Learning and Evolving Agents in User Monitoring and Training

Stefania Costantini[1], **Pierangelo Dell'Acqua**[2], **Luís Moniz Pereira**[3], and **Francesca Toni**[3]

[1] Dip. di Informatica, Università di L'Aquila, Coppito 67010, L'Aquila, Italy
stefcost@di.univaq.it
[2] Dept. of Science and Technology - ITN, Linköping University, Norrköping, Sweden
pierangelo.dellacqua@itn.liu.se
[3] Departamento de Informática, Centro de Inteligência Artificial (CENTRIA), Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
lmp@di.fct.unl.pt
[4] Dept. of Computing, Imperial College London, South-Kensington Campus, London, UK
ft@doc.ic.ac.uk

Abstract. We propose a general vision for agents in Ambient Intelligent applications, whereby agents monitor and un-intrusively train human users, and learn their patterns of behavior not only by observing and generalizing their observations, but also by "imitating" them. Agents can also learn by "imitating" other agents, when being told by them. Within this vision, agents need to evolve to take into account what they learn from or about users, and as a result of monitoring the users.

1 Introduction and Motivations

We envisage a framework where agents may improve the "quality of life" of human users, by efficiently supporting their activities. Our agents are supposed to interact with users: (i) with the aim of monitoring them for ensuring some degree of consistence and coherence in user behavior and, possibly, (ii) with the objective of training them in some particular task.

In our view, a system which is a realization of the envisaged framework will bring to a user the following potential advantages: the user is relieved of some of the responsibilities related to her behavior, as directions about the "right thing" to do are constantly and punctually provided. She is assisted in situations where she perceives herself as inadequate, in some respect, to perform her activities or tasks. She is possibly told how to cope with unknown, unwanted or challenging circumstances. She interacts with a "Personal Assistant" that improves in time, both in its understanding of the user needs, cultural level, preferred kinds of explanations, etc. and in its ability to cope with the environment.

To this aim, we assume that "our" agents are able (iii) to elicit (e.g. by inductive learning) the behavioral patterns that the user is adopting, and (iv) to learn rules and plans from other agents by imitation (or by "being told"). Here, on some aspects related

to learning we take inspiration from recent evolutionary cultural studies of human societal organization to collectively cope with their environment. We believe in fact that some principles emerging from these studies can equally apply to societies of agents. This is especially so when agents cooperate with humans so as to help them to adapt to environments that are new to them and/or when their ability to cope with the environment is too costly, non-existent or impaired. The envisaged agents will try to either modify or reinforce the rules/plans/patterns they hold, based on appropriate evaluation performed by an internal meta-control component. The evaluation might also convince the agent to modify its own behavior by means of advanced evolution capabilities.

This overall agent model is in accordance with the vision of Ambient Intelligence as that of a digitally augmented environment centered around the needs of humans, where appliances and services proactively and un-intrusively provide support and assistance.

We consider it necessary for an agent to acquire knowledge from other agents, i.e. learn "by being told" instead of learning only by experience. Indeed, this is a fairly practical and economical way of increasing abilities, widely used by human beings, as widely studied in evolutionary biology [1]. In fact, avoiding the costs of learning is an important benefit of imitation. An agent that learns and re-elaborates the learned knowledge becomes in turn an information producer, from which others can learn in turn. On the other hand, an agent that just imitates blindly can be a burden for the society to which it belongs.

Evolutionary biology shows that the long-run of evolution of human societies is a mixture of learners and copiers, in which both types have the same fitness as purely individual learners in a population without copiers. To understand this result, think of imitators as information scroungers and of learners as information producers. Information producers bear a cost to learn. When scroungers are rare and producers common, almost all scroungers will imitate a producer. If the environment changes, any scroungers that imitate scroungers will get caught out with bad information, whereas producers will adapt. Thus, an agent will be able to increase its fitness in such a society in two ways: if it is capable of usefully exploiting learned knowledge thus deriving new knowledge and becoming an information producer; if it is capable to learn selectively, learning when learning is cheap and accurate, and imitating otherwise.

