An Ontology-Based Approach to Interoperability for Bayesian Agents

Elder Rizzon Santos
ersantos@inf.ufrgs.br

Moser Silva Fagundes
msfagundes@inf.ufrgs.br

Rosa Maria Vicari
rosa@inf.ufrgs.br

Instituto de Informática - Universidade Federal do Rio Grande do Sul (UFRGS)
Po. B 15.064 - 91.501-970 - Porto Alegre - RS - Brazil. Phone/Fax: +55 51 33166161

ABSTRACT
This paper presents an ontology-based approach to promote the interoperability among agents that represent their knowledge through Bayesian networks. This research relies on semantic web foundations to achieve knowledge interoperability in the context of multiagent systems. Our first step was the specification of an ontology that formalizes the structures of the Bayesian network representation. Once handled the issue of the knowledge representation, we specify how a Bayesian agent operates such representation. Thus, we define a model of internal architecture to support Bayesian agents in the knowledge sharing and maintenance tasks.

Categories and Subject Descriptors
I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence—intelligent agents; I.2.4 [Artificial Intelligence]: Knowledge Representation Formalisms and Methods—semantic networks

General Terms
Design, Experimentation, Standardization

Keywords
semantic web, ontology, interoperability, bayesian networks, agent architecture

1. INTRODUCTION
Studies on interoperability on the context of Artificial Intelligence have been done mostly for the communication among intelligent agents. Today, such researches can be applied for the development of the semantic web, which is the mainstream on Internet technology. Considering the semantic web as an open system, populated by autonomous agents carrying out activities in behalf of its owners, interoperability issues (i.e. how these agents from different domains and with different goals will share their knowledge, co-operate and maximize the utility of the whole system) arise.

This paper presents an agent architecture that allows the interoperability of knowledge among Bayesian agents. Specifically, we are concerned on how heterogeneous Bayesian agents may exchange their knowledge. We consider Bayesian agents those that have their knowledge expressed through Bayesian networks. The fact that the agents use the same knowledge representation (i.e. Bayesian networks) does not guarantee that it is implemented in an interoperable way.

2. BAYESIAN NETWORK ONTOLOGY
One of the contributions of this research is the specification of an ontology to formalize the Bayesian network knowledge representation. Our ontology specification extended the concepts defined in [2], allowing a more broad utilization of the ontology. One of these utilizations is an agent architecture for interoperability, described in the section 3.

A discrete Bayesian network consists of a DAG (Directed Acyclic Graph) and a set of conditional probability distributions [1]. Each node in the network, called chance node, corresponds to exactly one discrete random variable which has a finite set of mutually exclusive states. The directed arcs specify the causal relation between the random variables. Each random variable associated with a chance node has a conditional probability distribution.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

AAMAS’07 May 14–18 2007, Honolulu, Hawai’i, USA.
Copyright 2007 IFAAMAS.

978-81-904262-7-5 (RPS) ©2007 IFAAMAS
value of these properties is an individual of the ChanceNode class. A ChanceNode individual has a chance variable associated to its definition. Such variable is specified by the hasChanceVariable property. This property allows only individuals of the classes PriorChanceVariable and ConditionalChanceVariable. Such constraint is necessary in order to differentiate prior nodes variables from non-prior nodes.

Before defining a chance variable it is necessary to define a state and its related concepts (Figure 2). A state is represented by the State class, which has only the hasLabel property responsible for the node identification. The State class has two direct subclasses. The first denotes a chance associated with a state and it is called StateProbability. It is defined by the inherited hasLabel property and the probability property (float data type). The second specialization is named ConditionalState and it specifies the multiple conditional chances associated with a state. A set of ConditionalState individuals constitutes a Conditional Probability Table (CPT). The ConditionalState class has two properties: the inherited hasLabel, which represents the label of the state and the hasConditionProbability property, which references multiple individuals of the ConditionalProbability class.

The ConditionalProbability class represents the conditional chances associated with a state. This class is defined by the probability and the hasCondition properties. The former is a float data type property that represents the numerical probability of a variable’s state under the conditions specified in the hasCondition property. This property references multiple individuals of the Condition class, and it denotes the conditions imposed in the probability of a state. The Condition class is constituted by a conditioning node and a state of this node, respectively referenced by the properties hasNode and hasState. The individual referenced by the hasNode property must be a ChanceNode since only chance nodes have random variables. The hasState property references an individual of the State class that indicates the specific state of the conditioning chance variable.

The ChanceVariable class represents a set of mutually exclusive states. The states, also called events or choices, correspond to the domain of the variable, which can be discrete or continuous. In this work we consider only discrete variables (finite sets). The ChanceVariable class specifies the hasLabel, hasState and hasMarginalDistribution properties. The first property identifies and provides an unique name for the variable. The second specifies the necessity of at least one state (represented by State individuals) associated with a variable (i.e. true or false in the context of a boolean variable). The last property returns the computed marginal distribution for the chance variable, and it references multiple StateProbability individuals. Each individual represents a state and its computed chance of occurrence.

It was necessary to differentiate prior node variables from non-prior ones, since a non-prior node has a CPT, and a prior node has only states and probabilities without conditioning variables. Thus, the classes PriorChanceVariable and ConditionalChanceVariable were created as subclasses of ChanceVariable. The difference between these two subclasses lies in the hasState property constraint. In the PriorChanceVariable class the hasState property has been restricted and it can reference only StateProbability individuals. As stated earlier, a state probability represents a state and its chance of occurrence. The set of StateProbability individuals referenced by the hasState property denotes all possible states associated with a prior chance variable. The hasState property of ConditionalChanceVariable class has also been constrained and only ConditionalState individuals can be assigned to it. The ConditionalState individuals represent a Conditional Probability Table of a variable associated with a non-prior node.

