[WORK DOCUMENT] Towards interactive causal relation discovery driven by an ontology

Mélanie Munch, Juliette Dibie-Barthelemy, Cristina Manfredotti, Pierre-Henri Wuillemin

Towards interactive causal relation discovery driven by an ontology

Melanie MUNCH[1], Juliette DIBIE[1], Cristina MANFREDOTTI[1], and Pierre-Henri WUILLEMIN[2]

[1] UMR MIA-Paris, AgroParisTech, INRA, University of Paris-Saclay, 75005 Paris, France

[2] Sorbonne University, UPMC, Univ Paris 06, CNRS UMR 7606, LIP6, 75005 Paris, France

Abstract. In complex domains, users need to be able to find causal relations between the different attributes that compose them. Our work offers a way to help users to uncover such relations by combining the representativity of ontologies and the flexibility of probabilistic relational models, and provides them with an interactive and iterative process in order to validate or modify the obtained results.

Keywords: Ontology, Probabilistic Relational Model, Causal Relation

1 Introduction

While analyzing complex domains (e.g. in biology), users[3] might need to be able to draw dependencies between the different attributes of these domains. More specifically, they might want to be able to formulate and verify hypothesis such as the influence of one parameter on another (e.g. to check if a person's birthplace has a causal influence on his/her place of study).

In this paper we present a new approach for discovering causal relations between data in a knowledge base guided by expert knowledge, represented by an ontology and an assumption about the possible causal links that exist.

We present an interactive and iterative method to help the user who has formulated such hypothesis. It guides him through three different steps: a first, based on gathering the information present in the knowledge base; a second, that learns the probabilistic dependencies between the selected data; and a third that helps analyzing the causal relations that have emerged from this information.

We propose to combine ontologies with probabilistic relational models (PRMs) and their essential graphs (EGs). PRMs extend Bayesian networks (BNs) with the notion of class of relational databases [15]. They allow to better express the relations between the different attributes and offer to the user a better readability during the verification of the results. An EG is a semi-directed graph to which multiple BNs can correspond that allows to identify arcs between variables. An EG allows the user to deduce causal dependencies between the attributes.

[3] In this article we denote by *user* all experts on the domain of study.

The paper is structured as follows. Sect. 2 presents the background on PRM and related works. Sect. 3 presents our approach to learn a PRM from an ontology. Sect. 4 presents our experiments on DBPedia. Sect. 5 concludes this paper.

2 Background

2.1 BNs and PRMs

PRMs extend BNs with the notion of class of relational databases. A BN is the representation of a joint probability over a set of random variables that uses a Directed Acyclic Graph (DAG) to encode probabilistic relations between variables. However, in the case of numerous random variables with repetitive patterns, it cannot efficiently represent every probabilistic relations.

PRMs extend the BN representation with a relational structure between potentially repeated fragments of BN called classes [15]. They define the high-level, qualitative description of the structure of the domain and the quantitative information given by the probability distribution over the different attributes [5]. A **class** is defined as a DAG over a set of attributes. These can be inner attributes or attributes from other classes referenced by so-called **reference slots**. The high level structure of a PRM (i.e. its **relational schema**) describes a set of classes C, associated with attributes $A(C)$ and reference slots $R(C)$. A slot chain is defined as a sequence of reference slots that allows one to put in relation attributes of objects that are indirectly related. The probabilistic models are defined on the low level structure (i.e. at the class level) over the set of inner attributes, conditionally to the set of outer attributes and represent generic probabilistic relations inside the classes. This is the **relational model** of the PRM. Classes can be instantiated for each specific situation. A **system** in a PRM provides a probability distribution over a set of instances of a relational schema [16] and, once instantiated, is equivalent to a BN.

2.2 Learning PRMs

Learning a PRM consists of two tasks: (1) defining the **relational schema** that represents the high level layer where all the classes are defined and their attributes without any probabilistic dependencies; (2) defining the **relational model** that represents the probabilistic dependencies over the previously defined relational schema. Due to these multiple layers, the number of free parameters is high and the target model is not unique: selecting one requires making subjective choices.