In the sequel we shall outline a model so inspired for the construction of logical agents that are able to learn and adapt their behavior in interaction with humans.

Let us emphasize that, to engage with humans, agents should have a description of how humans normally function. Clearly, the description will in general be initially limited to the "normal" user behavior in that ambient setting. We assume that the agents are deliberately designed and originally primed with some ambient setting in mind, and that the humans are new to the setting and/or experience difficulties or impairments in coping with it. As deep learning (i.e., learning from scratch) is time consuming and costly, and thus needs not be repeated by one and all, an agent may apply a hybrid combination of both deep learning and imitation. The view is that all agents and the society as a whole will eventually take profit from the learning/imitation process, that can here be seen as a form of cooperation.

Each agent will thus initially be equipped either with sibling agents or with a structured agent society with abilities related to its "role", i.e., with the supervision task it

will have to perform. These initial capabilities may be enhanced by means of internal learning in consequence of the interaction with both the user and the environment, and with other similar agents. However, when some piece of knowledge is missing and a task cannot be properly carried out by an agent, that piece may eventually be acquired from the society, if extant there, for the agent may be unable or unwilling to deep learn it. Then it will exercise it in the context at hand, subsequently evaluate it on the basis of such experience, and report back to the society. This evaluation of imparted knowledge builds up a network of agents' credibility and trustworthiness, where the learning producers benefit from the more extensive testing performed by scroungers.

A flexible interaction with the user will be made easier by adopting a multi-layered underlying agent model [2] where there is a base level, that we call PA for "Personal Assistant", and one (or more) meta-layer, that we call MPA. While the PA is responsible for the direct interaction with the user, the MPA is responsible for correct and timely PA behavior. Thus, while the PA monitors the user, the MPA monitors the PA. The actions that the PA will be able to undertake will include, for instance, behavioral suggestions, appliance manipulation, enabling or disabling user manipulation of an appliance. The actions the MPA will be able to undertake will include the modification of the PA in terms of adding/removing knowledge (modules) in the attempt at correcting inadequacies and generating a more appropriate behavior. In our framework, both the PA and the MPA will largely base their behavior upon verification of temporal-logic rules that describe expected and un-expected/unwanted situations. Whenever all rules are complied with, the overall agent is supposed to work well. Whenever some rule is violated, suitable actions are to be undertaken, to restore correct functioning. Here, temporal rules are checked at run-time [3] (at a certain frequency and with certain priorities) and necessary actions are immediately executed.

In Section 2 we discuss learning in agents, and we introduce the temporal logic rules that we employ for implementing monitoring and training. In Section 3 we first review previous work on training, and then we present our perspective. In Section 3.1 we show, mainly by means of examples, how the features of our framework can be profitably exploited for both monitoring and training a user. Finally, concluding remarks are set forth in Section 4.

2 On learning, evolution and self-checking

We assume that our agents do not act in isolation: rather, they are part of a society of agents. This society in its simplest version can be a set of sibling agents. More generally, it can be a structured society of agents sharing common knowledge and possibly common objectives. We assume that the agents belonging to this society are benevolent and willing to cooperate.

An agent that performs activities of monitoring/training a user must perform at least three kinds of different learning activities:

– **First Stage, Initialization**: in order to start its monitoring/training activities, the agent must receive either from a sibling agent or from the society a set of basic facts and rules that define:

- the *role* that the agent will impersonate with the user
- the basic behavior of the agent

This is clearly a form of *Learning by being told*.

– **Subsequent stages, Observation**: the agent will be able to observe the user's behavior along time and in different situations. The agent will collect the observations and will try to *classify* them with the aim of eliciting general rules or at least being able to expect with reasonable confidence what the user will do in the future.
– **Subsequent stages, Interaction**: whenever the monitoring/training agent will have to cope with a situation for which it has no sufficient knowledge/expertise, the agent will try to obtain, from either other agents of from the society, the necessary knowledge and rules. The agent will however in general evaluate the actual usefulness of the acquired knowledge.

