Learning by Knowledge Exchange in Logical Agents

Stefania Costantini and Arianna Tocchio

Artificial Intelligence Group

Università degli Studi dell'Aquila
Dipartimento di Informatica
I-67100 L'Aquila
Intelligent Agents

Intelligent Agents:

» An agent is a system that tries to fulfill a set of goals in a complex, dynamic environment. An agent is called autonomous if it operates completely on its own, deciding by itself what action it should take, considering the input coming from its sensors, in order to achieve its goals. An agent capable of improving at achieving its goals is said to be adaptive (Maes).

The context:

» The environmental context changes, cooperative or competitive agents can appear or disappear, ask for information, require resources, propose unknown goals and actions.

» Agents may try to improve their potentiality by interacting with other entities so as to perform unknown or difficult tasks.
How learning process can improve agents behavior

The usefulness of experience

» An intelligent system is not only situated in space but also in time. This implies that the system can develop itself so as to become better at its task if time and the particular task permit (through learning from experience) (Maes)

» So, adaptive agents can take profit of past experience using some learning approach.

What’s learning?

» Learning is the process of acquiring knowledge, skills, attitudes, or values, through study, experience, or teaching, that causes a change of behavior and allows an individual to formulate a new mental construct or revise a prior mental construct.

» It is a process that depends on experience and leads to long-term changes in behavior potential. (Wikipedia)
How learning process can improve agents behavior

» How can an agent improve its performance over time based on its experience?

> **reinforcement learning**: consists in assigning rewards (weights) to actions that contribute to the resolution of a problem;

> **models learning**: agents try to find casual relations between their actions and the events occurring in the environment via probabilistic or logical models;

> **classifier systems**: an agent can try to classify applicable rules by setting priorities and then updating these priorities according to the results achieved.

> **memory-based reasoning (MBR)**: is based on the idea that if a given action took place in the past in a given situation $s$ and gave good results, it will be useful in a new situation $s'$ similar to $s$;

> others...
Learning and MAS:

» One of the key features of MAS is the ability of “sub-contracting” computations to agents that may possess the ability to perform them.

» More generally, agents can try to achieve a goal by means of cooperative distributed problem-solving.

» However, on the one hand not all tasks can be delegated and on the other hand agents may need or may want to acquire new abilities to cope with unknown situations.
Need of new knowledge:

» The need of acquiring new knowledge can be recognized by an agent at least in relation to the following situations:

> There is an objective that the agent has been unable to reach.

*Use case.* An agent located in a network router for security management has to recognize and respond to attacks, and perform repairs. It is impossible to delegate this tasks, it is instead possible in some cases to learn suitable behaviors from trusted agents. For instance, the agent may perceive an unknown external event that presumably corresponds to an unknown attack: it can try to learn how to respond.

> There is some kind of computation that the agent is unable to perform.

*Use case.* An agent which acts as a mediator in a data integration context is responsible for reformulating at runtime queries defined on a (virtual) mediated schema into the local (actual) schemata of the underlying data sources. If a new data source is added to the system, a suitable wrapper may be missing. The agent can try to acquire the wrapper and thus “learn” how to cope with the new data format.
Learning and MAS:

» In our view, an improvement in the effectiveness of MAS may consist in introducing a key feature of human societies, i.e., cultural transmission of abilities.

» For us the effects of abilities transmission should include at least one of the following:

  > the range of behaviors is expanded: the agent can do more;
  > the accuracy on tasks is improved: the agent can do things better;
  > the speed is improved: the agent can do things faster.
Adopting Trust Evaluation for Secure Learning:

» Agents should try to learn new knowledge from trusted agents only.

» Successful evaluation of the acquired knowledge can lead to an increase of the level of trust of the sending agent, while a decision to discard that knowledge can also result in a decrease of the level of trust.

» We introduced a directory agent employed to inform agents of where to find the required knowledge and to maintain a value corresponding to agents reputation.

» This directory agent may in principle be notified of the updates of the level of trust performed by agents.

» How the mediator should compute the reputation of agents is a critical issue.

