Vers un apprentissage subquadratique pour les mélanges d'arbres

F. Schnitzler[1] P. Leray[2] L. Wehenkel[1]

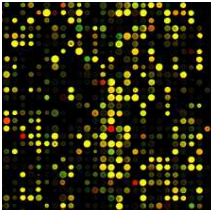
fschnitzler@ulg.ac.be

[1]Université deLiège

[2]Université de Nantes

10 mai 2010

The goal of this research is to improve the learning of bayesian networks in high-dimensional problems.



This has great potential in many applications :

- Bioinformatics
- Power networks

1. Motivation

2. Algorithms

3. Experiments

4. Conclusion

The choice of the structure search space is a compromise.

Sets of all bayesian networks

- Ability to model any density
- Superexponential number of structures
  $\Rightarrow$ Structure learning is difficult
  $\Rightarrow$ Overfitting
- Inference is difficult

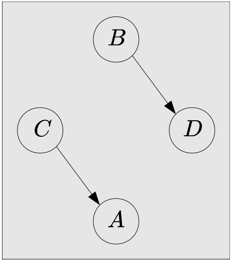
Sets of simpler structures

- Reduced modeling power
- Learning and inference potentially easier

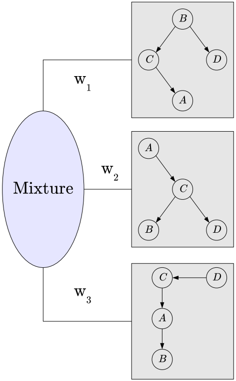
A tree is a graph without cycle where each variable has at most one parent.

Mixtures of trees combine qualities of bayesian networks and trees.

A forest is a tree missing edges :



A mixture of trees is an ensemble method :

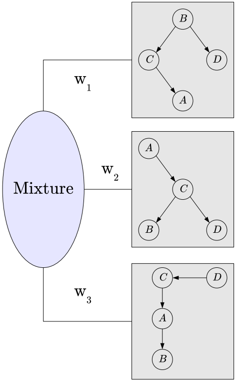


$$P_{MT}(\mathbf{x}) = \sum_{i=1}^{m} w_i P_{T_i}(\mathbf{x})$$

Mixtures of trees combine qualities of bayesian networks and trees.

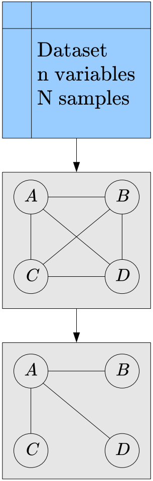
- Several models → large modeling power
- Simple models → low complexity :
  - inference is linear,
  - learning : most algorithms are quadratic.

Quadratic complexity could be too high for very large problems.
In this work, we try to decrease it.



Learning with mixtures of Trees, M. Meila & M.I. Jordan, JMLR 2001.

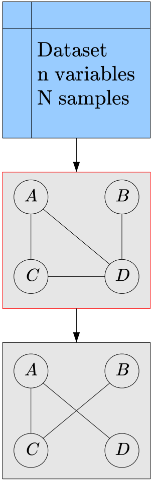
Quadratic scaling is due to the Chow-Liu algorithm.



- Maximize data likelihood
- Composed of 2 steps :
  - Construction of a complete graph whose edge-weight are empirical mutual informations ($\mathcal{O}(n^2 N)$)
  - Computation of the maximum width spanning tree ($\mathcal{O}(n^2 \log n)$)

Approximating discrete probability distributions with dependence trees, C. Chow & C. Liu, IEEE Trans. Inf. Theory 1968.

We propose to consider a random fraction δ of the edges of the complete graph.

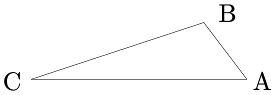


- No longer optimal
- Reduction in complexity (for each term) :
  - Construction of an uncomplete graph : $\mathcal{O}(\delta n^2 N)$
  - Computation of the maximum width spanning tree ($\mathcal{O}(\delta n^2 \log n)$)

Intuitively, the structure of the problem can be exploited to improve random sampling.

