EVA: An evolutionary approach to mutual monitoring of learning information agents

D. Rosaci and G.M.L. Sarné

DIMET, Università "Mediterranea" di Reggio Calabria, Via Graziella Loc. Feo di Vito, 89060 Reggio Calabria (Italy)
E-mail: {domenico.rosaci,sarne}@unirc.it

£

Abstract

A learning agent system is composed of agents able to autonomously enrich their knowledge and improve their performances, using learning strategies. The idea underlying this paper is that individual improvements obtained by the learning capabilities of an agent should be exploited to advantage the other agents, and a natural way of obtaining such a result is represented by evolutive processes. However, the biological evolutive mechanisms are often too complex to be reproduced in a software environment. In this context, we argue that the cloning, due to its very simple mechanism of reproduction, can be usefully used. In our approach, a user in a virtual community can substitute an unsatisfactory agent cloning an existing agent having both similar interests and a good reputation in the community. This mechanism induces an evolutionary process in the community, such that the less satisfactory agents are replaced by more performative agents. The key issue of this proposal is that of suitably selecting the agent to be cloned in presence of a user's request, and to this purpose we propose an evolutionary model of reputation. Our evolutionary approach has been implemented on the top of a leaning agent-based recommender system, and a number of experiments show that this novel strategy introduces significant improvements in the effectiveness of the recommendations.

Key words: Adaptivity, Multi-Agent Systems, E-Learning

1 Introduction

An intelligent information agent (Klusch M., Bergamaschi S., Edwards P. and Petta P., 2003; Rosaci D. and Sarné G.M.L., 2006; Rosaci D., Sarnè G.M.L. and Garruzzo S., 2009) is a software application that autonomously and proactively operates on behalf of its human owner, acquiring and maintaining relevant information from distributed and heterogeneous information sources. Generally, an information agent exploits the acquired knowledge to provide its human owner with useful recommendations, e.g. suggesting the next Web sites to visit, proposing new products to buy in e-commerce activities, etc. Thanks to the contributions provided by different research domains (eg. AI, Machine Learning, etc.), the issue of realizing intelligent information agents with learning capabilities has received a great deal of attention in the recent past. Indeed, it has been widely recognized that an information agent should be able to autonomously improve its knowledge and performances through experience to be regarded as "intelligent". For this reason, some recent approaches (Rosaci D., 2007; Reisinger J., Miikkulainen R., 2007; Mosqueira-Rey E., Alonso-Rios D., Vazquez-Garcia A., Baldonedo Del Rio B., Alonso Betanzos A. and Moret-Bonillo V., 2007) proposed the idea of designing communities of agents where each agent is able to modify both its behaviour and its internal knowledge through the use of learning methodologies. A key problem in such a context (Buccafurri F., Palopoli L., Rosaci D. and G.M.L. Sarné, 2004) is that of realizing an effective mutual monitoring among the agents, providing each agent with useful suggestions about other agents that could be contacted for obtaining a fruitful cooperation. The importance of the mutual monitoring derives from the consideration that a learning agent needs to interact with the other agents in its environment in order to improve their individual performances. A typical example of this necessity is given by the newcomer agents, that join for the first time with an agent community. In order to speed-up their effective integration in the community it appears suitable to use, besides individual learning, a cooperation with other more expert agents. As a second example, an agent might not completely satisfy its owner with its recommendations, and therefore the owner itself should have the possibility to search in the community for a possible help. For instance, in (Buccafurri F., Palopoli L., Rosaci D. and G.M.L. Sarné, 2004) and in (Rosaci D., 2007), it is proposed that the owner can integrate the individual knowledge of his agent with that of other agents that have similar interests in the community. A limitation of these approaches is given by the relatively simple cooperation mechanism, that is based only on a similarity measure. Instead, most of the recent approaches to cooperation in multi-agent systems use reputation models to select the best candidates for collaboration (Sabater J. and Sierra C., 2004; Schillo M., Funk P. and Rovatsos M., 2000; Carbo J., Molina J.M. and Davila J., 2002; Yu B. and Singh M.P., 2002). However, these reputation models are conceived for general multi-agent systems, and do not take into account the particular characteristics of a system composed by learning agents. In this paper, we propose to face the problems described above by an *evolutionary* framework, called *EVolutionary Agents* (EVA).

Why do we need an evolutionary approach to improve learning agent-based systems? To answer such a question, consider that the strong point of a learning agent system is that it is composed of agents able to autonomously enrich their knowledge and improve their performances, by using learning strategies. In other words, the capabilities of the single learning agents "improves" in time. However, past systems exploit this agent's improvement to produce better recommendations only for the owner of that agent, without considering the possibility of exploiting it to also improve the effectiveness of the whole multi-agent system. The idea underlying this paper is that individual improvements obtained by the learning capabilities of an agent should exploited to advantage the other agents, and a natural way of obtaining such a result is represented by the *evolution*.

Evolution is defined in biology as a process that results in heritable changes in a population spread over many generations (Gould S.J., 2002; Mayr E., 2001). More in particular, as highlighted in (Futuyma D.J., 2005), evolution consists in a modification of the properties of populations that transcends the lifetime of a single individual. The changes of an individual is not considered evolution, i.e. individual entities do not evolve. Furthermore, the changes in populations that are considered evolutionary are those that are inheritable via the genetic material from one generation to the next.

In this paper, we try to adapt the biological definitions above to the case of a population of learning agents.In particular, our approach is based on the following ideas:

- A user that is not satisfied by his agent, or that is a newcomer in the community, can require the system to provide him with a new agent.
- In order to satisfy the user's request, the system selects in the community that agent having the best score, where the score is computed taking into account both the similarity with the requester user and the reputation in the community.
- The selected agent is *cloned* and the created copy (clone) is sent to the requester user.
- The mechanism to compute the reputation of an agent takes into account the cloning mechanism, and considers a genetic component of the reputation.