Included in the initialization stage are general temporal-logic meta-rules to be included in the MPA. The following sample interval temporal logic rules (for a formal description, of the A-ILTL logic that we previously defined and adopt here, the reader may refer to [3]) state for instance that the user should eventually perform necessary actions within the associated time-threshold, and should never perform forbidden actions.

$FINALLY(T)\ A\ ::\ action(A), mandatory(user, A), timeout(A, T)$

$NEVER\ A\ ::\ action(A), forbidden(user, A)$

These meta-rules are checked dynamically, i.e., at run-time, at a certain (customizable) frequency. Meta-rules can be themselves customized by the agent, by means of learning, after a relevant number of interactions with a user. Assume for example that an agent is required to act as a baby-sitter. The kind of knowledge it will be equipped with can consist for instance of the following. A mandatory rule should, say, state that children should always go to bed within a certain time period.

$ALWAYS\ go\_to\_bed(P, T), early(T)\ ::\ child(P)$

The agent might later try to learn, by means of observations, what "early" means, according to children's age and family habits and will possibly elicit a rule such as:

$USUALLY\ (go\_to\_bed(P, T), 9:00 \leq T \leq 10:30 ::\ child(P), age(P, E), 10 \leq E \leq 13$

Vice versa, each agent will give its contribution to the society. As the case may be, the rule above might be communicated to the society and might (after suitable evaluation by the society itself) be integrated into the society's common knowledge and then communicated to other agents. An agent may contribute to the society's "common belief set" under several respects: (i) Provide the others with its own knowledge when required. (ii) In case of a structured society, insert into a repository whatever it has been able to learn. (iii) Provide a feedback about the usefulness/effectiveness, in its own context, of the knowledge it has been told by others. (iv) Participate in possible "collective evaluations" of learned knowledge.

Facts and rules that a monitoring/training agent is able to learn from the interaction with the user can be very important for the society, in that they can constitute knowledge that they will acquire "by being told". The agent will later on verify the adequacy of learned rules and will promptly revise/retract them in face of new evidence. The

principle to be used here is the *Unknown World Assumption* (UWA) where everything is unknown or undefined until we have some solid evidence of its truth or falsity. This principle differs from the more usual *Closed World Assumption* (CWA) where everything is assumed false until there is solid evidence of its truthfulness. The UWA stance is more skeptical, cautious, and even more realistic than the CWA.

Hopefully, after some iterations along this building/refinement cycle, the knowledge built is "good enough" in the sense that the predictions it makes are accurate "enough" concerning the environment observations resulting from experience. At this point, the theory can be used both to provide explanations to observations as well as to produce new predictions.

In computational logic, several approaches to learning rules and facts have been developed, also by (some of) the authors of this paper. As mentioned, in real-world problems, complete information about the world is impossible to achieve and it is necessary to reason and act on the basis of the available partial information. In situations of incomplete knowledge, it is important to distinguish between what is true, what is false, and what is unknown or undefined.

In [4] the authors showed that various approaches and strategies can be adopted in Inductive Logic Programming (ILP, henceforth) for learning with Extended Logic Programs (ELP) — including explicit negation — under an extension of well-founded semantics. As in [5, 6], where answer-sets semantics is used, the learning process starts from a set of positive and negative examples plus some background knowledge in the form of an extended logic program. Positive and negative information in the training set are treated equally, by learning a definition for both a positive concept p and its (explicitly) negated concept $\neg p$. Coverage of examples is tested by adopting the *SLX* [7] interpreter for ELP under the Well-Founded Semantics with explicit negation (*WFSX*) defined in [8, 9], and valid for its paraconsistent version [10].

