Data & Knowledge Engineering 73 (2012) 73–88

Contents lists available at SciVerse ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

An ontology-based approach for constructing Bayesian networks

Stefan Fenz *

SBA Research, Favoritenstrasse 16, 2nd floor, 1040 Vienna, Austria
Vienna University of Technology, Favoritenstrasse 9-11/E188, 1040 Vienna, Austria

ARTICLE INFO

Article history:
Received 21 April 2011
Received in revised form 13 December 2011
Accepted 14 December 2011
Available online 22 December 2011

Keywords:
Bayesian network
Ontology
Knowledge re-use

ABSTRACT

Bayesian networks are commonly used for determining the probability of events that are influenced by various variables. Bayesian probabilities encode degrees of belief about certain events, and a dynamic knowledge body is used to strengthen, update, or weaken these assumptions. The creation of Bayesian networks requires at least three challenging tasks: (i) the determination of relevant variables (nodes), (ii) the determination of relationships between the identified variables (links), and (iii) the calculation of the conditional probability tables (CPTs) for each node in the Bayesian network. Based on existing domain ontologies, we propose a method for the ontology-based construction of Bayesian networks. The method supports (i) the construction of the graphical Bayesian network structure (nodes and links), (ii) the construction of CPTs that preserve semantic constraints of the ontology, and (iii) the incorporation of already existing knowledge facts (findings). The developed method enables the efficient construction and modification of Bayesian networks based on existing ontologies.

1. Introduction

Bayesian networks (also referred to as probabilistic networks or belief networks) have been applied to several real-world applications such as medical diagnosis and prognosis, spam analysis, information retrieval, and natural language processing (cf. [8,24,16,30,21]). According to Neapolitan [24], Bayesian networks are*'graphical structures for representing the probabilistic relationships among a large number of variables and doing probabilistic inference with those variables'*. According to Heckerman et al. [15], the Bayesian probability of an event *x* is a person's degree of belief in the very event. Therefore, the frequentistic probability class (e.g., the probability that a coin lands heads or tails, based on a large number of trials tending to infinity) is distinguished from the Bayesian (personal) probability, in which probabilities encode degrees of belief about certain events and data is used to strengthen, update, or weaken these assumptions (e.g., the degree of belief that the coin will land heads at the next throw) [27]. The following challenges arise at the construction of Bayesian networks:

- identification of variables that are relevant to the considered domain (nodes),
- identification of relationships between the identified variables (links), and
- creation of the conditional probability table for each variable.

The conditional probability table (CPT) is used to express how the potential states of the node's parents affect the posterior probability of the considered node. In general, two different methods or a combination of both are used to construct a Bayesian network: (i) automated construction of Bayesian networks from existing data (cf. [3]), and (ii) the domain expert-based construction of Bayesian networks covering complex knowledge domains with insufficient or non-existing empirical data regarding relevant variables. Both approaches have their specific shortcomings. While the domain expert-based approach is time-consuming and error-prone (cf. [8]), the automated construction of Bayesian networks from existing data faces a bias problem.

* Tel.: +43 1 5053688.
E-mail address: sfenz@sba-research.org.

0169-023X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.datak.2011.12.001

The main problem is that the Bayesian network is constructed from a limited subpopulation (e.g., a medical database of a specific hospital covering the treated patients) to be applied to a large target population (e.g., the entire population of a country or continent). Therefore, automated construction methods based on existing data are often insufficient in practice [9].

To address these problems, Druzdzel et al. emphasize in [8] the need for approaches aimed at reducing the number of probabilities to be assessed and tools for supporting the quantification task in the construction of Bayesian networks.

According to the ontology definition given by Neches, ontologies are a potential solution to support the construction of Bayesian networks: *an ontology defines the basic terms and properties comprising the vocabulary of a topic area as well as the rules for combining terms and properties to define extensions to the vocabulary* [25]. An ontology's classes, properties, and individuals are widely used to capture and represent a given knowledge domain. Since Bayesian networks use nodes and links to represent a given domain, we can use the formally defined semantics of ontology classes, individuals, and properties to automatically construct the graphical structure (nodes and links) of Bayesian networks. Furthermore, ontologies can be used to automatically construct CPTs based on expert knowledge modeled in the ontology. Although experts would still be required to transfer their knowledge into a machine-readable format, the ontology enables them to model the knowledge in a way that is more aligned to the human way of thinking and reasoning engines can be used to infer new knowledge from already known facts. Furthermore, we may be able to reduce the time required to assess the necessary knowledge by using already existing domain ontologies. As previously noted, it is important to stress that the concepts of ontologies and Bayesian networks are not equivalent. While ontologies describe domain concepts and their interrelationships, Bayesian networks describe probability distributions and their interrelationships. For this reason, fully automated 1:1 mapping is not possible. However, real-world scenarios such as the use case described in this article have shown that ontologies already describing a certain domain can be valuable in the construction of Bayesian networks to represent causalities in that same domain. This work does not aim at constructing ontologies solely for the construction of Bayesian networks. Instead, it provides an approach that efficiently reuse knowledge that has already been modeled in an ontology in the construction of Bayesian networks. In such a setting, ontology tools can be used to manage the actual knowledge and therefore make the collaborative editing of the Bayesian network structures (e.g., the introduction of new variables) easier. To exploit the capabilities of ontologies for the efficient construction of Bayesian networks, a method for the ontology-based construction of Bayesian networks is required.

Most approaches in the field of ontology-based construction of Bayesian networks do not provide a method for using existing ontologies without specific extensions for the Bayesian network construction (cf. [6,28,34,5,35]). Approaches that do not require these extensions do not support the automated incorporation of findings (concrete knowledge) and conditional probability tables (CPTs) (cf. [17]). While general approaches for constructing CPTs exist (cf. [33,7,29,10]), existing ontology-based approaches require substantial ontology modifications to enable automated CPT computation (see Section 2 for further details). Therefore, the research questions are:

- RQ1: How can existing ontologies be used to construct the graphical structure of Bayesian networks?
- RQ2: How can existing ontologies be used to enrich Bayesian networks with CPTs and concrete findings while preserving the semantic constraints of the ontology?

Using existing domain ontologies, this article presents a generic method for the ontology-based construction of Bayesian networks by (i) using ontology classes/individuals to create the nodes of the Bayesian network, (ii) using ontology properties to link the Bayesian network nodes, (iii) utilizing the ontological knowledge base to support the conditional probability table calculation for each node, and (iv) enriching the Bayesian network with concrete findings from existing domain knowledge. The developed method enables the semi-automatic construction and modification of Bayesian networks based on existing ontologies. The method is demonstrated on the example of threat probability determination, which uses a security ontology as its underlying formal knowledge base.

This paper is organized as follows: The first part reviews related work and introduces the fundamentals of ontologies and Bayesian networks. The second part describes the proposed ontology-based construction of Bayesian networks in detail. The paper concludes with a prototype implementation of the proposed approach and a proof of concept conducted within the information security domain.

2. Related work

Before describing existing ontology-based Bayesian network approaches, we will provide a brief overview of approaches that can be used to learn Bayesian networks from data. Both fields share the same goal but differ substantially in the way they achieve that goal.

Lam and Bacchus [22] developed a Minimal Description Length (MDL) principle-based approach to learn Bayesian networks from raw data. The approach does not require any prior assumptions about the distributions and allows a trade-off between accuracy and complexity in the learned model. Larrafiaga et al. [23] proposed a Bayesian network learning approach that uses a database of cases as input and generic algorithms to search for the best ordering of the system variables. Friedman and Koller [13] presented a Bayesian approach to structure discovery in Bayesian networks. Based on a given data set the approach expresses posterior distributions over Bayesian network structures and evaluates the posterior probability of important structural features of the distribution. Hruschka et al. [20] proposed a low computational complexity method based on feature ranking algorithms that can be used to define efficient variables ordering in the Bayesian network learning context.

The described approaches are just a small subset of available approaches to learn Bayesian networks from data. However, this brief overview shows that this type of approach relies on raw data to construct valid Bayesian networks. In some cases there is no raw data but concepts and relations between these concepts have already been defined. Ontologies are an example of such a case.

