A hybrid approach for probabilistic relational models structure learning

Mouna Ben Ishak, Philippe Leray, Nahla Ben Amor

A hybrid approach for Probabilistic Relational Models structure learning

Mouna Ben Ishak[1], Philippe Leray[2], and Nahla Ben Amor[1]

[1] LARODEC Laboratory, ISG, Université de Tunis, Tunisia
[2] DUKe research group, LINA Laboratory UMR 6241, University of Nantes, France

Abstract. Probabilistic relational models (PRMs) extend Bayesian networks (BNs) to a relational data mining context. Just like BNs, the structure and parameters of a PRM must be either set by an expert or learned from data. Learning the structure remains the most complicated issue as it is a NP-hard problem. Existing approaches for PRM structure learning are inspired from classical methods of learning the BN structure. Extensions for the constraint-based and score-based methods have been proposed. However, hybrid methods are not yet adapted to relational domains, although some of them show better experimental performance, in the classical context, than constraint-based and score-based methods, such as the Max-Min Hill Climbing ($MMHC$) algorithm. In this paper, we present an adaptation of this latter to relational domains and we made an empirical evaluation of our algorithm. We provide an experimental study where we compare our new approach to the state-of-the art relational structure learning algorithms.

Keywords: Probabilistic relational model, Relational structure learning, Relational Max-Min Hill Climbing

1 Introduction

Probabilistic relational models (PRMs) [9,16] are an extension of Bayesian networks (BNs) [15] which allow to work with relational database representation rather than propositional data representation. PRMs are interested in manipulating structured representation of the data, involving objects described by attributes and participating in relationships, actions, and events. The probability model specification concerns classes of objects rather than simple attributes.
In order to be used, PRMs have to be constructed either by an expert or using learning algorithms. PRM learning implies finding a graphical structure as well as a set of conditional probability distributions that fit the best way to the relational training data. PRM structure learning remains the most challenging issue, as it is considered as a NP-Hard problem [7]. Only few works have been proposed to learn PRMs [6] or almost similar models [12,13,10] from relational data. Proposed algorithms are inspired from standard BNs learning approaches. Those latter are divided into three families, namely, constraint-based, score-based and hybrid approaches [5]. PRM structure learning approaches are adaptations of either constraint-based or score-based approaches. However, it has been shown that, for BNs, some hybrid approaches provide better experimental results than constraint-based and score-based methods [17]. In this paper we

present a new hybrid algorithm to learn the structure of a PRM from a complete relational dataset. Our proposal is an adaptation of the Max-Min Hill Climbing ($MMHC$) algorithm [17]. We call it Relational Max-Min Hill Climbing algorithm, $RMMHC$ for short. Also, we provide an experimental study where we compare the $RMMHC$ algorithm to state-of-the-art methods. The remainder of this paper is as follows: Section 2 presents useful background and discusses related work. Section 3 details the $RMMHC$ algorithm. Section 4 provides the empirical study. Finally, Section 5 concludes and outlines some perspectives.

2 Background

We start by providing a brief recall on PRMs and presenting methods to learn BNs and PRMs structure from data.

2.1 Probabilistic Relational models

A PRM is defined through two components: a graphical one, a dependency structure defined over the attributes of a relational structure (i.e., an entity-relationship model or a relational schema) containing classes ans class attributes, and a numerical component that quantifies probabilistic dependencies between variables of the relational structure.
Relational model. A relational structure consists of a set of classes $\mathcal{X} \equiv \mathcal{E} \cup \mathcal{R}$, where $\mathcal{E}$ is a set of entity classes and $\mathcal{R}$ is a set of relationship classes. Each $R \in \mathcal{R}$ links a set of entity classes $R(E_i \dots E_j)$. Each $X \in \mathcal{E} \cup \mathcal{R}$ has a set of attributes denoted by $\mathcal{A}(X)$. Every attribute takes on a range of values $\mathcal{V}(X.A)$.
A relational skeleton σ is a partial specification of an instance of a relational structure. It specifies the set of class objects that exist in a domain and the relations that hold between them.

