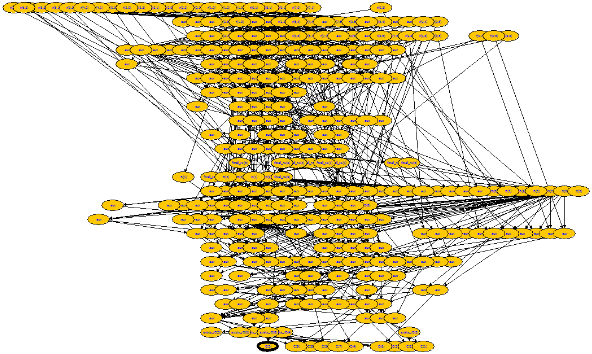
And an application on knowledge representation

- *Description logics* are used to create terminologies.
- An example of *probabilistic description logic*:

$$P(A) = \alpha_1, \quad B \sqsubseteq A, \quad C \sqsubseteq B \sqcup \exists r.D, \quad P(B|A) = \alpha_2,$$

$$P(C|B \sqcup \exists r.D) = \alpha_3, \quad P(D|\forall r.A) = \alpha_4.$$

- These sentences encode a large credal network.



Agenda

Some motivation

Graph-theoretical statistical models, directed and undirected
- Bayesian networks
- Markov random fields and other graph-theoretical models

Credal networks
- Other models related to credal networks
- Learning and Reasoning
- Software packages

Computational Complexity

Final remarks

Applications
- Computer vision problems
- Military planning

First, the unstructured approach

- Take 5 random binary variables A, B, C, D, E assuming values x and $\neg x$.
- To specify the joint distribution, we need 2^5 probability values:

$$p(a,b,c,d,e), p(a,b,c,d,\neg e), p(a,b,c,\neg d,e), \ldots$$

- We can compute the probability of events by marginalization:

$$p(a) = \sum_{B,C,D,E} p(a,B,C,D,E)$$

Inferences

- Conditional probabilities are obtained by Bayes rule:

$$p(a|d,\neg e) = \frac{p(a,d,\neg e)}{p(d,\neg e)} = \frac{\sum_{B,C} p(a,B,C,d,\neg e)}{\sum_{A,B,C} p(A,B,C,d,\neg e)}$$

- Drawbacks:
  - Exponential number of parameters to elicit.
  - Exponential number of terms in the summation to perform inferences.
- Bayesian networks, Markov random fields, etc, provide a way to alleviate these issues through graph-theoretical tools.

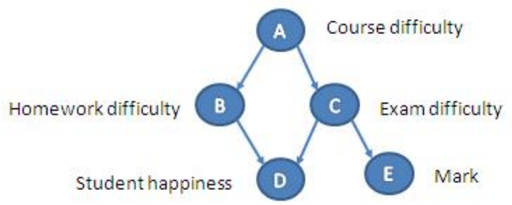
Why graphs

- Compact.
- Easy to handle, easy to visualize.
- Efficient algorithms.
- Plausible models of causal relations.

- Computer scientists love graphs!

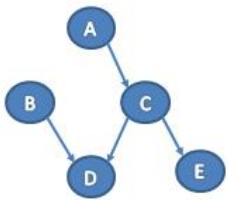
Directed Acyclic Graphs (DAGs)

- Nodes and arcs without directed cycles.
- Arcs define parents, children, descendants,...

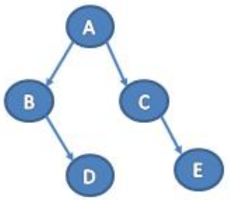


Polytrees

- The subjacent graph has no cycles.
- Trees are polytrees with maximum in-degree equals to 1.

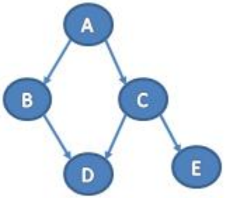


(e) Polytree



(f) Tree

Markov condition



- Suppose we have a DAG where each variable is independent of its non-descendants non-parents given its parents (Markov condition).
  - $B \perp\!\!\!\perp (C, E) | A$,
  - $D \perp\!\!\!\perp (A, E) | (B, C)$,
  - $E \perp\!\!\!\perp (A, B, D) | C$.

Exercise: Factorization

- Show: The Markov condition implies

$$p(X_1, \ldots, X_n) = \prod_i p(X_i | pa(X_i)).$$

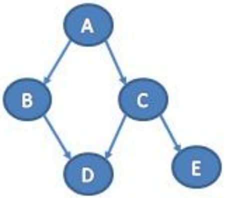
- E.g., in the previous network,

$$\begin{aligned} p(A, B, C, D, E) &= p(E|C)p(A, B, C, D) \\ &= p(E|C)p(D|B, C)p(A, B, C) \\ &= p(E|C)p(D|B, C)p(B|A)p(A, C) \\ &= p(E|C)p(D|B, C)p(B|A)p(C|A)p(A) \end{aligned}$$

Factorization

- Consequence: just a “few” parameters specify the whole joint distribution.
  - Linear in the number of variables if in-degree is bounded.

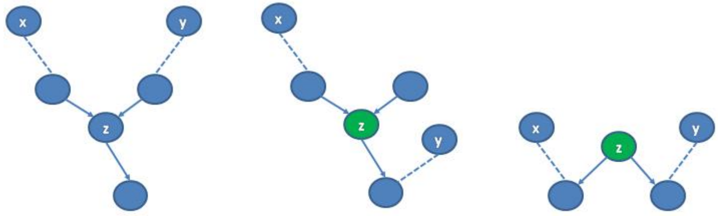
Bayesian networks



- A DAG with the Markov condition.
- For each variable (node), a “local” probability distribution: variable conditional on its parents.
- In a Bayesian network, all probability values are unique.
  - $p(A)$,
  - $p(B|a), p(B|\neg a)$,
  - $p(C|a), p(C|\neg a)$,
  - $p(D|b, c), p(D|\neg b, c), p(D|b, \neg c), p(D|\neg b, \neg c)$,
  - $p(E|c), p(E|\neg c)$,

Independence: d-separation

X is independent of Y given Z if all paths (using arcs in both directions) between X and Y are blocked by an element of Z. A path is blocked if *an element of Z is observed and has not two converging arcs* (in the path being analyzed) or *an element of Z is not observed (neither any of its descendants) and has two converging arcs*.



(g) Converging (h) Directed path (i) Diverging

Inferences

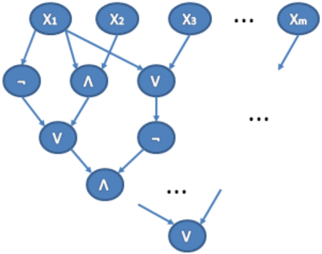
- Conditional probabilities are obtained by Bayes rule:

$$p(a|d,\neg e) = \frac{\sum_{B,C} p(a)p(B|a)p(C|a)p(d|B,C)p(\neg e|C)}{\sum_{A,B,C} p(A)p(B|A)p(C|A)p(d|B,C)p(\neg e|C)}$$

- In many situations, we can do much better (for example because of d-separation).

$$p(a|c) = \frac{p(a,c)}{p(c)} = \frac{p(a)p(c|a)}{\sum_A p(A)p(c|A)}$$

Inference is #P-hard

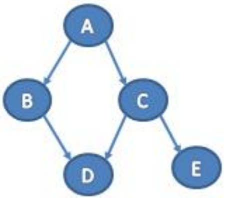


- Local conditional probability distributions encode the true tables of the logical operators.
- Let γ be the only operator without children.

$$p(\gamma) = \sum_{\mathbf{x}} p(\gamma|\mathbf{x})p(\mathbf{x}) = 2^{-m} \sum_{\mathbf{x}} p(\gamma|\mathbf{x})$$

- Note that $p(\gamma|\mathbf{x}) = 1$ if $\mathbf{x}$ satisfy the whole sentence, and zero otherwise.
- Hence, $2^m \cdot p(\gamma)$ is the number of instantiations of $\mathbf{x}$ that satisfy the logical sentence.

Bayesian network example



- $p(a) = 0.2, p(\neg a) = 0.8,$
- $p(c|a) = 0.1, p(\neg c|a) = 0.9, p(c|\neg a) = 0.8, p(\neg c|\neg a) = 0.2,$

$$p(a|c) = \frac{p(a,c)}{p(c)} = \frac{0.2 \times 0.1}{0.2 \times 0.1 + 0.8 \times 0.8} = 0.0303\ldots$$

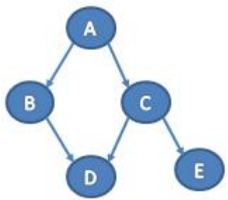
More on inferences

- Even the hardest belief updating inference is a matter of choosing the best ordering for evaluations.

$$p(y) = \sum_{\mathcal{X} \setminus Y} \prod_i p(x_i | pa(X_i))$$

Some of these summations can be interchanged with some multiplications.