It is worth to highlight that our evolutionary technique is based on a cloning mechanism, while in biological contexts evolution generally exploits different mechanisms to improve the characteristics of a population of individuals. For instance, in the human reproduction, two individuals merge their DNA to obtain a new individual having characteristics that are different from those of both the parents. Instead, the cloning does not induce any variation in the characteristic of the new individual with respect to that of the parent. As a matter of facts, in a biological context the cloning does not result as effective in improving the characteristics of individuals, and is rarely used as reproduction mechanism by biologic organisms. However, also the cloning, if completed by a suitable mechanism of selection, generate a change in a population of individuals, and the advantage of the cloning with respect to other biological mechanism of reproductions, is its extreme simplicity. Therefore, our idea is that, in the case of software agents, the changes induced by this simple mechanism can result in an improvement of the agent performances.

This idea derives from the possibility, offered by the current agent technology, of moving agents from a user to another one (agent mobility). It's appear very suitable to exploit agent mobility to face the problems defined above for learning agents, since this allows a user to profitably substitute an unsatisfactory agent with an agent having both similar interests and a good reputation in the community. The agent cloning that we propose induces an evolutionary process in the community, such that the less satisfactory agents are replaced by more performative agents. The key issue in this proposal is that of suitably selecting the agent to be cloned in presence of a user's request, and to this purpose we propose the evolutionary model of reputation described in Section 4.

1.1 Plan of the paper

The paper is organized as follows. Section 2 introduces the research context in which our proposal is placed, i.e. mutual monitoring among agents. Section 3 describes the EVA framework, and Section 4 introduces our reputation model. Section 5 present some experiments while in Section 6 we draw our final conclusions.

2 Related Work

In (Stone P., 2007), some current research directions in machine learning, multi-agent reasoning, and robotics are presented, discussing the problem of designing fully autonomous agents able to learn and keep up with a dynamically changing world, also interacting with one another.

The important issue of mutual monitoring among learning agents has been faced in several recent works. Some proposals address this issue generally providing each agent with an internal representation of both interests and behaviour of the associated human user, usually called ontology (Buccafurri F., Rosaci D., Sarné G.M.L. and Ursino D., 2002; Wooldridge M., 2000). In such a context, the next step for implementing mutual monitoring is to detect inter-ontology properties that can support an agent to choose the most promising agents to be contacted for knowledge-sharing purposes. Some of these approaches use as inter-ontology properties the similarity between ontology concepts (Bowers S., Lin K. and Ludäscher B., 2004), also by determining synonymies and homonymies (Buccafurri F., Rosaci D., Sarné G.M.L. and Ursino D., 2002) between concepts, while other approaches exploit, in addition to similarity, also other properties involving information about the whole agent community, as the *reputation* that an agent has gained inside the community (Buccafurri F., Palopoli L., Rosaci D. and G.M.L. Sarné, 2004). Similarly to our proposals, these approaches try to introduce a form of cooperation in a multi-agent system, based on a mutual monitoring on the agents. However, differently from our approach, none of the aforementioned proposals consider the possibility that the effectiveness of the agents can evolve in time, in consequence of a learning process.

Another approach (Rosaci D., 2007) proposes to induce logical rules representing agent behaviour in the ontology by means of a connectionist ontology representation, based on neural-symbolic networks. This mechanism exploits a new ontology representation, that derives from (d'Avila Garcez A.S. and Zaverucha G., 1999) the idea of using a neural symbolic network for representing a logic program. In this context, the mutual monitoring is realized by introducing a similarity measure of agent ontology that takes into account also the logical representation of the agent's behaviour. This approach, similarly to our one, exploits a learning activity to improve in time the effectiveness of the agent, but such an improvement is only individual and it does not induce a global improvement of the whole system via a cooperative behaviour, as our system does.

Some evolutionary approaches have been proposed in the context of learning agents. For example, in (Reisinger J., Miikkulainen R., 2007), authors study how adaptive representations allow evolution to explore the space of phenotypes by choosing the most suitable set of genotypic parameters. As another example, the approach presented in (Mosqueira-Rey E., Alonso-Rios D., Vazquez-Garcia A., Baldonedo Del Rio B., Alonso Betanzos A. and Moret-Bonillo V., 2007) describes an evolutionary multi-agent system to study of the usability of Web sites and navigation paths. Such a system builds a model of the users trying to reach one URL from another URL, simulates the browsing process, and analyses the Web pages that make up possible paths between source and destination.

These last approaches propose, similarly to our one, the exploitation of evolutionary techniques to make the behaviour of the multi agent-system adaptive. However, any of them supports mutual monitoring among agents, that is the main characteristic of our system, and that appears necessary to implement an effective form of cooperation.

In the literature, the notions of trust and reputation have been widely proposed in the context of information agents (see (Sabater J. and Sierra C., 2004) for an exhaustive survey about them), but the issue of considering reputation in learning agent-based recommender systems has been largely neglected. Recently, in (Garruzzo S. and Rosaci D., 2008) a reputation-based approach has been proposed to lead the evolution of a community of information software agents with the purpose of improving the agent communication. Although this approach is similar to our one in that the agent evolution is driven by a reputation mechanism, however it does not consider any evolutionary component, but it uses a traditional reputation framework.

3 The EVA Framework

Our evolutionary approach is based on the following idea. Assume that U is a community of users, and denote by u the generic user belonging to this community. Also suppose that u is assisted by a set of information software agents able to provide him with recommendations. In particular, we define a function $n(u)$ that yields as output the number of agents that support the user u and we call *agent-set* $A_u = a_1, a_2, .., a_{n(u)}$ the set of these $n(u)$ agents.

