An exact approach to learning Probabilistic Relational Model

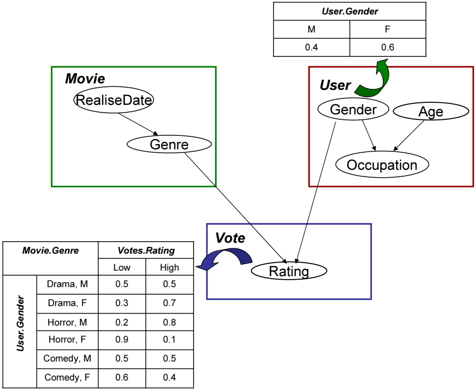
Nourhene Ettouzi[1], **Philippe Leray**[2], Montassar Ben Messaoud[1]

[1] LARODEC, ISG Tunis, Tunisia
[2] LINA, Nantes University, France

International Conference on Probabilistic Graphical Models
Lugano (Switzerland), Sept. 6-9, 2016

Motivations

- **Probabilistic Relational Models** (PRMs) extend Bayesian networks to work with relational databases rather than propositional data



Motivations

- Our goal : **PRM structure learning** from a relational database
- Only few works, inspired from classical BNs learning approaches, were proposed to learn PRM structure

Our proposal

- an **exact approach** to learn (guaranteed optimal) PRMs

Motivations

- Our goal : **PRM structure learning** from a relational database
- Only few works, inspired from classical BNs learning approaches, were proposed to learn PRM structure

Approaches		BNs	PRMs
Score-based	Approximate	✔	✔
	Exact	✔	✘
Constraint-based		✔	✔
Hybrid		✔	✔

Our proposal

- an **exact approach** to learn (guaranteed optimal) PRMs

Motivations

- Our goal : **PRM structure learning** from a relational database
- Only few works, inspired from classical BNs learning approaches, were proposed to learn PRM structure

Approaches		BNs	PRMs
Score-based	Approximate	✔	✔
	Exact	✔	✘
Constraint-based		✔	✔
Hybrid		✔	✔

Our proposal

- an **exact approach** to learn (guaranteed optimal) PRMs

Outline ...

Exact score-based approaches

Main issue : find the **highest-scoring** network

1. decomposable scoring function (BDe, MDL/BIC, AIC, ...)
2. optimal search strategy

Exact score-based approaches

Main issue : find the **highest-scoring** network

1. decomposable scoring function (BDe, MDL/BIC, AIC, ...)
2. optimal search strategy

Decomposability

Let us denote V a set of variables. A scoring function is *decomposable* if the score of the structure, $Score(BN(\mathcal{V}))$, can be expressed as the sum of local scores at each node.

$$Score(BN(\mathcal{V})) = \sum_{X \in \mathcal{V}} Score(X \mid Pa_X)$$

Each local score $Score(X \mid Pa_X)$ is a function of one node and its parents

Exact score-based approaches

Main issue : find the **highest-scoring** network

1. decomposable scoring function (BDe, MDL/BIC, AIC, ...)
2. optimal search strategy

- Spanning Tree [Chow et Liu, 1968]
- Mathematical Programming [Cussens, 2012]
- Dynamic Programming [Singh et al., 2005]
- **A* search** [Yuan et al., 2011]
  - variant of Best First Heuristic search (BFHS) [Pearl, 1984]
  - BN structure learning as a shortest path finding problem
  - evaluation functions g and h based on local scoring function

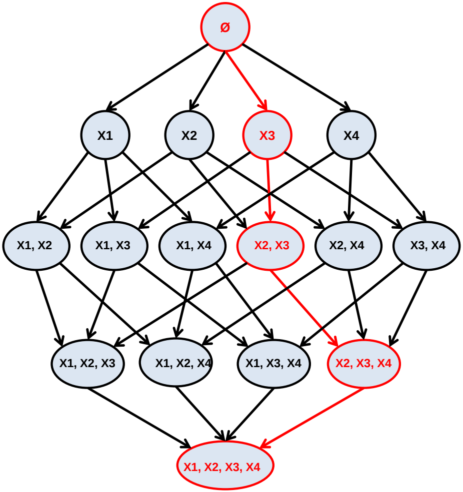
Exact score-based approaches

Main issue : find the **highest-scoring** network

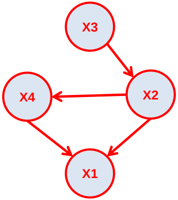
1. decomposable scoring function (BDe, MDL/BIC, AIC, ...)
2. optimal search strategy