Belief Updating Inferences



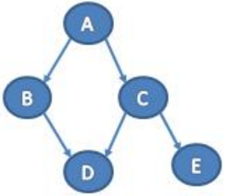
$$p(d) = \sum_{A,B,C} p(d|B,C)p(B|A)p(C|A)p(A) =$$

$$= \sum_{B,C} p(d|B,C) \sum_{A} p(B|A)p(C|A)p(A).$$

Exact inference: Variable elimination

- Choose an ordering for the variables where queries come last.
- Create a pool of functions with all local probability distributions.
- For each variable X in the ordering:
  - Insert all functions that contain X in a structure called bucket of X and remove them from the pool.
  - Multiply these functions and sum the result over X.
  - Insert the resulting functions in the pool.

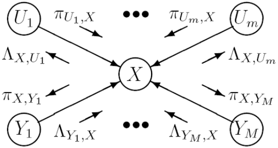
Variable elimination



Suppose the ordering A, B, C, D and we want $p(D)$.

- The pool contains $p(A), p(B|A), p(C|A), p(D|B, C)$.
- Bucket of A: $\sum_A p(B|A)p(C|A)p(A) = p(B, C)$. Insert the function $p(B, C)$ in the pool.
- Bucket of B: $\sum_B p(D|B, C)p(B, C) = p(D, C)$. Insert the function $p(D, C)$ in the pool.
- Bucket of C: $\sum_C p(D, C) = p(D)$. Insert $p(D)$ in the pool.
- Bucket of D: just get $p(d)$ from the pool.

Exact inference in polytrees: Belief Propagation



Local calculations:

$$p(x|e) = \alpha \Lambda(x) \pi(x),$$

$$\Lambda(x) = \Lambda_X(x) \prod_j \Lambda_{Y_j}(x),$$

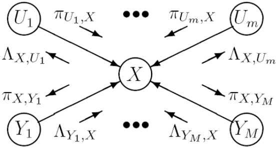
$$\pi(x) = \sum_u p(x|u) \prod_k \pi_X(u_k),$$

Messages to propagate:

$$\Lambda_X(u_i) = \alpha \sum_x \Lambda(x) \sum_{u_k : k \neq i} p(x|u) \prod_{k \neq i} \pi_X(u_k),$$

$$\pi_{Y_j}(x) = \alpha \pi(x) \Lambda_X(x) \prod_{k \neq j} \Lambda_{Y_k}(x).$$

Approximate inference: Loopy Belief Propagation



Local calculations:

$$p(x|e) = \alpha \Lambda(x) \pi(x),$$

$$\Lambda^{(t)}(x) = \Lambda_X(x) \prod_j \Lambda^{(t)}_{Y_i}(x),$$

$$\pi^{(t)}(x) = \sum_u p(x|u) \prod_k \pi^{(t)}_X(u_k),$$

Messages to propagate:

$$\Lambda^{(t+1)}_X(u_i) = \alpha \sum_x \Lambda^{(t)}(x) \sum_{u_k : k \neq i} p(x|u) \prod_{k \neq i} \pi^{(t)}_X(u_k),$$

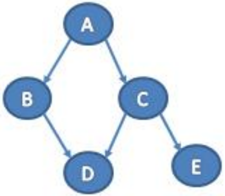
$$\pi^{(t+1)}_{Y_j}(x) = \alpha \pi^{(t)}(x) \Lambda_X(x) \prod_{k \neq j} \Lambda^{(t)}_{Y_k}(x).$$

Other inferences

Maximum a posteriori:

$$\arg\max_{A,D} p(A, D|e) = \arg\max_{A,D} p(A, D, e).$$

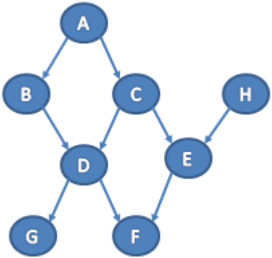
Bayesian network example



- $p(a) = 0.2, p(\neg a) = 0.8,$
- $p(b|a) = 0.4, p(\neg b|a) = 0.6, p(b|\neg a) = 0.5, p(\neg b|\neg a) = 0.5,$
- $p(c|a) = 0.1, p(\neg c|a) = 0.9, p(c|\neg a) = 0.8, p(\neg c|\neg a) = 0.2,$
- $p(d|b,c) = 0.75, p(\neg d|b,c) = 0.25, p(d|\neg b,c) = 0.6, p(\neg d|\neg b,c) = 0.4, p(d|b,\neg c) = 0.45, p(\neg d|b,\neg c) = 0.55, p(d|\neg b,\neg c) = 0.4, p(\neg d|\neg b,\neg c) = 0.6,$
- $p(e|c) = 0.7, p(\neg e|c) = 0.3, p(e|\neg c) = 0.2, p(\neg e|\neg c) = 0.8,$

Exercises

- Evaluate $p(a|e)$ using the Bayesian network just defined. Count the number of multiplications that you need to find the solution.
- Find $\arg\max_{A,C|d} p(A, C|d)$ using the same Bayesian network.
- Using the following Bayesian network, verify which are true: $(A \perp\!\!\!\perp H|F)$, $(G \perp\!\!\!\perp E|C)$, $(G \perp\!\!\!\perp E|A)$, $(B \perp\!\!\!\perp E|A, D)$, $(G \perp\!\!\!\perp H|F)$.

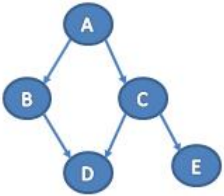


Learning problem

It is possible to learn

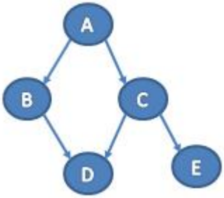
- The graph structure (that is, the (in)dependence relations).
- The parameters (that is, probability values that define the local conditional distributions).
  - Given data and structure.

A	B	C	D	E
a	¬b	¬c	¬d	¬e
a	b	¬c	d	¬e
a	¬b	c	d	¬e



Parameter learning

A	B	C	D	E
a	¬b	¬c	¬d	¬e
a	b	¬c	d	¬e
a	¬b	c	d	¬e



Maximum likelihood estimation

- $p(x|pa(X)) = \frac{n_{x,pa(X)}}{n_{pa(X)}}$
- $p(a) = 1, p(\neg a) = 0, p(c|a) = \frac{1}{3}, p(\neg c|a) = \frac{2}{3}, p(c|\neg a) = ?, \ldots$

Dirichlet process using posterior expectation as estimation

- $p(x|pa(X)) = \frac{n_{x,pa(X)} + s \cdot \tau(x|pa(X))}{n_{pa(X)} + s}$, where $\sum_x \tau(x|pa(X)) = 1$ and s are hyper-parameters. Suppose $s = 2$ and uniform priors. Then
- $p(a) = \frac{4}{5}, p(\neg a) = \frac{1}{5}, p(c|a) = \frac{2}{5}, p(\neg c|a) = \frac{3}{5}, p(c|\neg a) = \frac{1}{2}, \ldots$

Limitations

Suppose we have $A \perp\!\!\!\perp D | B, C$ and $B \perp\!\!\!\perp C | A, D$. It is not possible to define a Bayesian network that encodes such independence relations.

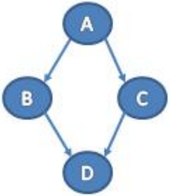


Figure: Unsuccessful try

Undirected graphs

- Nodes and edges (undirected arcs). E.g. Markov Random Fields.
- Markov condition: X and Y are independent given Z if any path between X and Y contains an element of Z.
- Joint distribution:

$$p(\mathbf{x}) = \frac{1}{\alpha} \prod_k \phi_k(x_{\{k\}}), \quad \text{where } \alpha = \sum_{\mathbf{x}} \prod_k \phi_k(x_{\{k\}}).$$

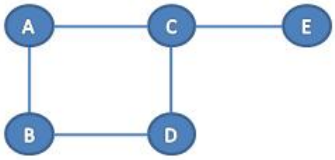
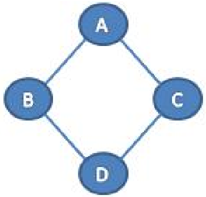


Figure: Markov Random Field

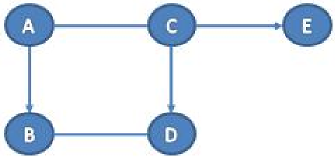
Undirected graphs

MRFs can encode $A \perp\!\!\!\perp D | B, C$ and $B \perp\!\!\!\perp C | A, D$.

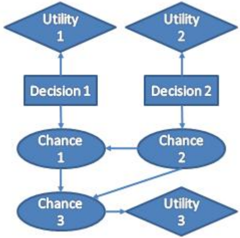


Others...

- Different types of nodes. E.g. influence diagrams, decision trees.
- Mixed graphs (directed and undirected arcs). E.g. chain graphs.