3.1 Evaluation of the user's satisfaction

When the user u accesses a Web page, each agent a_i of his agent set provides him with some recommendations, where each recommendation is a Web link. Consequently, during its whole life a_i suggested to u a set of recommendations that we denote by R_i. Moreover, we denote by L_u the set of Web links considered actually interesting by u during the life of the agent a_i. Obviously, we desire that R_i coincides with L_u. However, only a part of the provided recommendations are considered as *relevant* by u and thus belong to L_u. To evaluate the quality of the recommendation set R_i, a lot of measures have been proposed in the past (see (Herlocker J.L., Konstan J.A., Terveen L.G. and Riedl J.T., 2004)). Among those, *precision* and *recall* are the best known measures used in information retrieval (Raghavan V., Bollmann P. and Jung G.S., 1989). Precision of R_i is the fraction of the recommendations that are considered as relevant by the user u, and it can be formally defined as:

$$Pre(R_i) = \frac{|R_i \bigcap L_u|}{|R_i|}$$

In order to have a high precision, the agent has to provide a *high fraction* of relevant recommendations. However, to have a high precision does not mean to be a good recommender, since it is possible that the produced recommendations are only a little fraction of the total possible relevant recommendable links contained in L_u. To this reason, it is also defined the recall of R_i, that is is the fraction of the links actually selected by u that are successfully recommended by the agent a_i. Formally:

$$Rec(R_i) = \frac{|R_i \bigcap L_u|}{|L_u|}$$

In binary classification, recall is called *sensitivity*. So it can be looked at as the probability that a relevant link is suggested by the agent. It is trivial to achieve recall equal to 1 by returning all possible links as recommendations. Therefore recall alone is not enough but it is necessary to measure the number of non-relevant links also, for example by computing the precision.

A well-known approach to take into account both recall and precision is represented by the *F-measure* defined in (Van Rijsbergen C.J., 1979). F-measure is the harmonic mean of precision and recall, that is:

$$F(R_i) = 2 * \frac{Pre(R_i) * Rec(R_i)}{Pre(R_i) + Rec(R_i)}$$

This is also known as the F_1-measure, because recall and precision are evenly weighted. It is possible to use a more general general formula for a F_β-measure, where β is a non-negative real that weights the recall with respect to the precision:

$$F_\beta(R_i) = (1 + \beta^2) * \frac{Pre(R_i) * Rec(R_i)}{\beta^2 * Pre(R_i) + Rec(R_i)}$$

In this paper we use the F_β-measure of R_i to compute the satisfaction of the user u for the recommendations provided by his agent a_i.

Analogously, we can define precision, recall and F-measure for the whole agent-set A_u, considering that the recommendations provided by the agent-set is the union of the sets R_i relative to each agent $a_i \in A_u$. Formally:

$$Pre(A_u) = \frac{|\bigcup_{i=1}^{n(u)} R_i \bigcap L_u|}{|\bigcup_{i=1}^{n}(u) R_i|}$$

$$Rec(A_u) = \frac{|\bigcup_{i=1}^{n(u)} R_i \bigcap L_u|}{|L_u|}$$

$$F_\beta(A_u) = (1+\beta^2) * \frac{Pre(A_u) * Rec(A_u)}{\beta^2 * Pre(A_u) + Rec(A_u)}$$

In the remaining of the paper, we will use the F-measure to synthetically evaluate the user's satisfaction. We remark that other measures as, for instance MAE and ROC, could be used for the same purpose. However, these measures request the user to explicitly rate its satisfaction, while the F-measure can be computed simply observing the acceptance of the provided recommendations. Although we use F-measure as evaluation metrics, in Section 5 we will analyze the performances of our approach also in terms of other measures, by requesting test users to provide explicit feedbacks, and we will show that improving the user's satisfaction in terms of F-measure leads to analogously improvements also in terms of other estimators.

3.2 Evolutionary strategies to improve user's satisfaction

EVA framework introduces an evolutionary strategy to improve the satisfaction of the agents composing the MAS. In particular, as graphically depicted in Figure 1, in the EVA multi-agent architecture two types of agent are appositely conceived to manage such a strategy. First, each user u is provided with an agent called *Local Evolution Manager* (LEM_u). Moreover, the whole MAS is provided with a second type of agent, called *global evolution manager* (GEM). The evolutionary strategy is based on the following six ideas:

(1) The satisfaction of each user u with respect to the recommendations provided by his agent-set A_u is represented by $F_\beta(A_u)$.
(2) Each user u can arbitrarily set the coefficient β to be used in computing $F_\beta(A_u)$.
(3) Each user u can fix a *satisfaction threshold* ρ_u for $F_\beta(A_u)$, under which the quality of the recommendations provided by his agent-set is considered unsatisfactory.
(4) The agent LEM_u of each user periodically computes the measure $F_\beta(A_u)$, and verifies that $F_\beta(A_u)$ is greater than or equal to the threshold ρ_u. In the case $F_\beta(A_u) < \rho_u$, LEM_u determines which agents are individually unsatisfactory, that is which agents a_i have a measure $F_\beta(a_i)$ lesser than

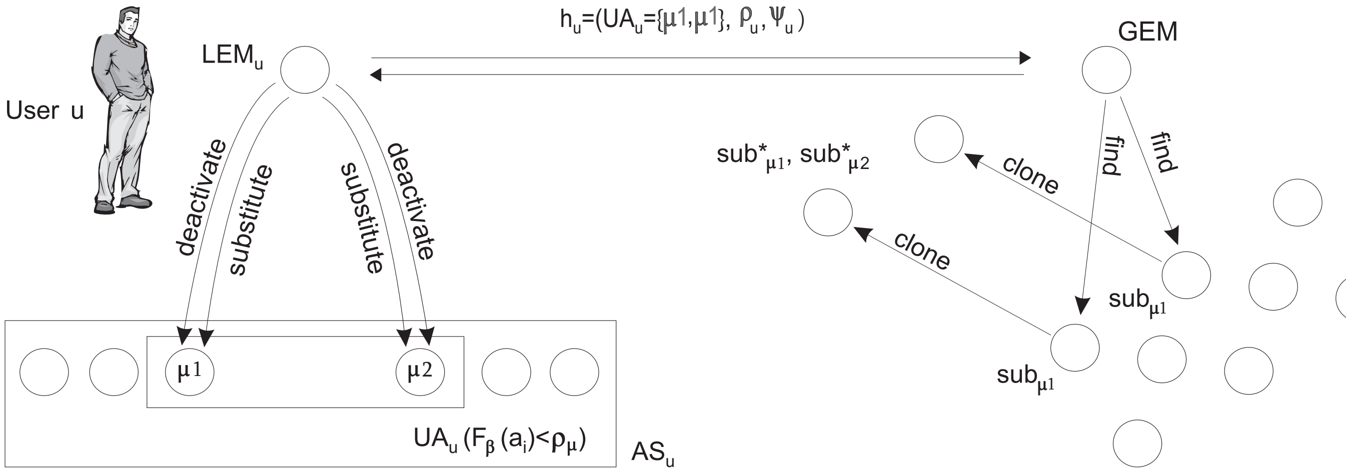

Fig. 1. The evolutionary strategy of EVA framework

ρ_u. We denote by UA_u the set of those unsatisfactory agents. If UA_u is not empty, then LEM_u de-activates the agents belonging to UA_u and sends a *help request* to the global evolution manager GEM. This help request is a message that informs GEM that the agent u deactivated the k agents belonging to the set UA_u and then it needs to substitute them with other k agents, presumably more satisfactory.