They define the basic concepts and relationships of a certain domain and can therefore be reused to construct the nodes and links of a Bayesian network without the need for substantial amounts of raw data. In contrast to learning Bayesian networks from data, the research field of ontology-based construction of Bayesian networks is quite new. It differs from learning Bayesian networks from data in that it relies on a well-defined knowledge base and a domain expert who selects the right ontological concepts for the Bayesian network construction. Currently, the following approaches exist:

Helsper et al. [17] construct Bayesian networks from ontologies in four main phases: First, the graphical structure of the Bayesian network is derived from classes and properties from the ontology. Then classes are translated into statistical variables with an exhaustive state space of mutually exclusive, discrete values. In the third step, properties are translated into arcs between variables. Finally, a manual consolidation step ensures that unnecessary arcs and state spaces are removed from the Bayesian network. The method does not support the quantification of the graphical structure (i.e., conditional probability tables).

Ding et al. [6] propose probabilistic OWL markups that can be attached to individuals, classes and properties in an OWL ontology. They define a set of translation rules to convert this probabilistically annotated OWL ontology into the directed acyclic graph of a Bayesian network. Conditional probability tables are constructed for each network node based on the logical properties of its parent node. The main problem of this approach is that it requires the extension of existing domain ontologies with the proposed probabilistic OWL markups.

Sadeghi et al. [28] use the descriptive information captured in an ontology to construct Bayesian decision models. The authors assume that the underlying ontology is specifically created to model the considered decision problem and that one hypothesis can explain the states of evidence observed in the domain. While the authors describe their overall approach – (i) build the ontology, (ii) implement the Bayesian network model, (iii) acquire the knowledge, and (iv) build the knowledge base – they give no detailed description regarding the ontology-based construction of the Bayesian network.

Similar to Ding et al., Yang et al. [34] propose probability and dependency-annotated OWL extensions to construct Bayesian networks from ontologies. While their approach allows dealing with multivalued random variables in the CPT construction, the approach still requires the extension of the ontology with the proposed probabilistic OWL constructs.

Devitt et al. [5] propose the following phases for automating Bayesian network construction from ontologies: (i) identification of the variables of interest, (ii) specification of the values these variables can take, (iii) definition of the properties between the variables, and (iv) assignment of a conditional probability distribution. The main difference between our approach and the approach of Devitt et al. is that we construct the Bayesian network directly from existing domain ontologies and do not require any Bayesian network-specific ontology extensions.

Zheng et al. [35] encode Clinical Practice Guidelines into an ontology that contains uncertainty features, and propose an algorithm that uses these features to construct the structure and the conditional probability tables of Bayesian networks. The proposed approach requires the creation of an ontology with the properties necessary for Bayesian network construction and does not consider the use of already existing domain ontologies.

Costa et al. [4] developed PR-OWL as a Bayesian ontology language for the semantic web. PR-OWL provides means of modeling uncertainty in ontologies and serves as a supporting tool for applications that benefit from probabilistic inference. PR-OWL extends the OWL standard by probabilistic features and thereby allows legacy OWL ontologies to interoperate with the probabilistic PR-OWL ontologies.

Bucci et al. [2] used ontologies and Bayesian networks to support medical diagnosis. Medical domain knowledge is extracted from the ontology and filled into Bayesian network templates to enable probabilistic reasoning. The predefined templates have been used to decrease the complexity at constructing the Bayesian network.

Ishak et al. [18] developed a set of mapping rules which support the construction of an Object Oriented Bayesian network structure by exploiting the knowledge stored within the ontology. The CPT construction is not considered by the proposed method.

In contrast to existing approaches, the approach presented in this article

- works with existing ontologies, i.e., no specific extensions are required to construct the graphical Bayesian network structure
- conducts CPT computation without invasive extensions, i.e., the necessary ontology extensions do not affect existing classes and individuals of the ontology
- preserves semantic constraints that have been defined in the ontology in the CPT calculation
- supports the incorporation of findings that are modeled in the ontology into the Bayesian network
- is designed as a generic approach and has been implemented as a prototype to construct Bayesian networks based on existing OWL ontologies

3. Ontologies and Bayesian networks

As the ontology-based construction of Bayesian networks requires an understanding of how ontologies and Bayesian networks work, this section introduces the most important aspects of both concepts. According to Gruber [14] an ontology is an explicit specification of a conceptualization. In ontology engineering we use the constructs of classes, individuals, and properties to create a formal knowledge model.

Classes group resources with similar characteristics. Class descriptions (axioms) describe the class. Potential class descriptions include property restrictions, which are subdivided into (i) value constraints (allValuesFrom, someValuesFrom, hasValue) and (ii) cardinality constraints (Max, Min). The different constraint types are explained by the following examples (derived from

[32]). The following allValuesFrom example describes a class of individuals for which the hasParent property only has values of class Human. Please note that the allValuesFrom property does not state that the property always must have values of this class.

```
<owl:Restriction>
   <owl:onProperty rdf:resource="#hasParent"/>
   <owl:allValuesFrom rdf:resource="#Human"/>
</owl:Restriction>
```

The following someValuesFrom example defines a class of individuals that have at least one parent who is a woman.

```
<owl:Restriction>
   <owl:onProperty rdf:resource="#hasParent"/>
   <owl:someValuesFrom rdf:resource="#Woman"/>
</owl:Restriction>
```

The following hasValue example describes the class of individuals who have the individual referred to as female as gender.

```
<owl:Restriction>
   <owl:onProperty rdf:resource="#Gender"/>
   <owl:hasValue rdf:resource="#female"/>
</owl:Restriction>
```

The following Max cardinality example describes a class of individuals that have at most two parents.

```
<owl:Restriction>
   <owl:onProperty rdf:resource="#hasParent"/>
   <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger">2</owl:maxcardinality>
</owl:Restriction>
```

The following Min cardinality example describes a class of individuals that have at least two siblings.

```
<owl:Restriction>
   <owl:onProperty rdf:resource="#hasSibling"/>
   <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">2</owl:minCardinality>
</owl:Restriction>
```

For further class descriptions, in-depth definitions and detailed examples see [32].

Individuals are instances of a specific class. To be an instance of a class, individuals must match the class descriptions of the class. The following individual example shows an instance of the class Person (named *Lina*) matching the previously explained class descriptions.

```
<Person rdf:ID="Lina">
   <hasParent rdf:resource="#LinasMother"/>
   <hasParent rdf:resource="#LinasFather"/>
   <Gender rdf:resource="#female"/>
   <hasSibling rdf:resource="#LinasSister"/>
   <hasSibling rdf:resource="#LinasBrother"/>
</Person>
```

LinasMother is an instance of class Human, Woman, and Parent. *LinasFather* is an instance of class Human, Man, and Parent. The gender of Lina is female.

Properties link individuals to individuals (object properties) or individuals to data values (data type properties). Please see the previously described individual example for how properties link different individuals (e.g., linking individual Lina to individual LinasSister via the hasSibling property).

By using classes, individuals, and properties, we can create a formal knowledge base that contains concrete domain knowledge. The main advantages of modeling and representing knowledge by an ontology is the explicit specification of the domain and the corresponding reasoning possibilities.

Bayesian networks have some similarities to ontologies, which we want to exploit for the ontology-based construction of Bayesian networks. Bayesian networks represent the probabilistic relationships within a certain domain by a directed acyclic graph (DAG). Basically, this graph consists of nodes, which represent relevant variables of the domain, and directed links connecting dependent nodes. The probability distribution over each node's states is defined by its conditional probability table (CPT) and the probability distribution of its parent nodes (i.e., nodes that are connected to it by a directed arc).

Two fundamental steps are required to build a Bayesian network: (i) create the graphical structure including relevant nodes and links, and (ii) create the conditional probability table for each node.

This article proposes a method that uses declarative knowledge modeled in the ontology to increase the efficiency of the Bayesian network construction.

We use the W3C Web Ontology Language (OWL) to model and present the ontology. The Norsys Netica Bayesian Network Software[1] was used to construct and present the actual Bayesian network. The proof of concept (including the actual user interface) was implemented as a Java-coded Protege 4.0[2] plug-in.

4. Ontology-based construction of Bayesian networks

The proposed ontology-based approach for constructing Bayesian networks consists of four main phases:

1. Selection of relevant classes, individuals, and properties
2. Creation of the Bayesian network structure
3. Construction of the CPTs
4. Incorporation of existing knowledge facts

While the first phase requires the input of a domain expert, the remaining three phases are conducted automatically based on the output of phase 1. Each phase is described in detail in the following subsections.