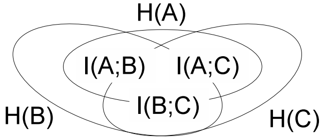
In an euclidian space, similar problems can be approximated by sub-quadratic algorithms. When 2 points B and C are close to A, they are likely to be close as well.

$$d(B, C) \leqslant d(A, B) + d(A, C)$$



Mutual information is *not* an euclidian distance. However the same reasoning can be applied. If the pairs A ;B and A ;C have high mutual information, I(B ;C) may be high as well.

$$I(B; C) \geqslant I(A; B) + I(A; C) - H(A)$$

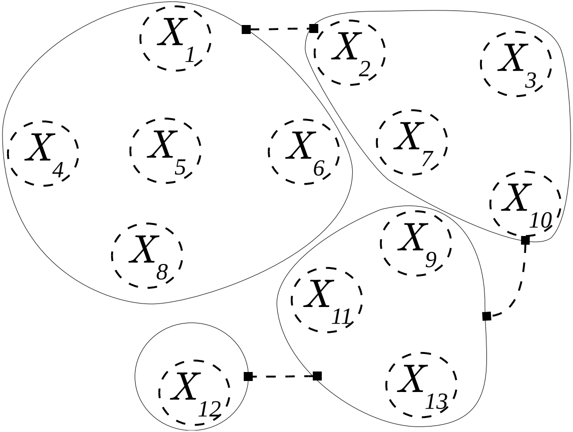


We want to obtain knowledge about the structure.

The algorithm aims at building :

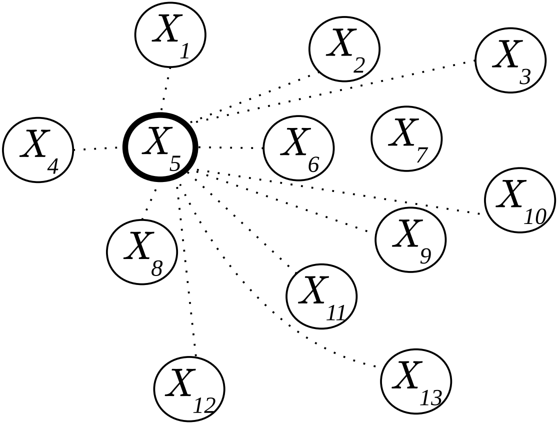
- a set of clusters on the variables,
- relationships between these clusters,

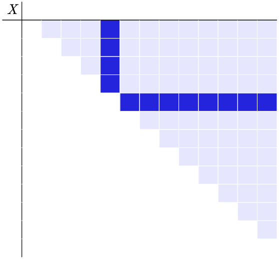
and then exploit it to target interesting edges.



We build the clusters iteratively :

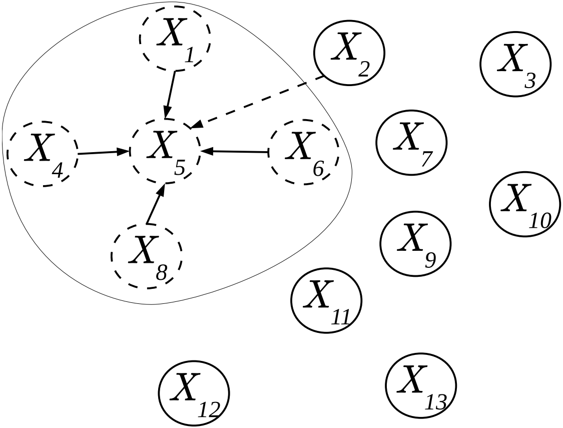
A center (X_5) is randomly chosen and compared to the 12 other variables.

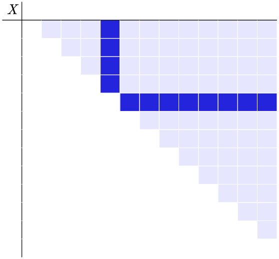




We build the clusters iteratively :

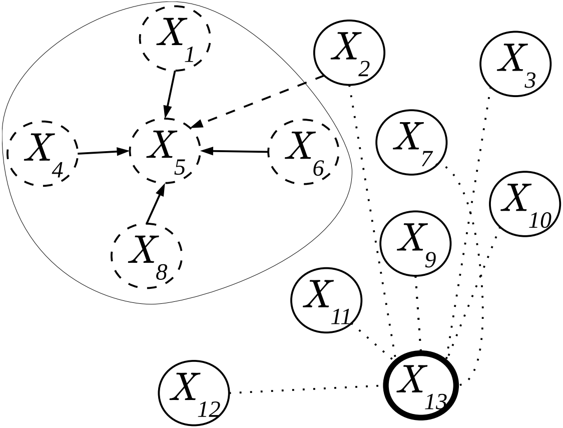
First cluster is created : it is composed of 5 members and 1 neighbour. Variables are assigned to a cluster based on two thresholds and their empirical mutual information with the center of the cluster.