(a) Chain graph



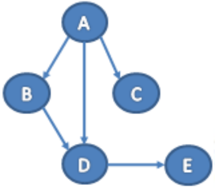
(b) Influence Diagram

Exercises

- Give a set of independence relations that can be encoded using a Bayesian network (and show such network) but cannot be encoded using a Markov Random Field. The network must encode exactly such relations (nothing else).
- Use maximum likelihood to estimate the parameters of the following Bayesian network.
- Repeat using a Dirichlet model with $s = 1$ and uniform $\tau(X|pa(X))$.

$$p(x|pa(X)) = \frac{n_{x,pa(X)} + s \cdot \tau(x|pa(X))}{n_{pa(X)} + s}$$

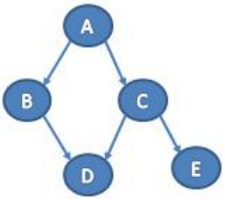
A	B	C	D	E
a	¬b	¬c	¬d	¬e
a	b	¬c	d	¬e
a	¬b	c	d	¬e



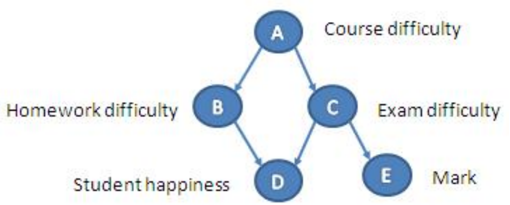
Agenda

Introduction

A credal network is a Bayesian network where some (or all) parameters are not precisely known, but instead constrained by convex constraints.



Constraints



- $p(a) \in [0.1, 0.3]$.
- $p(b|a) \in [0.5, 0.8]$.
- $p(d|b, c) = 0.4$.
- $p(d|b, \neg c) + p(d|\neg b, c) \leq 0.75$.
- ...

Constraints

- Constraints of credal networks are assumed to be local, that is, they contain parameters of a single node. E.g.
  - $p(b|a) \geq p(b|\neg a)$
  - $p(d|b,c) + 2 \cdot p(\neg d|\neg b, c) - 3 \cdot p(d|\neg b, \neg c) = 0.3$
  - $p(b|a) \leq p(c|a) \longleftarrow$ non-local!
- and if they have parameters of a single conditional distribution, then we say that they are separately specified.
  - $p(b|a) \leq 2 \cdot p(\neg b|a)$
  - $p(e|c) \geq p(\neg e|c)$
  - $p(b|a) \geq p(b|\neg a) \longleftarrow$ non-separate!
- Note: a credal network may or may not be separately specified.

Credal sets

- A *credal set* is a set of probability distributions, denoted as $K(\mathcal{X})$. In our example, $\mathcal{X} = \{A, B, C, D, E\}$.
- A conditional credal set is obtained by applying the Bayes rule to each distribution in a (joint) credal set.
- Credal sets $K(X|pa(X))$ are specified in the credal network for every X.

Markov condition

In this talk we adopt the strong independence concept.

Strong independence

All extreme distributions in the convex hull of $K(\mathcal{X})$ satisfy standard stochastic independence.

Strong Markov Condition

Every variable is strongly independent of its non-descendants given its parents.

Consequence of Markov condition

- The extreme distributions of $K(\mathcal{X})$ factorize as a standard Bayesian network:

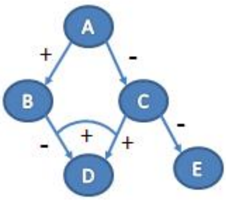
$$p(\mathcal{X}) = \prod_i p(X_i | pa(X_i))$$

  and the credal network can be viewed as a collection of Bayesian networks that share the same graph.
- This concept is the most adopted in the literature.
- But there are other concepts (SEE TALK TOMORROW...!).

Qualitative Probabilistic Networks

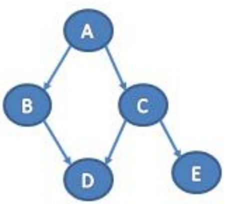
Sub-case of credal networks where constraints have some pre-defined format:

- Qualitative influences: effect of a parent to a child. A positive influence means that occurrence of the parent increases the chance of occurrence of the child.
  - E.g. $p(b|a) \geq p(b|\neg a)$, $p(e|c) \leq p(e|\neg c)$,...
- Qualitative synergies: effect of two parents acting together to influence a child.
  - E.g. $p(d|b,c) + p(d|\neg b, \neg c) \geq p(d|\neg b, c) + p(d|b, \neg c)$.

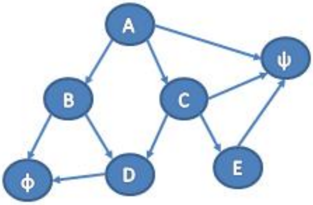


Probabilistic Propositional Logic Networks

- Nodes are associated to propositions.
- The graph encodes (in)dependence relations among propositions.
- Probabilistic propositional sentences are not restricted to parameters of the network.
  - E.g. $p(\phi) + p(\psi) \leq 0.7$, where $\phi = b \vee d$ and $\psi = (a \vee \neg e) \wedge c$.



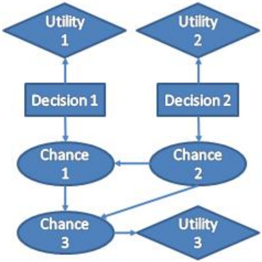
(c) Network structure



(d) Augmented network

Influence Diagrams

- Graphical model used for decision making.
- They extend Bayesian networks with decision and utility nodes (besides chance nodes as in Bayesian nets).
  - Ellipses represent probabilistic dependencies.
  - Rectangles represent decisions.
  - Diamonds represent profits or costs.



Credal inferences

An inference in a credal network is the task of finding a distribution that complies with the credal network constraints and maximize (or minimize) an objective function.

Inferences

Examples:

- Distribution that minimizes a marginal query given evidence (known as belief updating):

$$\underline{p}(a|d,e) = \min_{p \in K(\mathcal{X})} p(a|d,e).$$

- Distribution that maximizes likelihood function (given data):

$$\max_{p \in K(\mathcal{X})} \sum_{ijk} n_{ijk} \cdot \log p(x_{ik}|pa(X_i)_j),$$

where x_{ik} is a value of X_i, $pa(X_i)_j$ is a configuration for the parents $PA(X_i)$, and n_{ijk} are the counts from data.

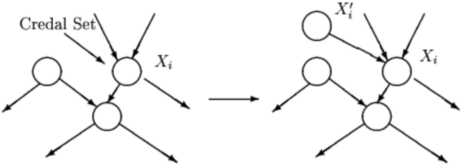
- Distribution that maximizes the (local) entropy:

$$\max_{p \in K(\mathcal{X})} -\sum_{ijk} p(x_{ik}|pa(X_i)_j) \cdot \log p(x_{ik}|pa(X_i)_j).$$

Cano–Cano–Moral transformation

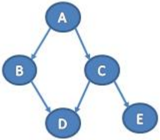
Translate a belief updating in credal network into a MAP inference in Bayesian network:

- Precondition: extreme points of credal sets must be available.
- For each node X_i, create an additional parent X_i' that has X_i as its sole child and where X_i' has uniform distribution over t values, where t is the number of extreme distributions in the credal set of X_i. The distribution of X_i becomes $p(X_i|pa(X_i), X_i') = p_l(X_i|pa(X_i))$, where $1 \leq l \leq t$ represents each extreme distribution of $K(X_i|pa(X_i))$.
- Now a MAP inference on the X_i' variables is enough.



Exercises

- Evaluate $\overline{p}(a|e)$ using the following credal network.



  $p(a) \in [0.1, 0.3]$, $p(c|a) = 0.5$, $p(c|\neg a) = 0.8$, $p(e|c) \in [0.6, 0.9]$, $p(e|\neg c) = 0.5$, $p(b|\neg a) \in [0.1, 0.5]$, $p(d|b, c) \in [0.1, 0.5]$, $p(d|\neg b, c) = 0.2$ and other parameters are vacuous.
- Translate the credal network into a Bayesian network using the CCM transformation.
- Find a parameterization that respects the credal network and maximizes the entropy in each local conditional distribution.

Bayesian network learning

- Given
  - Complete data set with samples of the variables, and
  - A credal network,

  parameter learning (in Bayesian networks) is the problem of selecting a distribution that complies with the credal network constraints and fits the data.
- The result is a Bayesian network.
- For example, standard maximum likelihood estimation is just the credal network inference that looks for the distribution that maximizes the likelihood function over a completely vacuous credal network.

Credal network learning

- Given
  - Complete data set with samples of the variables, and
  - A prior credal network,

  parameter learning is the problem of updating the credal network constraints using the information provided by the data.
- The result is a credal network.

Imprecise Dirichlet Model

Using an Imprecise Dirichlet Model, the resulting credal network is composed of:

- The same graph as the prior credal network.
- A collection of constraints that contains constraints of the prior credal network to restrict the values of $\tau_{x_{ik},pa(X_i)}$, for all x_{ik} and all parent configurations $pa(X_i)$, plus

$$p(x_{ik}|pa(X_i)) = \frac{s\tau_{x_{ik},pa(X_i)} + n_{x_{ik},pa(X_i)}}{s + n_{pa(X_i)}}.$$

Full Imprecise Priors

When no information about the priors are known, then

$$p(x_{ik}|pa(X_i)) \in \left[\frac{n_{x_{ik},pa(X_i)}}{s + n_{pa(X_i)}}, \frac{s + n_{x_{ik},pa(X_i)}}{s + n_{pa(X_i)}} \right],$$

Note that

$$\sum_{k \neq k'} \frac{n_{x_{ik},pa(X_i)}}{s + n_{pa(X_i)}} + \frac{s + n_{x_{ik'},pa(X_i)}}{s + n_{pa(X_i)}} = \frac{s + \sum_k n_{x_{ik},pa(X_i)}}{s + n_{pa(X_i)}} = 1$$

Binary networks

Another interesting case happens when the variables are binary and the prior network parameters are specified through intervals.

