Modeling Intelligent Agents through Causality Theory

Hector G. Ceballos and Francisco J. Cantu
Tecnologico de Monterrey, Mexico
{ceballos, fcantu}@itesm.mx

Abstract

We introduce Causal Agents*, a methodology and agent architecture for modeling intelligent agents based on Causality Theory. We draw upon concepts from classical philosophy about metaphysical causes of existing entities for defining agents in terms of their formal, material, efficient and final causes and use computational mechanisms from Bayesian causal models for designing causal agents. Agent's intentions, interactions and performance are governed by their final causes. A* Semantic Bayesian Causal Model*, which integrates a probabilistic causal model with a semantic layer, is used by agents for knowledge representation and inference. Agents are able to use semantic information from external stimuli (utterances, for example) which are mapped into the agent's causal model for reasoning about causal relationships with probabilistic methods. Our theory is being tested by an operational multiagents system implementation for managing research products.*

1 Introduction

The design of intelligent agents includes the problem of finding schemes flexible enough for representing knowledge and inference mechanism so that agents are capable of perceiving their environment and acting upon it. Agents are either software programs or moving robots which are characterized by being autonomous, proactive and rational entities. Agent behavior is driven by predefined rules or models and even represents preferences among goal states. Additionally, intelligent agents can be capable of learning from experience, improve its performance according to a utility function and respond under uncertainty conditions.

Several approaches have been proposed to model such intelligent agents. Some methodologies have focused in agent and task modeling and have proposed standard inference structures[1]. Others have proposed to use non-monotonic logics and probabilistic reasoning to tackle the frame problem[2] that limits agent capacity to learn and evolve.

In the case of probabilistic reasoning, development of Bayesian Causal Models [3] has allowed designers to formalize some aspects of causality theory. And even when its semantics is flexible enough to model complex problems and support uncertainty, it has to deal with the problem of interoperability between models. It has been pointed out in the research community the necessity of having contextual information to associate meaning to elements of the models[4].

On this sense, the Description Logics community has proposed the use of ontologies to represent shared meaning. The Tarskian semantic used to interpret symbols and relations makes it possible to ground them to real world objects.

This paper is organized as follows: Section 2 presents the theory that supports the agent architecture. Section 3 describes Causal Agents and their architecture. Section 4 presents an overview of system implementation. Section 5 summarizes related work and section 6 presents conclusions and future work.

2 Background

Our proposal is inspired in the theory of metaphysics and causality proposed by Aristotle and revised by Thomas Aquinas. Some aspects of this theory has been recently formalized in the form of Bayesian causal models.

2.1 Metaphysics, Causality and Intentionality

Metaphysics [5] developed by Aristotle, and revised by Aquinas, provides a general conceptualization of reality. It conceives reality constituted by *entities* or *beings* that have an essence people can recognize. Entity essence is defined by its characteristics or accidents and is captured by human mind through abstraction.

Aristotle classifies accidents in intrinsic, extrinsic and mixed. *Intrinsic accidents* includes quantitative (age, size, etc.), qualitative (color, shape, etc.) and relational (fatherhood, nationality, etc.) accidents, that is, what internally identify an entity. *Extrinsic accidents* are relative to time (birth date, duration, etc.), place (position), possession (property) and disposition (sit, stand, etc.). *Mixed accidents* explain interaction among entities: action is present in an entity when originates movement or change in another, meanwhile passion is present in entities that receive passively the action of another.

Aristotle considers *change* as a transition of an individual from one state to another, whenever the individual be able to reach the final state. He defined *potence* as the entity capacity to show certain accident. *Act*, opposite to potence, is the actual presence of the accident on the entity. Having certain accident in potence doesn't imply that the entity presents it actually, but just denotes possibility.

Causality refers to the set of all particular "causal" or "cause-effect" relations. Most generally, causation is a relationship that holds between events, properties, variables, or states of affairs. Causality implies at least some relationship of dependency between the cause and the effect. Cause chronologically precedes the effect.

According to Aristotle's theory, all possible causes fall into several wide groups, the total number of which amounts to the ways the question "why" may be answered; namely, by reference to the matter or the substratum (*material cause* or part-whole causation); to the essence, the pattern, the form, or the structure (*formal cause* or whole-part causation); to the primary moving change or the agent and its action (*efficient cause* or agent causation); and to the goal, the plan, the end, or the good (*final cause* or agent intention).