» We may observe that the trust level should be associated to context information such as the field of expertise and other specifications of the knowledge that agent is available to exchange.
Our learning approach:

» We discuss a learning approach useful to improve adaptive behavior in computational logic agents.

» Our approach is centered on the possibility of exchanging set of rules between agents.

» These sets of rules can either define a procedure, or constitute a module for coping with some sort of situation, or be just a segment of a knowledge base.

» However, agents should then be able to evaluate how useful the new knowledge is. To this extent, we propose two techniques:

  > The first technique associates to the acquired knowledge a specific objective, meaning that the new rules should help the agent to reach that objective.

  > The second technique consists in acquiring the same knowledge from several other agents, and then comparing the results.
Our approach in practice

» We decided to develop a prototypical implementation of our learning approach in the agent-oriented programming language DALI.

» DALI is an Active Logic Programming language for executable specification of logical agents.

» DALI includes several agent-oriented features:
  > autonomy
  > reactivity
  > pro-activity
  > social ability
  > children generation capability,...
Basic learning mechanism

» Agents that adopt forms of learning to improve their behavior can in perspective deal with more complex jobs, but expose themselves to some risks.

» Agents knowledge is generally divided into a set of facts and rules: the former represent the agent “beliefs” about itself and the world, while the latter determine the entity behavior.

» If learning one or more beliefs (plain facts) implies a certain degree of risk, adding rules coming from other agents to the knowledge base can very dangerous.

» Thus, in our view it is necessary to elaborate different learning strategies for beliefs and rules, reserving to the latter case a more sophisticated acquisition process.
Basic learning mechanism: beliefs learning

» The belief base of DALI agents is composed of the facts which are present in the agent logic program, dynamically augmented by past events.

» A DALI agent can ask for some facts by using the primitives:

  > is_a_fact(Fact, Ag)
  > query_ref(Fact, Matches_number, Ag)

where Fact is the desired information, Matches_number represents the number of matches that the agent intends to receive and Ag is the name of the agent asking for the fact.

» A direct acquisition beliefs method in DALI agent is based on the confirm primitive. This method allows an agent to send a fact to another one. Also in this case, the fact will be added to the agent beliefs only if the message will overcome the told filter.

  > messageA(Ag₁, confirm(Fact, Ag₂)).
Basic learning mechanism: rules learning

» The need to acquire new knowledge arises whenever an agent receives a communication act whose content is unknown and the meta-level does not succeed in searching for a semantically equivalent content recognized by the entity.

» Having no internal means to cope with this situation, the agent activates the learning rules process.

» This process is risky enough, so the agent must try to search a suitable information source.

» The solution is based on the introduction of a mediator agent that we call **yellow_rules_agent**, keeping track of the agents specialization and reliability.

» When an entity needs to learn something, it asks the **yellow_rules_agent** for the names agents having the same specialization and being more reliable.
First learning step: yellow_rules_agent

» Each agent living in the environment is identified by the tuple:

\[ \text{source}(A_i, S_i, KR_i, Q_i) \]

where:

> the first parameter represents the agent identification;

> the second one is a string synthesizing the agent role in the environment;

> the third one is a list of rules keys that the agent \( A_i \) is willing to transfer to other agents;

> The fourth one is the reliability value, computed by yellow rules agent according to positive and negative feedbacks.

» Agents that receive rules from agent \( i \), at the end of the verification phase send a message to yellow rules agent rating that knowledge.

» According to current and past values average, the \textit{yellow\_rules\_agent} computes agent \( i \) reliability by means of some kind of evaluation.
Second learning step: asking for missing rules

» In order to get the needed piece of knowledge, the agent can choose one of two techniques: the first one allows an agent to learn all required rules by specifying their heads.