After a theory has been built, it can be exploited on the one hand to analyze observations and to provide explanations for them, and on the other hand to foresee the user behavior. Notice that in practical situations several possible alternative rules might be learned. The MPA should include suitable Integrity Constraints (IC's) and preferences for choosing among alternatives. Moreover, learned rules should then be compared with subsequent observations, and thus might be refined, revised or even dropped. In this sense, the role of the society can be crucial.

The role of the society Finding possible alternative explanations is one problem; finding which one(s) is(are) the "best" is another issue altogether. In the next section we assume "best" means minimal set of hypotheses and we describe the method we use to find a best. Another interpretation of "best" could be "most probable", and in this case the theory inside the agents must contain the adequate probabilistic information.

Ex contradictione quodlibet. This well-known Latin saying means "Anything follows from contradiction". But contradictory, oppositional ideas and arguments can be combined together in different ways to produce new ideas. Since "anything follows from contradiction" one of the things that might follow from it is a solution to a problem to which several alternative positions contribute.

One well known method for solving complex problems widely used by creative teams is that of 'brainstorming'. In a nutshell, every agent participating in the 'brainstorm' contributes by adding one of his/hers idea to the common idea-pool shared by all the agents. All the ideas, sometimes clashing and oppositional among each other, are then mixed, crossed and mutated. The solution to the problem arises from the pool after a few iterations of this evolutionary process. The evolution of alternative ideas and arguments in order to find a collaborative solution to a group problem is the underlying inspiration of this work.

Evolutionary inspiration Darwin's theory is based on the concept of natural selection: only those individuals that are most fit for their environment survive, and are thus able to generate new individuals by means of reproduction. Moreover, during their lifetime, individuals may be subject to random mutations of their genes that they can transmit to offspring. Lamarck's theory, contrastingly, states that evolution is due to the process of adaptation to the environment that an individual performs in his/her life. The results of this process are then automatically transmitted to his/her offspring, via its genes. In other words, the abilities learned during the life of an individual can modify his/her genes.

Lamarckian evolution has recently received a renewed attention because it can model cultural evolution. In this context, the concept of "meme" has been developed. A meme is the cognitive equivalent of a gene and it stores abilities learned by an individual during his lifetime, so that they can be transmitted to his/her offspring.

In the field of genetic programming, Lamarckian evolution has proved to be a powerful concept and various authors have investigated the combination of Darwinian and Lamarckian evolution. In [11], some of the authors of this paper propose a genetic algorithm for belief revision that includes, besides Darwin's operators of selection, mutation and crossover, a logic based Lamarckian operator as well.

3 User Monitoring and Training

The term "training" refers in general to the acquisition of knowledge, skills and competencies as a result of teaching. In recent years, investigators in educational research have sought how to build artificial systems that can train users, and which properties such systems must satisfy. This research effort led to the notion of Intelligent Tutoring Systems (ITS) and Intelligent Learning Environments (ILE). Within ITS, one line of research has investigated how to exploit intelligent agents empowered with pedagogical skills. Giraffa and Viccari [12] argue that the fundamental reason for introducing these agents is their capability of communication and interaction as well as adaptation and learning.

Basically, pedagogical agents have a set of normative teaching goals and plans (teaching strategies) for achieving those goals, and associated resources in the learning environment. Four different types of pedagogical agents have been considered: tutors, mentors, assistance and web agents [12]. Pedagogical agents may be further divided into goal driven (tutor, mentor and assistance agent) and utility driven (web agents).

In order to create a system that adapts itself to each individual user, it must be designed to record the user's actions and deduce from that how the user's behavior evolves. Given such a user model, a system, in order to adapt, needs some mechanisms that allows it to analyze a possibly changed user model to derive adaptation recommendations based on various pre-defined rules.

To this aim, Kinshuk et al. [13] for example proposed the use of an approach based on fuzzy backpropagation techniques that build upon a neuro-fuzzy model combining neural networks with fuzzy logic. More recent approaches to adaptation propose to enrich the user model, by incorporating the ability to reason about intention, belief and action into the system. Along this line of research, Baldoni et al. [14] interpreted the notion of adaptation and proposed a form of it based on users' intentions and needs.