Example 1. An example of a relational structure is depicted in Figure 1(a), with three classes $\mathcal{X} = \{Movie, Vote, User\}$. $\mathcal{E} = \{Movie, User\}$. $\mathcal{R} = \{Vote\}$ The entity class $User$ has three attributes $\mathcal{A}(User) = \{Gender, Age, Occupation\}$. The linked entities of the relationship $Vote$ are $Movie$ and $User$ (Dotted links).
Figure 1(b) shows an example of a relational skeleton for the relational schema of Figure 1(a). It consists of three $User$ objects and five $Movie$ objects. User $user1$ has voted for two movies $M = movie1$ and $movie2$.

Probabilistic model. A PRM $\mathcal{M} = (\mathcal{S}, \Theta)$ brings together the strengths of probabilistic graphical models and the relational representation of data. A dependency structure $\mathcal{S}$ is constructed by adding probabilistic dependencies between class attributes, $\forall X.A \in \mathcal{A}(X)$, there is a set of parents $Pa(X.A) = \{U_1, \dots, U_l\}$. The numerical component is composed of the conditional probability distributions (CPD) of the attributes in the context of their parents in the dependency structure $P(X.A|Pa(X.A))$.
Probabilistic dependencies may be intra or inter classes, this depends on the path that connects the child to its parent. Several paths may be found depending on the way how the relational structure has been traversed. Friedman et al. [6] specify the path between

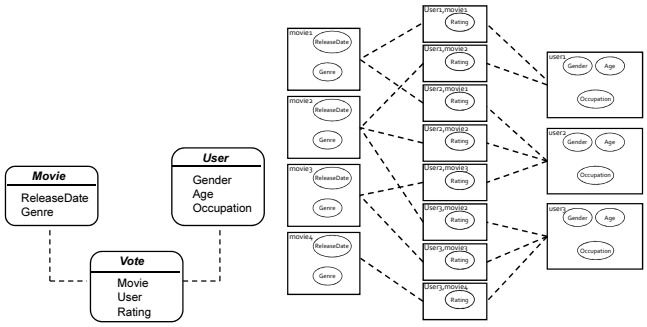


Fig. 1: Example of a relational structure (a) and a relational skeleton (b) for the movie domain inspired from the MovieLens dataset `http://grouplens.org/datasets/movielens/`

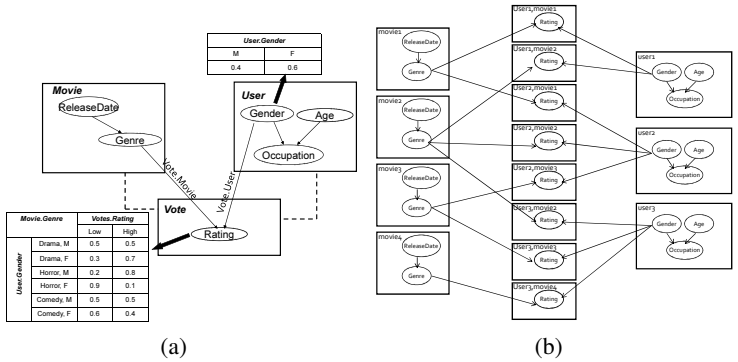


Fig. 2: Example of a probabilistic relational model (a) and a ground graph (b) for the movie domain of Figure 1

the parent and child variables using a slot chain. Heckerman et al. [8] refer to as constraint and Maier et al. [12] call it relational path.

Moreover, depending on the cardinality (i.e., the number of items an entity can participate in a relationship), it is possible for an attribute object to have multiple parents objects (i.e., a $Many$ cardinality). This number of parents is finite but not known in advance and it varies from one object to another. Whereas, there is only one CPD shared among all objects of a given parent attribute $X.A$. To address this issue, the notion of aggregation has been adopted from database theory: *An aggregate* γ takes a multiset of values of some ground type, and returns a summary of it. γ can be the MAX, MIN, MODE, etc.