4.1. Selection of relevant classes, individuals, and properties

Although ontology classes, individuals, and properties are semantically well defined, a domain expert is required to select those classes, individuals, and properties that are relevant to the considered problem and should be represented in the Bayesian network. In this context, classes, individuals, and properties are relevant if they influence the state of the Bayesian network's final output node(s). In selecting relevant classes, individuals, and properties, the domain expert has to ensure that the selection does not produce any redundant edges in the Bayesian network. E.g., if A is affected by B and B is affected by C, only edges from B to A and from C to B are allowed. It would not be allowed to incorporate an additional edge from C to A (e.g., through transitivity). Please see Section 6.2 for an example from the information security risk management domain.

4.1.1. Class/individual selection

One advantage of ontologies is that they allow us to generalize classes in the considered knowledge domain. This simplifies the task of selecting relevant classes and individuals. Based on an existing ontology, the expert has to select those classes and individuals that are relevant to the considered problem and should be represented by Bayesian network nodes. Whether the expert selects solely classes or both classes and individuals depends on the specific ontology. If the ontology does not contain any individuals, the selection is only conducted at class level. To simplify this assessment process, the domain expert is able to choose only high-level classes and all of their sub-classes or individuals are marked as the entities that are actually to be represented by nodes in the final Bayesian network. We identified and defined three different class/individual types that the domain expert has to select.

- Node Class/Individual: directly relevant to the problem domain, i.e., the identified classes/individuals represent input, output, or intermediate nodes in the Bayesian network.
- State Space Class/Individual: defines state spaces for Node Classes/Individuals (e.g., “high, medium, low” or “present, absent”). We use State Space Classes/Individuals to define a mutually exclusive and discrete state space in the ontology and use that knowledge in the automated Bayesian network construction.
- Weight Class/Individual: defines variable weights for Node Classes/Individuals in a given parent/child node combination.

The selection of Node and State Space Classes/Individuals is required to construct a valid Bayesian network. The selection of Weight Classes/Individuals is optional. Please see Section 4.3 for how the selection of Weight Classes/Individuals influences the CPT computation.

4.1.2. Property selection

The class/individual selection in the previous step restricts the number of potential properties that can be incorporated into the Bayesian network. For the automatic construction of Bayesian networks we use the previously defined class/individual types to identify four different types of properties.

- Link Property: potential Link Properties are those properties that connect the Node Classes/Individuals selected in the previous step. From this set, the expert selects properties that are relevant to the considered problem domain (i.e., properties affecting the probability of the selected classes/individuals). An example of a link property would be the *organization_implements_policy* property used in the use case at the end of this article. It connects implemented policies to the organization, i.e., if a certain policy is implemented it lowers the threat probability of this specific organization.
- State Value Property: each node within the Bayesian network must have a certain state space. State Value Properties assign numerical values to the mutually exclusive, discrete states defined by State Space Classes/Individuals (e.g., “2, 1, 0” for “high,

[1] Norsys Netica: http://www.norsys.com
[2] Protege Ontology Editor: http://protege.stanford.edu

medium, low"). The numerical state values are required to compute the CPTs of the Bayesian network nodes. Please see Section 4.3 for further details regarding the CPT computation.

- Weight Property: it may be necessary to assign weights to relevant child/parent node combinations for the automated CPT calculation. Therefore, the user has to specify three different types of Weight Properties that connect each selected Weight Individual to selected Node Individuals in the given child/parent combination: (i) Weight - child connects the child node to the weight individual, (ii) Weight - parent connects the parent node to the weight individual, and (iii) Weight - weight assigns a numerical weight ranging from 0 to 1 (e.g., 0.05) to the parent/child combination stored in the weight individual. With this tertiary pattern, a numerical weight value can be assigned to any given parent/child node relation.
- Finding Property: as the ontology provides a dynamic knowledge base in addition to the static knowledge model, we can exploit the knowledge base to incorporate existing knowledge facts as findings in the Bayesian network. Properties connecting Node and State Classes/Individuals are used to incorporate the findings (in the form of State Space Classes/Individuals) into the Bayesian network.

The selection of Link and State Value Properties is required to construct a valid Bayesian network. The selection of Weight and Finding Properties is optional. If weights are to be used in the Bayesian network construction, a domain expert has to model and specify Weight Classes/Individuals/Properties in the ontology. While this is a very time-consuming process, it enables the domain expert to encode different parent node weights to influence the CPT computation. Please see Section 4.3 for how the selection of Weight Properties influences the CPT computation.

4.2. Creation of the Bayesian network structure

This step automatically constructs the Bayesian network structure based on the manually selected classes and properties of the previous step. This step results in the graphical structure of the final Bayesian network, i.e., it establishes a directed acyclic graph, containing nodes and directed links.

First, the Node Classes/Individuals selected in the previous phase are used to populate the Bayesian network with nodes that are directly relevant to the problem domain. If the user selected the sub-classes of a specified Node Class, only the sub-classes of the Node Class are added to the graph. If the user selected the individuals of the Node Class, all directly asserted individuals are added to the graph.

For each constructed node we use (i) the corresponding State Space Classes/Individuals to build the state space of the node and assign numerical values for each state by using the selected State Value Property, and (ii) the Link Properties to connect the node to its parent nodes. The numerical state values are required to compute the CPTs of the Bayesian network nodes. Please see Section 4.3 for further details regarding the CPT computation. The following two subsections show how the graphical structure is constructed in the case of (i) class-based and (ii) individual-based ontology modeling.

4.2.1. Classes

For classes, we obtain for each selected Link Property all ontology classes that reference the considered Link Property. We check the class type for each of the returned classes. In the context of classes, the OWL class types *Equivalent Class* and *Super Class* are used to connect the considered class *A* via the selected Link Properties to appropriate parent nodes. Each equivalent and super class can contain several OWL restrictions. Each restriction is defined with a quantifier in the context of a specific property. For the ontology-based construction of Bayesian networks we are especially interested in the OWL restrictions: *someValuesFrom*, *allValuesFrom*, *Min*, *Max*, and *hasValue*. The class/individual stated within the restriction affects the probability of the class in which the restriction is located.

In the case of *someValuesFrom* and *allValuesFrom* restrictions, we link class *B*, stated in the *owl:someValuesFrom* and *owl:allValuesFrom* element, as parent node to the node of class *A*. The followingcode snippets have been taken from the OWL specification of class *A*. The "network view" shows how the nodes would be represented and connected in the Bayesian network (Fig. 1).

```
<owl:Restriction>
   <owl:onProperty rdf:resource="#A_affectedBy_B"/>
   <owl:someValuesFrom rdf:resource="#B"/>
</owl:Restriction>
<owl:Restriction>
   <owl:onProperty rdf:resource="#A_affectedBy_B"/>
   <owl:allValuesFrom rdf:resource="#B"/>
<owl:Restriction>
```

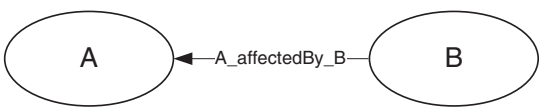

Fig. 1. someValuesFrom and allValuesFrom — network view.

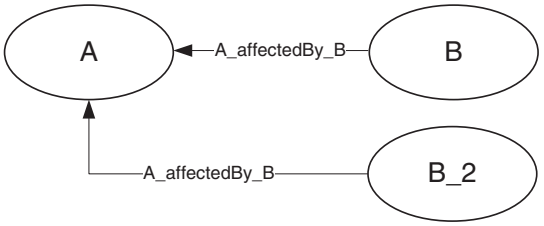

Fig. 2. Min and Max cardinality restrictions — network view.

Please note that the someValuesFrom and allValuesFrom restrictions express completely different semantic constraints when it comes to classifying individuals (cf. Section 3). In the context of Bayesian network construction, however, we use both constraints to determine basic relations at class level. As the semantic difference is only relevant in the individual classification, we ignore it in the class-based creation of the Bayesian network structure. The difference of both constraints is taken into account for the CPT calculation (cf. Section 4.3). If individuals are used to construct the Bayesian network structure (cf. Section 4.2.2), a reasoner classifies all individuals of the ontology according to existing class descriptions. In this case we also take the semantic difference between someValuesFrom and allValuesFrom constraints into account.