» If we consider the agent $A_k$ having selected the couple $(A_s, Q_s)$ and the head $H_l$, the following message will propose to the receiver agent the exchange of rules having the head $H_l$:

\[
messageA(A_s, ask\_rules\_head(H_l,A_k))
\]

» The second technique allows an agent to ask for a specific key that can match with either the head or the body of rules in the agent program. The message syntax will be:

\[
messageA(A_s, ask\_rules\_key(Key,A_k))
\]
Third learning step: rules exchanging

» Agents accepting the proposal to exchange rules that match with either the Head or the Key will pack all retrieved rules and will send them back to the Ak entity.

\[ \text{messageA}(A_k, \text{sent rules}([R_1, \ldots, R_n], C_h, A_s)) \]

» The parameter \( C_h \) represents the goal that must be invoked in order to activate the rules.

» In particular, if the agent \( \text{bob} \) receives from the agent \( \text{dave} \) the rules:

\[ \text{[dangerE :> call policeA, call police :< have a phoneP]} \]

the parameter \( C_h \) will correspond to \( \text{dangerE} \).
Fourth learning step: adding and managing rules

» As soon as these rules will be received by the learner agent, they will be unpacked and asserted as past events in its knowledge base with the suffix

\[
\text{learnP (Rule, Time, Sender, Objective)}
\]

where the parameter Objective is useful to remember what was the goal for which the request had been issued.

» Rules added as past events are managed by a specific internal event, gest\_learning(Rule), that implements the first filter level.

\[
\begin{align*}
\text{gest\_learning(Rule)} : & - \text{learnP (Rule, Time, Sender)} , \\
& \text{learn\_if(Rule, Time, Sender)}, \\
& \text{properties\_true(Rule)}. \\
\text{gest\_learningl (Rule)} : & \text{accept\_at\_presentA(Rule)}. 
\end{align*}
\]
Fourth learning step: adding and managing rules

» The first condition, learn_if(Rules, Time, Sender), defines a set of constraints that the considered Rules, the Time and the Sender agent must respect:

\[
\text{learn}_\text{if}(\text{Rules}, \text{Time}, \text{Sender}) : -\text{constraint}_1, \ldots, \text{constraint}_n.
\]

» The second condition, properties_true(Rule), takes more specific properties of the Rules into account:

> the syntactic correctness according to prolog and DALI language;
> the absence of procedure calls without a corresponding procedure;
> the overlap of rules originating from different agents;
> the rule consistence with respect to previously learned clauses.
Fifth learning step: estimating rules behavior

We propose two sample methods to estimate partially learned rules:

> **On objectives:** once introduced in the agent program, each piece of knowledge is used by the entity during its life, keeping always track of its performance with respect to the corresponding objective. The evaluation is performed considering:

  > the degree of correspondence between the *Objective* and past events generated by the *Rules* application;

  > given the state snapshot of the program execution, the degree of correspondence between the saved state and the declared *Objective*.

> **On comparison:** the agent can generate children, and can assign each child a different set of rules acquired by different sources for the same calculation. This check will consider also performance, time and resources spent.
Semantics of Learning by Rule Exchange

» Given the program $P_{Ag}$, the semantics can be based on the following.

  > An *initialization step* where $P_{Ag}$ is transformed into a corresponding program $P_0$ by means of some sort of knowledge compilation;

  > A sequence of evolution steps, where reception of each event is understood as a transformation of $P_i$ into $P_{i+1}$ where the transformation specifies how the event affects the agent program.

Program Evolution Sequence $PE = [P_0,..., P_n]$

Semantic Evolution Sequence $[M_0,...,M_n]$ where $M_i$ is the semantic account of $P_i$ ($M_i$ is the least Herbrand model of $P_i$).

Evolutionary semantics
Semantics of Learning by Rule Exchange

This semantic account can be adapted by transforming the initialization step into a more general knowledge compilation step, to be performed:

At the initialization stage

Upon reception of new knowledge

In consequence to the decision to accept/reject the new knowledge.
Conclusions

» We have proposed a form of cooperation among agents that consists in improving each agent’s skills by acquiring new knowledge from the others.

» More experimental work is needed for proving the effectiveness of the approach, and for putting various methods of verification of the usefulness of learning at work.

» Our goal is to complete the implementation and to test our approach in a real application.