If learning both the relational schema and the relational model is a complex problem, it can be eased by learning the PRM with a given relational schema. This can be compared to learn a BN since we only have to learn the probabilistic dependencies [5]. Learning the relational model can be achieved by different BN learning algorithms such as Greedy Hill Climbing[4]. In our approach we focus on

[4] This is a standard well known method for learning BN. It is described alongside other methods in [11].

statistical methods based on the frequency evaluation of the different values: the more an attribute presents a value in the database, the more likely this value is.

As for the first task, the construction of the relational schema can be guided by an ontology and its semantic knowledge, which reduces the number of free parameters for the learning. A PRM can then be learned combining the semantic knowledge of the ontology and a statistical learning method [10].

2.3 Essential Graph

An instantiated system of a PRM is equivalent to a BN. As a consequence, alongside the construction of the PRM, we also learn an Essential Graph (EG) An EG is a semi-directed graph associated to a BN and composed of edges and oriented arcs. They both share the same skeleton, but the orientation of the EG's edges can vary. If the orientation of an edge is the same for all the Markov equivalent graphs of the BN, this edge is also oriented in the EG; if not, the edge remains unoriented. All directed edges in the EG are called essential arcs [8]. The EG expresses whether an orientation between two nodes can be reversed without modifying the probabilistic relations encoded in the graph. It is useful when presenting results to the user as it helps him visualizing the causal relations learned: since the model has been learned with causal constraints, if an edge is oriented in the EG, it means that it represents a causal dependence.

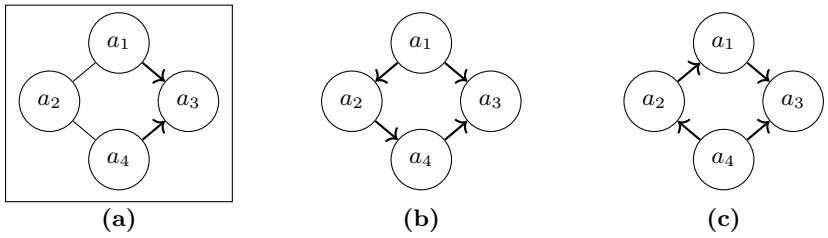

Fig. 1. Example of an essential graph **(a)** and two BNs **(a)** and **(b)** representing possible interpretations.

2.4 Related works

Our goal is to build a PRM's relational schema from an ontology and expert knowledge in order to learn a model expressing the probabilistic dependencies between the attributes. Related works have established that using constraints while learning BNs brings more efficient and accurate results. **Parameters learning** can be improved by allowing users to specify their knowledge through constraints estimations and priors [12]. In [2] an exact **structure learning** algorithm that uses data and expert's knowledge is presented by defining two types of constraints. In particular one of those identifies where arcs may or may not be

included. In [6] it is argued that combining analogical generalization and structure mapping with statistical machine learning methods allows state-of-the-art performances on standards tasks.

Using knowledge from both experts and ontologies to guide this structure learning, is, thus, possible to construct a BN closer to the studied domain than one learnt directly from data. Four main frameworks have already been defined (BayesOwl [3], PR-OWL [1], HyProb-Ontology [9] and OntoBayes [17]). It is also possible to build a BN guided by a specific task or need [7]. In [4] the authors present a method for automating the BN construction with ontologies. This method addresses four tasks: (i) identification of variables of interest, (ii) specification of the possible values for these variables, (iii) definition of the relations between them and (iv) determination of the conditional probability distribution.

In our approach we chose to work with PRM as they allow both a better expression of the probabilistic relations and a good information visualization. To the best of our knowledge, no method for a semi-automatic approach combining the knowledge of ontologies and experts has been proposed yet for PRM learning.

3 Learning a PRM from an ontology

Guided by a knowledge base and an assumption about causal links between data formulated by the user, our approach allows to build a valid model representing these causal relations. In order to do so we propose to use a PRM to help users check their assumptions and find new causal relations. This model is constructed through an **iterative** approach. First, the knowledge base and the assumption allow us to build a PRM where the attributes are semi-automatically classified in two distinct classes (the explaining and the consequence attributes). Then, the essential graph is proposed for validation through an **interactive** approach to the user who can modify the attributes classification. This confrontation helps the user, as he becomes often more aware of his preferences when the proposed solution violates them [13]. If not satisfied, a new PRM is generated and proposed for another confrontation. The iteration is repeated till complete satisfaction of the user.