As we will see below, the global evolution manager GEM will determine a set of substitutes agents based on both the similarity with the unsatisfactory agents contained in UA_u and the *reputation* that the community gives to these agents. Therefore, it is important that in the help request the agent LEM_u provides to GEM, besides of the set UA_u and the threshold satisfaction ρ_u, also a parameter ψ_u (a real value belonging to the interval $[0, 1]$) that represents how much the user u weights the similarity with respect to the reputation. For example, if $\psi_u = 0.3$ the user gives a thirty percent of relevance to the similarity and a seventy percent of relevance to the reputation. In conclusion, the help request of the agent u is a triplet $h_u = (UA_u, \rho_u, \psi_u)$.

(5) The evolution manager GEM maintains a similarity matrix $\Sigma = \{\Sigma_{ab}\}$, $a, b \in MAS$ where each element represents the similarity between two agents a and b belonging to the MAS. The similarity between two agents is computed as described in (Rosaci D., 2007), and it is a real value belonging to $[0, 1]$. Moreover, GEM also stores, for each agent a belonging to the MAS, a *reputation coefficient* r_a. This coefficient is a real value ranging in $[0, 1]$, computed as described in Section 4, and represents a measure of how much the whole community considers the performances of a as satisfactory.

When GEM receives a help request $h_u = (UA_u, \rho_u, \psi_u)$ by an agent u, it determines a substitute for each agent μ contained in UA_u. To this purpose, GEM examines as candidate to the substitution each agent a of the MAS to which is associated a F-measure greater than ρ_u. We denote by C_μ the set of these candidate agents. Then, GEM computes for each agent a belonging to AS_μ, the score:

$$s(a, \mu) = \psi_u \cdot \Sigma_{a,\mu} + (1 - \psi_u \cdot r_a)$$

Finally, GEM chooses as substitute of μ the agent sub_μ to which is associated the maximum score $max(s(a, \mu))$, $a \in MAS$[2].

(6) For each agent μ belonging to UA_u, the global evolution manager GEM clones the substitute agent sub_μ. The procedure of cloning consists of the creation of an agent sub^*_μ containing the same personal ontology of the agent sub_μ. As we can see in (Rosaci D., 2007), where a representation of the ontology of an information agent is provided, such an ontology contains both the categories of interests for the agent and the causal implications learnt by the agent during its life. This personal ontology can be thus viewed as a sort of *base information* about the agent, storing those categories the agent is interested in and the logic relationships the agent considers existing between the events. Therefore, cloning can be considered as a duplication of this base information, similarly to the duplication of the genetic material in biological evolutive processes. The clone agent sub^*_μ is then transmitted to the local evolution manager of u, that adds it to the agent-set of u in substitution of the unsatisfactory agent μ. From now on, the *evolution* of sub^*_μ will be completely independent on the *parent agent* sub_μ, since the model encoded in it will be applied to the environment of the user u. This way, monitoring the activity of u, the agent sub^*_μ will learn new information that could possibly modify its initial personal ontology.

It is important to point out that the core of the idea described above is represented by the last idea of the strategy. An agent μ that assisted a user u in a unsatisfactory manner is *killed* by the agent LEM_u and substituted by another agent sub^*_μ, provided via the activity of the global evolution manager GEM. The reason that leads to believe that this substitution will lead some advantages to the user u is based on the fact that sub^*_μ is the clone of an agent that presents the following features:

(a) It has a F-measure (as computed by its own user) that is greater than the satisfaction threshold required by u.
(b) It has a good score, computed by taking into account both the similarity with the substituted agent and the global reputation in the community.

Feature (a) alone is not sufficient to conjecture that the new agent sub^*_μ will produce an F-measure greater than the satisfaction threshold of u but only assures that the parent agent of sub^*_μ is goodly evaluated by its own user, that

[2] In case more than one agents are associated to the maximum score, then the agent having the maximum F-measure is chosen.

could obviously have a different perception of the satisfaction with respect to u. However, feature (b) assures that the parent agent of sub^*_μ has a personal ontology similar enough to that of substituted agent μ and, in addition, it has a good reputation in the community. Considered together, the two features gives a reasonable motivation to believe in a possible improvement of the satisfaction of u with respect to the original situation in which the unsatisfactory agent μ was present.

4 A novel approach to determine agent's reputation in an evolutionary environment

In multi-agent systems, the term *reputation* is commonly used to indicate the opinion of one agent about something (Sabater J. and Sierra C., 2001). The study of reputation in multi-agent systems has generated several approaches and a lot of reputation models have been proposed in the literature (Schillo M., Funk P. and Rovatsos M., 2000; Carbo J., Molina J.M. and Davila J., 2002; Yu B. and Singh M.P., 2002) (for an exhaustive review, see (Sabater J. and Sierra C., 2004)). Accordingly with (Sabater J. and Sierra C., 2001) we recognize three main issues in the study of the reputation in multi-agent systems:

- Reputation on an agent, rather than being a single concept, is a multi-dimensional concept, since it has to take into account different aspects. For instance, the reputation of being a good seller in ebay summarizes the reputation of having good products, the reputation of applying suitable prices, the reputation of giving appropriate descriptions of the products and the reputation of providing fast and secure delivery.
- Moreover, each agent usually has a different way of combining the single aspects of the reputations, weighting each aspect by its personal point of view (this is called the *ontological* dimension of the reputation).
- Finally, when an agent belongs to a group, besides the personal evaluation of the reputation (the *individual* dimension of the reputation), it has to take into account the opinion of the whole community (the *social* dimension of the reputation).