Posterior is also defined by intervals

The resulting network can also be expressed by intervals and there is no need to keep additional constraints.

$$p(x_i|pa(X_i)) \in \left[\frac{s\underline{\mathcal{T}}_{x_i,pa(X_i)} + n_{x_i,pa(X_i)}}{s + n_{pa(X_i)}}, \frac{s\overline{\mathcal{T}}_{x_i,pa(X_i)} + n_{x_i,pa(X_i)}}{s + n_{pa(X_i)}} \right],$$

$$p(\neg x_i|pa(X_i)) \in \left[\frac{s\underline{\mathcal{T}}_{\neg x_i,pa(X_i)} + n_{\neg x_i,pa(X_i)}}{s + n_{pa(X_i)}}, \frac{s\overline{\mathcal{T}}_{\neg x_i,pa(X_i)} + n_{\neg x_i,pa(X_i)}}{s + n_{pa(X_i)}} \right].$$

Non-binary networks

- If we keep only the intervals of previous slide, then we are eventually losing information about the credal sets.

Solution

Learning in non-binary credal network may be achieved by incorporating the Dirichlet expectation equations and all constraints of the prior credal network to define the posterior network.

- If one wants to remove the hyper-parameters τ from equations of the posterior network, a simple replacement suffices:

$$\tau_{x_{ik},pa(X_i)} = \frac{(s + n_{pa(X_i)}) \cdot p(x_{ik}|pa(X_i)) - n_{x_{ik},pa(X_i)}}{s}.$$

Incomplete data

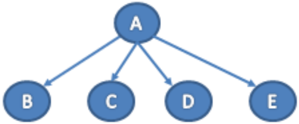
A	B	C	D	E
a	¬b	¬c	¬d	?
¬a	?	¬c	?	e
a	¬b	?	d	e

A good approach to deal with incomplete data is to consider all possible completions (conservative updating rule).

On the other hand, standard Bayesian network approach

Expectation–Maximization is an algorithm that iterates until convergence performing two steps: an expectation step where data are completed (using expectations) and a maximization step where standard maximum likelihood is employed.

Naive Credal Classifier



A	B	C	D	E
a	¬b	¬c	¬d	?
¬a	?	¬c	?	e
a	¬b	?	d	e

- Learning idea using the IDM:

$$p(X|A) \in \left[\frac{\underline{n}_{X,A}}{s + n_A}, \frac{s + \overline{n}_{X,A}}{s + n_A} \right],$$

for all X. A is the classification variable and is supposed to be observable.

Naive Credal Classifier

Credal dominance

Let a_i and a_j be classes of A. a_i dominates a_j if the posterior probability of a_i is greater than that of a_j everywhere.

Simple case: binary A implies that

$$\overline{p}(a|B, C, D, E) = 1 - \underline{p}(\neg a|B, C, D, E).$$

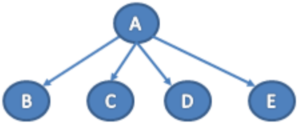
Thus $\neg a$ dominates a if $\overline{p}(a|B, C, D, E) < \underline{p}(\neg a|B, C, D, E)$, that is,

$$\underline{p}(\neg a|B, C, D, E) > 1/2.$$

Set-valued classification

Naive Credal Classifier returns all non-dominated classes.

Naive Credal Classifier



- General formulation: a' dominates a" if

$$\min_{t_{a'}, t_{a''} > 0} \left[\left(\frac{n_{a''} + s t_{a''}}{n_{a'} + s t_{a'}} \right)^{m-1} \prod_{j=1}^{m} \frac{\underline{n}_{a', x_j}}{\overline{n}_{a'', x_j} + s t_{a''}} \right] > 1$$

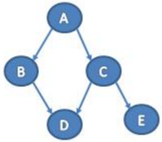
subject to $t_{a'} + t_{a''} = 1$, where A is the classification variable and $\mathcal{X}$ are the m features.

NCC is covered again later today.

Exercise

- Use Imprecise Dirichlet Model to learn new intervals for the credal network. Use $s = 1$.

A	B	C	D	E
a	¬b	¬c	¬d	¬e
a	b	¬c	d	¬e
a	¬b	c	d	¬e



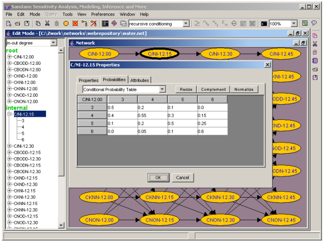
$p(a) \in [0.1, 0.3]$, $p(c|a) = 0.5$, $p(c|\neg a) = 0.8$, $p(e|c) \in [0.6, 0.9]$, $p(e|\neg c) = 0.5$, $p(b|\neg a) \in [0.1, 0.5]$, $p(d|b, c) \in [0.1, 0.5]$, $p(d|\neg b, c) = 0.2$ and other parameters are vacuous.

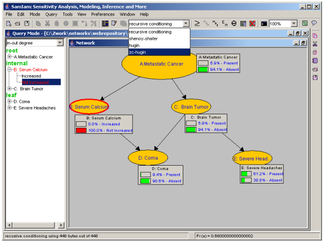
Software packages

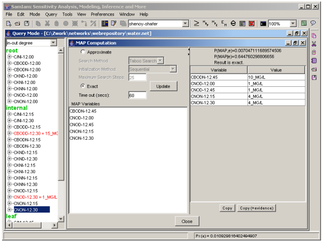
- SamIam
  - Performs MAP queries, so it is possible to solve credal inferences with CCM transformation (it is up to the user to perform the transformation).
- JavaBayes
  - Supports credal networks defined through extreme points.
- BNT (inside matlab)
  - Extensions have been developed to make BNT work with some credal networks.
- ...

Samlam

Developed by Darwiche at UCLA.

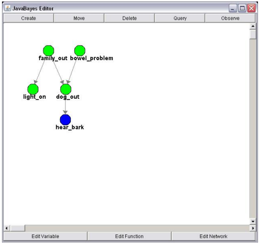


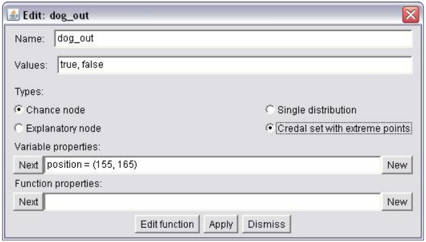


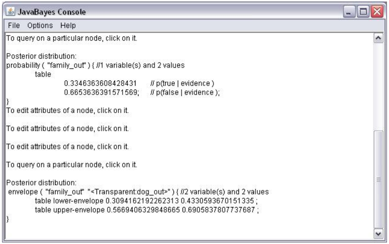


JavaBayes

Developed by Cozman at CMU and USP.

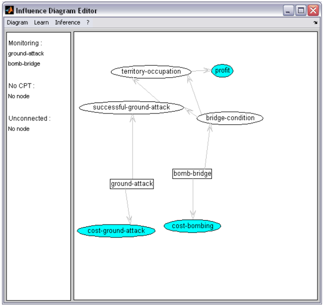


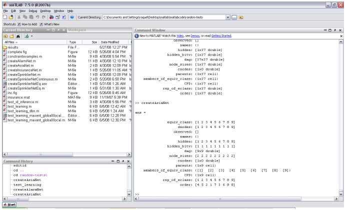


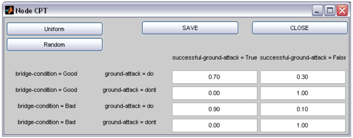


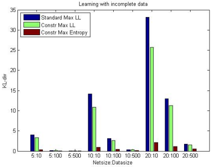
BNT

Developed by Murphy at UBC (except the graphical interface).