Each parent U_i has then the form $X.B$ if it is a simple attribute in the same class. $X.K.B$ or $\gamma(X.K.B)$ otherwise, where K is a path and γ is an aggregation function.

Example 2. Figure 2(a) shows a PRM for the relational structure of Figure 1(a). $User.Occupation$ has two parents from the same class $User$. $Vote.Rating$ has two parents: $Vote.User.Gender$ from the $User$ class and $Vote.Movie.Genre$ from the $Movie$ class. $Vote.Movie.genre \rightarrow Vote.rating$ is an example of a probabilistic dependency derived from a path of length one where $Vote.Movie.genre$ is the parent and $Vote.rating$ is the child as shown by Figure 2(a). Also, varying the path length may give rise to other dependencies. For instance, using a path of length three, we can have a probabilistic dependency from $\gamma(Vote.User.User^{-1}.Movie.genre)$ to $Vote.rating$. In this case, $Vote.rating$ depends probabilistically on an aggregate value of all the genres of movies rated by a particular user.

Given a PRM $\mathcal{M}$ and a relational skeleton σ, we can construct a ground Bayesian network (GBN) by applying the probabilistic dependencies specified in $\mathcal{M}$ to the object attributes of σ. The CPD for each $x.A$ is inherited from the CPD $P(X.A|Pa(X.A))$ defined in the PRM. An example of the graphical structure of a GBN is shown by Figure 2(b).

2.2 From BN to PRM structure learning

A wealth of literature has been produced that seeks to understand and provide methods for BN structure learning from data [5]. Some of the proposed approaches have been extended to learn from relational domains. In this section we start by a brief survey on BN structure learning approaches, then we present existing approaches for PRM structure learning.

BN structure learning. is known as an NP-Hard problem [3]. BN structure learning methods are divided into three main families. The first family tackles this issue as a constraint satisfaction problem. Constraint-based algorithms look for independencies (dependencies) in the data, using statistical tests then, try to find the most suitable graphical structure with this information. The second family treats structure learning as an optimization problem. They evaluate how well the structure fits to the data using a score function. So, these Score-based algorithms search for the structure that maximizes this function. The third family presents hybrid algorithms which combine the main features of both techniques, for instance, by using local conditional independence tests and global scoring functions. Tsamardinos et al. [17] proposed the max-min hill climbing ($MMHC$) hybrid algorithm and provided a wide comparative study among several algorithms from three algorithm families, using several benchmarks and metrics (e.g., execution time, SHD measure). Following this study, they showed that their proposal outperforms other algorithms included in the study. The $MMHC$ algorithm consists of two phases:

- The first phase, ensured by the max-min parents and children ($MMPC$) algorithm, aims to find, for each node in the graph, the set of candidate nodes that can be connected to it. At this stage there is no distinction between children and parents nodes and links orientation is not of interest. $MMPC$ discovers the set of candidate parents and children (CPC) for a target variable T. It consists of a raw neighborhood identification step ensured by the $\overline{MMPC}$ algorithm and an additional symmetrical correction step, where $MMPC$ removes from each set $CPC(T)$ each node

X for which $T \notin CPC(X)$. $\overline{MMPC}$ consists of a forward phase where for each variable T of the graph, a set of variables are added to $CPC(T)$, and a backward phase whose role is to remove false dependencies detected in the forward phase. Dependency is measured using an association measurement function such as mutual information or χ^2.

– The second phase allows the construction of the graph $\mathcal{G}$ using the greedy search heuristic constrained to the set of candidate parents and children of each node resulting from the first phase.

PRM structure learning. aims at finding the dependency structure $\mathcal{S}$ for a given relational structure and a relational observational dataset that instantiates this structure. As we have seen in Section 2.1, paths may be arbitrary large and give rise to complicated models. So that a user specified value, a maximum path length (K_{max}), is required to limit the length of possible paths that one can cross in the model. Only few works have been proposed to learn PRM structure from relational data [6,12,13,10]. These latter are inspired from classical methods for BN structure learning.