- Spanning Tree [Chow et Liu, 1968]
- Mathematical Programming [Cussens, 2012]
- Dynamic Programming [Singh et al., 2005]
- **A* search** [Yuan et al., 2011]
  - variant of Best First Heuristic search (BFHS) [Pearl, 1984]
  - BN structure learning as a shortest path finding problem
  - evaluation functions g and h based on local scoring function

A* search for BNs [Yuan et al., 2011]

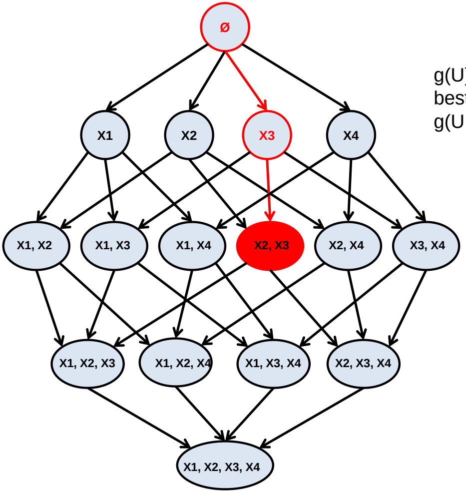


Order graph: the search space



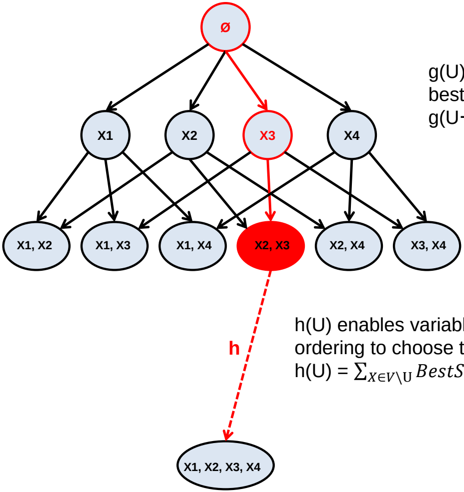
Optimal BN

A* search for BNs [Yuan et al., 2011]



g(U) : sum of edge costs on the best path from the start node to U.
g(U→{U∪X}) = BestScore(X,U)

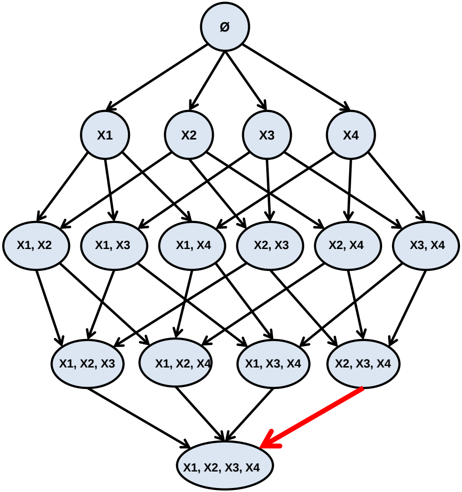
A* search for BNs [Yuan et al., 2011]

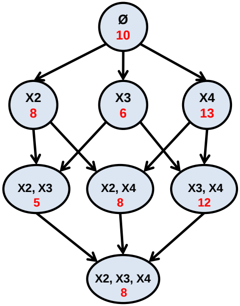


g(U) : sum of edge costs on the best path from the start node to U.
g(U→{U∪X}) = BestScore(X,U)

h(U) enables variables not yet present in the ordering to choose their optimal parents from V.
h(U) = $\sum_{X \in V \setminus U} BestScore(X|V \setminus \{X\})$

A* search for BNs [Yuan et al., 2011]

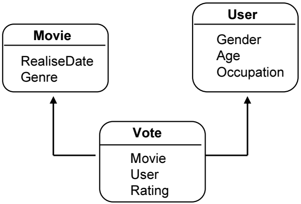




Parent graph of X1

Outline ...