In the case of *Min* and *Max* cardinality restrictions, we use the range of the property on which the restriction is defined to connect *A* and *B*. Based on the cardinality value *n* we construct *n* nodes of class *B* and connect them to node *A*. In the following *Min* cardinality example *B* would be the range of property *A_affectedBy_B*. Since the cardinality in our example is set to 2, we construct the nodes *B* and *B_2* based on class *B* and connect them to node *A* (Fig. 2).

```
<owl:Restriction>
   <owl:onProperty rdf:resource="#A_affectedBy_B"/>
   <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">2</owl:minCardinality
</owl:Restriction>
```

Value restrictions connect the class to the individual level as restrictions. We link individual *B_1*, stated in the owl:hasValue element, as parent node to child node *A* (Fig. 3).

```
<owl:Restriction>
   <owl:onProperty rdf:resource="#A_connectedTo_B"/>
   <owl:hasValue rdf:resource="#B_1"/>
</owl:Restriction>
```

4.2.2. Individuals

In addition to class restrictions connecting classes with classes or individuals, OWL property assertion axioms indicate which individuals are connected to each other by which properties. For each Link Property selected by the user, we obtain a list of OWL property assertions that reference the considered property. The object of the property assertion indicates the parent node, which is connected by a directed link to its child node, represented by the subject of the assertion. The following code snippet shows individual *A_1* connected by property *A_affectedBy_B* to individual *B_1* (Fig. 4). In this case *A_1* is the object and *B_1* is the subject: *B_1*→*A_1*.

```
<A rdf:ID="A_1">
   <A affectedBy_B rdf:resource="#B_1"/>
</A>
```

4.3. Construction of CPTs

To construct the CPTs for all relevant nodes, we run through the entire network and select those nodes that have more than zero parents. For each selected node we use the corresponding Weight Classes/Individuals and Properties to determine the weights of its parent nodes. A tertiary pattern is used to define the parent node weight for each child–parent node combination. The CPT construction at each node is influenced by (i) the state space of its parent nodes, (ii) the context-specific weight of its

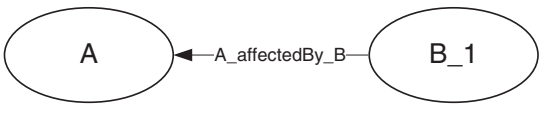

Fig. 3. hasValue restrictions — network view.

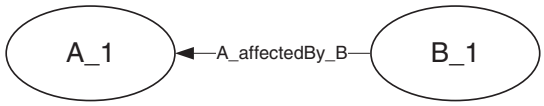

Fig. 4. Individuals — network view.

parent nodes, and (iii) a distribution function that determines how the parent states are used to determine the state of the considered node. We reduce calculation complexity by limiting the number of states for nodes with parents to 2, i.e., each node that has a parent node has two states (e.g., "true, false", "yes, no", "present, absent", etc.). Nodes without parents, i.e., input nodes, can have more than two states (e.g., "high, medium, low"). Each state must be associated with a numerical value, starting at 0 for the lowest state. As we are using Boolean nodes, we calculate their CPT with the following generic function:

$$P(N|X_1,\ldots,X_n)=\left(\frac{\frac{S_{X_1}}{h_{X_1}}+\ldots+\frac{S_{X_n}}{h_{X_n}}}{n}\right)*C\left(S_{X_1},\ldots,S_{X_n}\right) \tag{1}$$

$$C\left(S_{X_1},\ldots,S_{X_n}\right)=\begin{cases}1 & , \prod_{i=1}^{n} S_{X_i}\neq 0 \vee T=0\\ 0 & , \prod_{i=1}^{n} S_{X_i}=0 \wedge T=1\end{cases} \tag{2}$$

Depending on parent nodes $X_1 \ldots X_n$ and their states $S_{X_1} \ldots S_{X_n}$, probability function P calculates the state of Boolean node N. h_{X_i} describes the highest possible numerical state of node X_i. P sums up all parent states (divided by h_{X_i}), divides them by the total number of parent nodes n, and multiplies the result by the return value of function C. Given the current state values of nodes $X_1 \ldots X_n$, C returns 1 if the product of all state values is not equal to zero or T is equal to zero. C returns 0 if the product of all state values is equal to zero and T is equal to 1. T is set to 1 if N's parent nodes are the result of an ontological Some or Min constraint (e.g., at least one or n relations have to be present to fulfill the constraint). If the state of one parent node equals zero in a Min or Some constraint relationship, the probability of the considered node will also be set to zero. In all other settings, such as All constraints, function C returns 1 and does not affect the final probability of the considered node. We use the Netica API and the described function to create the CPT for each node depending on all possible parent node states. In its current form, the function assumes equal weights for all parent nodes. If the user has selected different weights for each parent node in a given node set of Weight Classes/Individuals/Properties, we use the following function:

$$P(N|X_1,\ldots,X_n)=\left(\frac{S_{X_1}}{h_{X_1}}*w_{X_1}\right)+\ldots+\left(\frac{S_{X_n}}{h_{X_n}}*w_{X_n}\right)*C\left(S_{X_1},\ldots,S_{X_n}\right) \tag{3}$$

where w_{X_i} is the weight of parent X_i in the context of node N. The sum of all w must be 1. To use weights in the CPT construction, the user must define:

- a Weight Class that contains all weight individuals
- a Weight Property (OWL object property) that connects weight individuals to their child individual
- a Weight Property (OWL object property) that connects weight individuals to their parent individual
- a Weight Property (OWL data property) that contains the actual weight (0.0–1.0) of the weight individual

In the CPT construction of each node, all individuals of the selected Weight Class are iterated. If the properties of one weight individual match a given parent/child combination, the value of its data property is incorporated into the CPT construction function. The following requirements exist for using weights in the CPT construction: (i) each parent node of a given child node must provide a weight value, and (ii) all weight values must add up to 1.

4.4. Incorporation of existing knowledge facts

The previous phases have created a Bayesian network that includes relevant nodes, node links, node scales, and node weights supporting the CPT calculation. The current phase uses the ontological knowledge base and the selected Finding Properties to efficiently incorporate already modeled findings into the Bayesian network. This becomes useful if findings are mainly managed in the ontology and are to be reused in the Bayesian network. The selection of Finding Properties is optional, i.e., the network can be kept general if Finding Properties is not selected.

To incorporate findings, we obtain all Finding Properties selected by the user and get the corresponding referencing axioms of the ontology. Again, we consider all OWL object property, property assertion, some restriction, all restriction, min restriction, max restriction, and value restriction axioms that reference the considered Finding Property. The OWL class/individual referenced

within the axioms must be within the selected State Space Classes/Individuals, i.e., the state space of the nodes. Otherwise it would not be possible to enter the finding in the corresponding node of the Bayesian network.

The following code snippet shows class *A* individual *A_1*, with property *A_has_Rating* connecting *A_1* to State Space Individual *medium*:

```
<A rdf:about='#A_1'>
   <A_has_Rating rdf:resource="#medium"/>
</A>
```

To incorporate the finding, a new node *A_1_A_has_Rating* is created, connected to existing node *A_1*, and the finding is entered into the newly created node. If *A_1* has only one parent node (the newly created node), the state of *A_1* corresponds to the parent node's state. Please note that node *A_1_A_has_Rating* has a different state space (e.g., high [2], medium [1], and low [0]) than node *A_1* (Boolean scale). Therefore, the newly added parent node is required to transform findings with non-Boolean state spaces to the Boolean scale used at the remaining network nodes. The CPT construction functions described in Section 4.3 are used to map the different scales and to determine the state of Boolean node *A_1* given the state of node *A_1_A_has_Rating*:

$$P(A\_1|A\_1\_A\_has\_Rating) = \left(S_{A\_1\_A\_has\_Rating}/h_{A\_1\_A\_has\_Rating}\right)/n$$
$$P(A\_1|A\_1\_A\_has\_Rating) = (1/2)/1$$
$$P(A\_1|A\_1\_A\_has\_Rating) = 0.5$$

5. Prototype implementation

A prototype using the developed method was implemented as a Protege 4.0 plug-in using the Netica Java API to construct and modify the actual Bayesian network.