Friedman et al. [6] proposed the Relational Greedy Hill-Climbing Search (RGS) algorithm. For each path length $k \in \{0, K_{max}\}$ RGS defines a hypothesis space of potential PRM structures (i.e., neighbors) it is willing to consider, using the add_edge, $delete\_edge$ and $reverse\_edge$ operators. Then, it computes the score of each neighbor, and keeps the graph that has the best score, until it reaches a structure that has the highest score in the list of neighbors. As score function, they used a relational extension of the Bayesian Dirichlet (BD) score [4]. In this process, the neighborhood search space could be super-exponential.

Maier et al. proposed two constraint-based approaches. The first is a relational extension of the PC algorithm to learn PRM structure from relational data [13]. Yet, unlike the PC algorithm which is sound and complete the RPC algorithm did not satisfy these criteria. The second approach comes to refine the RPC algorithm [12]. They proposed the relational causal discovery (RCD) algorithm and proved that this approach is sound and complete for causally sufficient relational data. The RCD algorithm performs on two phases. In the first phase, given a maximum path length, RCD starts by providing the set of all potential dependencies. Then continues by removing conditional independences found using conditional independence tests. Because of asymmetry caused by the use of aggregate functions, RCD verifies whether a statistical association is detected between two variables in both directions and it leaves the dependency if a statistical association exists in at least one direction, but omits this information about orientation. In the second phase, RCD determines the orientation of the dependencies discovered previously. Orientation rules are similar to those used by the PC algorithm. In [10] the authors proposed a refined version of the RCD algorithm in term of time complexity and space.

Hybrid approaches combine both techniques and some algorithms, such as the $MMHC$, experimentally outperforms the classical approaches. Yet, no hybrid algorithm has been proposed for PRMs. In the next section, we will provide a new hybrid approach to learn PRM structure from relational data. Our proposal is a relational extension of the $MMHC$ algorithm detailed at Section 2.2, that we refer to as relational max min hill climbing ($RMMHC$).

Algorithm 1 $\overline{RMMPC}$

Require: $schema$: A relational model, $\mathcal{D}$: A database instance
$Current\_Path\_length$: A path length, T: A target attribute
Ensure: CPC: The set of parents and children of T, $CPC(T) = CPC_T^{sym} \cup CPC_T^{asym}$
$Pot_{list} = Generate\_potential\_list(T, Current\_Path\_length)$
% Phase I: Forward
repeat
  $\langle F.assocF \rangle = MaxMinHeuristic(T, CPC(T), Pot_{list})$
  if $assocF \neq 0$ **then**
    if $Current\_path\_length = 0$ OR $does\_Not\_Contains\_Many\_Relationship(F)$ **then**
      $CPC_T^{sym} = CPC_T^{sym} \cup F$
    else
      $CPC_T^{asym} = CPC_T^{asym} \cup F$
    end if
    $CPC(T) = CPC_T^{sym} \cup CPC_T^{asym}$
    $Pot_{list} = Pot_{list} \backslash F$
  end if
until CPC has not changed or $assocF = 0$ or $Pot_{list} = \emptyset$
% Phase II: Backward
for all $A \in CPC(T)$ **do**
  if $\exists S \subseteq CPC, s.t. Ind(A; T|S)$ **then**
    $CPC(T) = CPC(T) \backslash \{A\}$
  end if
end for

3 $RMMHC$: The Relational Max Min Hill Climbing Algorithm

$RMMHC$ preserves the same phases as the $MMHC$ algorithm (cf. Section 2.2). The neighborhood identification phase, ensured by the $RMMPC$ algorithm, handles asymmetry caused by the use of aggregators and leads to a partially oriented neighborhood (cf. Section 3.1). This latter is then used to simplify the global structure identification phase (cf. Section 3.2).