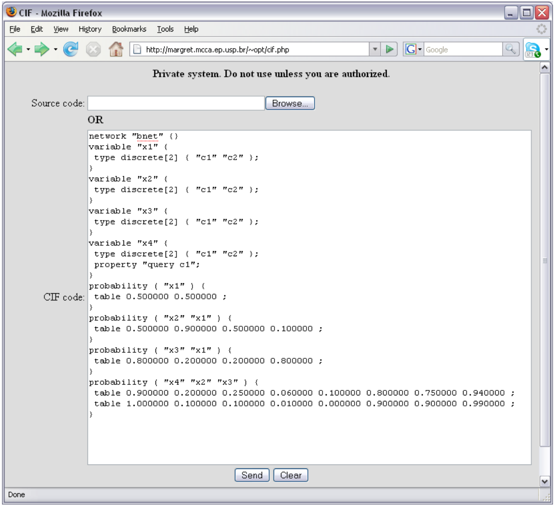








cif2ampl


CIF - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://margret.mcca.ep.usp.br/~opt/cif.php
Google
Private system. Do not use unless you are authorized.
Source code:
Browse...
OR
CIF code:
network "bnet" {}
variable "x1" {
type discrete[2] { "c1" "c2" };
}
variable "x2" {
type discrete[2] { "c1" "c2" };
}
variable "x3" {
type discrete[2] { "c1" "c2" };
}
variable "x4" {
type discrete[2] { "c1" "c2" };
property "query c1";
}
probability ("x1") {
table 0.500000 0.500000 ;
}
probability ("x2" "x1") {
table 0.500000 0.900000 0.500000 0.100000 ;
}
probability ("x3" "x1") {
table 0.800000 0.200000 0.200000 0.800000 ;
}
probability ("x4" "x2" "x3") {
table 0.900000 0.200000 0.250000 0.060000 0.100000 0.800000 0.750000 0.940000 ;
table 1.000000 0.100000 0.100000 0.010000 0.000000 0.900000 0.900000 0.990000 ;
}
Send
Clear
Done

Exercises

We are going to use the software packages to model the credal networks described in this presentation and perform some inferences with them. Hence, let's take a look into the softwares...

Agenda

Networks with respect to topology

- Bounded induced-width: moralized graph has induced-width bounded by $O(\log(s))$, where s is the size of input.
- Polytrees: a bounded induced-width network where the subjacent graph has no cycles.
- Multiply-connected: the general case.

Complexity classes of interest

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PP} \subseteq \mathrm{NP}^{\mathrm{PP}}$$

Some important problems:

- `SAT` is NP-Complete ($\exists x\ \phi(x)$?)
- `MAJSAT` is PP-Complete
  (majority of x satisfies $\phi(x)$?)
- `E-MAJSAT` is $\mathrm{NP}^{\mathrm{PP}}$-Complete
  ($\exists x$ such that majority of y satisfies $\phi(x, y)$?)

Summary of complexity results

Bayesian Networks

Problem	*Polytree*	*Bounded induced-width*	*Multiply-connected*
Belief updating	Polynomial	Polynomial	PP-Complete
MPE	Polynomial	Polynomial	NP-Complete
MAP	NP-Complete	NP-Complete	$\mathrm{NP}^{\mathrm{PP}}$-Complete
MmAP	Σ_2^{P}-Complete	Σ_2^{P}-Complete	$\mathrm{NP}^{\mathrm{PP}}$-Hard

Credal Networks with strong independence

Problem	*Polytree*	*Bounded induced-width*	*Multiply-connected*
MPE	Polynomial	Polynomial	NP-Complete
Belief updating	NP-Complete	NP-Complete	$\mathrm{NP}^{\mathrm{PP}}$-Complete
MAP	Σ_2^{P}-Complete	Σ_2^{P}-Complete	$\mathrm{NP}^{\mathrm{PP}}$-Hard

MAP for Bayesian networks

Evidence $E = e$, set of variables Q, rational r, is there an instantiation q for Q such that $p(q|e) > r$?

- NP-Complete for polytrees and for bounded induced-width networks.
  - Difficult comes from reduction of SAT. Pertinence is immediate, because given q, evaluating $p(q|e) > r$ is polynomial.
- $\mathrm{NP}^{\mathrm{PP}}$-Complete in the general case.
  - Pertinence is immediate as the PP oracle is enough to compute $p(q|e)$. Difficult comes from E-MAJSAT.

Belief updating in credal networks

Evidence $E = e$, query $Q = q$, rational r, is is there a distribution that comply with the credal network constraints and such that $\overline{p}(q|e) > r$?

- NP-Complete for polytrees and for bounded induced-width networks
- $\mathrm{NP}^{\mathrm{PP}}$-Complete in the general case (by a reduction from `E-MAJSAT`)

PT and BIW-CN-Pr are NP-Complete

- `PT-CN-Pr`: Belief updating in polytree credal networks.
- `BIW-CN-Pr`: Belief updating in BIW credal networks.

Pertinence of `BIW-CN-Pr` (which ensures pertinence of `PT-CN-Pr`) is immediate, as given a distribution (that is, choosing a vertex of each set of probabilities) we have a Bayesian network updating problem to solve, which is polynomial.

To show hardness we reduce the `MAX-3-SAT` problem to `PT-CN-Pr`. It can be formulated as follows: *Given a set of boolean variables* $\{X_1, \ldots, X_n\}$*, a 3CNF formula with clauses* $\{C_1, \ldots, C_m\}$ *and an integer* $0 \leq k \leq m$*, is there an assignment for the variables that satisfies at least* k *clauses?*

PT and BIW-CN-Pr are NP-Complete

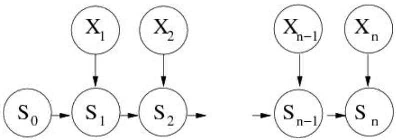


Figure: Polytree used in the network of Theorem.

We are going to show that

$$p(S_n = c) \quad = \quad \begin{cases} 0 & \text{if } c \text{ is satisfied} \\ \frac{1}{m+1} & \text{if } c \text{ is not satisfied} \end{cases}$$

Summing over all categories of S_n, we obtain

$$p(S_n = 0) = 1 - \sum_{c \in \{1,\ldots,m\}} p(S_n = c)$$

and thus $\overline{p}(S_n = 0)$ minimizes this sum.

PT and BIW-CN-Pr are NP-Complete

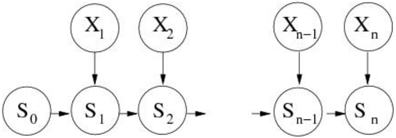


Figure: Polytree used in the network of Theorem.

$p(X_i = x_i)$ and $p(X_i = \neg x_i)$ are in $[0, 1]$. S_i may assume $m + 1$ categories (from 0 to m) and is defined as

$$
\begin{aligned}
p(S_i = c | S_{i-1} = c, x_i) &= 0 \text{ if } x_i \in C_c, \text{ or } 1 \text{ otherwise} \\
p(S_i = c | S_{i-1} = c, \neg x_i) &= 0 \text{ if } \neg x_i \in C_c, \text{ or } 1 \text{ otherwise} \\
p(S_i = c | S_{i-1} \neq c, X_i) &= 0 \text{ for } X_i \in \{x_i, \neg x_i\},
\end{aligned}
$$

for $c \neq 0$. The probability values for $c = 0$ guarantee coherency in probabilities (that is, make the sums equal to 1); note that we include a dummy node S_0 with $p(S_0 = c) = \frac{1}{m+1}$ for all c.

PT and BIW-CN-Pr are NP-Complete

Consider $p(S_n = c)$ for $c \neq 0$. Note that

$$p(S_n = c) = p(S_0 = c) \prod_i p(S_i = c | S_{i-1} = c)$$

where $p(S_i = c | S_{i-1} = c)$ equals to

$$\sum_{X_i \in \{x_i, \neg x_i\}} p(S_i = c | S_{i-1} = c, X_i)\, p(X_i) = \begin{cases} 0 & \text{if } X_i \text{ satisfies } c \\ 1 & \text{otherwise} \end{cases}$$

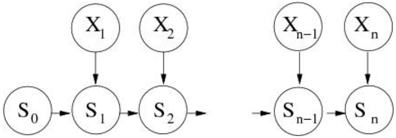


Figure: Polytree used in the network of Theorem.

PT and BIW-CN-Pr are NP-Complete

$$p(S_n = c) \quad = \quad \begin{cases} 0 & \text{if } c \text{ is satisfied} \\ \frac{1}{m+1} & \text{if } c \text{ is not satisfied} \end{cases}$$

Summing over all categories of S_n, we obtain

$$p(S_n = 0) = 1 - \sum_{c \in \{1,\ldots,m\}} p(S_n = c)$$

and thus $\overline{p}(S_n = 0)$ minimizes this sum. The number of unsatisfied clauses is

$$(m+1)(1 - \overline{p}(S_n = 0))$$

Now we just have to compare this number with k.

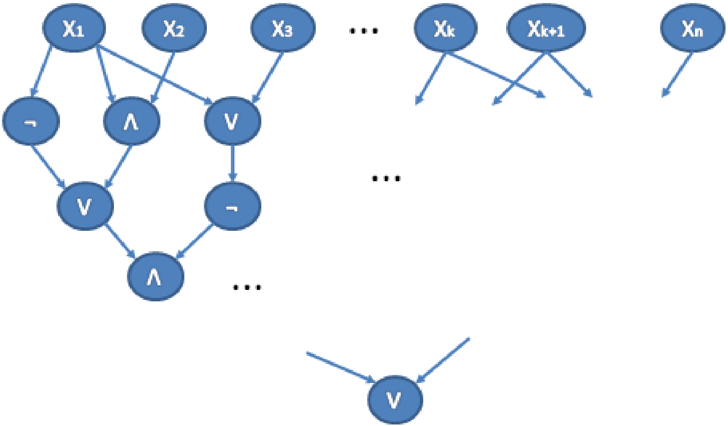
CN-Pr is $\mathrm{NP}^{\mathrm{PP}}$-Complete

- Pertinence is immediate, because when we fix a distribution that comply with the credal network constraints, the PP oracle is enough to verify if $\overline{p}(q|e) > r$. (If credal sets are defined by extreme points, then CCM transformation also shows the same pertinence result as MAP in Bayesian networks is $\mathrm{NP}^{\mathrm{PP}}$-Complete.)
- Hardness comes from a redution of E-MAJSAT. (A transformation from BN-MAP to CN-Pr would suffice, but it uses a non-separate credal network. With the reduction, the result is stronger.)