Fig. 5 shows the user interface of the developed plug-in. The following steps are performed by the user to construct the Bayesian network:

1. The user selects the class that contains the State Space Classes/Individuals and selects the State Value Property, which provides a numerical value for each state.
2. The user selects Node and Weight Classes/Individuals from the panel in the top left corner. For each class, the user can specify the type and whether individuals, sub-classes or only the selected class should be included in the Bayesian network. The 'Add Class' button adds the classes/individuals to the list located in the top right corner and the corresponding nodes are created in the Bayesian network.
3. The user selects Link, Weight, and Finding Properties from the middle panel. For each property, the user can specify the type and whether the nodes should be summarized or links should be reversed. The 'Add Property' button adds the properties to the list

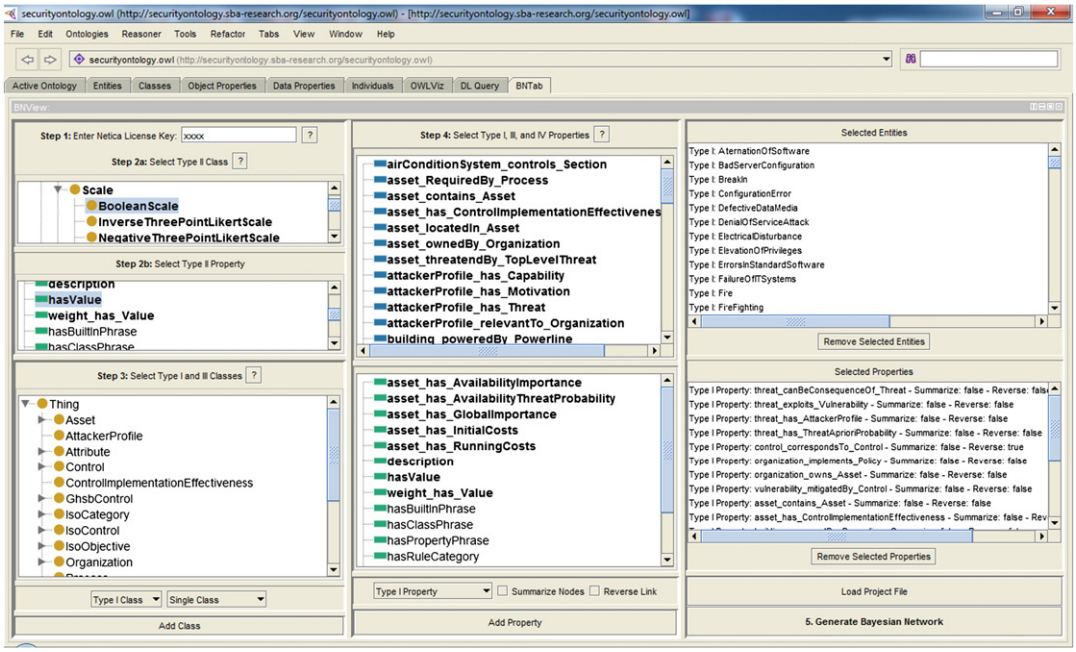

Fig. 5. Prototype implementation — user interface.

located in the bottom right corner. For Weight Properties, the user can select 'Weight - child', 'Weight - parent', and 'Weight - value' to define the necessary properties for the weight individuals.

- If the user checks the 'Summarize Nodes' check box, individuals or sub-classes of the property's range class will not be connected directly to its domain sub-classes/individuals. Instead, an intermediate node is created, connected to the domain sub-classes/individuals and to the range sub-classes/individuals. This ensures that the CPT in the domain sub-classes/individuals can be limited in its complexity.
- If the user checks the 'Reverse Link' check box, the link direction of the selected property is simply reversed in the Bayesian network. This feature is for convenience purposes only and enables the user to quickly compensate for missing inverse properties in the ontology.

4. As the last step, the user constructs the Bayesian network by clicking the 'Generate Bayesian Network' button. The plug-in will iterate over the selected properties and obtains all referencing axioms from the ontology for each property. The links between the created nodes are established according to the axiom descriptions.

The Protege 4.0 plug-in and installation instructions can be downloaded from REMOVED FOR DOUBLE-BLIND REVIEW

6. Use case – threat probability determination

This use case is based on an existing ontology from the information security domain. We show how the proposed method and its implementation can be used to establish a Bayesian network for threat probability determination. This section (i) introduces the security ontology, (ii) shows which classes/individuals and properties are used to establish the Bayesian network, and (iii) shows the final Bayesian network and demonstrates its capabilities.

6.1. The security ontology

The security ontology [12] was proposed based on the security relationship model described in the National Institute of Standards and Technology Special Publication 800-12 [26]. Fig. 6 shows the high-level classes and corresponding properties of the ontology. A threat gives rise to follow-up threats, represents a potential danger to the organization's assets and affects specific security attributes (e.g., confidentiality, integrity, and/or availability) as soon as it exploits a vulnerability in the form of a physical, technical, or administrative weakness. Additionally, each threat is described by potential threat origins (human or natural origin) and threat sources (accidental or deliberate source). Each vulnerability is assigned a severity value and the asset on which the vulnerability could be exploited. Controls have to be implemented to mitigate an identified vulnerability and to protect the respective assets by preventive, corrective, deterrent, recovery, or detective measures (control type). Each control is implemented by an asset class or a combination of several asset classes. Controls are derived from and correspond to best practice and information security standard controls (e.g., the German IT Grundschutz Manual [1] and ISO/IEC 27001 [19]) to ensure the incorporation of widely accepted knowledge. The controls are modeled on a highly granular level and are, therefore, reusable for different standards. When implementing the controls, compliance with various information security standards is implicit. To enrich the knowledge model with concrete information security knowledge, the German IT Grundschutz Manual was superimposed on the security ontology and more than 500 information security classes and 600 corresponding formal axioms were integrated into the ontological knowledge base (cf. [11]). The coded ontology follows the OWL-DL (W3C Web Ontology Language) [32] standard and ensures that the knowledge is represented in a standardized and formal way. The entire ontology is available as an online version at http://sec.sba-research.org and as an OWL file at http://securityontology.sba-research.org/securityontology.owl.

As the security ontology provides detailed knowledge about threat, vulnerability, and control dependencies, this knowledge can be utilized to build the Bayesian network for threat probability determination. The following section provides an in-depth

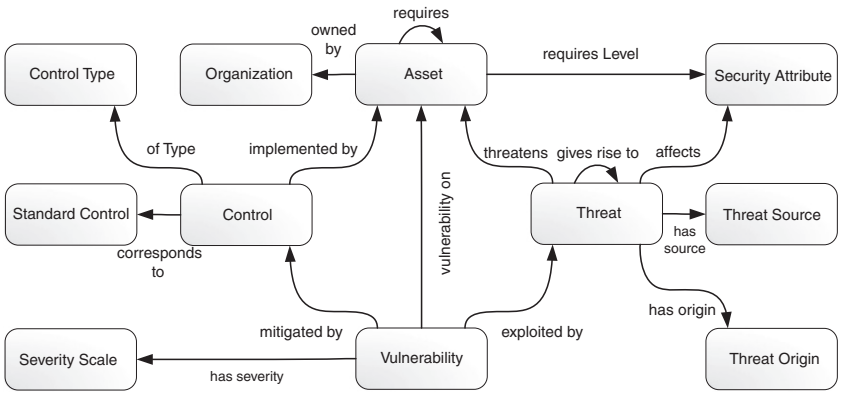

Fig. 6. Security ontology — main classes and properties.

Table 1
Selected Classes/Individuals.

Type	Class	Superclass	Includes	Scale
Node class	LowLevelThreat	Threat	5 Individuals	Boolean
Node class	TopLevelThreat	Threat	26 Individuals	Boolean
Node class	Vulnerability	–	93 Individuals	Boolean
Node class	AttackerProfile	–	1 Individual	Boolean
Node class	ThreatAprioriProbability	–	17 Individuals	Boolean
Node class	Control	–	114 Individuals	Boolean
Node class	ControlImplementationEffectiveness	–	14 Individuals	Boolean
Node class	CompliantControl	Control	114 Sub-classes	Boolean
Node class	Policy	Asset	108 Sub-classes	Boolean
Node class	Software	Asset	5 Sub-classes	Boolean
Node class	Movable asset	Asset	12 Sub-classes	Boolean
State space class	BooleanScale	Scale	2 Individuals	–
Weight class	Weight	-	Single class	–

description of the proposed method and shows how the method has been applied to create the Bayesian threat probability determination network.