3.1 Definition of a relational schema

In order to construct and select a PRM as close as possible to the assumption provided by the user, we define a general relational schema composed of two distinct classes, the **explaining** and the **consequence**. This will serves as a guide during the construction of our relational schema. The **explaining** class contains the attributes (called **explaining attributes**) whose values are defined beforehand and are always known by the user. The **consequence** class contains the attributes (called **consequence attributes**) whose values are determined after the problem resolution, they are often result attributes.

Distinguishing between explaining and consequence attributes influences the causal dependencies detection: if a relation is found between an explaining and

a consequence attribute, the causality is automatically determined from the explaining attributes to the consequence ones. To respect this order during the PRM learning, we impose the constraint that if an inter-class link is found between two attributes, the source of the arrow must be in the explaining class.

The probabilistic dependencies of the relational model are learned using a classical Bayesian learning method we denote M. Bayesian learning methods are based on statistical evaluation. In order to be able to find causal relations between attributes using a PRM, the values of the attributes in a given database have to respect some constraints: if all values are identical or, on the contrary, all different (e.g. an ID), or cannot be discretized in a reasonable number of categories, they are not pertinent for the learning.

Definition 1. ***Useful learning attribute.*** *Let B be a database containing the values of an attributes a. a is a* ***useful learning attribute*** *for B if:*

1. *its values are quantitative or qualitative;*
2. *there exist in B at least two different values for it;*
3. *there is in B at least one of its values being repeated at least once;*
4. *each of its values can be discretized in maximum ε categories (ε being experimentally fixed).*

3.2 Definitions of the expert knowledge

Let us first formalize the user's assumption $\mathcal{H}$ on causal links between the data used in the PRM's construction. It can be represented by the following pattern:

$$\mathcal{H}\text{: } E_1, ..., E_n \text{ have a causal influence on } C_1, ..., C_p$$

with E_1, ..., E_n the explaining attributes and C_1, ..., C_p the consequence ones whose causal relations we want to model with a PRM. We denote the sets of the attributes of $\mathcal{H}$ by $A_E^{\mathcal{H}} = \{E_1, ..., E_n\}$ and $A_C^{\mathcal{H}} = \{C_1, ..., C_p\}$.

Example 1. In a knowledge base about students, an user's assumption could be: "$\mathcal{H}_1$: One's birthplace and social standing have a causal influence on the university where a student studies". Where P = "birthplace", S = "social standing" and U = "university", $A_E^{\mathcal{H}} = \{P, S\}$ and $A_C^{\mathcal{H}} = \{U\}$.

Let us now introduce our definition of a knowledge base. We consider a knowledge base $\mathcal{KB}$ where the ontology is represented in OWL[5] and the data in RDF[6]. $\mathcal{KB}$ is defined by the couple $(\mathcal{O}, \mathcal{F})$, where:

- the ontology $\mathcal{O} = (\mathscr{C}, DP, OP, A)$ is defined by a set of classes $\mathscr{C}$, a set of *owl:DataTypeProperty* DP in $\mathscr{C} \times T_D$ with T_D being a set of primitive datatypes (e.g. integer, string), a set of *owl:ObjectProperty* OP in $\mathscr{C} \times \mathscr{C}$, and a set of axioms A (e.g. property's domains and ranges, subsumption).

[5] https://www.w3.org/OWL/
[6] https://www.w3.org/RDF/

– a collection of triples $\mathcal{F} = (s, p, o)$, called instances, where s is the subject of the triple, p is a property that belongs to $DP \cup OP$ and o is the object of the triple; for a triple (s, p, o), we note $domain(p) = s$ and $range(p) = o$.

