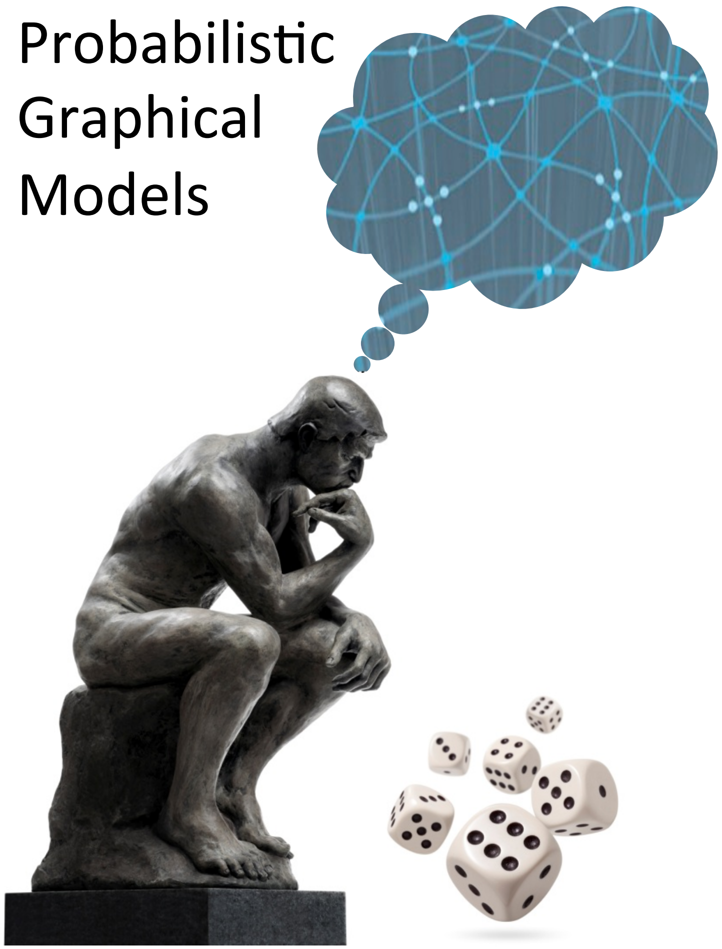
Probabilistic
Graphical
Models



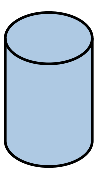
Learning

BN Structure

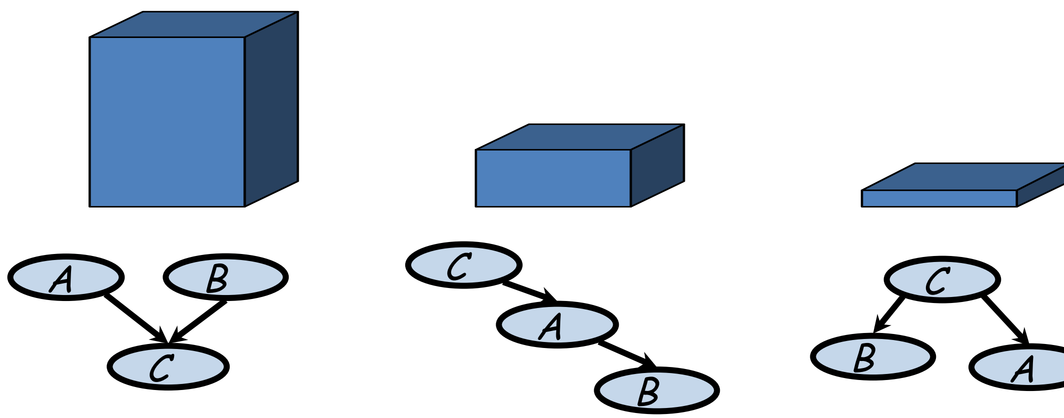
Structure Learning In Trees

Score-Based Learning

Define scoring function that evaluates how well a structure matches the data



A,B,C
<1,0,0>
<1,1,1>
<0,0,1>
<0,1,1>
.
.
<0,1,0>



Search for a structure that maximizes the score

Optimization Problem

Input:

- Training data
- Scoring function (including priors, if needed)
- Set of possible structures

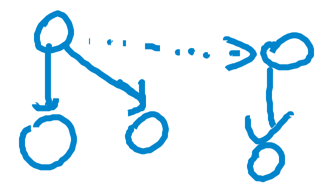
Output: A network that maximizes the score

Key Property: Decomposability

$$\text{score}(\mathcal{G} : \mathcal{D}) = \sum_i \text{score}(X_i \mid \mathbf{Pa}_{X_i}^{\mathcal{G}} : \mathcal{D})$$

Learning Trees/Forests

- Forests
  - At most one parent per variable



- Why trees?
  - Elegant math
  - Efficient optimization
  - Sparse parameterization ⇒ overfit less

    M is small relative to n

Learning Forests

- *p(i)* = parent of X_i, or 0 if X_i has no parent

$$\text{score}(\mathcal{G} : \mathcal{D}) = \sum_i \text{score}(X_i \mid \mathbf{Pa}^{\mathcal{G}}_{X_i} : \mathcal{D})$$

have parent

no parent

$$= \sum_{i:p(i)>0} \text{score}(X_i \mid X_{p(i)} : \mathcal{D}) + \sum_{i:p(i)=0} \text{score}(X_i : \mathcal{D})$$

same for all trees

$$= \sum_{i:p(i)>0} \left(\text{score}(X_i \mid X_{p(i)} : \mathcal{D}) - \text{score}(X_i : \mathcal{D})\right) + \sum_{i=1}^{n} \text{score}(X_i : \mathcal{D})$$

$p(i) \to i$

Improvement over "empty" network

Score of "empty" network

- Score = sum of edge scores + constant

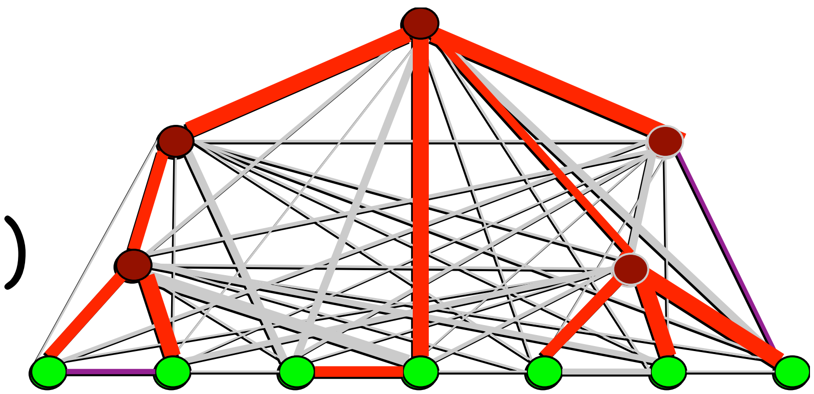
Learning Forests I

- Set $w(i \rightarrow j) = \boxed{Score(X_j \mid X_i) - Score(X_j)}$
- For likelihood score, $w(i \rightarrow j) = \overset{\text{mutual information}}{\boxed{M\,I_{\hat{P}}(X_i; X_j)}}$, and all edge weights are nonnegative
  - ⇨ Optimal structure is always a tree
- For BIC or BDe, weights can be negative
  - ⇨ Optimal structure might be a forest

Learning Forests II

- A score satisfies score equivalence if I-equivalent structures have the same score
  - Such scores include likelihood, BIC, and BDe
- For such a score, we can show $w(i \rightarrow j) = w(j \rightarrow i)$, and use an undirected graph