In the envisaged approach, where we consider our agent as advanced personal assistants rather than teachers, we do not reproduce the above-mentioned features of E-learning systems. Rather, we see an agent as supervising what the user does or intends to do, and provides permissions or denials on the one hand and assistance and support on the other hand. In particular, in the envisaged framework we mean: by monitoring, the activity of supervising the user by preventing incorrect or forbidden actions, suggesting correct ones and checking whether they have been performed (in the correct/expected order and/or at the correct/expected time); by training, the activity of providing suggestions about which are the actions to perform (indicating the correct/expected order and/or the correct/expected timing) and of answering questions and providing explanations; the explanations can be at different levels, according to user profile and user request; the training activity is joint with the monitoring activity in order to verify user response to the system's stimuli.

Our agents are meant to be personal assistants that provide guidance, directions and illustrations so that a user can be: (i) Relieved of some of the responsibilities related to her behavior, as directions about the "right thing" to do are constantly and punctually provided (ii) Assisted in situations where the user is "disabled" either physically or psychologically, in that the environment is either unknown or for some reason difficult to cope with. Recent work in the study of disability and personal/social difficulties [15] has emphasized that a person sometimes cannot put in practice what she is supposed to be able to do in her present condition because of either subjective or external factors that induce confusion and difficulty: this situation, that everybody has at least partially experienced on some occasion, is now being seriously taken into account by specialists, as subjective difficulties may lead to a sense of un-safeness or inadequacy, or even to accidents of various kinds. Therefore, a system such as the one we envisage can be potentially useful to everybody in some real-life situation.

The fact that training and monitoring are far from unrelated is practically demonstrated by the DALICA system [16] where a user is followed (through exploiting a satellite system) by a personal assistant agent in her visit to either a museum or an archeological area: the agent, on the basis of a constantly updated and improved user profile, suggests routes, provides personalized explanations and checks that the user does not enter forbidden areas or perform forbidden/dangerous actions. Explanations are taken from a repository and customized according to user profile and user desires and preferences.

3.1 User monitoring and training: case study

The following scenario illustrates the dynamic aspects of the knowledge base of a PA/MPA whose knowledge evolves to reflect changes in user behavior as well as in the environment.

Suppose we have a user who must undergo treatment for some illness and therefore needs to take medicine. He/she asks his/her personal assistant about what to do during treatment, e.g., "Can I drink a glass of wine if I have to take this medicine?" Or, more generally, the user may just ask "Can I drink a glass of wine now?" where the personal assistant should give advice based on whether there is medicine to be taken (or other related matters).

As discussed in the previous section, the agent and in particular the PA will have been equipped by the society with initial knowledge about its task. However, if the available knowledge turns out to be either missing or inadequate, then the PA is able to resort to the MPA. Suppose the user asks: "Can I drink a glass of wine now?" and the agent can find no answer in its present state of beliefs.

The PA may have been equipped with a rule such as:

$ALWAYS\ asks(user, do(action, A)), known(A) \div lookup(A)$
where it is stated that if the rule is not verified, and this can only be because action A is not known, then the agent should try to find out what is A by means of $lookup(A)$. In this case, the corresponding reactive rule will belong to MPA, and may be defined as follows:

$lookup(A) \leftarrow check(A)$
$check(A) \leftarrow found\_module(A, M), assert(M)$
$check(A) \leftarrow not\ found\_module(A, M), learn(A, M), assert(M)$

Here, the reactive rule performs a check by invoking $check(A)$: if it will be possible to find in the MPA knowledge base a module M coping with A, then it is asserted (added to the PA); otherwise, the MPA triggers a learning process (by invoking $learn(A, M)$) that should return the module to be asserted. In this case, the learning will presumably be "by being told", i.e, the MPA will obtain the needed module M by the society.