6.2. Constructing the Bayesian network

The objective of this example Bayesian network, which was built to demonstrate the proposed ontology-based construction method, is to determine the probability of threats taking various influencing factors into account. The following influencing factors were identified by the domain expert:

- *Threats* influence each other. Predecessor threats influence the probability of successor threats (e.g., an active fire gives rise to the smoke threat).
- The probability of deliberate threats is influenced by an attacker's motivation and capability (*attacker profile*). Probabilities of non-deliberate threats are affected by *a priori probabilities*.
- The threat probability is influenced by the exploitation probability of *vulnerabilities* that can be exploited by the threat.
- The vulnerability exploitation probability is affected by available *control implementations* and their *effectiveness*.
- Whether a control is implemented or not depends on the implementation of *policies*, certain *movable assets*, and *software*.

Based on these dependencies, the user selects relevant classes/individuals from the security ontology (cf. Table 1).

Depending on the selected classes/individuals, the user selects relevant properties from the ontology (cf. Table 2). A property is relevant if it connects the selected classes/individuals and contributes to the final objective of the Bayesian network: determining threat probabilities.

The user has selected a Boolean scale state space for each node and defined the numerical value of each state space with the *hasValue* property. First, all selected individuals/sub-classes are converted into Bayesian network nodes. Second, the selected Link Properties are used to establish directed node links. Findings are incorporated by using the Finding Properties. The *attackerProfile_has_Capability*

Table 2
Selected properties.

Type	Property	Option
Link property	threat_canBeConsequenceOf_Threat	
Link property	threat_exploits_Vulnerability	Summarize
Link property	threat_has_AttackerProfile	
Link property	threat_has_ThreatAprioriProbability	
Link property	control_correspondsTo_Control	Reverse
Link property	organization_implements_Policy	
Link property	organization_owns_Asset	
Link property	vulnerability_mitigatedBy_Control	
Link property	asset_contains_Asset	
Link property	asset_has_ControlImplementationEffectiveness	
Link property	building_poweredBy_Powerline	
Link property	section_controlledBy_AirConditionSystem	
State value property	hasValue	
Weight property	weight_has_Child	Child
Weight property	weight_has_Parent	Parent
Weight property	weight_has_Value	Value
Finding property	attackerProfile_has_Capability	
Finding property	attackerProfile_has_Motivation	
Finding property	threatAprioriProbability_has_Probability	
Finding property	controlImplementationEffectiveness_has_Effectiveness	

Table 3
Weight Individuals.

Individual	has_Child	has_Parent	has_Value
Weight_1	Theft	Break In	0.1
Weight_2	Theft	Default Attacker Profile	0.6
Weight_3	Theft	Unauthorized Physical Access	0.2
Weight_4	Theft	No Regularly Reviewed Resource Inventory	0.05
Weight_5	Theft	No Use of Kensington Locks	0.05

property, for example, refers to the capability finding relevant to the selected Attacker Profile Node Individual. In this concrete case the *DefaultAttackerProfile* individual has capability *medium* and motivation *low*. Both findings are entered into the network. Findings for a priori probabilities and control implementation effectiveness values are entered into the Bayesian network in the same way. At this stage the network contains relevant nodes, links, and findings. All data has been gathered from the security ontology.

The next step addresses the CPT construction of each node. At each node, all individuals of the selected Weight Class (*Weight*) are iterated. At each weight individual, the values of the selected child and parent Weight Properties *weight_has_Child* and *weight_has_Parent* are compared to the current child and its parent nodes. If the combination matches, the value stored in property *weight_has_Value* is used as the weight for the parent/child combination. The following example shows how we determine the parent weights of the *Theft* node.

The parents of child node *Theft* are *Break In*, *Unauthorized Physical Access*, *No Use of Kensington Locks*, *No Regularly Reviewed Resource Inventory*, and *Default Attacker Profile*. Table 3 shows the weight individuals and individual values for properties *weight_has_Child*, *weight_has_Parent*, and *weight_has_Value*. All weights add up to 1 and all child/parent combinations provide a weight value. Therefore, the requirements for the CPT calculation are met. We calculated the conditional probability for each parent node combination using the following function (cf. Section 4.3):

$$P(T|B,D,P,R,U) = (B/1) * 0.1 + (D/1) * 0.6 + (P/1) * 0.2 + (R/1) * 0.05 + (U/1) * 0.05$$

T = Theft, B = Break In, D = Default Attacker Profile, P = Unauthorized Physical Access, R = No Regularly Reviewed Resource Inventory, and U = No Use of Kensington Locks. If the parent node is False, its variable is set to 0. If the parent node is True, we set its variable to 1. Table 4 shows the final CPT for the theft node. The final Bayesian network can be downloaded from REMOVED FOR DOUBLE-BLIND REVIEW

6.3. Evaluation

A workshop-based evaluation of the final Bayesian network was conducted. Three security domain experts with several years of experience in the field of information security risk management evaluated the resultant Bayesian network. The following questions were addressed: (i) Is the final Bayesian network correct in terms of both structure and CPTs? (ii) Is the final Bayesian network useful to the actual user? and (iii) Does reasoning over the final Bayesian network provide any benefit to the user?

6.3.1. Correctness

The workshop participants checked the correctness of the Bayesian network by comparing its nodes and links to the concepts and properties modeled in the ontology. Due to time constraints, only the threats *Inadmissible Temperature and Humidity*, *Theft*, and *Unauthorized Physical Access* and their parent nodes (threats, attacker profiles, vulnerabilities, and a priori probabilities) were considered. For each parent node, the evaluation team recursively checked the correctness of its parent nodes (e.g., in the case of a vulnerability node, they checked whether it was connected to the correct control nodes). In addition to node links, the workshop participants also checked the CPT of each node. The main question in the CPT evaluation was whether the given parent node state combinations result in realistic states for the considered node. Due to time constraints and the large amount of conditional probabilities (30,687), the experts were instructed to evaluate 10 randomly chosen parent node state combinations at every analyzed node.

As an example, we describe the evaluation of the *Inadmissible Temperature and Humidity* threat: First, its parent nodes are checked. *Inadmissible Temperature and Humidity* is connected to one predecessor threat (*Fire*), two vulnerabilities (*No Heat Detector* and *No Air Conditioning In Server Room*) and one a priori threat probability (*Threat A Priori Threat Probability 15*). The *No Heat Detector* vulnerability is connected to the *Heat Detector In Server Room Control*, which is connected to the *Heat Detector In Server Room Control Compliant Server Room* control specification, which is connected to its implementation node (*Heat Detector*). The heat detector node is connected to its control implementation effectiveness node, which, finally, is connected to the node containing the actual effectiveness value. The interested reader can follow these steps in the Bayesian network provided at REMOVED FOR DOUBLE-BLIND REVIEW

The evaluation showed that each of the analyzed nodes and links corresponded to the original ontological structure. The CPT evaluation revealed two findings: (i) the CPTs were calculated correctly according to the calculation function and the provided weight values, and (ii) more work is necessary to fine-tune the weight values in the ontology. We addressed the second CPT evaluation finding by, together with the domain experts, incorporating reasonable weights for the threat nodes *Inadmissible Temperature and Humidity*,

Table 4
CPT for theft node.

False	True	P	B	U	R	D
1	0	False	False	False	False	False
0.4	0.6	False	False	False	False	True
0.95	0.05	False	False	False	True	False
0.35	0.65	False	False	False	True	True
0.95	0.05	False	False	True	False	False
0.35	0.65	False	False	True	False	True
0.9	0.1	False	False	True	True	False
0.3	0.7	False	False	True	True	True
0.9	0.1	False	True	False	False	False
0.3	0.7	False	True	False	False	True
0.85	0.15	False	True	False	True	False
0.25	0.75	False	True	False	True	True
0.85	0.15	False	True	True	False	False
0.25	0.75	False	True	True	False	True
0.8	0.2	False	True	True	True	False
0.2	0.8	False	True	True	True	True
0.8	0.2	True	False	False	False	False
0.2	0.8	True	False	False	False	True
0.75	0.25	True	False	False	True	False
0.15	0.85	True	False	False	True	True
0.75	0.25	True	False	True	False	False
0.15	0.85	True	False	True	False	True
0.7	0.3	True	False	True	True	False
0.1	0.9	True	False	True	True	True
0.7	0.3	True	True	False	False	False
0.1	0.9	True	True	False	False	True
0.65	0.35	True	True	False	True	False
0.05	0.95	True	True	False	True	True
0.65	0.35	True	True	True	False	False
0.05	0.95	True	True	True	False	True
0.6	0.4	True	True	True	True	False
0	1	True	True	True	True	True

Theft, *Unauthorized Physical Access*, and their parent nodes. To eliminate any risk of bias, we created a golden data set with the experts, which includes 18 predefined network states. Based on the golden data set and according to [31] we checked the monotonicity[3] of the Bayesian network. For our threat probability example this means that higher a priori probabilities, higher vulnerability exploitation probabilities and higher attacker effectiveness values result in higher threat probabilities. Each of the 18 predefined network states is defined by different output probability distributions of the mentioned threat nodes given different input probability distributions of their parent nodes. We set the parent nodes in the Bayesian network according to the golden data set for each predefined network state and compared the resulting threat probabilities to the values defined by the experts in the golden data set. As the 18 predefined network states were not exhaustive, the experts tested additional states by varying the input parameters of the threat nodes. The network behaved as defined in the golden data set and, thus, as expected by the experts.