Brentano defined *intentionality* as a characteristic of "mental phenomena", by which they could be distinguished from "physical phenomena". Every psychical, or mental, phenomenon has a content, and is directed at an object (the *intentional object*). Every belief, desire, etc. has an object that it is about: the believed, the wanted. The property of being intentional, of having an intentional object, is the key feature to distinguish mental phenomena and physical phenomena, because physical phenomena lack intentionality altogether.

2.2 Bayesian Causal Models

Pearl[3] proposes a *semi-markovian model* to represent a probabilistic causal model, i.e. a model where some variables are observed and others don't. Probabilistc causal model can be expresed by:

$$M = \langle V, U, G_{VU}, P(v_i|pa_i, u_i)\rangle \tag{1}$$

where V is the set of observed variables, U is the set of unobserved variables, G_{VU} is a causal graph consisting of variables in $V \times U$ and $P(v_i|pa_i, u_i)$ is the probabilistic function of V_i which value depends on the value of its parents (PA_i) in the graph and the value of unobserved variables (U_i) affecting it. A *markovian causal model* is a special case of probabilistic causal models where it doesn't exist unobserved variables, i.e. $U = \emptyset$

The simplest operation on causal models is *prediction*, which consists on calculate the *a priori* probability of a set of variables Y , i.e. $P(y)$. *Intervention* operation consists on setting a variable or set of variables to a given value and to calculate the probability of the rest of the variables in the new model. Atomic interventions are performed over a single variable and is equivalent to lifting X_i from the influence of the old mechanism $x_i = f(pa_i, u_i)$ and placing it under the influence of a new mechanism that sets the value x_i while keeping all other mechanisms unperturbed. Pearl represents atomic intervention like $do(X_i = x_i)$, $do(x_i)$ or $\hat{x}_i$.

A model modified by an intervention $do(x_i)$ can be solved for the distribution of other variable X_j, yielding to the notion of *causal effect* of X_i on X_j, which is denoted $P(x_j|\hat{x}_i)$. The question of *causal effect identifiability* is whether a given causal effect of a given set of variables X on a disjoint set of variables Y,$P(y|\hat{x})$, can be determined uniquely from the distribution $P(v)$ of the observed variables, and is thus independent of the unknown quantities, $P(u)$ and $P(v_i|pa_i, u_i)$, that involve elements of U.

Pearl characterizes *plan identification* as the probability of a variable Y given a set of control variables X, a set of observed variables Z (often called *covariates*), and a set of unobserved variables U. Control variables are ordered ($X = X_1, X_2, ..., X_n$) so that every X_k is a nondescendant of $X_{k+j}(j > 0)$ in G and Y is descendant of X_n. N_k is the set of observed nodes that are nondescendants of any element in the set of control variables, i.e. previous evidence. A *plan* is an ordered sequence $(\hat{x}_1, \hat{x}_2, ..., \hat{x}_n)$ of value assignments to control variables, where $\hat{x}_k$ means "X_k is set to x_k".

Pearl and Robins provide a general criterion for plan identification: the probability $P(y|\hat{x}_1, \hat{x}_2, ..., \hat{x}_n)$ is identifiable if, for every $1 \leq k \leq n$, there exists a set Z_k of covariates satisfying

$$Z_k \subseteq N_k \tag{2}$$

and

$$(Y \perp\!\!\!\perp X_k|X_1, ..., X_{k-1}, Z_1, Z_2, ...Z_k)_{G_{\underline{X}_k, \overline{X}_{k+1}, ..., \overline{X}_n}}, \tag{3}$$

that is, Y is conditionally independent of X_k given previous actions and their respective covariates. $G_{\underline{X}}$ denotes the graph obtained by deleting from G all arrows emerging from nodes in X, $G_{\overline{X}}$ denotes the graph obtained by deleting from G all arrows pointing to nodes in X.