Given a propositional formula ϕ over the boolean variables $X_1, \ldots, X_n$ and an integer $1 \leq k \leq n$, E-MAJSAT is the task to decide if there is an instantiation to variables $X_1, \ldots, X_k$ such that the majority of all instantiations to $X_{k+1}, \ldots, X_n$ satisfy ϕ.

CN-Pr is NP^{PP}-Complete

Build a credal network that models ϕ.



CN-Pr is $\mathrm{NP}^{\mathrm{PP}}$-Complete

- $X_1, \ldots, X_n$ are nodes without parents such that $X_{k+1}, \ldots, X_n$ have uniform distributions and $X_1, \ldots, X_k$ have vacuous distributions.
- Operators $\vee, \wedge, \neg$ become binary nodes which parents are the elements they connect. Conditional probability distributions are the true-table of the operators.
- Let γ be the only operator without children, x an instantiation to $X_1, \ldots, X_k$, and y an instantiation to $X_{k+1}, \ldots, X_n$.

$$p(\gamma) = \sum_x \sum_y p(\gamma|x, y)p(x)p(y) = \frac{1}{2^{n-k}} \sum_x \sum_y p(\gamma|x, y)p(x).$$

CN-Pr is NP^{PP}-Complete

- When maximizing $\max_p p(\gamma)$ with respect to the vacuous distributions of $X_1, \ldots, X_k$, we have a solution in the extremes and so $p(x) = 1$ for exactly one x (call it as x'). Hence

$$\max_p \frac{1}{2^{n-k}} \sum_x \sum_y p(\gamma|x,y)p(x) = \frac{1}{2^{n-k}} \sum_y p(\gamma|x',y).$$

- $p(\gamma|x',y)$ is one if γ is satisfied by x' and y and zero otherwise (note that this does not involve imprecise probabilities).
- $\max_p p(\gamma) > \frac{1}{2} \iff \sum_y p(\gamma|x',y) > 2^{n-k-1}$, which means that the majority of the instantiations y satisfy ϕ.

Bayesian MAP inference and Credal Belief Updating

- Belief updating in credal networks is solvable by MAP (CCM transformation)
- The opposite direction is also possible: MAP is solvable by belief updating in credal networks with joint queries without changes in topology.

MPE for credal networks

Evidence e, rational r, is there a complete instantiation x for the variables such that $\underline{p}(x, e) > r$?

- Polynomial for polytrees and for bounded induced-width networks
- NP-Complete in the general case

Same complexity of its Bayesian sibling!

Exercises

- Show how to translate a Bayesian network MAP problem to a credal network belief updating inference.
- Prove that credal MPE in separately specified polytrees can be solved by MPE in polytree Bayesian networks.

Agenda

Conclusion

We have discussed

- Basic concepts of Bayesian and credal networks.
- Why credal networks are interesting.
- Which challenges have to be faced.
- Some applications. Hopefully more to come.

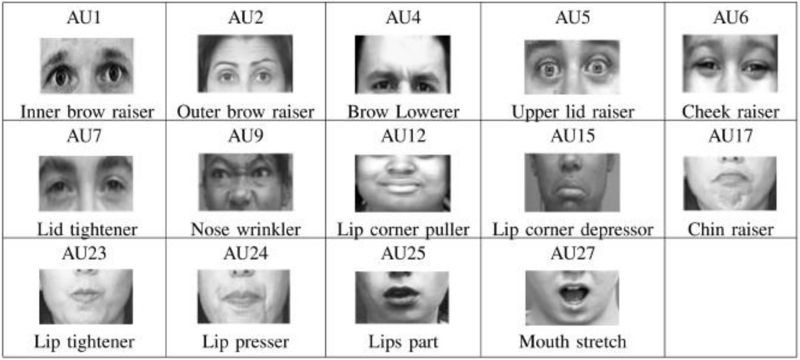
Agenda

Facial expression recognition

- 8000 images from DFAT-504 data set.
- Facial expressions can be defined through Action Units (AUs), which represent muscle contractions.
  - AU1: inner brow raiser
  - AU2: outer brow raiser
  - AU5: upper eyelid raiser
  - AU9: nose wrinkle
  - AU17: chin raiser

Facial expression recognition

Facial action unit coding system:



Parameterization of the SQPN

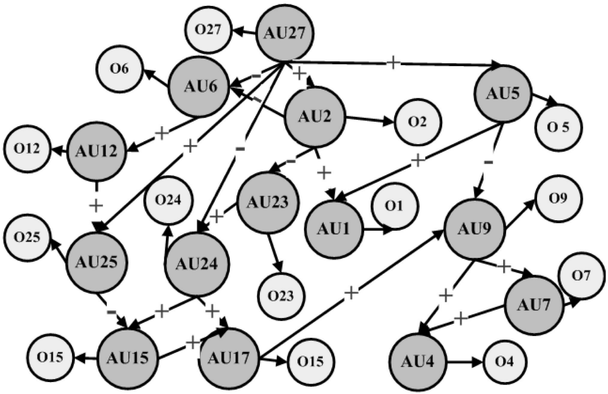
- Parameters of observed nodes are defined by the expert using the errors of the measurement technique.
- Parameters of hidden nodes are learned from data.
  - Data contains 28 columns: 14 measurements from Computer Vision techniques and 14 manually labeled AUs.
  - Prior SQPN and Imprecise Dirichlet Model are employed.

AUs have relations

- Mouth stretch increases the chance of lips apart; it decreases the chance of cheek raiser and lip presser.
- Nose wrinkle increases the chance of brow lowered and lid tightened.
- Eyelid tightened increases the chance of lip presser.
- Lip presser increases the chance of chin raiser.

Facial expression recognition

Semi-qualitative Probabilistic Network:



Inference Approaches

Two approaches are tested:

1. After learning, we perform a query in the credal network to select the distribution of maximum entropy.
   - Then standard Bayesian network belief updating is performed for each AU, given the observations: $p(AU_i|\mathcal{O})$.
   - Main advantage: performance.
2. Inference is performed directly in the credal network, and only cases with interval dominance are analyzed, that is, the maximum probability of AU occurrence (or absence) is less than the minimum of absence (or occurrence). So, we classify only if $\overline{p}(AU_i|\mathcal{O}) \leq \underline{p}(\neg AU_i|\mathcal{O})$ or $\overline{p}(\neg AU_i|\mathcal{O}) \leq \underline{p}(AU_i|\mathcal{O})$.
   - Inference algorithm is slower, but gain is greater.

Facial expression recognition

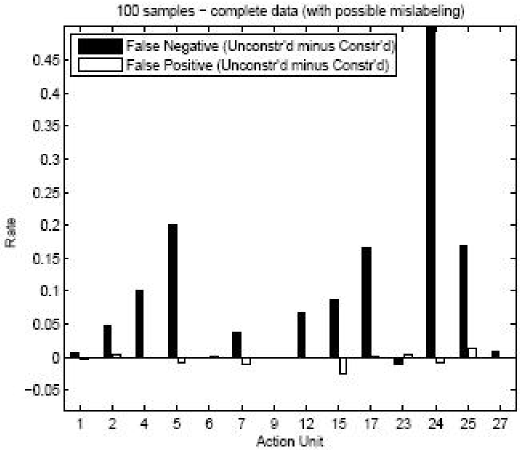


Figure: Benefits of using a prior SQPN

Facial expression recognition

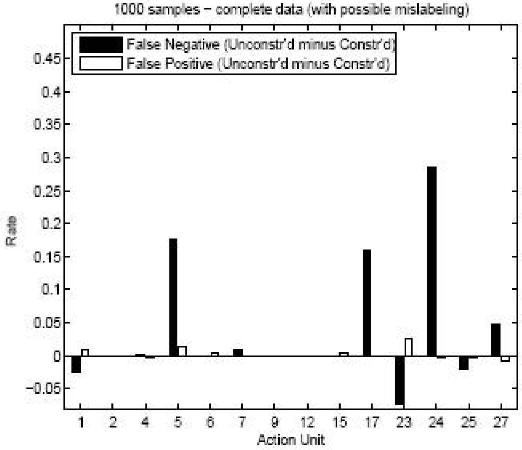


Figure: Benefits of using a prior SQPN

Facial expression recognition

Dataset Size	Maximum Entropy		Interval Dominance	SQPN gain	
	Positive	Negative		Positive	Negative
100	9.8%	-0.1%	49.2%	9.6%	-0.7%
1000	4.0%	0.3%	54.8%	11.4%	0.4%

Table: Percentage of improvement with Maximum entropy and SQPN+IDM approaches against standard maximum likelihood.