6.3.2. Usefulness

We evaluated the usefulness of the developed Bayesian network construction tool and the Bayesian network that was constructed in our case study. The workshop team, consisting of information security risk management domain experts, was asked to rate the usefulness of the tool and the constructed network.

The team confirmed the usefulness of the tool because (i) it allowed them to efficiently construct the Bayesian network without any external support, (ii) the necessary knowledge can be centrally managed in the ontology and is automatically transferred to the Bayesian network (e.g., in the case of newly discovered vulnerabilities), and (iii) it allowed them to use existing collaborative ontology editing tools to change the structure of the Bayesian network.

The workshop team also confirmed the value of the Bayesian network in the field of threat probability determination. The network shows how changes in control implementation effectiveness, a priori threat probability and attacker effectiveness affect the final threat probability. Dependencies between the node types are already modeled and fine-tuning of weight values is possible.

6.3.3. Reasoning

The reasoning capabilities of the network were tested by checking how the output changed with varying input parameters (control effectiveness, a priori probability, and attacker effectiveness). For example: according to existing findings, the initial network indicates an initial probability of 29.3% for the inadmissible temperature and humidity threat. Changing the effectiveness of

[3] The main variable of interest behaves monotonically in the observable variables, in the sense that higher values for the variable of interest become more likely with higher-ordered observations [31].

the heat detector from medium to high instantly results in a decreased threat probability of 21%. Changing the a priori probability from low to high and keeping the high effective heat detector results in a threat probability of 46%. Using a low effective heat detector at a high a priori threat probability results in 62.6% threat probability.

The workshop team manipulated control effectiveness, a priori probability, and attacker effectiveness values at the threats *Inadmissible Temperature and Humidity*, *Theft*, and *Unauthorized Physical Access*. The resultant changes in the threat probability values showed the behavior expected by the evaluation team. The team confirmed that the Bayesian network provides correct threat probability ranges for the given input parameters. However, the experts noted that the correctness of these values is solely confirmed based on the nodes' CPTs and their parent node states. As threat probabilities are fictitious values, their correctness cannot be confirmed with real-world reference values.

6.3.4. Usability

The workshop team consisting of risk management domain experts evaluated the user interface usability of the developed tool. Users were asked to select relevant classes, individuals, and properties to construct a simple Bayesian network without any explanation. As the workshop team did not include any Bayesian network experts, it first failed at constructing a valid Bayesian network. In a second step we explained the tool and Bayesian network basics to the users. With this additional knowledge, the team was able to construct a valid Bayesian network. The usability evaluation results show that a short but solid documentation is required to enable regular users to construct valid Bayesian networks with the developed tool.

7. Discussion

The use case presented in this article produced a basic Bayesian network containing 557 nodes, 579 directed links, and 30,687 conditional probabilities. Based on an existing ontology, it enables a domain expert to create a Bayesian network within 5 min. While we did not try to manually construct such a large network, we estimate the manual construction time required to be several hours. The estimation is based on real-world experiments in which we manually constructed 25 nodes. The construction steps for each node are: (i) checking the node name and type in the ontology, (ii) creating a new node within the Netica application, (iii) defining state spaces (e.g., "true, false") including numerical value assignment, (iv) checking to which nodes the current node should be connected, and (v) connecting the node to each of its parent and child nodes. In an experiment we were able to construct 25 nodes in 6 min and 38 s, resulting in an average node building time of 15.92 s. If the user were able to construct one node including its links within 16 s, the entire network would be finalized in about 2.5 h. This estimation does not include the calculation of 30,687 conditional probabilities. The proposed method saves time, especially in the construction of Bayesian networks with more than 18 nodes.

Fig. 7 shows the estimated time effort for constructing a Bayesian network by manual means and with the proposed method. Relevant assumptions and settings: (i) both construction methods derive the required knowledge from an existing ontology, (ii) a virtualized Intel Core 2 Duo 2.13 GHz processor with 2 GB memory and Microsoft Windows XP Professional (32-bit) operating system is used to run the automated Bayesian network construction method, (iii) the user requires 16 s to manually construct one node, (iv) the proposed method requires a network size-independent setup time of 2 min, and (v) a factor of 0.006 min was added for each node constructed with the proposed method (derived from the 3 min excluding setup time that were required to construct the 557 nodes). It has to be stressed that this is an estimation and that the real time for constructing a Bayesian network with the proposed method depends very much on the underlying ontology. What we can say with certainty is that the complexity of manually constructed Bayesian networks grows at least linearly with the number of nodes and links. The proposed method enables us to significantly reduce the construction time by reusing existing ontologies. Please note that this comparison is only valid in a setting in which we want to construct Bayesian networks based on already existing ontologies. As the main purpose of the developed approach is to efficiently reuse already modeled knowledge, it cannot be used to reduce the time required for constructing Bayesian networks from scratch.

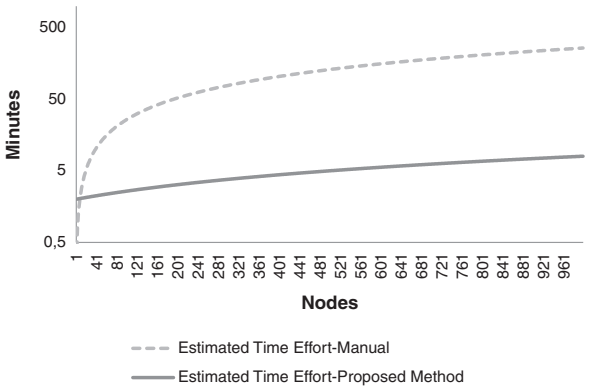

Fig. 7. Bayesian network construction — estimated time effort.

7.1. Limitations

The Boolean node assumption in the CPT construction is a limitation of the proposed approach. Currently, Boolean parent node states and their context-specific weights are used to calculate the first Boolean node state (e.g., "true") of the child node. The child node's second Boolean state (e.g., "false") is calculated by subtracting the value of the first state from 1. However, general approaches for constructing CPTs exist (cf. [33,7,29,10]). Fenton et al. [10] most recently proposed an approach to constructing CPTs for ranked nodes with a minimal amount of expert elicitation. CPTs are represented by means of parametric probability functions. The probability of the child node is defined as a weighted function of the parent node values. The approach by Fenton et al. is not restricted to Boolean nodes. Ranked nodes with any number of states can be used. Further research may lead to the incorporation of Fenton's approach into the proposed ontology-based approach for constructing Bayesian networks. For this, it would be necessary to store a probability distribution function for each parent/child node combination in the ontology. In the same way we currently store node weights for specific parent/child combinations in the ontology, we would be able to use the stored distribution function to construct CPTs as described by Fenton et al. However, domain experts would have to agree on appropriate distribution functions for all parent/child node combinations. As the definition and incorporation of these distribution functions into the ontology would increase the work load in the ontology engineering process significantly, the developed method provides a trade-off between complexity and accuracy by using the Boolean node assumption.

8. Conclusion

When creating Bayesian networks, analysts are confronted with the following challenges: (i) What are the relevant variables for my problem? (ii) How do these variables relate to each other? (iii) What are potential states of the identified variables? and (iv) What is the weight of each variable in the context of its siblings? To address these challenges, this paper has proposed an ontology-based method for the construction of Bayesian networks and has shown its applicability with a use case in the field of threat probability determination. As the method chosen to address these challenges was the use of existing ontologies, the research questions of this article were:

- RQ1: How can existing ontologies be used to construct the graphical structure of Bayesian networks?
- RQ2: How can existing ontologies be used to enrich Bayesian networks with CPTs and concrete findings while preserving the semantic constraints of the ontology?