Example 2. Fig. 2 gives an excerpt of an ontology about students and universities: *Student* $\in \mathscr{C}$, *Name* $\in T_D$, *isRegisteredIn* $\in OP$, *hasForName* $\in DP$.

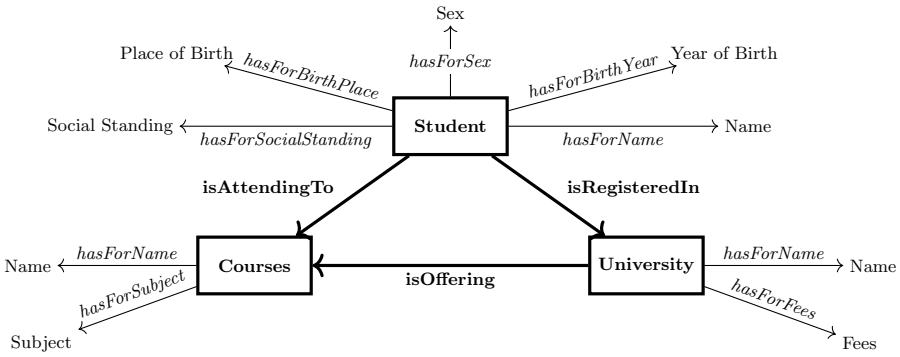

Fig. 2. An excerpt of an ontology about students and their universities

In order to construct a PRM from an ontology, we need to find in the ontology to what correspond the useful learning attributes of Def. 1. We chose to consider datatype properties $p \in DP$ whose range are literal values.

Definition 2. ***Useful datatype property for a knowledge base.*** *A datatype property $p \in DP$ is a useful datatype property for the knowledge base $\mathcal{KB}$=($\mathcal{O}$, $\mathcal{F}$) if the set of all its values v_i in $\mathcal{KB}$ respects the constraints of Def. 1 where v_i is a literal value s.t. range(p) = v_i.*

Example 3. The datatype property *hasForSex* is a useful one: its literal values are different, repeated and can be instantiated in a reasonable number of categories. On the contrary, the datatype property *hasForName* for the class Student cannot be a useful one, because its different literal values can be hard to discretize.

3.3 Attributes identification from the causal links assumption

Let us first look at the attributes of the user's assumption $\mathcal{H}$. We denote the set of all ontology's entities corresponding to an attribute $a \in A_i^{\mathcal{H}}$ as S_i^a, with $i \in \{C, E\}$ and $A^{\mathcal{H}} = A_C^{\mathcal{H}} \cup A_E^{\mathcal{H}}$. Alg. 1, detailed below, allows one to construct $S_i^{\mathcal{KB}}$: the set of all S_i^a grouping the datatype properties of the ontology $\mathcal{O}$ corresponding to the explaining and the consequence attributes of $\mathcal{H}$.

Algorithm 1: Construction of the set $S_{\mathcal{H}}^{\mathcal{KB}} = S_E^{\mathcal{KB}} \cup S_C^{\mathcal{KB}}$

Input : $\mathcal{KB}$=($\mathcal{O}$, $\mathcal{F}$) a knowledge base, $A_C^{\mathcal{H}}$ and $A_E^{\mathcal{H}}$ the sets of the attributes of a causal links assumption $\mathcal{H}$

Output: the sets $S_C^{\mathcal{KB}}$ and $S_E^{\mathcal{KB}}$ of the datatype properties defined in the ontology $\mathcal{O}$ that correspond to the attributes of $A_C^{\mathcal{H}}$ and $A_E^{\mathcal{H}}$

$S_C^{\mathcal{KB}} = \emptyset$, $S_E^{\mathcal{KB}} = \emptyset$;

\# **Step 1:** Selection of the different attributes of $\mathcal{H}$

→ the sets S_E^a for each attribute $a \in A_E^{\mathcal{H}}$,

the sets S_C^a for each attribute $a \in A_C^{\mathcal{H}}$;

\# **Step 2:** Automatic validation of the attributes

→ the updated sets S_E^a and S_C^a that checks Def. 2;

If $\exists a \in A^{\mathcal{H}}$ s.t. $S_i^a = \emptyset$ **then** Exit ;