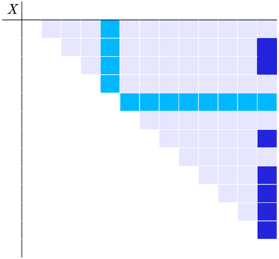




We build the clusters iteratively :

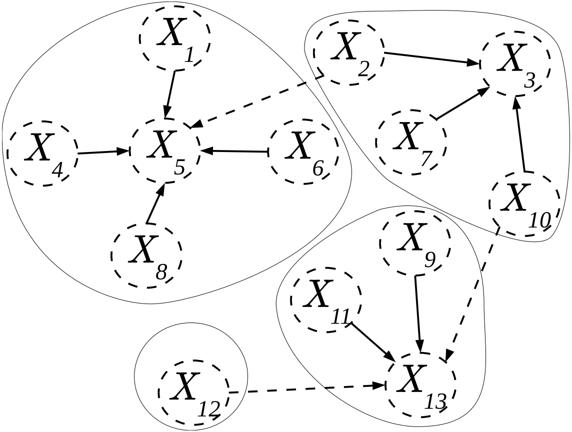
The second cluster is built around X_{13}, the variable the furthest away from X_5. It is only compared to the 7 remaining variables.

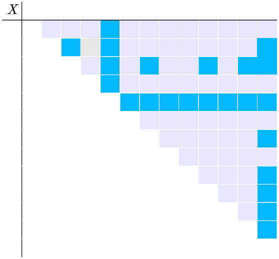




We build the clusters iteratively :

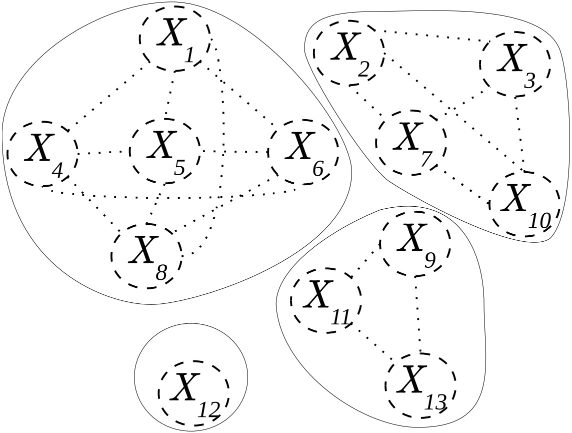
After 4 iterations, all variables belong to a cluster, the algorithm stops.

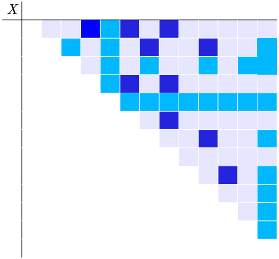




We build the clusters iteratively :

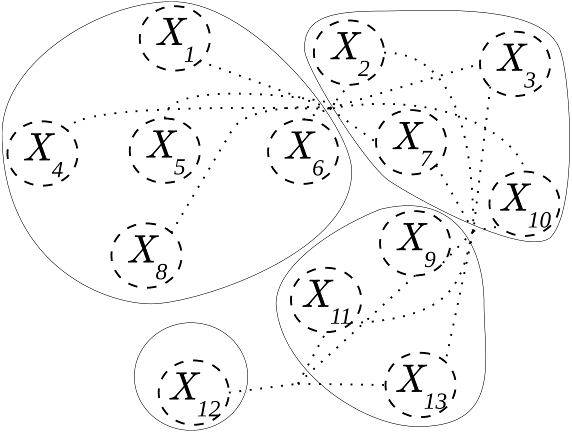
Computation of mutual information among variables belonging to the same cluster.

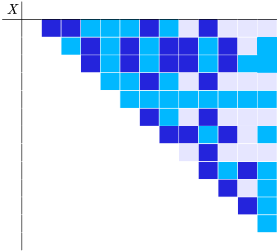




We build the clusters iteratively :

Computation of mutual information between variables belonging to neighboring clusters.





1. Motivation

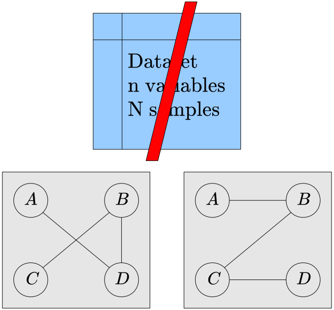
2. Algorithms

3. Experiments

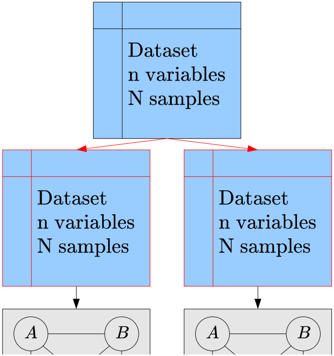
4. Conclusion

Our algorithms were compared against two similar methods.