We have shown with the proposed method how the use of existing ontologies (i) enables the semi-automatic creation of Bayesian networks, (ii) reduces the complexity of modeling Bayesian networks by using high-level classes and properties to integrate relevant sub-classes into the Bayesian network, and (iii) provides the possibility of maintaining the underlying knowledge body of Bayesian networks.

Compared to existing approaches, we provide a method that uses already existing ontologies without the requirement for specific extensions for Bayesian network construction. Furthermore, the method enables the incorporation of existing knowledge facts (findings) and the construction of CPTs, where the weights of the parent nodes are derived from the ontology and incorporated into the CPT construction function. In contrast to existing approaches, the presented method preserves the ontological semantics in CPT construction.

The limitations of the proposed method are: (i) functions for calculating conditional probability tables are not provided by the ontology and have to be modeled externally (currently only the parent node weights and states are derived from the ontology), (ii) Boolean node assumption in the CPT construction, and (iii) human intervention is still necessary to some extent if the ontology does not provide a knowledge model that fits the domain of interest exactly.

Further research could address these limitations by incorporating relevant CPT creation functions directly into the ontology.

The proposed method significantly reduces the effort required to create Bayesian networks by interpreting classes, individuals, and properties of existing ontologies.

References

[1] BSI, IT Grundschutz Manual, , 2004.
[2] G. Bucci, V. Sandrucci, E. Vicario, Ontologies and Bayesian networks in medical diagnosis, System Sciences (HICSS), 2011 44th Hawaii International Conference on, Jan. 2011, pp. 1–8.
[3] W. Buntine, A guide to the literature on learning probabilistic networks from data, IEEE Transactions on Knowledge and Data Engineering 8 (2) (Apr 1996) 195–210.
[4] P.C. Costa, K.B. Laskey, K.J. Laskey, Uncertainty reasoning for the semantic Web I. Chapter PR-OWL, A Bayesian Ontology Language for the Semantic Web, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 88–107.
[5] A. Devitt, B. Danev, K. Matusikova, Constructing Bayesian networks automatically using ontologies, Proceedings of Second Workshop on Formal Ontologies Meets Industry (FOMI 2006), 2006.
[6] Z. Ding, Y. Peng, A probabilistic extension to ontology language owl, Proceedings of the 37th Hawaii International Conference on System Sciences, 2004.
[7] M. Druzdzel, L. van der Gaag, Elicitation of probabilities for belief networks: combining qualitative and quantiative information, Proc 11th Annual Conference on Uncertainty in Artificial Intelligence, 1995, pp. 141–148.
[8] M. Druzdzel, L. van der Gaag, Building probabilistic networks: "Where do the numbers come from?" Guest editors' introduction, IEEE Transactions on Knowledge and Data Engineering 12 (4) (July-August 2000) 481–486.
[9] M.J. Druzdzel, F.J. Diez, Combining knowledge from different sources in causal probabilistic models, Journal of Machine Learning Research 4 (2003) 295–316.
[10] N. Fenton, M. Neil, J. Caballero, Using ranked nodes to model qualitative judgments in Bayesian networks, IEEE Transactions on Knowledge and Data Engineering 19 (2007) 1420–1432.

[11] S. Fenz. Ontology- and Bayesian-based Information Security Risk Management. PhD thesis, Vienna University of Technology, October 2008.
[12] S. Fenz, A. Ekelhart, Formalizing information security knowledge, Proceedings of the 4th ACM Symposium on Information, Computer, and Communications Security, ACM, New York, NY, USA, 2009, pp. 183–194, 978-1-60558-394-5.
[13] N. Friedman, D. Koller, Being Bayesian about network structure. a Bayesian approach to structure discovery in Bayesian networks, Machine Learning 50 (2003) 95–125.
[14] T. Gruber, A translation approach to portable ontology specifications, Knowledge Acquisition 5 (2) (April 1993) 199–220.
[15] D. Heckerman, A tutorial on learning with Bayesian networks, Technical Report MSR-TR-95-06, Microsoft Research, Advanced Technology Division, Redmond, WA 98052, November 1996.
[16] D. Heckerman, J. Breese, K. Rommelse, Troubleshooting under uncertainty, Technical Report MSR-TR-94-07, Microsoft Research, Redmond, Washington, January 1994.
[17] E.M. Helsper, L.C. van der Gaag, Building Bayesian networks through ontologies, in: F. van Harmelen (Ed.), ECAI 2002: Proceedings of the 15th European Conference on Artificial Intelligence, IOS Press, 2002, pp. 680–684.
[18] M.B. Ishak, P. Leray, N.B. Amor, Ontology-based generation of object oriented Bayesian networks, Proceedings of the 8th Bayesian Modelling Applications Workshop, 2011, pp. 9–17.
[19] ISO/IEC, ISO/IEC 27001:2005, Information technology — security techniques — information security management systems — requirements, , 2005.
[20] E.R.H. Jr., N.F. Ebecken, Towards efficient variables ordering for Bayesian networks classifier, Data & Knowledge Engineering 63 (2) (2007) 258–269.
[21] R. Kennett, K. Korb, A. Nicholson, Seabreeze prediction using Bayesian networks, PAKDD '01: Proceedings of the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer-Verlag, London, UK, 2001, pp. 148–153.
[22] W. Lam, F. Bacchus, Learning Bayesian belief networks: an approach based on the mdl principle, Computational Intelligence 10 (1994) 269–293.
[23] P. Larrafiaga, C. Kuijpers, R. Murga, Y. Yurramendi, Learning Bayesian network structures by searching for the best ordering with genetic algorithms, IEEE Transactions on Systems, Man, and Cybernetics — Part A 26 (1996) 487–493.
[24] R. Neapolitan, Learning Bayesian networks, Prentice Hall, 2003.
[25] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, W.R. Swartout, Enabling technology for knowledge sharing, AI Magazine 12 (3) (1991) 36–56.
[26] NIST, An introduction to computer security — the NIST handbook, Technical Report, NIST (National Institute of Standards and Technology), October 1995, Special Publication 800-12.
[27] J. Pearl, Causality: Models, Reasoning, and Inference, Cambridge University Press, March 2000.
[28] S. Sadeghi, A. Barzi, J. Smith, Ontology driven construction of a knowledgebase for Bayesian decision models based on umls, Studies in Health Technology and Informatics 116 (2005) 223–228.
[29] M. Takikawa, B. D'Ambrosio, Multiplicative factorization of noisy-max, Proceedings of the Uncertainty in AI conference, 1999.
[30] J.G. Torres-Toledano, L.E. Sucar, Bayesian networks for reliability analysis of complex systems, IBERAMIA '98: Proceedings of the 6th Ibero-American Conference on AI, Springer-Verlag, London, UK, 1998, pp. 195–206.
[31] L.C. van der Gaag, H.J.M. Tabachneck-Schijf, P.L. Geenen, Verifying monotonicity of Bayesian networks with domain experts, International Journal of Approximate Reasoning 50 (March 2009) 429–436.
[32] W3C, OWL — web ontology language, http://www.w3.org/TR/owl-ref/February 2004.
[33] M. Wellman, Fundamental concepts of qualitative probabilistic networks, Artificial Intelligence 44 (1990) 257–303.
[34] Y. Yang, J. Calmet, Ontobayes: an ontology-driven uncertainty model, International Conference on Intelligent Agents, Web Technologies and Internet Commerce, 2005.
[35] H.-T. Zheng, B.-Y. Kang, H.-G. Kim, An ontology-based Bayesian network approach for representing uncertainty in clinical practice guidelines, in: F. Bobillo, P.C.G. da Costa, C. d'Amato, N. Fanizzi, F. Fung, T. Lukasiewicz, T. Martin, M. Nickles, Y. Peng, M. Pool, P. Smrz, P. VojtÃ¡s (Eds.), CEUR Workshop Proceedings, vol. 327, URSW, 2007CEUR-WS.org.

Dr. Stefan Fenz is a researcher at Vienna University of Technology and SBA Research. In 2010 Stefan worked as a visiting scholar at Stanford Center for Biomedical Informatics Research at Stanford University. His primary research is on information security, with a secondary interest in semantic technologies and named entity recognition. Stefan received an MSc in software engineering and internet computing from Vienna University of Technology, an MSc in political science from University of Vienna, an MSc in business informatics from Vienna University of Technology, and a PhD in computer science from Vienna University of Technology. He is a member of the IFIP WG 11.1 — Information Security Management, the IEEE Systems, Man, and Cybernetics Society and ISC2.