\# **Step 3:** user validation of the attributes

→ the sets S_E^a and S_C^a validated by the user;

If $\exists a \in A^{\mathcal{H}}$ s.t. $S_i^a = \emptyset$ **then** Exit ;

\# **Step 4:** Integrity verification of the attributes ;

→ boolean *bool*;

If *bool* = True **then** $\forall i \in \{E, C\}, S_i^{\mathcal{KB}} = \cup_{a \in A_i^{\mathcal{H}}} S_i^a$;

If one of the sets S_E^a or S_C^a is empty at the end of Alg. 1 then there is no possible answer to the assumption in the knowledge base: all attributes of $\mathcal{H}$ have to be represented in the ontology by at least one datatype property.

Step 1. For each attribute $a \in A_i^{\mathcal{H}}$, its set S_i^a with the classes and properties that correspond to it in $\mathcal{KB}$ is built. Using a similarity measure (e.g. Jaccard measure) and α an experimentally fixed value in [0,1], if the similarity between an attribute's name and an entity's label in $\mathcal{KB}$ is higher than α, we have: (i) if the entity is a class or a datatype property, it is added to S_i^a; (ii) if the entity is an object property, its range and domain classes are added to S_i^a.

Example 4. Following $\mathcal{H}_1$ and the university ontology, we determine S_E^P ={datatype property *hasForBirthPlace*}, S_E^S ={datatype property *hasForSocialStanding*} and S_C^U ={class *University*}.

Step 2. The validity of each attribute $a \in A^{\mathcal{H}}$ for $\mathcal{KB}$ is checked. An attribute is **valid** if the set of its corresponding entities in $\mathcal{KB}$ respects three conditions: first, they all must have a sufficient number of instances (otherwise the learning would not be possible, as there would be too much missing values); second, all the datatype properties that compose it must be useful ones for $\mathcal{KB}$; finally, all the classes that compose it must be the *domain* of at least one useful datatype property. If an entity does not respect those conditions, it is removed from S_i^a. If the set becomes empty, then the algorithm ends. Moreover, all classes of S_i^a are replaced by their datatype properties. At the end, the sets S_E^a and S_C^a are only composed of datatype properties.

Example 5. The datatype properties *hasForBirthPlace* and *hasForSocialStanding* can be discretized by creating large categories (e.g. the region of the birth-

place). The class University has two useful datatype properties, *hasForName* and *hasForFees*. The attributes P, S and U are therefore valid.

Step 3. For each valid attribute $a \in A^{\mathcal{H}}$, its set S_i^a of corresponding datatype properties in $\mathcal{KB}$ is presented to the user who made the assumption. The user can then choose to remove the properties he judges inadequate. An attribute can be invalidated if all of its properties have been removed: the algorithms then ends.

Example 6. Let's consider the assumption $\mathcal{H}_2$: "One's social standing has an influence over their university name.". We would have $A_E^{\mathcal{H}} = \{\text{social standing } S\}$ and $A_C^{\mathcal{H}} = \{\text{university name } U\}$. During the construction of S_C^U, we could select the three datatype properties *hasForName* of the university ontology (see Fig. 2). However, the only one interesting for the problem is the one about the University.

Step 4. The consistency of the set of entities in $A^{\mathcal{H}}$ is checked: there must exist enough instances linking the entities in $A^{\mathcal{H}}$. From any datatype property p_1 of the set of all selected datatype properties, we start by looking at its domain class c_1. First, we look at c_1 datatype properties: if one of them has been selected, it means that it is linked to p_1. Then, we continue by looking at its object properties and their other classes if they have enough instantiations (for instance, if there exists an object property p_o with enough instantiations such that $domain(p_o) = c_1$), we look at $range(p_o)$. By successive iterations over the newly encountered class, the exploration continue until all selected entities have been linked to p_1. However, in case not all the entities have been linked, the assumption $\mathcal{H}$ cannot be answered and the step returns False.