3.1 Relational Max Min Parents and children: $RMMPC$

Neighborhood identification: $\overline{RMMPC}$. The $\overline{RMMPC}$ algorithm aims to find the list of neighbors of a target attribute T, that consists of either children or parents of T, from a set of potential variables. For BNs, $\overline{MMPC}$ does not make a difference between a node in the graph structure and a variable, and the potential set of parents and children of a node T is $\mathcal{V} \backslash T$, where $\mathcal{V}$ is the set of BN nodes. While, in a relational domain, and due to the horizon of crossed paths, the number of potential variables is not fixed. Thus, we have to make the difference between an attribute and a variable:

- An attribute is characterized by its name, domain, a set of possible aggregators and the class that it belongs to. A child is an attribute.
- A variable is characterized by its name, domain, the class that it belongs to, a specific aggregator type and the path that it is derived from. A parent is a variable and its path starts from the class to which the child belongs. This notion is defined in [12] as a canonical dependency.

Algorithm 2 $RMMPC$

Require: $schema$: A relational model, $\mathcal{D}$: A database instance, $Current\_Path\_length$: A path length, T: A target attribute
Ensure: CPC: The set of parents and children of T, $CPC = CPC^{sym} \cup CPC^{asym}$
1: **if** $Current\_Path\_length = 0$ **then**
2: $CPC_T^{sym} = \emptyset, CPC_T^{asym} = \emptyset$
3: $CPC(T) = CPC_T^{sym} \cup CPC_T^{asym}$
4: **end if**
5: $CPC(T) = \overline{RMMPC}(schema, \mathcal{D}, T, Current\_Path\_length)$
6: **for all** $A \in CPC(T)$ **do**
7: **if** $Current\_Path\_length = 0$ **then**
8: $CPC_A^{sym} = \emptyset, CPC_A^{asym} = \emptyset$
9: $CPC(A) = CPC_A^{sym} \cup CPC_A^{asym}$
10: **end if**
11: $CPC(A) = \overline{RMMPC}(schema, \mathcal{D}, A, Current\_Path\_length)$
12: **if** $A \in CPC_T^{sym}$ AND $T \notin CPC_A^{sym}$ **then**
13: $CPC(T) = CPC(T) \backslash \{A\}$
14: **end if**
15: **end for**

Consequently, each parent is a variable, while each target is an attribute. When searching the $CPC(T)$, T is a target attribute. $CPC(T)$ consists of the candidate parents and children of T, and $|CPC(T)|$ depends on the length of the traversed path $k \in \{0 \dots K_{max}\}$. For each value of k, a subset of potential parents and children can be generated. As the final generated $CPC(T)$ list may be very large, we adopt the same strategy as [6] and we proceed by phases. That is, suppose that we want to provide the list of children and parents of each attribute T given a maximum path length k_{max}, the neighborhood identification will be done on $k_{max}+1$ phases. At phase 0, we will search for the set of parents and children of attribute T from the same class as T, at phase 1, we will search for the set of parents and children of attribute T in classes related to T class using paths of length one. At phase 2, we will go through further classes and search for the set of parents and children of attribute T in classes related to T class by traversing paths of length 2 and so on. The neighborhood identification, for one specified value of path length, is described by Algorithm 1. The $Generate\_potential\_list$ method aims to identify the list of potential parents and children of a target attribute T given a path length k. Its result is a set of potential variables of the form $X_T.A$ for intra-class dependencies and $X_T.k.Y.A$ or $\gamma(X_T.k.Y.A)$ for inter-class dependencies.
On the other hand, as some dependencies may require aggregators, there is an inherent asymmetry and this list of candidate dependencies is closely related to the path composition. So that, we propose to divide the neighborhood list, CPC, into two sub-lists. Formally, $CPC(T) = CPC_T^{sym} \cup CPC_T^{asym}$, where:

- CPC_T^{sym} : The set of potential children and parents of target attribute T coming either from the same class as T, with path length equal to 0 or from paths that do not contain any $Many$ relationship.
- CPC_T^{asym} : The set of potential variables coming from the other paths. In this case, A could only be a potential parent of T [7].