When these conditions are satisfied, the plan causal effect is given by

$$P(y|\hat{x}_1, \hat{x}_2, ..., \hat{x}_n) = \sum_{z_1, ..., z_n} P(y|z_1, ..., z_n, x_1, ..., x_n) \times \prod_{k=1}^{n} P(z_k|z_1, ..., z_{k-1}, x_1, ..., x_{k-1}) \tag{4}$$

2.3 Knowledge Representation

Ontology Web Language (OWL) is a W3C recommendation [6] for ontologies definition built over the widespread de facto standards XML and RDF. Inspired on the Object Oriented paradigm, OWL has as primitive elements: classes, properties, Instances of classes and relationships between instances.

Classes identify types of individuals (essence) and have certain *properties* (accidents) associated to them. Inherence mechanism applies to classes and properties. Individuals are represented as instances of a class and inherence properties associated to the class (accidents in potence). Any element in the ontology is identified by an URL, which permits reference other ontologies definitions.

Properties are divided in two kinds: datatyped and objects. First uses the XMLSchema data types and second points to instances of certain class. Properties have a range (possible values) and domain (possible classes to be attained to). Properties characteristics that can be expressed are: transitivity, symmetry, functionality and inverse. Some local restrictions can be defined in the class specification such as: cardinality and restriction of values to certain class. The hasValue restriction allows to specify classes based on the existence of particular property values.

SPARQL[7] is a query language for getting information from RDF graphs. It provides facilities to: extract information in the form of URIs (blank nodes and literals), extract RDF subgraphs, and construct new RDF graphs based on information in the queried graphs.

Formally, a SPARQL query contains four components: the graph pattern (GP), the dataset being queried (DS), a set of solution modifiers (SM), and the result form (R). The graph pattern of a query is called the query pattern.

The Graph pattern is a set of triplets and constraints that generates a RDF subgraph (WHERE clause). The queried RDF dataset is indicated through namespaces; SPARQL permits the use of prefixes. Results produced by the query can be modified in several ways: be ordered, select some parts of the solution (projection), remove duplicates (distinct) and limit the number of results.

SPARQL has four query result forms. These result forms use the solutions from pattern matching to form result sets or RDF graphs. The query result forms are: SELECT (that returns the variables bound in a query pattern match), CONSTRUCT (that returns a RDF graph constructed by substituting variables in a set of triple templates), DESCRIBE (that returns an RDF graph that describes the resources found), and ASK (that returns a boolean value indicating whether a query pattern matches or not). Variables have a global scope. Use of a given variable name anywhere in a query identifies the same variable.

3 Causal Agent

We propose a methodology for modeling intelligent agents and an agent architecture based on causality theory. First we present main causes that originate an intelligent agent and describe the ontological framework used to represent it. Next we present the agent architecture and explain how causality and intentionality elements are represented on it. We introduce an extended causal model that controls agent behavior through probabilistic reasoning. Finally we comment our application test-bed and current development.

3.1 Agent Causality and Intentionality

Lets define a *Causal Agent* as an artificial, intentional entity which: *(i)* has a *formal cause* represented by an agent class (essence) that groups properties and methods (accidents in potence), *(ii)* has a *material cause* constituted by its properties values and implemented sensors and actuators (accidents in act), *(iii)* has an *effective cause* that identifies the software or human agent that create or instantiate it, *(iv)* has a *final cause* that represents the goal or state (intentional object) that the agent must reach or maintain, and *(v)* has a *causal model* that controls its behavior and accumulates experience in terms of causal relations.

On agent's instantiation it is necessary to specify the four main causes that originates it: *(i)* the agent class, *(ii)* initial parameters, *(iii)* the creator id, and *(iv)* DL statements that identifies agent creator's intention. Causal model is initialized considering optimal conditions, i.e. deterministic causal relations and null external or unknown causes.

For example, agent's instantiation can be made trough an OWL instance of the OWL class representing the formal cause on which the material cause is expressed as properties values and the efficient cause is expressed through the *createdBy* property. The final cause can be expressed by the DL statement $< I, P, V >$ where I represents the agent instance and V is the value that the P property must reach in order to finish its execution. An statement presenting not grounded values like $< I, P, ?v >$ would identify an infinite task.

Agent's class is used to instantiate a software agent and for modeling other agents. Creator ID is used in agent's causal model to recognize authority over it permitting to modify agent's behavior.