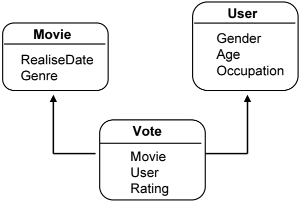
Relational schema $\mathcal{R}$



Definitions

- classes + attributes, $X.A$ denotes an attribute A of a class X
- reference slots = foreign keys (e.g. $Vote.Movie$, $Vote.User$)
- inverse reference slots (e.g. $User.User^{-1}$)
- slot chain = a sequence of (inverse) reference slots
  - ex: $Vote.User.User^{-1}.Movie$: al the movies voted by a particular user

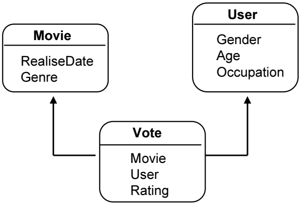
Relational schema $\mathcal{R}$



Definitions

- classes + attributes, $X.A$ denotes an attribute A of a class X
- reference slots = foreign keys (e.g. $Vote.Movie$, $Vote.User$)
- inverse reference slots (e.g. $User.User^{-1}$)
- slot chain = a sequence of (inverse) reference slots
  - ex: $Vote.User.User^{-1}.Movie$: all the movies voted by a particular user

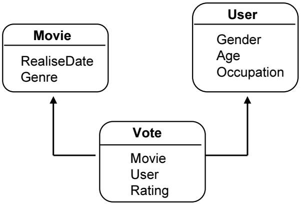
Relational schema $\mathcal{R}$



Definitions

- classes + attributes, $X.A$ denotes an attribute A of a class X
- reference slots = foreign keys (e.g. $Vote.Movie$, $Vote.User$)
- inverse reference slots (e.g. $User.User^{-1}$)
- slot chain = a sequence of (inverse) reference slots
  - ex: $Vote.User.User^{-1}.Movie$: all the movies voted by a particular user

Relational schema $\mathcal{R}$



Definitions

- classes + attributes, $X.A$ denotes an attribute A of a class X
- reference slots = foreign keys (e.g. $Vote.Movie$, $Vote.User$)
- inverse reference slots (e.g. $User.User^{-1}$)
- slot chain = a sequence of (inverse) reference slots
  - ex: $Vote.User.User^{-1}.Movie$: all the movies voted by a particular user

Probabilistic Relational Models

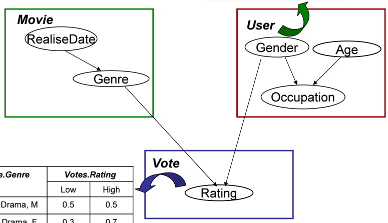
[Koller & Pfeffer, 1998]

Definition

A PRM Π associated to $\mathcal{R}$:

- a qualitative dependency structure $\mathcal{S}$ (with possible long **slot chains** and **aggregation functions**)
- a set of parameters $\theta_{\mathcal{S}}$

User.Gender	
M	F
0.4	0.6



Movie.Genre, User.Gender	Votes.Rating Low	Votes.Rating High
Drama, M	0.5	0.5
Drama, F	0.3	0.7
Horror, M	0.2	0.8
Horror, F	0.9	0.1
Comedy, M	0.5	0.5
Comedy, F	0.6	0.4

Aggregators

- $Mode(Vote.User.User^{-1}.Movie.genre) \rightarrow Vote.rating$
- movie rating from one user can be dependent with the most frequent genre of movies voted by this user

PRM structure learning

Relational variables

- finding new variables potentially dependent with each target variable, by exploring the relational schema and the possible aggregators
- ex: $Vote.Rating$, $Vote.user.user^{-1}.Rating$, $Vote.movie.movie^{-1}.Rating$, ...