If all verifications have been successful, we note $S_E^{\mathcal{KB}} = \cup_{a \in A_E^{\mathcal{H}}} S_E^a$ and $S_C^{\mathcal{KB}} = \cup_{a \in A_C^{\mathcal{H}}} S_C^a$ with $a \in A^{\mathcal{H}}$, $S_i^a \neq \emptyset$, the set of all datatype properties in $\mathcal{KB}$ that correspond to the explaining and consequence attributes of $\mathcal{H}$.

Most of the time the attributes expressed in the user's assumption $\mathcal{H}$ are not enough to find causal relations between data and it is necessary to find other meaningful attributes in order to improve the model that we want to build.

3.4 Enriching the set of PRM attributes

Thanks to the user's assumption $\mathcal{H}$, the datatype properties corresponding to attributes of a PRM can be "classified" as explaining or consequence attributes. However, this determination is more complex when analyzing the datatype properties added during the enrichment step, as there is no indication of their belonging. These datatype properties are considered as explaining by default.

Linking a set of concepts in a given context to explain them is similar to a connection graph problem [14]. The method we introduce is based on successive iterations on the ontology's graph over the properties, starting from the entities found previously, and following them to find new potential properties to enhance our set. We first set $s = S_E^{\mathcal{KB}} \cup S_C^{\mathcal{KB}}$, meaning that we start by looking at the

datatype properties that we know are relevant both to the user and the problem, as well as to the learning of the PRM. For each entity of s, depending on its nature, different scenarios are possible:

- if it is a **class**, we add all its datatype and object properties in s;
- if it is an **object property**, we add its domain and range to s;.
- if it is a **datatype property** that is useful and approved by the user, we keep it; if the user can determine if it is an explaining or a consequence attribute, it is added to its rightful set, otherwise it is added to $S_E^{\mathcal{KB}}$.

Once there are no more classes or properties to look at in s, we stop. At the end, $S_E^{\mathcal{KB}}$ and $S_C^{\mathcal{KB}}$ are the sets of all datatype properties that can be used to help the user checking $\mathcal{H}$, each of them corresponding to a PRM class.

3.5 PRM construction

Two ways of bringing information are proposed to the user: first the knowledge brought during the construction of the PRM relational schema creates causal constraints, which reduces the number of free parameters and helps to learn a model closer to what he intends; second the EG used to criticize the learned model helps to express causal dependences, meaning that if an edge has a certain orientation given the causal constraints, it is probably causal.

In order for the user to check the different relations and their validity, we propose an interactive and iterative method based on the study of the EG and its comparison with the learned PRM. Considering that the PRM has been learned under causality constraints (given by the user and the ontology), the EG helps to determine causal relations: if an edge is oriented in the EG, then it is certainly causal. Two verifications are to be made: a first one for the inter-class relations, and a second one for the intra-class relations.

First, the EG inter-class relations are presented to the user. We usually start by them as they are the one he had direct control over: if one of them appears to be wrongly oriented, it means that the relational schema (RS) has been badly constructed. It, thus, needs to be modified by moving one or several nodes between the consequence and explaining classes. If no problem has been detected, the intra-class relations are studied. If the relation is **non oriented** in the EG, it means that both cases are possible: the user can choose to keep this relation as it is in the learned PRM or inverse it. If it is inversed, a new PRM has to be learned. In order to do so, the user needs to specify RS by creating a new class. If the relation is **oriented**, it means that if the orientation between the two attributes is wrong according to the user, the RS needs to be modified. Two cases are possible: if one of the node only needs to change class, then we keep two classes in the RS's structure. Otherwise, if this change is not possible, the RS needs to be modified by introducing new classes.

Our method is interactive since it is supervised by the user. It is iterative since whenever the RS is modified, a new PRM has to be learned and a new set of verifications has to be done. The advantage of learning a PRM is that it allows a better constraint on the relations during the learning and eases the user's task of verification by structuring the result into classes.

4 Application

To illustrate our approach we have chosen to use the DBPedia ontology part dedicated to films. As experts, we have formulated the following hypothesis $\mathcal{H}_e$: *The origin country of a film has an influence on the number of awards it has won.*

The database used is composed of instances of movies selected from DBPedia, completed with other data from the website Internet Movie Database, IMDb[7].