Parameters that model agent intention are given in terms of agent ontology instances with grounded relations and values. Agent's causal model will drive agent behavior until reach a state on which all constraints are satisfied.

3.2 Agent Ontology

Three layers of OWL ontologies are used for modeling the agent and the application domain, as well as to annotate the agent causal model. The *Causal Ontology* is used to model real and reason entities in terms of accidents and causes. The *Agent Ontology*, which describes agent classes through characteristics and capabilities, is used to define a taxonomy of agents and publish agents' descriptions in the white pages. The *Domain Ontology* is used to model the application domain and permits to specialize agents in the system. The use of these three layers permits to reuse agents and processes in different application domains.

3.3 Agent Architecture

The causal agent's architecture is shown in Figure 1. Its core is a Bayesian Causal Model embedded in a semantic layer. In the causal model are represented agent beliefs and through probabilistic procedures is possible identify plans that lead to the agent's final cause achievement, learn new causal relations and update probabilistic distributions based on experience. Causal model structure and its operations are described in 3.4.

Sensors inputs are translated into perceptions understandable by the agent through a *parser* that uses *semantic descriptors*. These descriptors are used in white pages to describe agent characteristics and capabilities. They permit to receive and pass parameters to sensors and actuators implementation.

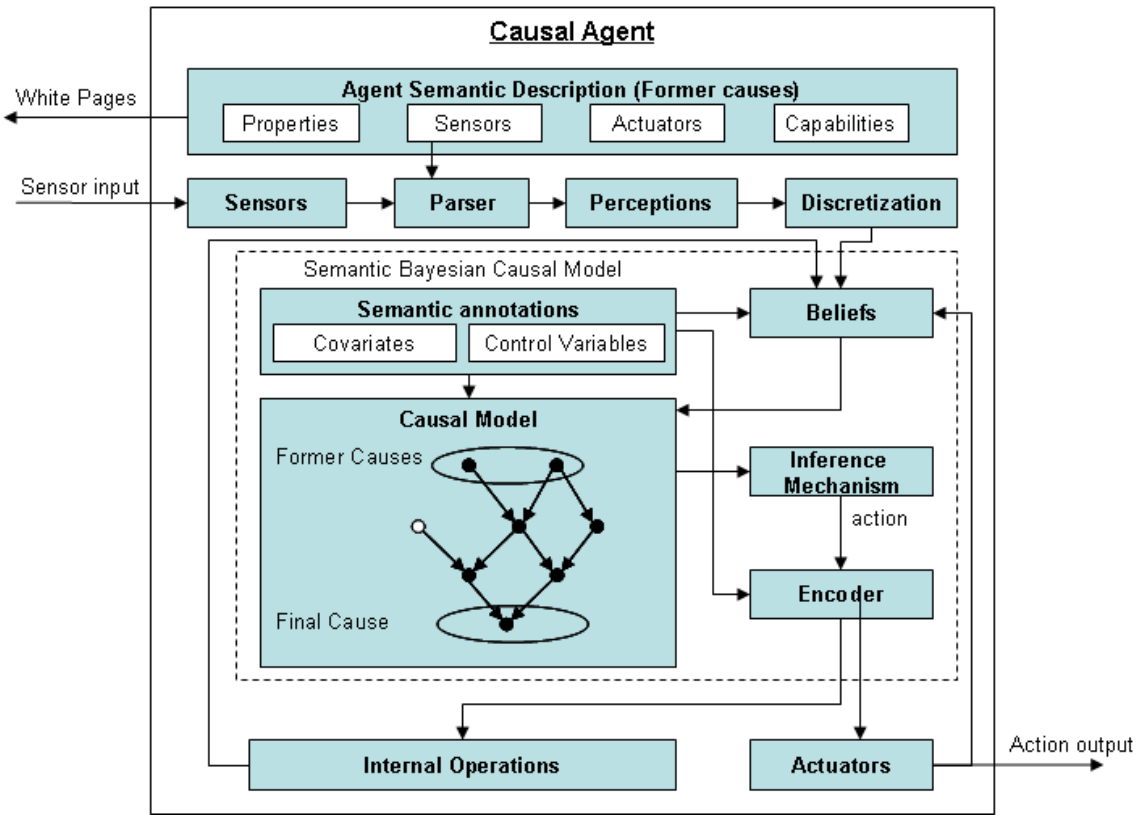

Figure 1. Causal Agent Architecture

Perceptions are transformed to discrete ranges to avoid the use of literals. The casual model performs a belief revision and chooses the best action from a set of possible plans generated through probabilistic methods.