In this work, we deal with the reputation of an agent, belonging to an agent community, with respect to its ability of providing good recommendations. We note that in our framework an agent, besides giving recommendations to its own owner, can be cloned and its clones will assist other owners. In this perspective it is reasonable to model both an individual and a social dimension of the reputation, where the social dimension is related to the cloning activity.

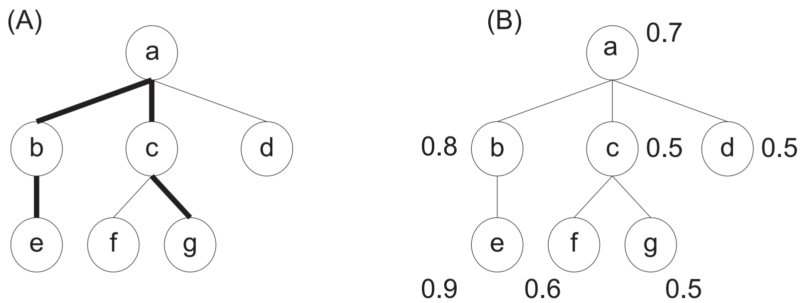

Fig. 2. An example of Descent Tree

4.1 Ontological and individual dimensions

Our reputation problem presents a unique individual dimension, i.e. the reputation of providing good recommendations to the agent's owner, where "good" means "relevant". Therefore, accordingly to the discussion provided in Section 3.1, it is reasonable to consider, as possible ontological dimensions of the individual reputation, both the precision and the recall of the recommendations. Consequently, in our approach we assume the F-measure $F_{\beta_{u_a}}(a)$ as a global measure of the individual reputation of the agent a, taking into account both the two ontological dimensions and where we denote as u_a the owner of a and as β_{u_a} the quantitative representation of the personal consideration of u_a for the precision with respect to the recall.

4.2 Social Dimensions

Since the evolutionary strategy we have described above implies that an agent can be cloned and its clone can be moved in a new environment, it is necessary to define the agent reputation under this novel, particular perspective.

To study this problem, it is useful to represent the relationships the cloning process introduces in the set of the agents, by using some definitions directly derived from the terminology used to represent relationships between entities that share a genealogical origin. Figure 2-(A) helps us to introduce these definitions. This figure represents a set of agents, that have been mutually involved in cloning activity, by using a sort of "genealogical" tree, where each node is associated to an agent, and each edge represents a cloning process between two agents. We call *parent* the agent that is cloned and *child* the agent that results from the cloning. For instance, in the example of Figure 2-(A), the agent a is the parent of the agent b. Obviously, while an agent has only one parent, it can have zero or more children. We call *siblings* two agents, like b and c in Figure 2-(A), that have the same parent. The parental relationship can be recursively applied, leading to introduce the notion of *ancestor* of an agent. For instance, a is an ancestor of e since it is the parent of b which is, in its turn, the parent of e. Already using the analogy with the human kinship system, we define *relatives* two agents that share a common ancestor, as all the agents in Figure 2-(A), and we can also define a kinship degree between two relatives, as the distance between them in the tree passing through the common ancestor. For instance, the agents e and g have kinship degree equal to 4, since the path linking them is composed by four edges. All these notions are formally defined below.

Definition 1 (Parent and Sibling Agent). Let a be an agent of the community. We denote by $children_a$ the set of clones of this agent (we remark that an agent can produce zero or more clones). Two agents b and c, both belonging to $children_a$, are called *sibling agents*. Correspondingly, a is called the *parent agent* of each agent belonging to $children_a$

Definition 2 (Ancestor Agent). Let a and p be two agents of the community. We say that p is an *ancestor agent* of a if either: (i) p is the parent agent of a, or (ii) there exists an agent c in the community such that p is the parent agent of c and c is an ancestor agent of a.

Definition 3 (Relatives, Descent Tree and Kinship Degree). Let a and b be two agents of the community. We say that a and b are *relatives* if a and b share a common ancestor agent p. We call *family* of a, denoted by $\mathcal{F}_a$ the set of all the relatives of a. We define the *Descent Tree* of a, a tree $DT_a = \langle V, E \rangle$ such that (i) each agent $x \in \mathcal{F}_a$ is associated with a unique vertex $v_a \in V$ and (ii) each pair (x, y), $x, y \in \mathcal{F}_a$, such that x is the parent agent of y, is associated with a unique edge $e_{x,y} \in E$. Finally, let a and b be two agents, such that a and b are relatives. We define the *kinship degree* of a and b, denoted by $k_{a,b}$, the length of the path that links a and b in the Descent Tree DT_a.

In this context, the following two observations should be remarked.

(1) If an agent a is cloned, then each of its clones b, where $b \in children_a$, being identical to a, has to inherit at the cloning time the same reputation of a.
(2) The inherited reputation of b can be considered as a sort of initial reputation. However, considering that b will moved to a user different from that of a, the reputation of b should evolve in time, taking into account, besides the initial inherited reputation, the individual satisfaction expressed by its current owner. It is also necessary to determine how these two components (i.e., inherited reputation and individual satisfaction) are weighted for determining a unique, global, measure of reputation.
(3) Due to the cloning process, each agent a belongs to a *family* of relatives, represented in the descent tree DT_a. Since a shares with all these relatives some similarities deriving from the cloning process, it seems reasonable

to consider the performances of all these relatives in order to determine the reputation of a. This consideration introduces a social component (deriving from the relatives' performances) in the computation of the reputation.