Complexity reduction :
Random tree sampling ($\mathcal{O}(n)$),
no connection to the data set.


~~Dataset~~
~~n variables~~
~~N samples~~

Variance reduction :
Bagging ($\mathcal{O}(n^2 \log n)$).


Dataset
n variables
N samples

Dataset
n variables
N samples

Dataset
n variables
N samples

Probability Density Estimation by Perturbing and Combining Tree Structured Markov Networks,
S. Ammar and al. ECSQARU 2009.

Experimental settings

Tests were conducted on synthetic binary problems :

- 1000 variables,
- Average on 10 target distributions × 10 data sets,
- Targets were generated randomly.

Accuracy evaluation :

- Kullback-Leibler divergence is too computationally expensive :

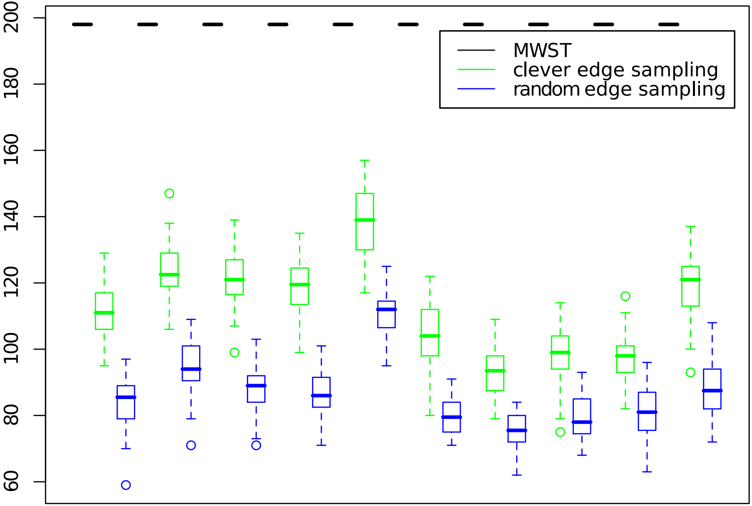
$$D_{KL}(P_t||P_l) = \sum_{\mathbf{x}} P_t(\mathbf{x}) \log \frac{P_t(\mathbf{x})}{P_l(\mathbf{x})}.$$

→ Monte carlo estimation :

$$\hat{D}_{KL}(P_t||P_l) = \sum_{\mathbf{x} \sim P_t} \log \frac{P_t(\mathbf{x})}{P_l(\mathbf{x})}.$$

The proposed algorithm succeeds in improving the random strategy.

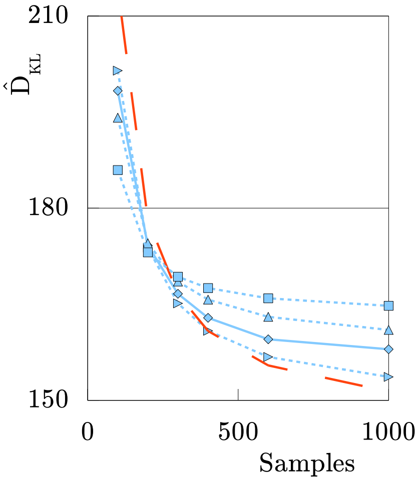
Edges similar to the MWST for single trees of 200 variables :



Variation of the proportion of edges selected

Results for a mixture of size 100 :

- Random edge sampling is :
  - better than the optimal tree for small data sets,
  - worse for bigger sets,
- The more edges considered, the closer to the optimal tree.

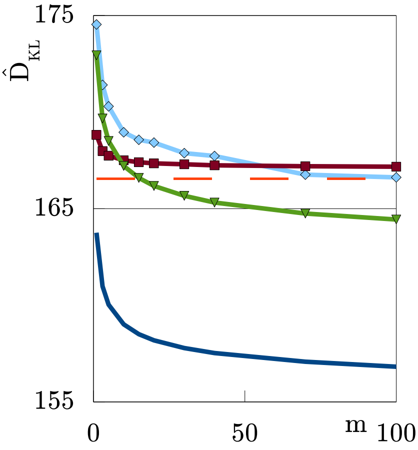


60%, 35%, 20%, 5% (▷, ◊, △, □)

The more terms in the mixture, the better the performance

300 samples :

- More sophisticated methods tend to converge slower,
- Random trees are always worse than an optimal tree,
- Other mixtures outperform CL tree.

