Learning Conditional Preference Networks: an Approach Based on the Minimum Description Length Principle

Pierre-François Gimenez[1], Jérôme Mengin[2]
[1]CentraleSupélec, University of Rennes, IRISA
[2]IRIT, University of Toulouse

IJCAI'24, August 8th, 2024

CentraleSupélec

IJCAI
JEJU 2024

IRIT

CentraleSupélec

Context

Context: recommendation in e-commerce

- Highly customizable items (e.g., cars, computers, travel) in large combinatorial spaces
- Classical recommendation algorithms are not scalable enough to be usable
- To help users find the product they prefer, we need to modelize their preference over this combinatorial space using a preference model class
- To learn preferences, sales history are generally plentiful

Model class	Recom. query complexity	Expressiveness	Learnable from...
Conditional Lexicographic Preferences	P	Low	Pairwise comparisons, sales history
Bayesian Networks	NP-hard	Maximum	Sales history
Acyclic CP-nets	**P**	**High**	**Pairwise comparisons**

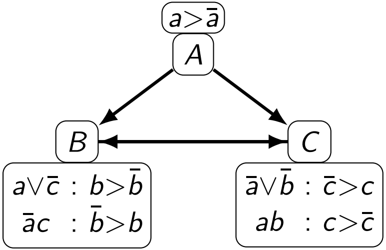
$\Rightarrow$ main contribution: **a learning algorithm for CP-nets from sales history**

CentraleSupélec

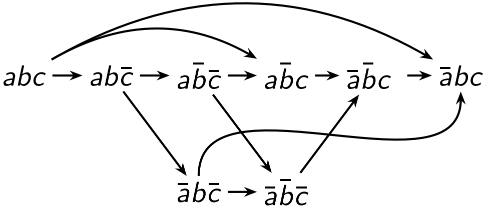
CP-net

CP-net

- A CP-net = a directed graph of features + local preference tables
- Each CP-net is associated to a partial order



A CP-net with 3 variables



Its associated partial order

Learning process

MDL principle: the best model is simple and explains the reality faithfully

- The best model ϕ minimizes $L(\phi) + L(D|\phi)$ where $L(\phi)$ is the size of the model and $L(D|\phi)$ is the size of the data compressed by ϕ

MDL learning of preference model

- Preference models can compute $\mathrm{opt}(u)$ the most preferred extension of a partial vector u
- Example: $\mathrm{opt}(\overline{b}) = a\overline{b}\overline{c}$
- We introduce $\mathrm{code}(o)$: the smallest u such that $\mathrm{opt}(u) = o$
- Example: $\mathrm{code}(a\overline{b}\overline{c}) = \overline{b}$
- We use $\mathrm{code}(\cdot)$ to compress and $\mathrm{opt}(\cdot)$ to uncompress data
- The learning algorithm is a hill-climbing search to maximize $L(\phi) + L(D|\phi)$

Losses

MDL loss equations:

$$L(\phi) = L_{\mathbb{N}}(n) + \sum_{N \in \mathcal{X}} L_{\mathbb{N}}(|Pa(N)|) + \log_2 \binom{n-1}{|Pa(N)|} + |\underline{Pa(N)}| \log_2 |\underline{N}|$$

$$L(D|\phi) = \sum_{o \in D} \left[\; L_{\mathbb{N}}(|\text{code}(o,\phi)|) + \log_2 \binom{n}{|\text{code}(o,\phi)|} + \sum_{X \in \text{code}(o,\phi)} \log_2(|\underline{X}| - 1) \; \right]$$

For the theoretical analysis, we use an approximation of $L(\phi) + L(D|\phi)$, the Normalized Mean Code Length (NMCL):

$$NMCL(\phi) = \frac{1}{n} E_p[|\text{code}(\cdot,\phi)|]$$

Contributions

Algorithm 1: Learning algorithm

Data: a dataset D, an initial CP-net ϕ'

```
score ← L(φ') + L(D|φ'); previous_score ← +∞
while score < previous_score do
    φ ← φ'
    neighbors ← transformations(φ)
    remove non-acyclic graphs from neighbors
    fit CPTs of neighbors from D
    φ' ← argmin_{φ'' ∈ neighbors} L(φ'') + L(D|φ'')
    previous_score ← score
    score ← L(φ') + L(D|φ')
return φ
```

Sample complexity

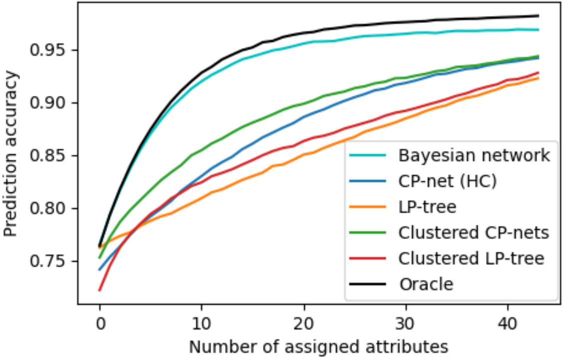
For the family of CP-nets with n nodes and whose nodes have at most k parents:

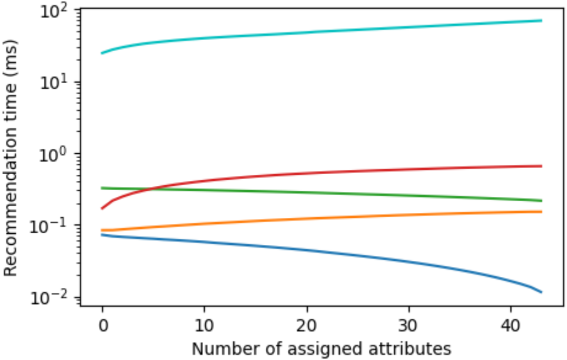
$$N(\delta, \epsilon) = O\left(\frac{d^{2k}}{\epsilon^2}\left(\ln\frac{1}{\delta} + k(\ln d + \ln(n+1))\right)\right)$$

Computational complexity

Finding the acyclic CP-net that minimizes the empirical score over D is NP-complete (reduced from the minimum feedback arc set problem)

Experiments





Experiments on a recommendation task

- Better accuracy than lexicographic preferences, similar speed
- Lower accuracy than Bayesian networks, but much faster
- Clustering helps with the limited expressivity

CentraleSupélec

Experiments and conclusion

Conclusion

- CP-nets can now be used for much more applications
- Low query complexity: they can be used in IoT
- Code is open-source (cf. QR code)

Future works

- Our experiments hint to a interesting connection between Bayesian networks and CP-nets
- This framework can be applied to any preference model class, not just CP-nets!

GitHub

pfgimenez.fr/ijcai24