# Learning Sum-Product Networks with Maximum Spanning Trees and Concave-Convex Procedure

Renato Lui Geh

Institute of Mathematics and Statistics — University of São Paulo

# Recall LearnSPN

**Algorithm 1** `LearnSPN`: Gens-Domingos structure learning schema

**Input** Set of instances $I$ and scope $X$

**Output** SPN structure learned from $I$ and $X$

1: **if** $|X| = 1$ **then**
2: **return** univariate distribution over $I[X]$
3: Partition $X$ into $P_1, P_2, \ldots, P_m$ st $\forall i, j,\ i \neq j,\ P_i \perp P_j$
4: **if** $m > 1$ **then**
5: **return** $\prod_i \texttt{LearnSPN}(D, P_i)$
6: Cluster $I$ such that $Q_1, Q_2, \ldots, Q_n$ are $I$'s clusters
7: **return** $\sum_i \frac{|Q_i|}{|I|} \texttt{LearnSPN}(Q_i, X)$

**Limitation:** restricted to tree structures.

# Prometheus

## Prometheus : Directly Learning Acyclic Directed Graph Structures for Sum-Product Networks

**Priyank Jaini** PJAINI@UWATERLOO.CA
*University of Waterloo, Waterloo AI Institute, Vector Institute, Ontario, Canada*

**Amur Ghose** AMUR@IITK.AC.IN
*Indian Institute of Technology, Kanpur, India*

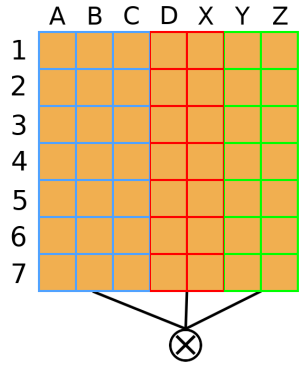
**Pascal Poupart** PPOUPART@UWATERLOO.CA
*University of Waterloo, Waterloo AI Institute, Vector Institute, Ontario, Canada*

### Abstract

In this paper, we present Prometheus, a graph partitioning based algorithm that creates multiple variable decompositions efficiently for learning Sum-Product Network structures across both continuous and discrete domains. Prometheus proceeds by creating multiple candidate decompositions that are represented compactly with an acyclic directed graph in which common parts of different decompositions are shared. It eliminates the correlation threshold hyperparameter often used in other structure learning techniques, allowing Prometheus to learn structures that are robust in low data regimes. Prometheus outperforms other structure learning techniques in 30 discrete and continuous domains. We also describe a sampling based approximation of Prometheus that scales to high-dimensional domains such as images.

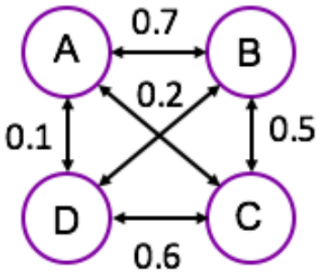
Takes ideas from LearnSPN: clustering and decomposition; but with a twist.

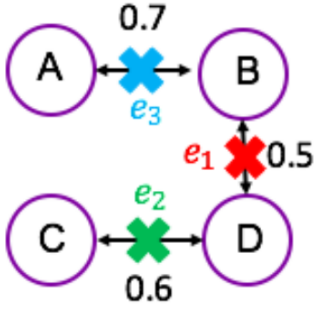
## LearnSPN:

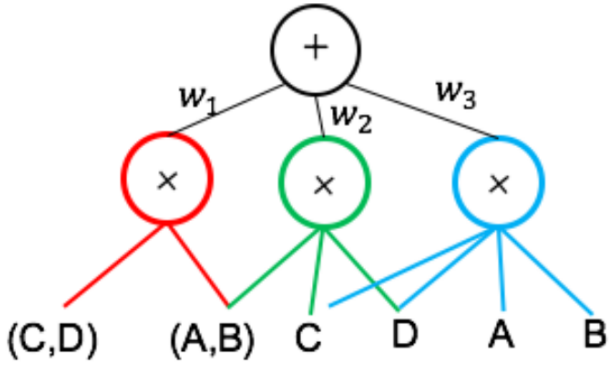



Partitions once greedily.