4.1 Attributes selection

Two attributes are represented in $\mathcal{H}_e$, the **origin country** O and the **number of awards won** W. Using text mining methods, we have determined that S_E^O is composed by the datatype property *imdb:hasForOriginCountry*, and that S_C^W is composed by the datatype property *imdb:hasForWonAwards*. Both datatype properties are instantiated a sufficient number of times and are useful datatype properties. Plus, both have for *domain* the same Film class, so the set $S_E^O \cup S_C^W$ is consistent.

In order to enrich $S_E^{\mathcal{KB}}$ and $S_C^{\mathcal{KB}}$, we have selected other datatype properties. We have added in the explaining group the **runtime** (how long a film lasts), the **budget** and the **release year**, since they describe the film before its release; in the consequence class the **number of nominations**, the **IMDb note** (average notation of the film given by the users of the website) and the **gross**, since they describe how the public reacted to the film after its release. Some attributes were refused because they could not be efficiently discretized (such as the different actors, directors, producers); other because they were not involved in the problem (such as the Wikipedia ID page). All of these properties are also directly linked to the Film class.

Once everything selected, this gave us a database of 10,000 films to study, composed of 90,000 RDF triplets.

4.2 PRM learning

First iteration. A PRM is learned and a EG corresponding to its instantiated system is presented (Fig. 3 (a)). Only one of its inter-class relation is oriented (i.e. budget has an impact on nomination); however, the two other are easily oriented, since they have been distributed in two different classes: their orientation has been constraint by the user. The study of the explaining inter-class relations is also pretty straight-forward, as they are all already oriented. On the contrary, none of the explaining relation are oriented: it means that all of the possibilities are Markov equivalent. However, as experts, it appeared to us that it seems inconsistent to consider the release year and the budget to have an influence on the country. As a consequence we split the explaining class into two new classes,

[7] http://www.imdb.com/

E_1 in which we put the Country attribute and E_2 in which we put the rest of the explaining class attributes.

Second iteration. A second PRM and its associated EG (Fig. 3 (b)) are learned with the new relational schema which adds the constraint that the country attribute must have an influence over the release year and the budget. This modifies greatly the relations inside the explaining class, as now they are all oriented.

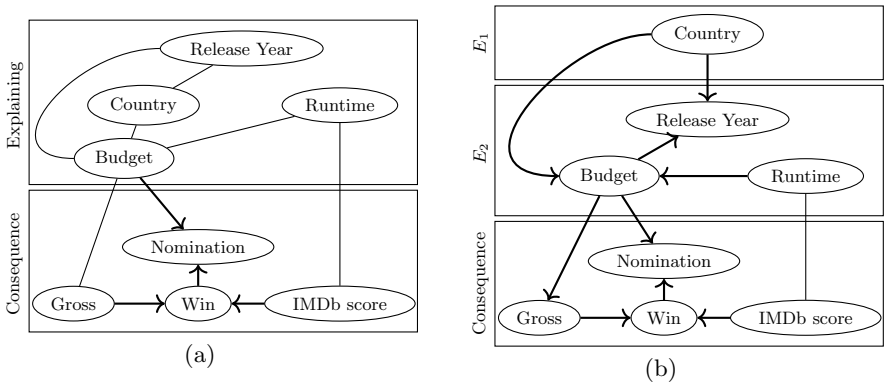

Fig. 3. First (a) and second (b) learned GE represented with their relational schema.

As a conclusion, the assumption has been verified: according to our database and expert knowledge, the country has an (indirect) influence over the number of awards won, through its attributes budget and gross.

5 Conclusion

Combining ontologies and PRMs is efficient when learning a model to represent and explain a complex domain. However, in order to fully represent the dependencies between data, an expert knowledge is often required. In this article, we present a generic interactive and iterative method, suitable for any ontology, that helps a user to find causal relations between data. Given $\mathcal{H}$ an hypothesis on the domain to impose a research context, our method (i) selects the interesting attributes of $\mathcal{H}$, (ii) enriches this selection by adding other attributes and (iii) learns a probabilistic model to capture the dependencies and identify causal relations.