In our definition, a situation is a particular configuration that a probabilistic network assumes given a set (possibly empty) of evidences of events occurrence. When the evidences of events are reflected in the network, a new situation arises. Such situations are useful to keep the history of modifications of a Bayesian network.

An evidence, represented by the Evidence class, corresponds to any information regarding the state of a variable from a probabilistic network. The Evidence class is composed by a node, a label, and a chance, represented by properties hasNode, hasLabel and probability, respectively. The hasNode property can reference only individuals of the ChanceNode class, since chance nodes are the only kind of node that represent random events.

A situation, represented by the Situation class, has two properties. The first is the hasBayesianNetwork property used to reference the network individual whose configuration corresponds to the given situation. The second property is the hasEvidence that corresponds to the set of evidences that originates the situation.

In order to establish a link between two sequential situations we created a class named SituationTransition. This class is described by hasPriorSituation property and hasPosteriorSituation property, which represents a prior and a posterior situation, respectively.

3. BAYESIAN AGENT INTERNAL ARCHITECTURE

The agent architecture for interoperability presented in this section extends and generalizes the architecture proposed in [2]. The main goal of our architecture is to enhance the interoperability of Bayesian network knowledge among agents. The interoperability is achieved by an ontology-based approach to represent the uncertain knowledge of the agent. Following, we detail the agents’ internal architecture, depicted in the Figure 3.

The Agent Implementation Specific Components are not specified by this architecture since they relate to the particular purpose of each agent design. However, we specify the way they interact with the other architecture components. Usually, the Agent Implementation Specific Components define the manner that the agent reasons about its goals and how it achieves them (i.e. planning and goal de-
The first component of the architecture is the Perception Handler, which receives and forwards the perceptions to the respective components capable of interpreting them. The characteristics of a perception are taken into account to decide which component will receive it. Since in the context of this work we are dealing with interoperability among Bayesian agents, we focus on two particular categories of perception: Bayesian Network Knowledge and Query. The first corresponds to individuals of the ontology presented in the section 3. The perceptions of this category are forwarded to the Knowledge Base (KB) Update component. The second corresponds to queries about the agent’s knowledge that are forwarded to the KB Query component.

The second component of the architecture is the KB Update. Its purpose is to evaluate the incoming Bayesian Network Knowledge, and insert the selected ones into the knowledge base as individuals of the Bayesian network ontology. The information to be inserted is selected following the criteria defined by the designer. A simple implementation of this component performs insertions in the KB without restrictions. A more sophisticated implementation interacts with the KB Query component to retrieve already inserted Bayesian information to constrain the information to be inserted.

Our knowledge base is constituted by the Bayesian network ontology, detailed in the section 2, and its individuals. It stores the Bayesian networks situations, the transitions between situations and the evidences. The base can contain multiple different Bayesian networks. Any modification in a Bayesian network characterizes a new situation, and the sequence of situations represents a history of a network. The history may be useful for an agent planning, for example.

In order to perform probabilistic reasoning in the Bayesian networks stored in the knowledge base, we specify the Bayesian Inference component. Its inputs are the Current Situation of a Bayesian network and a set of Evidences. The Bayesian Inference output is the New Situation with its probabilities recalculated considering the Evidences. It is worth to point out that both situations are individuals of the Bayesian-Network class and that the Evidences are Evidence class individuals. The New Situation resulting from the inference process constitutes the most up-to-date knowledge that the agent has about its domain. The presence of this component is indispensable since updated knowledge is necessary to support the agent decisions and actions.

The KB Query component receives queries from Agent Implementation Specific Components, Perception Handler and KB Update. These queries can return events (states) and their occurrence probabilities, causal relations between variables and other information that can be inferred from the Bayesian networks knowledge base. Queries from the agent specific components are usually performed to aid the agent in its decision making process. The queries forwarded by Perception Handler are related to knowledge that external agents need to be informed about. Finally, the queries from the KB Update component are executed with the purpose of selecting which information will be inserted on the KB.

The Agent Implementation Specific Components are not specified by the architecture since they relate to the particular purpose of each agent design. However, we specify the way they interact with the architecture components. Usually, this component define the manner that the agent reasons about its goals and how it achieves them (i.e. planning and goal deliberation).

The core of the interoperability relies on the Bayesian network ontology. It provides the fundamental domain concepts among the Bayesian agents making possible their knowledge exchange. The architecture supports the knowledge representation in a broader way, not only in the interaction with other Bayesian agents. The architecture also provides the means for the knowledge maintenance.

4. CONCLUSION

We define an internal architecture that provides support for knowledge intense agents to interoperate their knowledge with other agents. In this case, the interoperability is in the scope of Bayesian knowledge regarding an adequate way to express it. Besides that, we provide resources for maintaining such Bayesian knowledge. Maintenance features allow the execution of updates, queries and an inference process to propagate evidences to the corresponding networks present in the knowledge base. The interoperability provided by this architecture aids agent specific decision making since it facilitates the discovery of new knowledge, allowing the agent to consider evidences that were not part of its original KB.

5. REFERENCES