⇒ adding another dimension in the search space

⇒ limitation to a given maximal slot chain length

Constraint-based methods

- relational PC [Maier et al., 2010] relational CD [Maier et al., 2013], rCD light [Lee and Honavar, 2016]

PRM structure learning

Relational variables

⇒ adding another dimension in the search space

⇒ limitation to a given maximal slot chain length

Constraint-based methods

- relational PC [Maier et al., 2010] relational CD [Maier et al., 2013], rCD light [Lee and Honavar, 2016]

Score-based methods

- Greedy search [Getoor et al., 2007]

Hybrid methods

- relational MMHC [Ben Ishak et al., 2015]

PRM structure learning

Relational variables

- ⇒ adding another dimension in the search space
- ⇒ limitation to a given maximal slot chain length

Constraint-based methods

- relational PC [Maier et al., 2010] relational CD [Maier et al., 2013], rCD light [Lee and Honavar, 2016]

Score-based methods

- Greedy search [Getoor et al., 2007]

Hybrid methods

- relational MMHC [Ben Ishak et al., 2015]

PRM structure learning

Relational variables

⇒ adding another dimension in the search space

⇒ limitation to a given maximal slot chain length

Constraint-based methods

- relational PC [Maier et al., 2010] relational CD [Maier et al., 2013], rCD light [Lee and Honavar, 2016]

Score-based methods

- Greedy search [Getoor et al., 2007]

Hybrid methods

- relational MMHC [Ben Ishak et al., 2015]

Outline ...

Relational BFHS

- Strategy similar to [Yuan et al., 2011], but adapted to the relational context

Two key points

- search space : how to deal with "relational variables" ?
  ⇒ relational order graph
- parent determination : how to deal with slot chains, aggregators, and possible "multiple" dependencies between two attributes ?
  ⇒ relational parent graph
  ⇒ evaluation functions

Relational BFHS

- Strategy similar to [Yuan et al., 2011], but adapted to the relational context

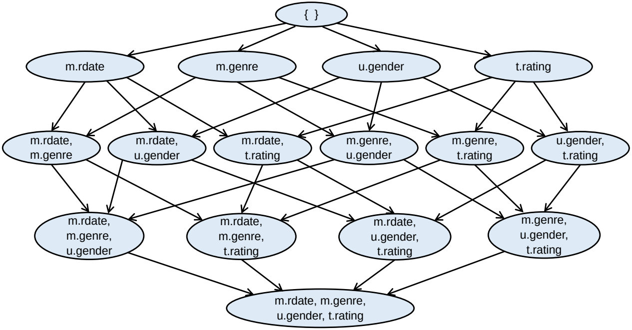
Two key points

- search space : how to deal with "relational variables" ?
  $\Rightarrow$ relational order graph
- parent determination : how to deal with slot chains, aggregators, and possible "multiple" dependencies between two attributes ?
  $\Rightarrow$ relational parent graph
  $\Rightarrow$ evaluation functions

Relational order graph

Definition

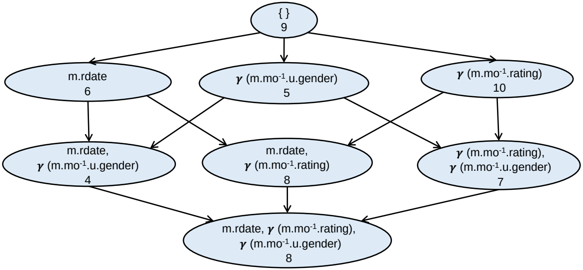
- lattice over $2^{\mathcal{X}.\mathcal{A}}$, *powerset* of all possible attributes
- no big change wrt. BNs



Relational parent graph

Definition (one for each attribute $X.A$)

- lattice over the candidate parents for a given maximal slot chain length (+ local score value)
- the same attribute can appear several times in this graph
- one attribute can appear in its own parent graph, e.g. *gender*