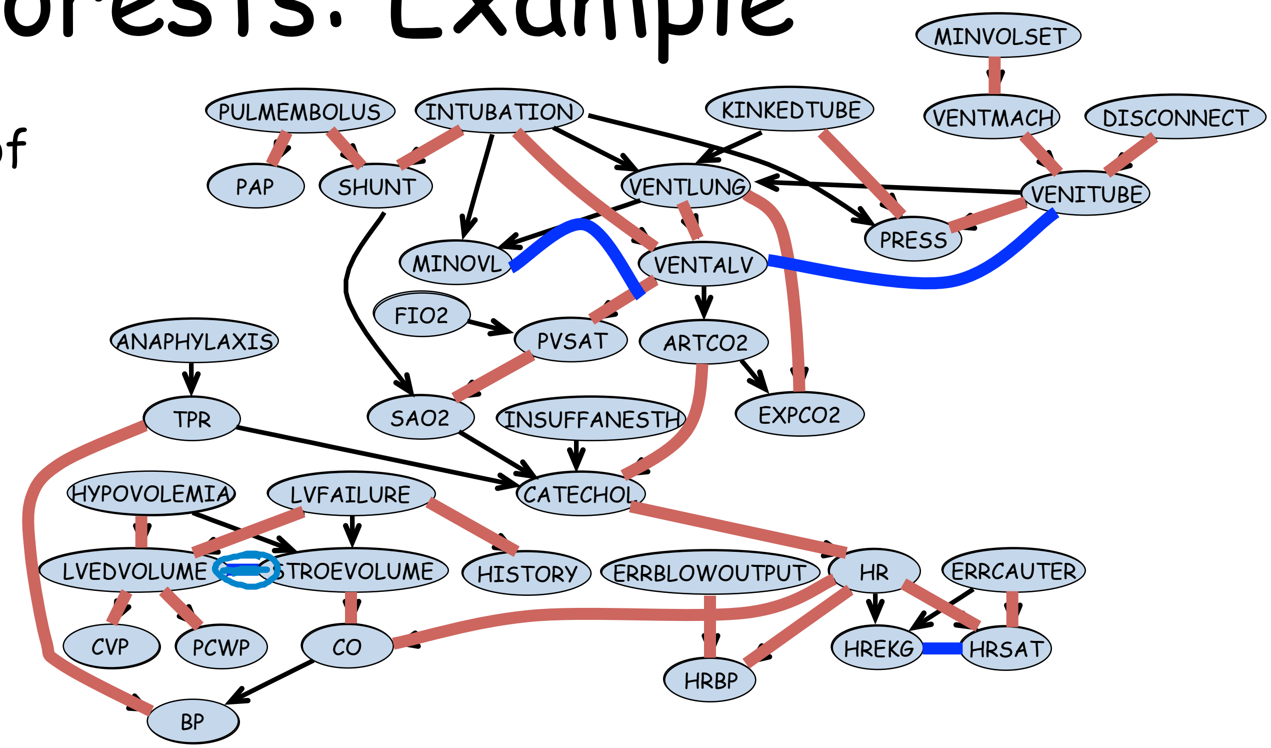
Learning Forests III
(for score-equivalent scores)



- Define undirected graph with nodes {1,…,n}
- Set $w(i,j) = \max[Score(X_j \mid X_i) - Score(X_j), 0]$
- Find forest with maximal weight
  - Standard algorithms for max-weight spanning trees (e.g., Prim's or Kruskal's) in $O(n^2)$ time
  - Remove all edges of weight 0 to produce a forest

Learning Forests: Example

Tree learned from data of Alarm network



Correct edges

Spurious edges

- Not every edge in tree is in the original network
- Inferred edges are undirected – can't determine direction

Summary

- Structure learning is an optimization over the combinatorial space of graph structures
- Decomposability ⇨ network score is a sum of terms for different families
- Optimal tree-structured network can be found using standard MST algorithms
- Computation takes quadratic time