The selected action is *encoded* using semantic annotations over the intervened control variable which is executed through internal or external actuators. Action execution is registered in agent beliefs in order update its beliefs, supporting this way reasoning in stochastic environments. The comparison between expected behavior and actual observations is used to update the model, i.e. learn from experience.

3.4 Semantic Bayesian Causal Model

The *Semantic Bayesian Causal Model* (SBCM) is an extension of a Bayesian causal model with a semantic layer that permits to represent causal relationships among events and to plan in order to achieve agent's intention [8].

A SBCM is represented by:

$$M = \langle V, U, G_{VU}, P(v_i|pa_i, u_i), P(u), C, Z, F, A, O, B \rangle \tag{5}$$

V is the set of endogenous variables that represents events and information that agent can be aware of. U is the set of exogenous variables and is used to represent unknown causes. G_{VU} is a causal graph consisting of variables in $V \times U$ that identifies cause-effect dependencies among events. $P(v)$ is the Bayesian probabilistic distribution that codifies

the likelihood of an event given certain conditions. $P(u)$ is a probabilistic distribution used to explain bias in the system or interference produced by external factors. $C \subset V$ represents endogenous variables that can be manipulated by the agent (control variables). $Z \subset V$ represents those events the agent observe but cannot alter (covariates). F is a set of interventions on V that identifies those conditions the agent must reach or maintain. A is a set of semantic annotations over V expressed in terms of the OWL ontology O. B is the set of interventions $(V_i = v_i)$[1] representing current agent's beliefs.

The agent inference process, shown in Figure 2, is performed at two levels: semantic and causal. Former enables common understanding between agents meanwhile the latter summarizes agent experience and guides its behavior through probabilistic methods. In the first phase, agent perceives the environment through its sensors and transforms its perceptions into DL assertions (A-Box) expressed in a given ontology O (T-Box).

Annotations associated to every variable, denoted A_i, are expressed as queries on SPARQL. Every query associated to a covariate Z_i is evaluated against the current perceptions A-Box (node instantiation phase). Covariates evaluated positively produce an intervention that later is revised with agent beliefs. A special variable in the query is bound to the variable value in the intervention. If A_i doesn't contain this special variable, Z_i is made true when perceptions match annotations, and false otherwise.

On the second phase, beliefs are revised with interventions generated from discrete perceptions. This revision is made by replacing old perceptions by new ones. In those cases where no information is given about certain sensors, current perception is estimated according to a dynamic causal model or remains unknown if there is not enough information. Actions recently performed by the agent are included in the set of beliefs. Performed actions and perceptions are used to train the model probabilistic distribution.

Once beliefs are revised, an instance of the causal model is generated replacing belief interventions and pruning those relations that no longer holds. Over the instantiated model, a set of possible plans to reach F is elaborated, and through a heuristic the most feasible plan is selected. The first action of this plan is selected for execution. This action is represented by an intervention over a control variable $(C_w = c_w)$. c_w is replaced in the C_w variable annotations to produce a set of triplets that encodes the command sent to the actuator.

3.5 A Causal Agent Example

Lets define instantiate

Agent's formal cause is expressed trough an OWL class that specifies those properties the agent can show, including the list of sensors and actuators it posses. Last represent agent's material cause. In order to instantiate the agent we must define an OWL instance of this class on which some properties are set and instances of sensors and actuators are associated to it. Instance properties setting constitute the way the agent's creator expresses the final cause. One of these properties identifies the efficient cause (the *createdBy* property) which is set to the system administrator ID or another agent ID.

Agent's causal model is ...