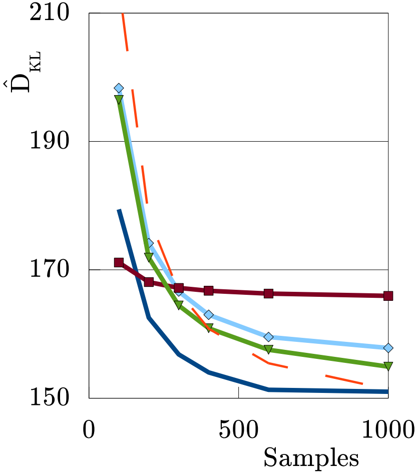


The fewer samples, the (relatively) better the randomized methods.

For high-dimensional problems, data sets will be small.

Results for a mixture of size 100 :

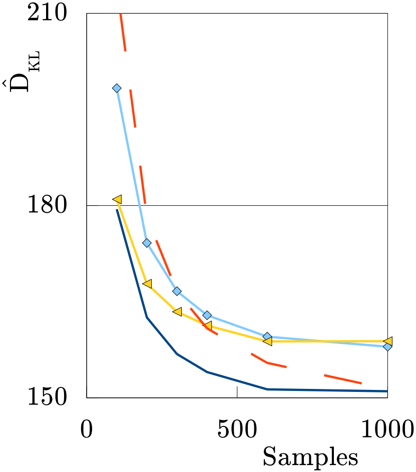
- Random trees (□) are better when samples are few,
- Bagging (-) is better for $N > 50$,
- Clever edge targeting (▽) is always better than random edge sampling (◇).



Methods can also be mixed :

A combination (◁) of bagging (-) and random edge sampling (◇, 35%) :

- Performance lies between base methods.
- Improve bagging complexity.
- The fewer the sample, the closer to bagging.



Conclusion

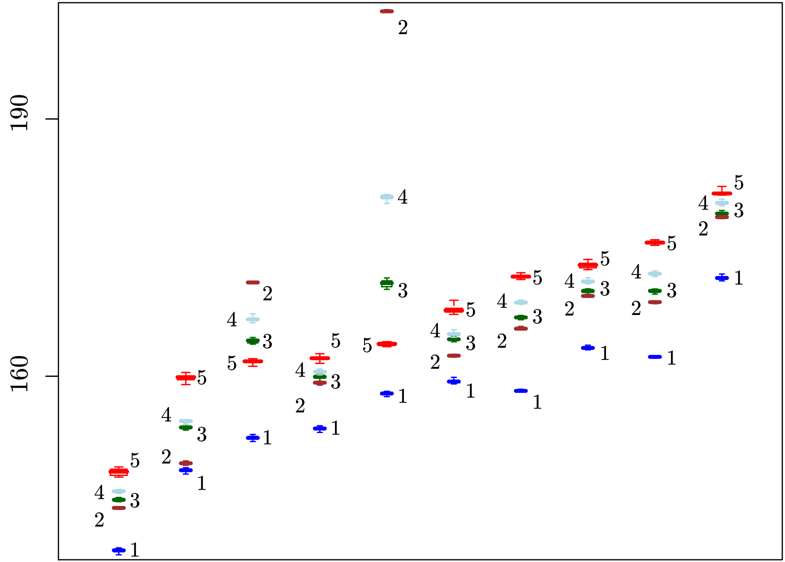
Our results on randomized mixture of trees :

- Accuracy loss is in line with the gain in complexity.
- The interest of randomization increases when the sample size decreases.
- Clever strategies improve results without hurting complexity
  → Worth developing.

Future work :

- Experiment other strategies,
- Include and test those improvements in other algorithms for building MT.

Significance of the curves



Computation time

Rand. trees	Rand. edge sampling	Clever edge sampling	Bagging
2,063 s	64,569 s	59,687 s	168,703 s

Table: Training CPU times, cumulated on 100 data sets of 1000 samples (MacOS X ; Intel dual 2 GHz ; 4GB DDR3 ; GCC 4.0.1)

$$\begin{aligned}
H(B,C,A) &\geqslant H(B,C)\\
H(A)+H(B|A)+H(C|AB) &\geqslant H(B,C)\\
H(A)+(B|A)+H(C|A) &\geqslant H(B,C)\\
H(B)+H(C)+2H(A) &\geqslant H(B,C)+H(B)\\
+H(B|A)+H(C|A) & \qquad +H(C)+H(A)\\
H(B)+H(C)-H(B,C) &\geqslant H(B)+H(A)-H(B,A)\\
& \qquad +H(C)+H(A)-H(C,A)-H(A)\\
I(B;C) &\geqslant I(A;B)+I(A;C)-H(A)
\end{aligned}$$