As for the standard case [17], $MaxMinHeuristic$ selects the variables that maximize the $MinAssoc$ with target attribute T conditioned to the subset of the currently esti-

Algorithm 3 $RMMHC$

Require: $schema$: A relational model, $\mathcal{D}$: A database instance, k_{max}: $Maximun\_Path\_Length$
Ensure: The local optimal dependency graph $\mathcal{S}$
% Local search
for $Current\_Path\_length = 0$ to k_{max} **do**
  for all T **do**
    $CPC(T) = RMMPC(schema, \mathcal{D}, T, Current\_Path\_length)$
  end for
end for
% Global search
$\mathcal{S} = RGS(schema, \mathcal{D}, CPC)$

mated $CPC(T) = CPC_T^{sym} \cup CPC_T^{asym}$.
Symmetrical correction: $RMMPC$**.** The $RMMPC$ algorithm (Algorithm 2) comes to refine the result of Algorithm 1 by applying a symmetrical correction to the $\overline{RMMPC}$ result. As $CPC(T)$ consists of two subsets, the symmetrical correction depends on the concerned subset.

- For each $A \in CPC_T^{sym}$, we must verify that $T \in CPC_A^{sym}$, otherwise, A has to be removed from CPC_T^{sym}. This symmetrical correction is equivalent to the symmetrical correction of standard $MMPC$.
- For each $A \in CPC_T^{asym}$, we cannot apply the symmetrical correction since the SQL queries involved in such a case are not equivalent and the resulting datasets on which we will apply statistical tests are not the same. However, $\forall A \in CPC_T^{asym}$, A can only be a parent of T. By this way, we can deduce the dependency direction, directly from the first phase of $RMMHC$.

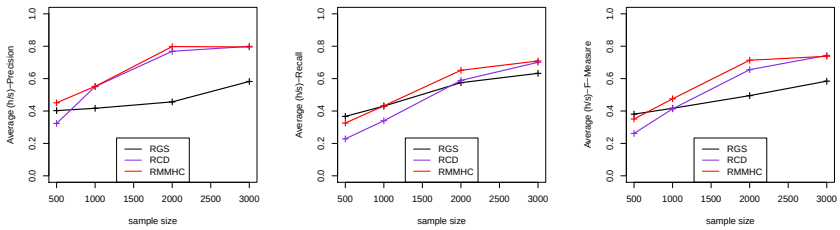
A detailed toy example on the various steps of this phase can be found in [1].

3.2 Global structure identification

The global structure identification is performed using a score-based algorithm only on the set of variables derived from the first local search phase. We choose to work with the RGS procedure, using the relational Bayesian score. In this case, $Pot_K(X.T)$ consists of the CPC list of attribute $X.T$ found on the local search step. As this set contains two subsets, the choice of the operator to be performed during the neighbors generation process depends on the concerned subset:

- For CPC_T^{sym}: each $A \in CPC_T^{sym}$ can be either a child or a parent of $X.T$ so all the operators, namely, add_edge, $delete\_edge$ and $reverse\_edge$ can be tested.
- For CPC_T^{asym}: each $A \in CPC_T^{asym}$ is a potential parent of $X.T$ so only the add_edge and $delete\_edge$ operators can be tested.

The global search step is expensive in term of complexity, since the size of the generated neighborhood may increase rapidly. $RMMHC$ performs the local search procedure in phases until reaching the K_{max} value. The result of this search procedure will be the CPC list of all variables for all path lengths. This partially directed result allows to



(a) Average precision (b) Average recall (c) Average F-measure

Fig. 3: The average values of Precision, Recall and F-Measure with respect to the sample size

further reduce the size of the search space during greedy search. It is used as input to the global search procedure that will be run only one time. The overall process is as presented by Algorithm 3.