PA will not contain the plain constraint that one should not drink alcohol and take medicine:

$\bot \leftarrow drink, take\_medicine$
as it provides no temporal information for returning a reliable answer. Rather, it may contain the A-ILTL rule stating that one should never drink alcohol within sixty minutes before or after the consumption of medicine:

$NEVER\ (drink : T1), (take\_medicine : T2),\ T1 - T2 < 60$

The rule can be exploited both to block an action if the other one has been performed already, or to provide explanations, should the user ask for advice.

If the user wishes to be trained in taking medicine, we might define for instance the following rule that states which medicine the user should take before dinner. The underlying assumption is that, towards getting trained, the user tells the system about which actions she is about to do.

$ALWAYS$

$(take\_medicine(M) : T1), (have\_dinner : T2),$
$T1 - T2 < 30 :: dinnertime(T1),$
$indication(M, beforedinner) \div train\_user\_md$

$train\_user\_md \leftarrow \ldots$

The $ALWAYS$ rule is false whenever one of the conjuncts is false. I.e., in the reaction to $train\_user\_md$ it should be checked whether dinner-time is near, and then it is appropriate to take the medicine or whether the user is going to have dinner, having forgot the medicine. It should be explained that this kind of medicine requires consumption before dinner. By slightly modifying the operational behavior, the system might check the context and tell the user what to do at the correct time.

However, the system might control if the treatment is effective by checking that the user has recovered after a certain time (say, one week). Otherwise, the treatment must be revised.

$FINALLY(T)\ recovered(T) :: T = 1week \div revise\_treatment$

4 Concluding Remarks

There are several future directions for the ideas that we discussed and sketched in this initial work. First, we intend to fully implement an instance of the proposed framework, starting from EVOLP [17] [18], DALI [19] [20] and KGP [21] [22] agents (which are fully-defined and fully-implemented approaches) that provide the main elements and can be exploited in combination in an implementation. We have discussed a semantic framework for such an integration in [23].

Next, we aim at designing the meta-meta level for controlling knowledge exchange. Particular attention should be dedicated to strategies involving reputation and trust for the evaluation of learned knowledge.

References

1. Richardson, P.J., Boyd, R.: Not by Genes Alone - How Culture Transformed Human Evolution. The University of Chicago Press (2005)
2. Costantini, S., Tocchio, A., Toni, F., Tsintza, P.: A multi-layered general agent model. In: AI*IA 2007: Artificial Intelligence and Human-Oriented Computing, 10th Congress of the Italian Association for Artificial Intelligence. LNCS 4733, Springer-Verlag, Berlin (2007)
3. Costantini, S., Dell'Acqua, P., Pereira, L.M., Tsintza, P.: Runtime verification of agent properties. In: Proc. of the Int. Conf. on Applications of Declarative Programming and Knowledge Management (INAP09). (2009)
4. Lamma, E., Riguzzi, F., Pereira, L.M.: Strategies in combined learning via logic programs. Machine Learning **38**(1-2) (January 2000) 63–87
5. Inoue, K., Kudoh, Y.: Learning extended logic programs. In: IJCAI (1). (1997) 176–181
6. Inoue, K., Haneda, H.: Learning abductive and nonmonotonic logic programs. In Flach, P.A., Kakas, A.C., eds.: Abduction and Induction: Essays on their Relation and Integration, Kluwer (2000) 213–231