In a short-term perspective, we want to extend our method on temporal ontology (e.g. an ontology of transformation processs) relying on the assumption that the temporal relations can be considered as constraints to classify the attributes in explainig or consequence PRM classes. On future works, we want to study in more details the validation and introspection of the learned model in order to improve our help to the user.

References

1. Paulo Cesar G. da Costa, Kathryn B. Laskey, and Kenneth J. Laskey. Pr-owl: A bayesian ontology language for the semantic web. In Paulo Cesar G. da Costa, Claudia d'Amato, Nicola Fanizzi, Kathryn B. Laskey, Kenneth J. Laskey, Thomas Lukasiewicz, Matthias Nickles, and Michael Pool, editors, *Uncertainty Reasoning for the Semantic Web I*, pages 88–107, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.
2. Cassio P. de Campos, Zhi Zeng, and Qiang Ji. Structure learning of bayesian networks using constraints. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 113–120, New York, NY, USA, 2009. ACM.
3. Zhongli Ding and Yun Peng. A Probabilistic Extension to Ontology Language OWL. In *Proceedings of the 37th Hawaii International Conference On System Sciences (HICSS-37).*, page 10, Big Island, Hawaii, January 2004.
4. S. Fenz, A. M. Tjoa, and M. Hudec. Ontology-based generation of bayesian networks. In *2009 International Conference on Complex, Intelligent and Software Intensive Systems*, pages 712–717, March 2009.
5. Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. Learning probabilistic relational models. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 99, Stockholm, Sweden, July 31 - August 6, 1999. 2 Volumes, 1450 pages*, pages 1300–1309, 1999.
6. Chen Liang and Kenneth D. Forbus. Learning plausible inferences from semantic web knowledge by combining analogical generalization with structured logistic regression. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 551–557. AAAI Press, 2015.
7. Eveline M. Helsper and Linda C. Gaag. Building bayesian networks through ontologies., 01 2002.
8. David Madigan, Steen A Andersson, Michael D Perlman, and Chris T Volinsky. Bayesian model averaging and model selection for markov equivalence classes of acyclic digraphs. *Communications in Statistics–Theory and Methods*, 25(11):2493–2519, 1996.
9. Abdul-Wahid Mohammed, Yang Xu, and Ming Liu. Knowledge-oriented semantics modelling towards uncertainty reasoning. *SpringerPlus*, 5(1):706, Jun 2016.
10. Melanie Munch, Pierre-Henri Wuillemin, Cristina Manfredotti, Juliette Dibie, and Stephane Dervaux. Learning probabilistic relational models using an ontology of transformation processes. In *On the Move to Meaningful Internet Systems. OTM 2017 Conferences*, pages 198–215, Cham, 2017. Springer International Publishing.
11. Richard E. Neapolitan. *Learning Bayesian Networks*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2003.
12. Cassio P. de Campos and Qiang Ji. Improving bayesian network parameter learning using constraints, 01 2009.
13. Pearl Pu and Boi Faltings. Enriching buyers' experiences: the smartclient approach. 01 2000.
14. Abhishek Sharma and Kenneth D. Forbus. Graph traversal methods for reasoning in large knowledge-based systems. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, AAAI'13, pages 1255–1261. AAAI Press, 2013.
15. Lionel Torti, Pierre-Henri Wuillemin, and Christophe Gonzales. Reinforcing the Object-Oriented Aspect of Probabilistic Relational Models. In *PGM 2010* -

The Fifth European Workshop on Probabilistic Graphical Models, pages 273–280, Helsinki, Finland, September 2010.

16. Pierre-Henri Wuillemin and Lionel Torti. Structured probabilistic inference. *Int. J. Approx. Reasoning*, 53(7):946–968, 2012.
17. Yi Yang and Jacques Calmet. Ontobayes: An ontology-driven uncertainty model. In *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce Vol-1 (CIMCA-IAWTIC'06) - Volume 01*, CIMCA '05, pages 457–463, Washington, DC, USA, 2005. IEEE Computer Society.