## Prometheus:



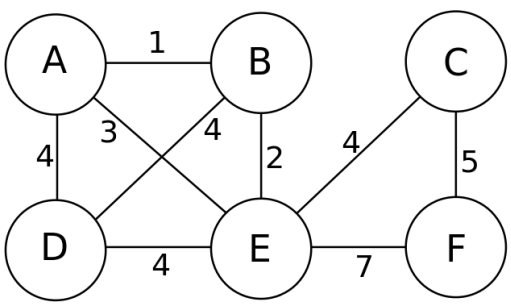



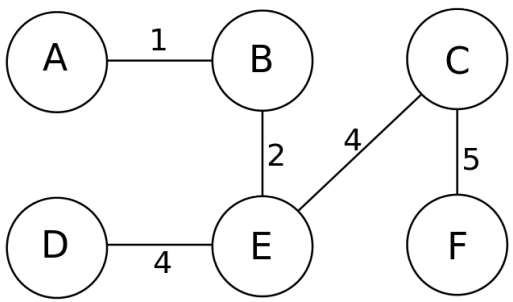

Partitions by Maximum Spanning Tree and reuses repeated scopes.

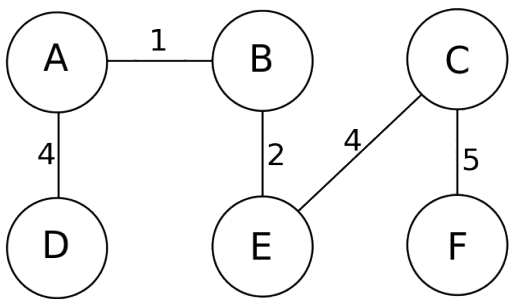
# Maximum spanning trees (MSTs)

**Definition 1 (Minimum spanning tree).**

Let $G = (V, E)$ be an undirected weighted graph. The minimum spanning tree is the subgraph $T \subseteq G$ where $V_T = V_G$ and $E_T \subseteq E_G$ st $\sum_{e \in E_T} w_e$ is minimum, where $w_e$ is the weight associated with edge $e$.




Maximum spanning tree $\equiv$ minimum spanning tree with negated weights

# Graph partitioning

Let $S = \{X_1, X_2, \ldots, X_n\}$ be the scope and $D_S$ be the available data restricted to scope $S$. Let $G$ be a complete undirected weighted graph, where $V_G = S$, and each weight $w_{x,y}$ of edge $e = \{x, y\} \in E_G$ is defined by the Pearson correlation between the two variables $x$ and $y$:

$$w_{x,y} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}}$$

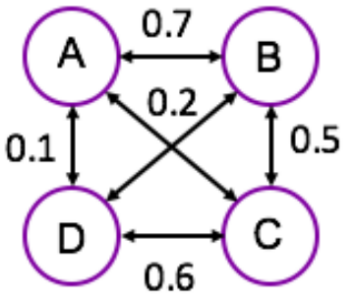
## What we want

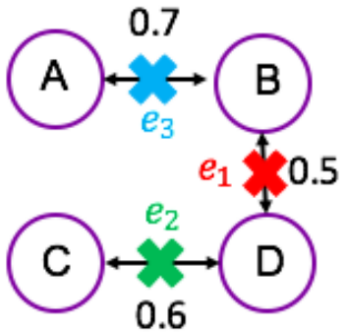
Partition $S$ in such a way that each subset has high correlation between their variables.

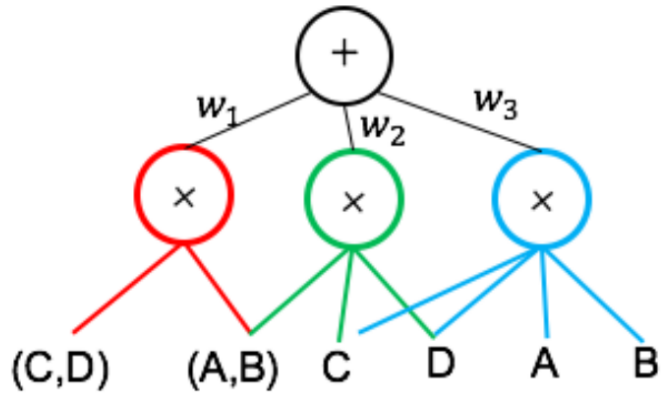
# Prometheus

## Their (Jaini, Ghose, and Poupart 2018) proposal:

Let $T$ be $G$'s maximum spanning tree. Remove the weakest edge from $T$. The components of the resulting graph are the different partitions. Add these partitions to a list $L$ and repeat until there are no edges left to remove.