7. Alferes, J.J., Pereira, L.M.: Extended logic programs under the well-founded semantics. Available at: http://xsb.sourceforge.net/manual2/node179.html
8. Alferes, J.J., Pereira, L.M.: Reasoning with Logic Programming. LNAI 1111. Springer, Berlin (1996)
9. Dix, J., Pereira, L.M., Przymusinski, T.: Prolegomena to logic programming and non-monotonic reasoning. Non-Monotonic Extensions of Logic Programming - Selected papers from NMELP'96 (1997) 1–36
10. Damásio, C.V., Pereira, L.M.: A survey of paraconsistent semantics for logic programas. In Gabbay, D., Smets, P., eds.: Handbook of Defeasible Reasoning and Uncertainty Management Systems. Volume 2, Reasoning with Actual and Potential Contradictions. Coordenad. Kluwer Academic Press (1998) 241–320
11. Lamma, E., Pereira, L.M., Riguzzi, F.: Belief revision via lamarckian evolution. New Generation Computing **21**(3) (August 2003) 247–275
12. Giraffa, L., Viccari, R.: The use of agents techniques on intelligent tutoring systems. In: 18th Int. Conf. of the Chilean Computer Science Society, IEEE Computer Society (1998) 76–83
13. Kinshuk, Nikov, A., Patel, A.: Adaptive tutoring in business education using fuzzy back-propagation approach. In M.J., S., G., S., D., H., R.J., K., eds.: Usability Evaluation and Interface Design: Cognitive Engineering, Intelligent Agents and Virtual Reality. Lawrence Erlbaum Associates Inc. Publishers (2001) 465–468
14. Baldoni, M., Baroglio, C., Patti, V., Torasso, L.: Using a rational agent in an adaptive, web-based tutoring system. In Brusilovsky, P., Henze, N., Millan, E., eds.: Workshop on Adaptive Systems for Web-Based Education, 2nd Int. Conf. on Adaptive Hypermedia and Adaptive Web-based Systems. (2002) 43–55
15. Giancola, M.: Icf e riabilitazione: dallo studio delle disabilitá alle scelte terapeutiche (towards a formal study of disability to enhance the rehabilitation process). *Laurea Specialistica in Scienze delle professioni sanitarie della riabilitazione (Master Degree in Science and Technologies of rehabilitation)*, Università degli Studi di L'Aquila (2007) in italian.
16. Costantini, S., Mostarda, L., Tocchio, A., Tsintza, P.: Dalica agents applied to a cultural heritage scenario. IEEE Intelligent Systems **3**(2) (2008) Special Issue on Ambient Intelligence.
17. Alferes, J.J., Brogi, A., Leite, J.A., Pereira, L.M.: Evolving logic programs. In: Logics in Artificial Intelligence, Proc. of the 8th Europ. Conf., JELIA 2002. LNAI 2424, Springer-Verlag, Berlin (2002) 50–61
18. J.Alferes, J., Brogi, A., Leite, J.A., Pereira, L.M.: An evolvable rule-based e-mail agent. In: Procs. of the 11th Portuguese Intl.Conf. on Artificial Intelligence (EPIA'03). LNAI 2902, Springer-Verlag, Berlin (2003) 394–408
19. Costantini, S., Tocchio, A.: A logic programming language for multi-agent systems. In: Logics in Artificial Intelligence, Proc. of the 8th Europ. Conf.,JELIA 2002. LNAI 2424, Springer-Verlag, Berlin (2002)
20. Costantini, S., Tocchio, A.: The DALI logic programming agent-oriented language. In: Logics in Artificial Intelligence, Proc. of the 9th European Conference, Jelia 2004. LNAI 3229, Springer-Verlag, Berlin (2004)
21. Kakas, A.C., Mancarella, P., Sadri, F., Stathis, K., Toni, F.: The KGP model of agency. In: Proc. ECAI-2004. (2004)
22. Bracciali, A., Demetriou, N., Endriss, U., Kakas, A., Lu, W., Mancarella, P., Sadri, F., Stathis, K., Terreni, G., Toni, F.: The KGP model of agency: Computational model and prototype implementation. In: Global Computing: IST/FET International Workshop, Revised Selected Papers. LNAI 3267. Springer-Verlag, Berlin (2005) 340–367
23. Costantini, S., Dell'Acqua, P., Pereira, L.M., Tocchio, A.: Conditional learning of rules and plans in logical agents. Draft