To take into account both the observations above, in our approach we introduce a *reputation coefficient* r_a associated with each agent a, belonging to the interval $[0, 1]$, where $r_a = 1$ means complete reliability of a. Such a reputation coefficient is computed as a weighted mean of n contributions, where each contribution is associated with one agent of the descent tree DS_a and where n is the number of relatives composing the descent number DT_a. The first contribution is associated with the agent a itself, being relative to the individual dimension of the reputation, and it is equal to the F-measure $F_{\beta_a}(a)$. Each of the other $n-1$ contributions is associated with one of the relatives of a, and it is equal to the F-measure $F_{\beta_a}(a)$. Each of this component, relative to an agent $x \in \mathcal{F}_a$, is weighted by a coefficient, equal to $k_{a,x} + 1$. This way, the contribution of the satisfaction $F_{\beta_a}(a)$ to the overall reputation is equal to $F_{\beta_a}(a)$, being $k_{a,a} = 0$, while the contribution of the satisfaction obtained by each other relative b is as smaller as higher is the kinship degree between a and b. More formally:

$$r_a = \frac{\sum_{b \in \mathcal{F}_a} \frac{F_{\beta_b}(b)}{k_{a,b}}}{\sum_{b \in \mathcal{F}_a} \frac{1}{k_{a,b}}}$$

As an example, consider the situation depicted in Figure 2-(B), where to each agent of the descent tree is associated the satisfaction (i.e. the F-measure) with respect to its own owner. The F-measure of the agent e is equal to 0.9. The reputation of e can be computed as follows:

$$r_e = \frac{0.9 + \frac{0.8}{2} + \frac{0.7}{3} + \frac{0.5}{4} + \frac{0.5}{4} + \frac{0.6}{5} + \frac{0.5}{5}}{1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{4} + \frac{1}{5} + \frac{1}{5}} = 0.696$$

5 Experimental results

In this section, we describe some experiments we have performed to evaluate the advantages introduced by our approach in a multi-agent recommender system. In particular, we have implemented EVA on the top of the CILIOS recommender system (Rosaci D., 2007), that can be used to recommend Web pages to users. CILIOS provides each user with a set of recommended Web pages, such that each page i is associated with a rate p_i that represents the degree of relevance for the user that the system assigns to i. In our experiment, we have request to the user to explicitly rate each page recommended by the system, after having visioned it. We denote by r_i the rate provided by the user for the page i. Both p_i and r_i are integer values ranging in the interval [1-5], where 1 (resp. 5) means minimum (resp. maximum) relevance.

5.1 Description of the experiment

In our experiment, we have used four test-sets of real users, that we denote as $S1$, $S2$, $S3$ and $S4$, having different cardinalities. More in particular, $S1$ (resp. $S2$, $S3$, $S4$) contains 30 (resp. 50, 70, 102) users, where each user u is monitored by a CILIOS agent A_u. Each agent is initially provided with a personal ontology, having the structure described in (Rosaci D., 2007), and where the concepts contained in the personal ontologies belong to the *literature ontology* publicly available at the site of the MUADDIB project (MUADDIB URL, 2009), an XML-schema ontology built for testing recommender systems. In addition, each user u is assisted by an EVA agent LEM_u, which periodically computes the measure $F_\beta(A_u)$, and verifies that $F_\beta(A_u)$ is greater than or equal to the threshold ρ_u, while a global evolution manager GEM is associated with the whole multi agent system and implements the evolutionary strategy described in 3.2. In our experiments we have used $\beta = 1$, and we have computed the average satisfaction $F(S_i)$ for each user set S_i, $i = 1, 2, 3, 4$, as follows:

$$F(S_i) = \frac{1}{|S_i|} \cdot \sum_{u \in S_i} F_1(A_u)$$

We have trained the CILIOS agents on the set of XML Web sites available at the MUADDIB site. After the training phase, we have performed a test phase, during which CILIOS agents, whitout exploiting the additional EVA agents, generated recommendations for their users for 5 days. The average satisfaction values after this test phase for the four sets are those reported in the first row of Table 1, denoted by *day* 5.

Starting from this initial situation, we have activated the EVA agents, leaving the system to generate recommendations exploiting our evolutionary approach.

5.2 Evaluation in terms of F-measure

The results of the experiment, in terms of average satisfaction F, computed in correspondence of the days 9,15,25,45 and 60, respectively, are reported in Table 1.

T	S_1	S_2	S_3	S_4
day 5	0.62	0.65	0.68	0.70
day 9	0.64	0.70	0.74	0.75
day 15	0.67	0.72	0.78	0.83
day 25	0.71	0.77	0.83	0.86
day 45	0.74	0.80	0.85	0.89
day 60	0.77	0.82	0.87	0.92

Table 1
The temporal evolution of the average F-measure

We remark that our approach significantly improves the performance of the recommender system, for each of the considered test-set. After 60 days, the improvement in terms of F is equal to about 24 percent for the set S_1, about 26 percent for the set S_2, about 28 percent for the set S_3 and about 31 percent for the set $S4$. Therefore, the larger is the number of users in the test set, the higher is the improvement of the satisfaction induced by the evolutionary approach. We argue that this result can be explained by considering that a larger size of the user space increases the possibility to determine satisfactory clones by the evolutionary approach.

5.3 Evaluation with other metrics

Although evaluation by means of precision, recall and F-measure has been widely used in Recommender Systems, other metrics have been proposed in the literature, capable of deeply analyzing the advantages and the limitations of a recommender system.

More in particular, three main categories of metrics have been proposed for evaluating the accuracy of a prediction algorithm, namely accuracy metrics, classification accuracy metrics, and rank accuracy metrics (see (Herlocker J.L., Konstan J.A., Terveen L.G. and Riedl J.T., 2004)). Predictive accuracy metrics measure how close the recommender systems predicted ratings are to the true user ratings. Predictive accuracy metrics are particularly important for evaluating tasks in which the predicting rating will be displayed to the user such as in the case of our experiment. To measure statistical accuracy we use the mean absolute error (MAE) metric, defined as the average absolute difference between predicted ratings and actual ratings. In our experiments we compute, for each test set, the MAE for each user, and then the average over all the users of the set. Formally, for each set S_j, $j = 1, 2, 3, 4$:

$$MAE_{S_j} = \frac{\sum_{i=1}^{N_{S_j}} |p_i - r_i|}{N_{S_j}} \tag{1}$$

where N_{S_j} is the total numbers of recommendations generated for all the users of S_j.