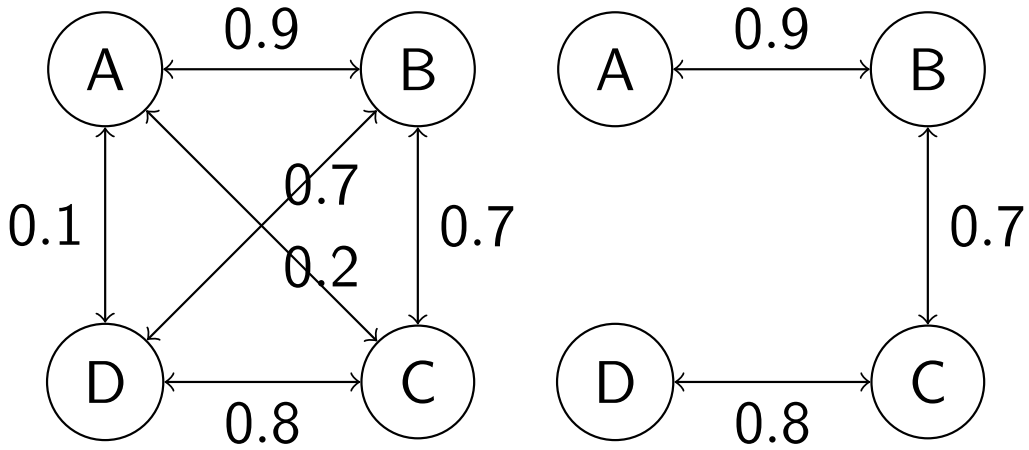




Each element in $L$ is a product decomposition between the partitioned scopes. If such a sub-scope already exists, reuse.

# The problem

An MST could potentially discard highly correlated pairs of variables.




The above graph would generate the same SPN as the last example, and completely discard B and D's high-correlation interaction.

# Our (initial) idea

Instead of partitioning by MST, try other graph partitioning methods.

## Max-flow min-cut

Given two vertices $s$ and $t$, the max-flow problem describes a minimum cut that completely disconnects $s$ from $t$.

## Problem:

How to choose $s$ and $t$? Should $\{s, t\} \subset S$? Different choices could favor certain variables. Should $s$ and $t$ be artificial nodes? If so, should $s$ and $t$ connect to all other vertices in $S$? In this case, either $(s, v) \in E_G$, $\forall v \in S$; or $(v, t) \in E_G$ is a cut. So should $s$ and $t$ only connect to certain nodes? We fall into the same problem of favoring certain variables.

**Too many problems! Try something else?**

# Our (current) idea

Graph partitioning is NP-hard, but we can try a heuristic algorithm.

## Kernighan-Lin algorithm

Bipartition a graph $G$ into two subsets of fixed size $X$ and $Y$ by greedily minimizing the sum of weights that go from $X$ to $Y$ and vice-versa.

## Problem:

If we only consider balanced partitions, the algorithm is fast, but the resulting SPN might be too simple. If we consider partitions of arbitrary lengths, then we have to try every possible bipartition of size in range $[1, \lfloor \frac{|S|}{2} \rfloor]$.

**Ideas?**

# Parameter learning

**Partial derivatives wrt internal node $j$**

$$\frac{\partial S}{\partial S_j} = \sum_{\substack{n \in \mathrm{Pa}(j) \\ n\text{: sum}}} w_{n,j} \frac{\partial S}{\partial S_n} + \sum_{\substack{n \in \mathrm{Pa}(j) \\ n\text{: product}}} \frac{\partial S}{\partial S_n} \prod_{k \in \mathrm{Ch}(n) \setminus \{j\}} S_k$$

**Partial derivatives wrt weight $w_{n,j}$**

$$\frac{\partial S}{\partial w_{n,j}} = S_j \frac{\partial S}{\partial S_n}$$

# Gradient descent

**Recapping gradient descent:**

| | |
|---|---|
| **Generative** | |
| **Soft** | $\Delta w_{n,j} = \eta \frac{\partial S(\mathbf{x}, \mathbf{y})}{\partial w_{n,j}}$ |
| **Hard** | $\Delta w_{n,j} = \eta \frac{c_{n,j}}{w_{n,j}}$ |
| **Discriminative** | |
| **Soft** | $\Delta w_{n,j} = \eta \left( \frac{1}{S(\mathbf{y}, \mathbf{x})} \frac{\partial S(\mathbf{y}, \mathbf{x})}{\partial w_{n,j}} - \frac{1}{S(\mathbf{x})} \frac{\partial S(\mathbf{x})}{\partial w_{n,j}} \right)$ |
| **Hard** | $\Delta w_{n,j} = \eta \frac{\Delta c_{n,j}}{w_{n,j}}$ |

Can we do something different (perhaps better)?