Image Segmentation

We worked with over-segmented images and applied Bayesian networks with imprecise probabilities to choose boundary vertices and edges using most probable explanation.

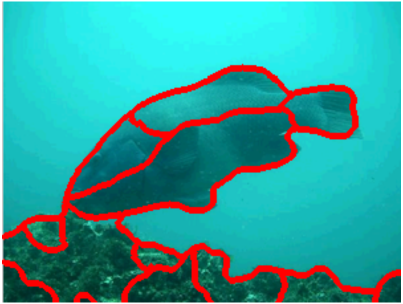
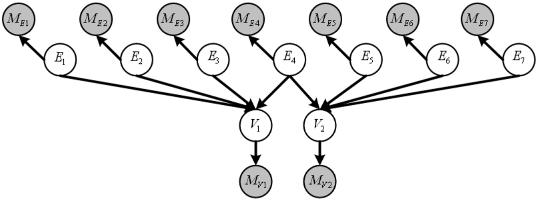


Image Segmentation



- Edges are denoted by E_j and vertices are denoted by V_t. Shadowed nodes are related to computer vision measurements.
- $p(m_{V_t}|v_t) = 0.99$ and $p(m_{V_t}|\neg v_t) = 0.1$. The same idea holds for edge measurements, but with distinct strengths.
- Border should be closed:

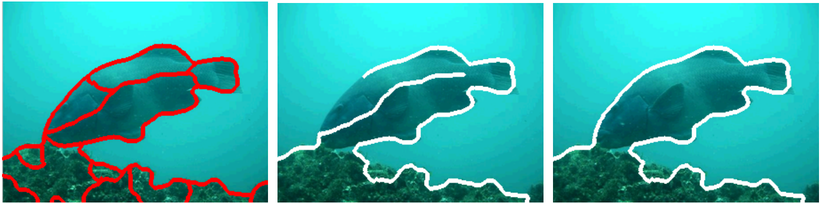
$$p(v_t|pa(V_t)) = \begin{cases} \geq 0.5, & \text{if exactly two parent nodes are true;} \\ 0.3, & \text{if none of the parent nodes are true;} \\ 0, & \text{otherwise.} \end{cases}$$

Inference

Given the SQPN, the goal of image segmentation is achieved by inferring the most probable categories of the variables given the observations (measurements), that is, we look for the categories of E given M_E, M_V that maximize $p(E|M_E, M_V)$. Unfortunately that is very time consuming, but it is much easier to compute categories of E, V that maximize

$$\max_p p(E, V, M_E, M_V) = \prod_t p(V_t|pa(V_t))p(M_{V_t}|V_t) \prod_j p(E_j)p(M_{E_j}|E_j).$$

Image segmentation



(a) Over-segmented (b) Bayesian network (c) Credal network

Image segmentation

(d) Bayesian network

(e) Credal network

Activity recognition

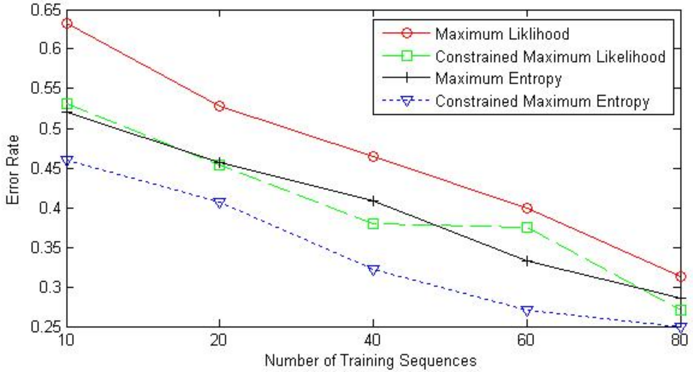
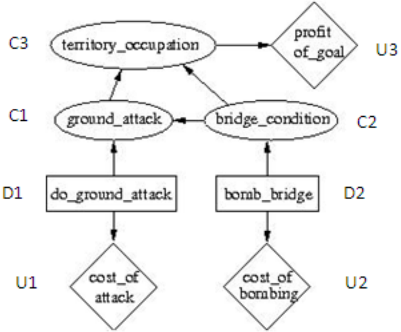


Figure: Comparison between learning ideas

Influence Diagram



$U_3(C3)$ → 2 parameters
P(C3|C1,C2) → 8 parameters
P(C1|C2,D1) → 8 parameters
P(C2|D2) → 4 parameters
$U_1(D1)$ → 2 parameters
$U_2(D2)$ → 2 parameters

Strategy selection is the problem of deciding the action to take at each decision node.

- Shall we do a ground attack?
- Bomb the bridge? Both? None?

Expected utility

Each strategy $s = (d_1, \ldots, d_N)$ (collection of local decisions) has an expected utility:

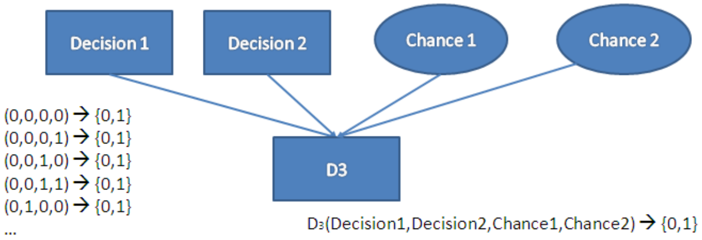
$$EU(s) = \sum_x p(x) \cdot \sum_U f_U(pa(U)),$$

where

$$p(x) = \prod_{x \downarrow C} p(C|pa(C)) \prod_{x \downarrow D} d_D(pa(D)).$$

In words, it is the sum of utility functions for each network configuration x, weighted by the probability of such configuration. We want to maximize the expected utility.

Hard problem



The number of distinct strategies may be huge. Supposing all nodes are binary, then D_3 has 2^4 distinct parent configurations. For each configuration, a decision must be taken. Thus just on D_3 there are 2^{2^4} distinct possibilities.

EBO example

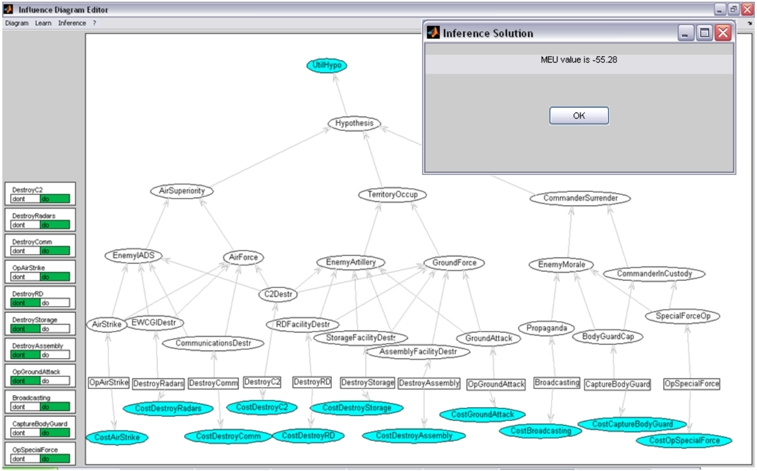


Figure: Approximate inference using *Single Policy Updating*.

EBO example

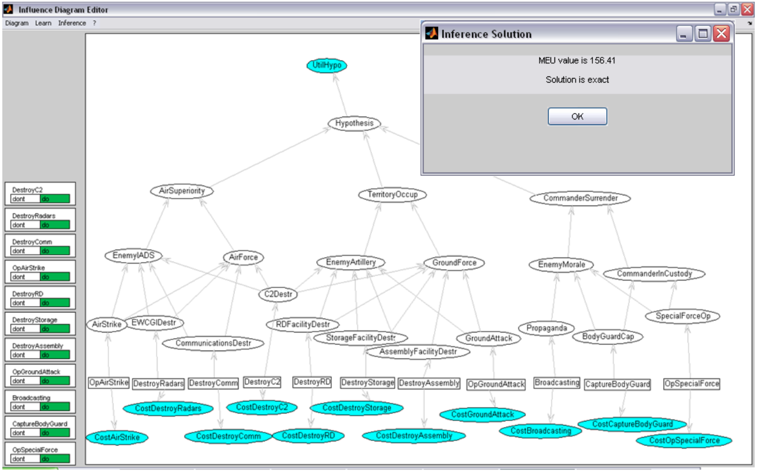
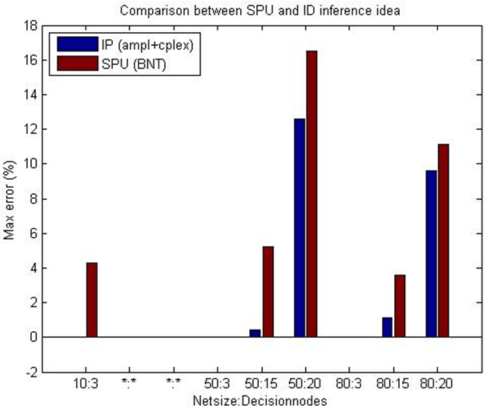


Figure: Exact solution using credal network reformulation.