3.3 Time complexity of the algorithms

The $MMPC$ algorithm consists of the $\overline{MMPC}$ algorithm of complexity $\mathcal{O}(|Pot_{list}|.2^{|CPC|})$ and an additional symmetrical correction. Thus, its overall complexity is $\mathcal{O}(|Pot_{list}|^2.2^{|CPC|})$. At each iteration of the classical greedy search algorithm, the number of possible local changes is bounded by $\mathcal{O}(\mathcal{V}^2)$, where $\mathcal{V}$ is the number of nodes in the graph [17].
Our $\overline{RMMPC}$ algorithm presents the same steps as for the standard case, augmented with the $Generate\_potential\_list$ procedure which is of complexity $\mathcal{O}(N^k)$, where N is the number of classes and k is the current path length. Thus, its time complexity, at each k value, $k \in \{0 \dots K_{max}\}$ remains equal to $\mathcal{O}(|Pot_{list}|.2^{|CPC|})$. Thus, augmented with the symmetrical correction, the time complexity of the $RMMPC$ algorithm is $\mathcal{O}(|Pot_{list}|^2.2^{|CPC|})$. For RGS, we have to iterate on attributes and for each attribute, we have to iterate on the list of all its potential parents. Let us consider β the number of potential parents that could be reached, then the number of possible local changes is bounded by $\mathcal{O}(\beta.\mathcal{V})$. Note that $\beta = |CPC|$ when the RGS is called after a local search step performed using $RMMPC$ algorithm. In $RMMHC$ algorithm, the local search step has been augmented with an outer loop presenting the current path length to consider at each iteration. Thus the final complexity of the local search is $\mathcal{O}(K_{max}.|Pot_{list}|^2.2^{|CPC|})$.

4 Experiments

We will compare the $RMMHC$ algorithm to the state-of-the-art approaches, namely, the RGS and RCD algorithms (cf. Section 2.2). The RCD is supposed to correct the theoretical problems of RPC and an experimental study on these two approaches can be found in [12]. Thus the RPC algorithm is excluded from the comparative study. In

term of specific implementations, we have re-implemented the RGS algorithm and used our version in the experimental study. We have used the source code of the RCD algorithm available in [3]. As both RCD and $RMMHC$ use statistical independence tests, we have implemented the linear regression test to fit the RCD implementation and we have used it to perform statistical tests during the local search phase of $RMMHC$. To judge conditional independence, we have run both RCD and $RMMHC$ using a threshold $\alpha = 0.05$.

Networks and Datasets. Unlike standard Bayesian networks, where a set of ground truth models (i.e., benchmarks) is available to perform experimentations, there is no such models defined in the context of PRMs. Consequently, we have used our generating process, already described in [2] to generate gold models and relational database instances. We have followed the same experimental protocol as [12] and we have generated relational models containing: 4 entity classes, one less than the number of entities as relationship classes. The number of attributes per class is drawn from $Poison(\lambda = 1) + 1$ and cardinalities are selected uniformly at random. The number of dependencies is from 1 to 15, limited by a maximum path length $= 3$ and at most 3 parents per variable. For each of the previously described networks, we have randomly sampled 5 relational observational complete datasets with 500, 1000, 2000 and , 3000 instances as an average number of objects per class for each.

Evaluation metrics. We have compared the algorithms in term of the quality of reconstruction. using the Precision, Recall and F-score measurement defined in [12].

Experimental results. Figure 3 presents the experimental results in term of Precision, Recall and F-score. RGS presents the worst result for all sample sizes ≥ 1000. $RMMHC$ outperforms RGS and RCD in term of Precision for all sample sizes and it presents the best Recall and F-score values for sample sizes ≥ 1000. For small sample size ($= 500$), $RMMHC$ and RGS have similar results, followed by the RCD algorithm. Figure reft4 shows that for sample size ≥ 1000, beyond 50% of the dependencies retrived by $RMMHC$ are relevant. Figure 3(b) shows that for sample size ≥ 1000, $RMMHC$ was able to find beyond 40% of the relevant dependencies. Both values are increased by raising the sample size.