Classification accuracy metrics measure the frequency with which a recommender system makes correct or incorrect decisions about whether an item is good. We use Receiver Operating Characteristic (ROC) sensitivity to measure classification accuracy. The ROC model attempts to measure the extent to which an information filtering system can successfully distinguish between signal (relevance) and noise. The ROC curve represents a plot of recall (percentage of good recommendations returned), versus fallout (percentage of bad recommendations returned). We consider a recommendation good if the user gave it a rating of 4 or above, otherwise we consider the recommendation bad. We refer to this ROC sensitivity with threshold 4 as ROC-4. ROC sensitivity ranges from 0 to 1, where 1 is ideal and 0.5 is random. Since comparing multiple systems using ROC curves is tedious and subjective, we provide as a single summary performance number the area underneath a ROC curve, also known as Swets A measure, that can be used as a single metric of the systems ability to discriminate between good and bad recommendations. Moreover, to complete our analysis, we compute, besides MAE and ROC curve, also the Customer ROC (CROC) curve, another metric introduced in (Schein A.I., Popescul A., Ungar L.H. and Pennock D.M., 2005), and we use as synthetic evaluation parameter the area under the CROC curve. Rank accuracy metrics measure the ability of a recommendation algorithm to yield a recommended ordering of items that matches how the user would have ordered the same items. It is important to point out that ranking metrics do not attempt to measure the ability of an algorithm to accurately predict the rating for a single item, therefore in our case, where we display to the user predicted rating values, it is important to additionally evaluate the system using a predictive accuracy metric as described above. We use the Normalized Distance-based Performance Measure (NDPM) as rank accuracy metric, that is computed as follows:

$$NDPM = \frac{2 \cdot C^{-} + C^{u}}{2 \cdot C^{i}} \tag{2}$$

where C^{-} is the number of contradictory preference relations between the system ranking and the user ranking. A contradictory preference relation happens when the system says that item 1 will be preferred to item 2, and the user ranking says the opposite. C^{u} is the number of compatible preference rela-

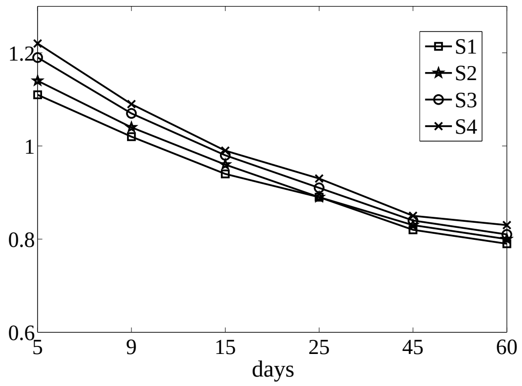

Fig. 3. Average mean absolute error (MAE) of the EVA recommender system

tions, where the user rates item 1 higher than item 2, but the system ranking has item 1 and item 2 at equal preference levels. C^i is the total number of preferred relationships in the users ranking (i.e. pairs of items rated by the user for which one is rated higher than the other). NPDM is a value ranging in [0.0..1.0], where 0.0 means best recommendations and 1.0 means worst recommendations. In our experiment, we have computed the average of NPDM on all the users.

Figure 3 shows how the average MAE decreases in time, and this means an improvement of the effectiveness due to the evolutionary strategy. This improvement is confirmed by the analysis of the Swet's A measure related to the ROC-4 curve (see Figure 4), where the advantage is represented by the increment in time of the measure. Analogous considerations can be done considering the area under the CROC curve in Figure 5. The analysis of the NPDM measure, represented in Figure 6 shows that this measure decreases in time, further confirming that the evolutionary strategy improves the effectiveness of the recommendations. All the measures show that, after 60 days of using our approach, the performances of the system improves for a 25-30 percent.

6 Conclusions

In this paper we have presented an evolutionary strategy that allows a user of a learning agent-based recommender system to substitute an unsatisfactory agent with a more effective one. The substitution is suggested by our system, operating a selection on the available agents that takes into account both the similarity with the requester user and the reputation of the agent in the whole community. The selected agent is cloned and assigned to the requester agent, and if it performs effectively enough, it is maintained, otherwise it is in its turn substituted and killed. This strategy leads to an evolution of the multi-agent

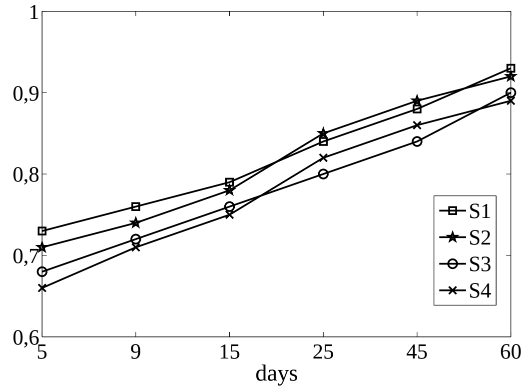

Fig. 4. Swet's A measure related to the Receiver Operating Characteristic ROC-4 of the EVA recommender system

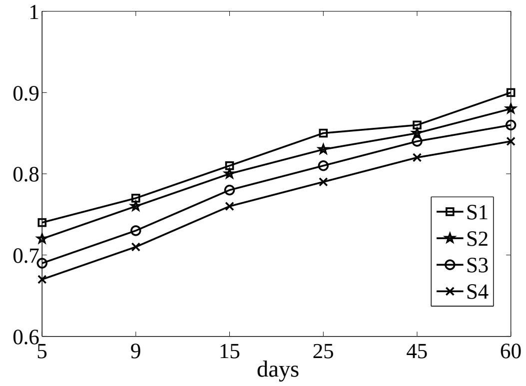

Fig. 5. Area under the Customer ROC (CROC) curve of the EVA recommender system

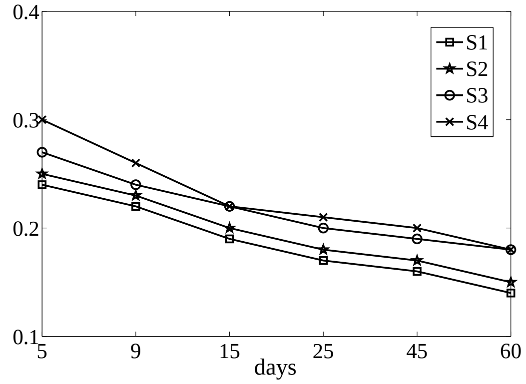

Fig. 6. Average Normalized Distance-based Performance Measure (NPDM) of the EVA recommender system