# A Unified Approach for Learning the Parameters of Sum-Product Networks

## A Unified Approach for Learning the Parameters of Sum-Product Networks

**Han Zhao**
Machine Learning Dept.
Carnegie Mellon University
han.zhao@cs.cmu.edu

**Pascal Poupart**
School of Computer Science
University of Waterloo
ppoupart@uwaterloo.ca

**Geoff Gordon**
Machine Learning Dept.
Carnegie Mellon University
ggordon@cs.cmu.edu

### Abstract

We present a unified approach for learning the parameters of Sum-Product networks (SPNs). We prove that any complete and decomposable SPN is equivalent to a mixture of trees where each tree corresponds to a product of univariate distributions. Based on the mixture model perspective, we characterize the objective function when learning SPNs based on the maximum likelihood estimation (MLE) principle and show that the optimization problem can be formulated as a signomial program. We construct two parameter learning algorithms for SPNs by using sequential monomial approximations (SMA) and the concave-convex procedure (CCCP), respectively. The two proposed methods naturally admit multiplicative updates, hence effectively avoiding the projection operation. With the help of the unified framework, we also show that, in the case of SPNs, CCCP leads to the same algorithm as Expectation Maximization (EM) despite the fact that they are different in general.

# A Unified Approach for Learning the Parameters of Sum-Product Networks

Zhao, Poupart, and Gordon 2016 show (amongst other things) that:

- Expectation-Maximization (EM) is equivalent to Concave-Convex Procedure (CCCP);
- CCCP is empirically better than (some other) methods;
- CCCP also converges faster;
- A simple structure + CCCP is able to achieve better results than a more complex structure with no parameter learning.

# Concave-Convex Procedure (CCCP)

**Theorem 2 (Yuille and Rangarajan 2002).**

*Let $E(\vec{x})$ be an energy function with bounded Hessian $\frac{\partial^2 E(\vec{x})}{\partial \vec{x} \partial \vec{y}}$. Then we can always decompose it into the sum of a convex function and a concave function.*

**Theorem 3 (CCCP, Yuille and Rangarajan 2002).**

*Consider an energy function $E(\vec{x})$ of form $E(\vec{x}) = E_{vex}(\vec{x}) + E_{cave}(\vec{x})$ where $E_{vex}(\vec{x})$, $E_{cave}(\vec{x})$ are convex and concave functions of $\vec{x}$ respectively. Then the discrete iterative CCCP algorithm $\vec{x}^t \mapsto \vec{x}^{t+1}$ given by:*

$$\vec{\nabla} E_{vex}\left(\vec{x}^{t+1}\right) = -\vec{\nabla} E_{cave}\left(\vec{x}^{t}\right)$$

*is guaranteed to monotonically decrease the energy $E(\vec{x})$ as a function of time and hence to a minimum or saddle point of $E(\vec{x})$.*

# CCCP for SPN parameter learning

CCCP weight update formula, which is equivalent to EM's:

$$w_{ij} \leftarrow w_{ij} \frac{\partial S}{\partial S_i}(\mathbf{x}) \frac{S_j(\mathbf{x})}{S(\mathbf{x})}$$

where $j$ is a child node of $i$.

- $\frac{\partial S}{\partial S_i}(\mathbf{x})$: partial derivative of the SPN $S$ wrt sub-SPN $S_i$ (i.e. the SPN rooted at $i$);
- $S_j(\mathbf{x})$: the value of $S_j$ given evidence $\mathbf{x}$;
- $S(\mathbf{x})$: value of root SPN $S$.

# References I


Jaini, Priyank, Amur Ghose, and Pascal Poupart (Nov. 2018). "Prometheus : Directly Learning Acyclic Directed Graph Structures for Sum-Product Networks". In: *Proceedings of the Ninth International Conference on Probabilistic Graphical Models*. Ed. by Václav Kratochvíl and Milan Studený. Vol. 72. Proceedings of Machine Learning Research. Prague, Czech Republic: PMLR, pp. 181–192. URL: http://proceedings.mlr.press/v72/jaini18a.html.

Yuille, Alan L and Anand Rangarajan (2002). "The Concave-Convex Procedure (CCCP)". In: *Advances in Neural Information Processing Systems 14*. Ed. by T. G. Dietterich, S. Becker, and Z. Ghahramani. MIT Press, pp. 1033–1040. URL: http://papers.nips.cc/paper/2125-the-concave-convex-procedure-cccp.pdf.

# References II


Zhao, Han, Pascal Poupart, and Geoffrey J Gordon (2016). "A Unified Approach for Learning the Parameters of Sum-Product Networks". In: *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee et al. Curran Associates, Inc., pp. 433–441. URL: http://papers.nips.cc/paper/6423-a-unified-approach-for-learning-the-parameters-of-sum-product-networks.pdf.