5 Conclusion

We proposed a first hybrid approach to learn PRMs structure from relational observational data. Our $RMMHC$ algorithm is based on a local search phase that allows to handle asymmetry and leads to a partially oriented neighborhood. This latter is used as input to simplify the global structure identification phase, optimize the search space and consequently enhance the scalability. We have also presented a first comparative study of state-of-the-art relational structure learning approaches and experiments showed that our approach presents good results in term of quality of reconstruction. However, this work is just the beginning for several challenging research tasks.

$RMMHC$ can be improved to deal with more complex structural uncertainty [7], or it can be adapted to learn PRM extensions [14]. Another avenues for future research is

[3] https://kdl.cs.umass.edu/display/public/Relational+Causal+Discovery

combining other theories to learn the model structure [11]. Also, one interesting perspective consists on the use of some prior knowledge, derived from knowledge representation frameworks such as ontologies, as input to the learning process.

References

1. Ben Ishak M.: Probabilistic relational models: learning and evaluation. Ph.D. dissertation, Université de Nantes, Ecole Polytechnique; Université de Tunis, Institut Supérieur de Gestion de Tunis, (2015)
2. Ben Ishak M., Leray P., Ben Amor N.: Probabilistic relational model benchmark generation: Principle and application. Intelligent Data Analysis International Journal (to appear), 615–635, (2016)
3. Chickering D. M., Geiger D., Heckerman D.: Learning Bayesian networks is NP-hard. Technical report, MSR-TR-94-17, Microsoft Research, (1994)
4. Cooper G.F., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Machine Learning, 9, 309–347, (1992)
5. Daly R., Shen Q., Aitken S.: Learning Bayesian networks: approaches and issues. The Knowledge Engineering Review, 26, 99–157, (2011)
6. Friedman N., Getoor L., Koller D., Pfeffer A.: Learning probabilistic relational models. In Proceedings of the International Joint Conference on Artificial Intelligence, pp. 1300–1309, (1999)
7. Getoor L., Koller D., Friedman N., Pfeffer A., Taskar B.: Probabilistic Relational Models. In Getoor, L., and Taskar, B., eds., Introduction to Statistical Relational Learning, MA: MIT Press, Cambridge, (2007)
8. Heckerman D., Meek C., Koller D.: Probabilistic models for relational data, Technical report, Microsoft Research, Redmond, WA, (2004)
9. Koller D., Pfeffer A.: Probabilistic frame-based systems. In Proc. AAAI, pp. 580–587, AAAI Press, (1998)
10. Lee S., Honavar V.: On learning causal models from relational data. In Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI 2016), pp. 3263–3270, (2016)
11. Li X-L., He X-D.: A hybrid particle swarm optimization method for structure learning of probabilistic relational models. Information Sciences, 283, 258–266, (2014)
12. Maier M., Marazopoulou K., Arbour D., Jensen D.: A sound and complete algorithm for learning causal models from relational data. In Proceedings of the Twenty-ninth Conference on Uncertainty in Artificial Intelligence, pp. 371–380, (2013)
13. Maier M., Taylor B., Oktay H., Jensen D.: Learning causal models of relational domains. In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, pp.531–538, (2010)
14. Marazopoulou K., Maier M., Jensen D.: Learning the structure of causal models with relational and temporal dependence. In Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence, pp. 572–581, (2015)
15. Pearl J.: Probabilistic reasoning in intelligent systems. Morgan Kaufmann, San Franciscos, (1988)
16. Pfeffer A. J.: Probabilistic Reasoning for Complex Systems. Ph.D. dissertation, Stanford University, (2000)
17. Tsamardinos I., Brown L. E., Aliferis C. F.: The max-min hill-climbing Bayesian network structure learning algorithm. Machine Learning, 65, 31–78, (2006)