Results on random diagrams



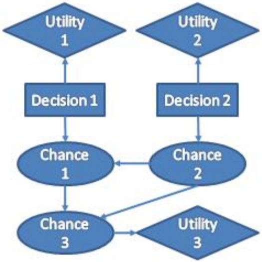
Experimental results

Nodes		Approx.# of	IP			SPU	
Total	Decision	Strategies	Time(sec)	Evals (B&B)	Max.Error(%)	Time(sec)	Max.Error(%)
10	3	2^{17}	0.66	5	0.000	0.10	0.740
20	6	2^{34}	1.73	125	0.000	0.39	2.788
50	10	2^{51}	30.42	4048	0.000	1.62	2.837
60	15	2^{52}	29.77	2937	0.000	2.99	1.964
70	20	2^{54}	125.06	7132	0.000	5.52	3.448
120	25	2^{102}	254.80	15626	0.544	11.58	2.193
120	30	2^{116}	403.13	5617	4.639	13.79	7.281
120	35	2^{120}	578.99	9307	5.983	16.87	11.584

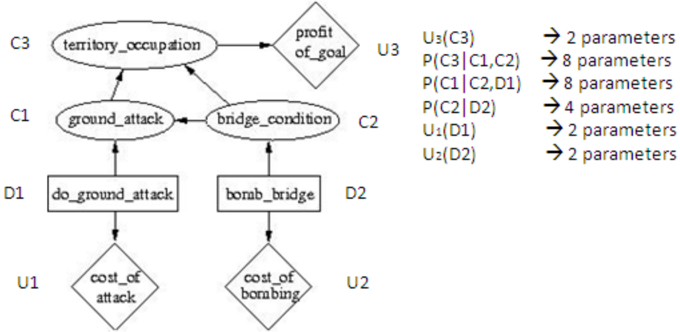
Table: Average results on 30 random influence diagrams.

Influence Diagrams

- Graphical model used for decision making.
- Extends the well known and widely used Bayesian networks with decision and utility nodes (besides chance nodes as in Bayesian nets).
  - Ellipses represent probabilistic dependencies.
  - Rectangles represent decisions.
  - Diamonds represent profits or costs.



Influence Diagrams



Strategy selection is the problem of deciding the action to take at each decision node.

- Shall we do a ground attack?
- Bomb the bridge? Both? None?

Expected utility

Each strategy $s = (d_1, \ldots, d_N)$ (collection of local decisions defined through functions $d_i : \Omega_{PA(X_i)} \to \{0, 1\}$) has an expected utility:

$$EU(s) = \sum_x p(x) \cdot \sum_U f_U(pa(U)),$$

where

$$p(x) = \prod_{x \downarrow C} p(C|pa(C)) \prod_{x \downarrow D} d_D(pa(D)).$$

In words, it is the sum of utility functions for each network configuration x, weighted by the probability of such configuration. We want to maximize the expected utility.

Reformulation

Influence Diagram → credal network

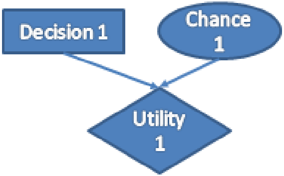
- Utility nodes → chance nodes
- Decision nodes → fully imprecise probability nodes (no constraints besides simplex).
- Chances nodes do not change

Important property

A solution to the new problem is also a solution to the original problem.

Reformulation: utility nodes

Coopers transformation makes utility nodes become binary chance nodes such as $f_U(pa(U))$ is replaced by $p(u|pa(U))$ using a simple normalization to deal with numbers that do not belong to [0,1].



- Before: $U_1(Decision1, Chance1) \in \mathcal{R}$
- Normalize: $[-100, 100] \rightarrow [0, 1]$.
  - $U_1(true, true) = 100 \rightarrow p(u_1|true, true) = 1$.
  - $U_1(true, false) = -50 \rightarrow p(u_1|true, false) = 1/4$.
  - $U_1(false, true) = -100 \rightarrow p(u_1|false, true) = 0$.
  - $U_1(false, false) = 0 \rightarrow p(u_1|false, false) = 1/2$.

Reformulation: objective function

$$EU(s) = \sum_x p(x) \cdot \sum_U f_U(pa(U))$$

$$= \sum_x \sum_U p(x) \cdot f_U(pa(U))$$

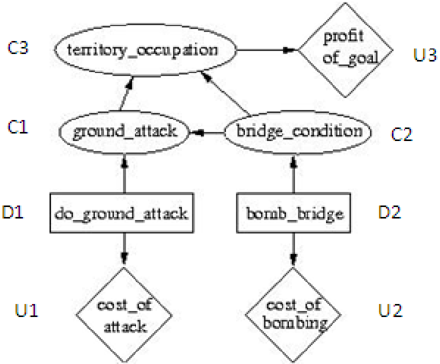
$$= \sum_x \sum_U p(x, u) = \sum_U \sum_x p(x, u) = \sum_U p(u)$$

Inference in credal network

The problem of strategy selection in influence diagrams can be solved with the query $\max_p \sum_U p(u)$ in a credal network.

Example

Create the bilinear programming problem (symbolically) for the follow problem:



$U_3(C3)$ → $p(U3|C3)$

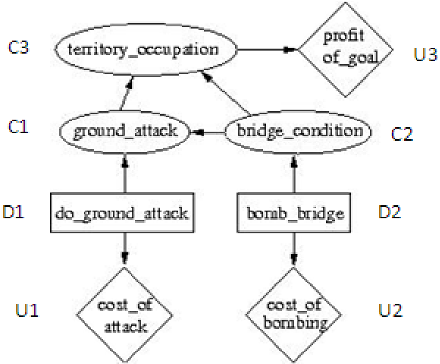
$U_1(D1)$ → $p(U1|D1)$

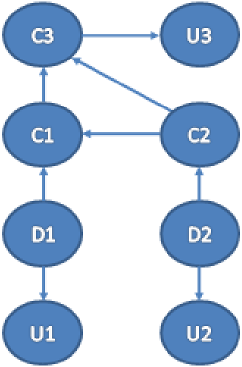
$U_2(D2)$ → $p(U2|D2)$

D_1 → $p(D1)$ (unknown)

D_2 → $p(D2)$ (unknown)

Example: equivalent credal network





$U_3(C3)$	→ $p(U3\|C3)$
$U_1(D1)$	→ $p(U1\|D1)$
$U_2(D2)$	→ $p(U2\|D2)$
D_1	→ $p(D1)$ (unknown)
D_2	→ $p(D2)$ (unknown)

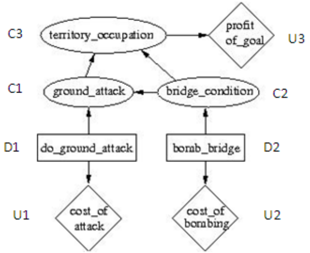
Now we need to find parameters that maximize the expected utility (Parameters of all nodes are known expect for D1 and D2)

Example: bilinear program

Objective:

$$\max_{p} p(u_1) + p(u_2) + p(u_3)$$

Constraints:



$$p(u_1) = \sum_{D_1} p(u_1|D_1)p(D_1)$$

$$p(u_2) = \sum_{D_2} p(u_2|D_2)p(D_2)$$

$$p(u_3) = \sum_{D_2} p(u_3|D_2)p(D_2) \quad \leftarrow \text{ non-linear}$$

$$\forall D_2 \quad p(u_3|D_2) = \sum_{C_2} p(C_2|D_2)p(u_3|C_2)$$

$$\forall C_2 \quad p(u_3|C_2) = \sum_{D_1} p(D_1)p(u_3|C_2, D_1) \quad \leftarrow \text{ non-linear}$$

$$\forall C_2, D_1 \quad p(u_3|C_2, D_1) = \sum_{C_1} p(C_1|C_2, D_1)p(u_3|C_1, C_2)$$

$$\forall C_1, C_2 \quad p(u_3|C_1, C_2) = \sum_{C_3} p(C_3|C_1, C_2)p(u_3|C_3)$$

Going further: linearizing bi-linear terms

In the previous program, for example the term $p(u_3|d_2)p(d_2)$ is non-linear, but one of the factors (in this case, $p(d_2)$ is known to be $\{0,1\}$). Then

- Replace $p(u_3|d_2)p(d_2)$ by y
- Include constraints about y:

$$0 \leq y \leq p(u_3|d_2)$$

$$p(u_3|d_2) + p(d_2) - 1 \leq y \leq p(d_2)$$

- The programs are equivalent!

Obtaining a linear integer program

Using the bilinear transformation previously described, all non-linear terms involve an auxiliary optimization variable and a parameter of the network that can be translated into integer variables if extreme points of credal sets are known.