[1]Capital letters represent variables (V_i) meanwhile small letters represent variable values(v_i)

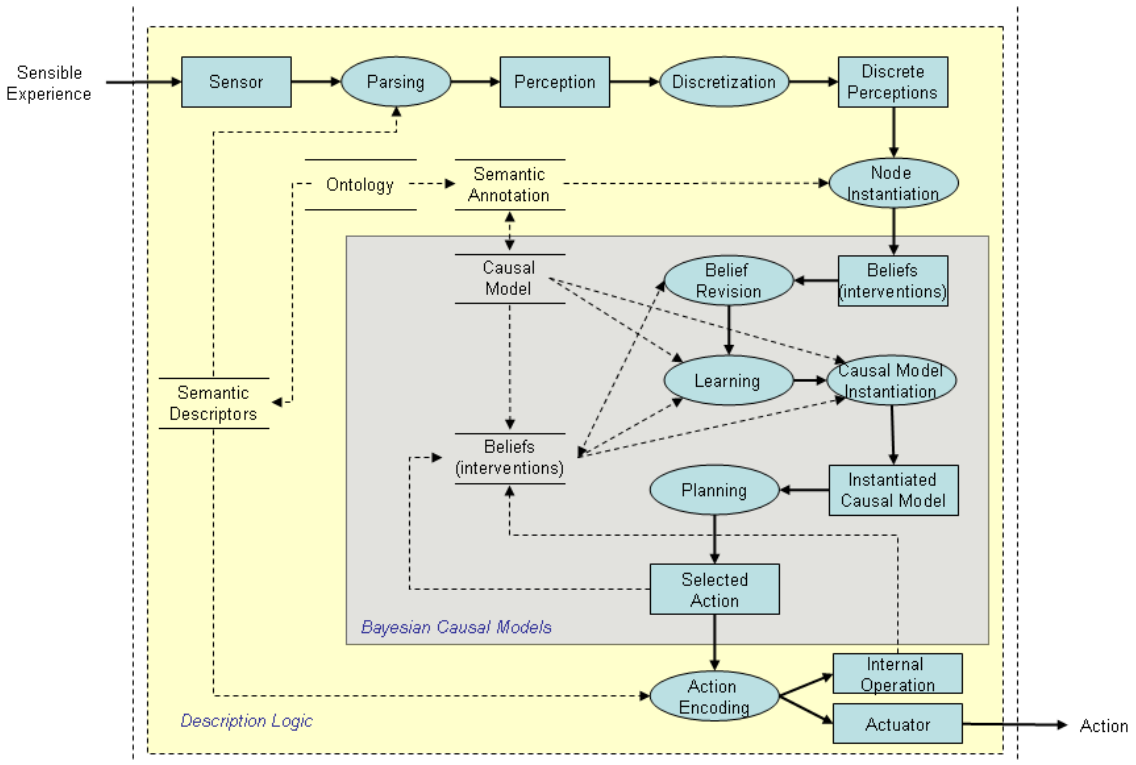

Figure 2. Causal Agent Inference Process

4 Current Development

We are testing our approach by incorporating intelligent agents to an information system that manages research products at a university [9]. This system has been operational for three years and offer services to researchers, students and research chairs in several modules that include publications, projects, research groups, thesis and graduate programs among others. Information stored in data repositories contains thousands of records organized according to a research ontology.

Currently, tasks are performed by humans through web interfaces provided by the system in roles clearly defined. For instance, there are auditors on charge of classify and validate information fed in the system. Users receive alerts whenever a close related knowledge asset is registered or updated. This relation is given by user roles and asset nature. Another part of the system operation is performed off-line; for example, information integration and loading.

Our agent architecture and methodology will be used on information integration and auditing, as well as users modeling. Uncertainty handling in information integration will permit to retrieve new data from web sources meanwhile auditor agents will validate its usefulness and correctness. In both cases, humans will validate agent's results training on this way agents' causal models.

Users modeling will permit to offer services to researchers and to generate a profile that improve their experience on the system by offering them shortcuts and performing repetitive tasks automatically.

5 Related Work

We recognize validity of beliefs, desires and intentions on an intelligent and autonomous agent, proposed in the DBI architecture[10]. Our approach maintains these elements and incorporates causality notions and formalisms as a mean to unify knowledge representation and reasoning mechanisms.