system, with the purpose of eliminating the unsatisfactory agents and leaving to evolve the most effective agents. The core of our strategy is an evolutionary reputation model, where the reputation of a parent agent is inherited by a child agent, that will autonomously evolve in its own environment, using its learning capabilities to increase this "genetic", initial contribution to its reputation. We have performed an experimental campaign to evaluate, using different well-known evaluation metrics, how our approach is capable to improve in time the effectiveness of the recommendations produced by a given recommender system. To this purpose, we have implemented our approach on the top of the CILIOS recommender system, and we have observed a significant improvements of the performances induced by the evolutionary strategy. As for our ongoing research, we are planning to more deeply study theoretical properties of our strategy as, for instance, the *robustness* with respect to the failure of some agent and the *adaptivity* to the changes in the external environments. We argue that the study of these properties could highlight other advantages introduced by this novel strategy, besides of the improvement in the effectiveness of the recommendations.

References

Bowers S., Lin K. and Ludäscher B., 2004. On Integrating Scientific Resources Through Semantic Registration. In: Proceedings of the Third International Semantic Web Conference. Springer, Hiroshima, Japan, pp. 620–634.

Buccafurri F., Palopoli L., Rosaci D. and G.M.L. Sarné, 2004. Modeling Cooperation in Multi-Agent Communities. Cognitive systems Research 5 (3), 171–190.

Buccafurri F., Rosaci D., Sarné G.M.L. and Ursino D., 2002. An Agent-Based Hierarchical Clustering Approach for E-Commerce Environments. In: Proceedings of the Third E-Commerce and web Technologies (EC-Web 2002). Lecture Notes in Computer Science. Springer-Verlag, Aix-en-Provence, France, pp. 115–118.

Carbo J., Molina J.M. and Davila J., 2002. Comparing predictions of sporas vs. a fuzzy reputation agent system. In: Proceedings of Third International Conference on Fuzzy Sets and Fuzzy Systems. Interlaken, pp. 147–153.

d'Avila Garcez A.S. and Zaverucha G., 1999. The Connectionist Inductive Learning and Logic Programming System. Applied Intelligence 11 (1), 59–77.

Futuyma D.J., 2005. Evolution. Sunderland: Sinauer Associates.

Garruzzo S. and Rosaci D., 2008. Agent clustering based on semantic negotiation. ACM Transaction on Autonomous Adaptive Systems 3 (2), 1–40.

Gould S.J., 2002. The Structure of Eveolutionary Theory. Belknap Press.

Herlocker J.L., Konstan J.A., Terveen L.G. and Riedl J.T., 2004. Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems (TOIS) 22 (1), 5–53.

Klusch M., Bergamaschi S., Edwards P. and Petta P. (Ed.), 2003. Intelligent Information Agents. Lecture Notes in Computer Science, Vol. 2586, Springer.

Mayr E., 2001. What Evolution is. Basic Books.

Mosqueira-Rey E., Alonso-Rios D., Vazquez-Garcia A., Baldonedo Del Rio B., Alonso Betanzos A. and Moret-Bonillo V., 2007. An Evolutionary Multiagent System for Studying the Usability of Websites. In: Proceedings of the International Workshop on Web-based Tools. Patrasso, Greece, pp. 34–41.

MUADDIB URL, 2009. http://muad.altervista.org.

Raghavan V., Bollmann P. and Jung G.S., 1989. A Critical Investigation of Recall and Precision as Measures of Retrieval System Performance. ACM Transactions on Information Systems (TOIS) 7 (3), 205–229.

Reisinger J., Miikkulainen R., 2007. Acquiring Evolvability Through Adaptive Representations. In: Proceedings of the Ninth Conference on Genetic and Evolutionary Computation (GECCO '07). ACM, London, England, pp. 1045–1052.

Rosaci D., 2007. CILIOS: Connectionist Inductive Learning and Inter-Ontology Similarities for Recommending Information Agents. Information Systems 32 (6).

Rosaci D. and Sarné G.M.L., 2006. MASHA: A Multi-Agent System Handling User and Device Adaptivity of Web Sites. User Modeling and User-Adapted Interaction 16 (5), 435–462.

Rosaci D., Sarnè G.M.L. and Garruzzo S., 2009. A Distributed Recommender System Supporting Device Adaptivity. ACM Transactions on Information Systems (TOIS). To Appear. Available from the authors.

Sabater J. and Sierra C., 2001. REGRET: A Reputation Model for Gregarious Societies. In: Proceedings of the Fourth Workshop on Deception Fraud and Trust in Agent Societies. Montreal, Canada, pp. 61–70.

Sabater J. and Sierra C., 2004. Review on Computational Trust and Reputation Models. Artificial Intelligence Review 24 (1), 33–60.

Schein A.I., Popescul A., Ungar L.H. and Pennock D.M., 2005. CROC: A New Evaluation Criterion for Recommender Systems. Electronic Commerce Research 5 (1), 51–74.

Schillo M., Funk P. and Rovatsos M., 2000. Using Trust for Detecting Deceitful Agents in Artificial Societies. Applied Artificial Intelligence 14, 825–848.

Stone P., 2007. Learning and Multiagent Reasoning for Autonomous Agents. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence. Hyderabad, India, pp. 12–30.

Van Rijsbergen C.J., 1979. Information Retrieval. Butterworths, London.

Wooldridge M., 2000. Reasoning About Rational Agents. MIT Press.

Yu B. and Singh M.P., 2002. An Evidential Model of Distributed Reputation Management. In: Proceedings of Autonomous Agent and Multi-Agent Systems. Bologna, Italy, pp. 294–301.