Evaluation functions

Relational cost so far **(more complex than BNs)**

- $g(U \to U \cup \{X.A\})$ = interest of having a set of attributes U (and $X.A$) as candidate parents of $X.A$
- $= BestScore(X.A \mid \{CPa_i / \mathcal{A}(CPa_i) \in \mathcal{A}(U) \cup \{A\}\})$

Relational heuristic function **(similar to BNs)**

$$h(U) = \sum_{X} \sum_{A \in \mathcal{A}(X) \setminus \mathcal{A}(U)} BestScore(X.A \mid CPa(X.A))$$

Evaluation functions

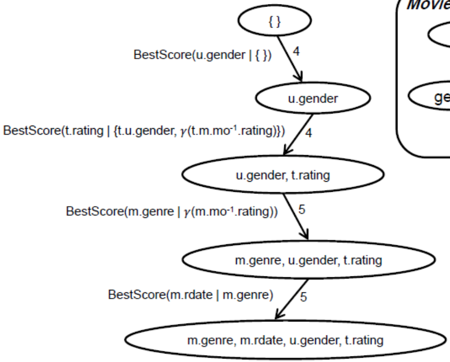
Relational cost so far **(more complex than BNs)**

- $g(U \rightarrow U \cup \{X.A\})$ = interest of having a set of attributes U (and $X.A$) as candidate parents of $X.A$
- $= BestScore(X.A \mid \{CPa_i / \mathcal{A}(CPa_i) \in \mathcal{A}(U) \cup \{A\}\})$

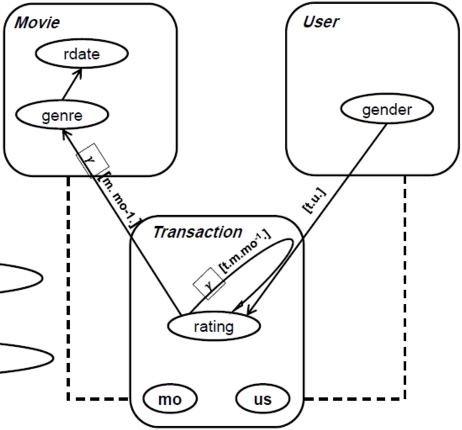
Relational heuristic function **(similar to BNs)**

$$h(U) = \sum_{X} \sum_{A \in \mathcal{A}(X) \setminus \mathcal{A}(U)} BestScore(X.A \mid CPa(X.A))$$

Relational BFHS : example



(a) The shortest path resulting from A* search



(b) An optimal PRM related to the relational schema

Outline ...

Conclusion and Perspectives

Visible face

An **exact approach** to learn optimal PRM, inspired from previous works dedicated to Bayesian networks [Yuan et al., 2011; Malone, 2012; Yuan et al., 2013] whose performance was already proven

To do list

- **Implement** this approach on our software platform PILGRIM
- Provide an **anytime** PRM structure learning algorithm, following the ideas presented in [Aine et al., 2007; Malone et al., 2013] for BNs

Thank you for your attention

Conclusion and Perspectives

Visible face

An **exact approach** to learn optimal PRM, inspired from previous works dedicated to Bayesian networks [Yuan et al., 2011; Malone, 2012; Yuan et al., 2013] whose performance was already proven

To do list

- **Implement** this approach on our software platform PILGRIM
- Provide an **anytime** PRM structure learning algorithm, following the ideas presented in [Aine et al., 2007; Malone et al., 2013] for BNs

Thank you for your attention

Conclusion and Perspectives

Visible face

An **exact approach** to learn optimal PRM, inspired from previous works dedicated to Bayesian networks [Yuan et al., 2011; Malone, 2012; Yuan et al., 2013] whose performance was already proven

To do list

- **Implement** this approach on our software platform PILGRIM
- Provide an **anytime** PRM structure learning algorithm, following the ideas presented in [Aine et al., 2007; Malone et al., 2013] for BNs

Thank you for your attention