In our approach, the *agent state* is given by a set of intervened causal variables, which together with probabilistic distribution and causal relationships constitute agent *beliefs*. Semantic annotations permit to communicate these beliefs to other agents.

Belief revision, in our case, pursues two objectives: to update current beliefs and to refine the model. A naive approach for static models forgets all events occurred in previous time frames and only considers current perceptions. These models can use negations as failure.

With a dynamic causal model, variables states at previous time frames are represented by variables in the causal model. A learned relationship between previous states would even make possible to predict the variable value at time $t - i$ in terms of values at time $t - j$ where $i < j$.

Final cause represents agent *intention*. Agent *options* or *desires* are obtained from plans generated from current beliefs and oriented to reach the final cause. *Filter* function is represented by the heuristic used to choose a plan. *Action selection* is made selecting the first action in the chosen plan.

6 Conclusions

In this paper we explore Aristotelian-Thomist Causality theory applications in agents design. We believe that an intentional agent can be modeled through the formal, material, efficient and final causes proposed by Aristotle and revised by Aquinas. This design permits an agent to develop in a stochastic environment supporting external or unknown causes existence and planning under uncertainty.

The agent causal model can be updated through experience and can manage change in the environment conditions. All the time, agent will be driven by its final cause and will be looking forward to optimize the way to accomplish it and collaborate with its creator intention.

Annotations over causal model variables enable matching variables among different causal models and calculating distributed causal effects [11]. This is possible due to semantic meaning associated to variables. Agents will be in position of exchange information about causal relationships influencing other agent's behavior enforcing cooperation.

Besides, semantic information associated to variables presenting an irregular behavior (noise) would lead to causal relationships discovery. Semantic information dismissed in the node instantiation phase can be used for this purpose. This way, we are in position of not just learn probabilistic distributions but the causal structure too [12].

As future work, we intend to learn causal relationships (structure learning) rather than just probabilistic distributions (parameter learning), to model other agents by observing their behavior and to communicate knowledge in the form of causal relations to other agents useful for their purposes.

References

[1] Schreiber, G.: Knowledge Engineering and Management: The CommonKADS Methodology. MIT Press (2000)

[2] Crockett, L.: The Turing Test and the Frame Problem: AI's Mistaken Understanding of Intelligence. Ablex, Norwood, New Jersey (1994)

[3] Pearl, J.: Causality. Models, Reasoning, and Inference. Cambridge University Press (2000)

[4] Benzi, M.: Contexts for causal models. In: Causality and Probability in the Sciences, University of Kent (2006)

[5] Alvira, T., Clavell, L., Melendo, T.: METAFISICA. 8th edn. EUNSA (2001)

[6] Smith, M.K., Welty, C., Deborah L. McGuinness, E.: OWL Web Ontology Language Guide (W3C Recommendation 10 February 2004)

[7] Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF. W3C working draft. http://www.w3.org/TR/2006/WD-rdf-sparql-query-20061004/ (2006)

[8] Ceballos, H., Cantu, F.: Integrating semantic annotations in bayesian causal models. In Calvanese, D., Franconi, E., Haarslev, V., Lembo, D., Motik, B., Tessaris, S., Turhan, A.Y., eds.: Proceedings of the 20th International Workshop on Description Logics DL'07, Bozen-Bolzano University Press (2007) 527–528

[9] Cantu, F., Ceballos, H., Mora, S., Escoffie, M.: A knowledge-based information system for managing research programs and value creation in a university environment. In: Proceedings of the Eleventh Americas Conference on Information Systems, Omaha NE, USA, Association for Information Systems (AIS) (2005)

[10] Weiss, G.: Multiagent Systems. MIT Press (1999)

[11] Maes, S., Meganck, S., Manderick, B.: Identification of causal effects in multi-agent causal models. In: IASTED International Conference on Artificial Intelligence and Applications. (2005) 178–182

[12] Flores-Quintanilla, J., Morales-Menendez, R., Ramirez-Mendoza, R., Garza-Castanon, L., Cantu-Ortiz, F.: Towards a new fault diagnosis system for electric machines based on dynamic probabilistic models. In: American Control Conference, 2005. Proceedings of the 2005. Volume 4., IEEE (2005) 2775–2780