Beyond tree-shaped credal sum-product networks
Tijn Centen, Cassio de Campos, Thomas Krak, Erik Quaeghebeur

1 Sum-product networks

A sum-product network (SPN) models a joint probability distribution.

Context
• AI and ML applications with a large number of random variables
• Requirement to efficiently calculate probabilities and expectations
• Graphical representations of joint probability distributions have intuition-building appeal

Definition of SPNs
• Built from typically univariate ‘leaf’ distributions \( (p_X^b, p_Y^b, \ldots) \)
• Combined by repeated application of
  – independent ‘product’ operations (\( \emptyset \) nodes);
  – ‘sum’ operations (\( \emptyset \) nodes), which are convex mixtures characterized by weight vectors \( (w_{a,b}, w_{abc,abd}) \);
• Resulting in a joint ‘root’ distribution

Properties of SPNs
• Efficient computation of expectations of ‘query’ functions that factorize over the random variables: \( f(X,Y) = f(X)/f(Y) \)
• An SPN’s underlying structure is an acyclic directed graph (DAG)

2 Credal sum-product networks

A credal sum-product network (CSPN) models a joint credal set.

Definition of CSPNs
• Use sensitivity analysis-style complete product for independent product operations
• Add imprecision by using sets of weight vectors \( (W_{a,b}) \) in sum nodes, resulting in credal sets \( (M_X^{ab}, M_X^{abc}) \)

Limitations of the state of the art (Mauá et al., 2018)
Efficient computation of lower and upper expectations demonstrated only for
• CSPNs with tree-shaped graphs, together with
• indicator-difference ‘focused’ query functions: factorizing functions where
  – all but one factor is an indicator (e.g., \( f(Y) = I_i(Y) \))
  – the non-indicator ‘focus factor’ is a difference of indicators (e.g., \( f(X) = I_i(X) - I_i(X) \));
  – it can be any function for general focused query functions

3 Theoretical results

We can relax the restriction to focused queries and trees if the combination of these relaxations is compatible with efficient inference of lower and upper expectations.

Our contribution: cases beyond the state of the art
1. Query: general factorizing function
   Graph: tree
2. Query: factorizing function with all factors non-negative
   Graph: DAG
3. Query: general focused function
   Graph: DAG for which no cycle edge is part of the paths from root to focus variable leafs (see figure)

4 Experimental results

• An implementation for non-tree-shaped CSPNs has been made
• Credal classification used as the test application
• DAG-shaped CSPNs sometimes perform better, for some datasets and parameters

Graph shapes used
• arbitrary tree (left example)
• class-discriminative tree (middle);
• focus (class) variable leafs below highest product nodes
• class-discriminative DAG (right)

Comparison between shapes, datasets, and evidence types
• Plots below are a function of imprecision (\( \epsilon \)-contamination of weight vectors)
• Show singleton classification accuracy (full lines) and fraction (dashed lines)

‘Gesture’ dataset: impact of shape
‘Diabetes’ dataset: impact of shape
‘Diabetes’ dataset: impact of soft evidence width