Modeling Cooperation in Multi-Agent Communities

F. Buccafurri, D. Rosaci, G. M. L. Sarnè and L. Palopoli

DIMET, Università “Mediterranea” di Reggio Calabria, Via Graziella Loc. Feo di Vito, 89100 Reggio Calabria (Italy)
E-mail: {bucca,rosaci,sarne,palopoli}@ing.unirc.it

Abstract

In Multi-Agent Systems the main goal is providing fruitful cooperation among agents in order to enrich the support given to user activities. Cooperation can be implemented in many ways, depending on how local knowledge of agents is represented and consists, in general, in providing the user with an integrated view of individual knowledge bases. But the main difficulty is determining which agents are promising candidates for a fruitful cooperation among the (possibly large) universe of agents operating in the net. This paper gives a contribution in this context, by proposing a formal framework for representing and managing cooperation in multi-agent networks. Semantic properties are here represented by coefficients, and adaptive algorithms permit the computation of a set of agents suggested for cooperation. Actual choices of the users modify internal parameters in such a way that the next suggestions are closer to users expectancy.

Key words: Multi-Agent Systems, Agent Cooperation, Adaptive Learning

1 Introduction

Coordinating the activities of multiple agents is a basic task for the viability of any system in which such agents coexist. Each agent in an agent community does not have to learn only by its own discovery, but also through cooperation with other agents, by sharing individual learned knowledge. Indeed,

cooperation is often considered as one of the key concepts of agent communities [18,11,12,17,10,35,4,14]. Moreover, the problem of integrating heterogeneous knowledge bases has to be considered in order to implement cooperation [2,3,25,24,21,31,5,36]. Researchers in Intelligent Agent Systems have recognized that learning and adaptation are essential mechanisms by which agents can evolve coordinated behaviours finalized to meet the knowledge of the interest domain and the requirements of the individual agents [32,6,28,34,29]. In order to realize such a cooperation, some techniques developed in the field of Machine Learning has been introduced in various Multi-Agent Systems (often denoted by MAS) [20,8,33].

In such a context, this paper describes a new multi-agent model, called SPY, able to inform the individual user’s agent of a multi-agent network about which agents are the most appropriate to be contacted for possible knowledge integration. The main contributions of this paper are the following:

- We point out which properties can be considered important for driving the integration of the knowledge coming from non local agents and give a formal model in which such properties are represented as a quantitative information by means of a number of real coefficients.
- We propose an adaptive method for determining, for a given agent $a$ of a multi-agent net, the most appropriate agents to cooperate with $a$. Such a method is adaptive in the sense that it takes into account reactions of the user (by exploiting some reactive properties) and, as such, its result depends on user behaviour.
- On the basis of this model we design a strategy for supporting cooperation of agents operating in a multi-agent network. The first step consists in providing the user with a number of agent lists, each containing the most appropriate agents for cooperation, from which the user can choose agents she/he wants to contact for supporting her/his activity. The multiplicity of such choice lists depends on the multiplicity of the properties that can be used as preference criteria. Users are free to use the suggested lists even partially, or to ignore them. In any case, user’s behaviour induces a modification of some coefficients (describing reactive properties) in such a way that lists suggested in the future are (hopefully) closer to real user needs. Therefore, the system learns from user’s behaviour about how to provide her/him with suggestions meeting as much as possible their expectancy.

The plan of the paper is the following. In the next section we present related work. In Section 3 we describe how we represent agent knowledge. Section 4 includes the core of our proposal: after the definition of the semantic properties we show how to extract such properties and exploit them in order to detect good candidates for cooperation. Cooperation is implemented by merging agent knowledge bases through a technique presented in Section 5. The model is validated by a number of experiments and examples reported in Sec-
Finally, we draw our conclusion and sketch a description of a system implementation in Section 7.

2 Related Work

In the context of Machine Learning approaches, [34] illustrates the progress made in the available work on learning and adaptation in Multi-Agent Systems, and provides a general survey of Multi-Agent Systems using adaptation and learning. In [32], a demonstration of how reinforcement-learning agents can learn cooperative behavior in a simulated social environment is provided, specifying that if cooperation is done intelligently, each agent can benefit from other agents instantaneous information, episodic experience and learned knowledge. [6] concerns with how a group of intelligent agents can work together in order to solve a problem or achieve a common goal, by using Machine Learning techniques to refine their knowledge. An example of a practical situation that needs to be modeled in a group of agents is presented in [28], where a probabilistic reciprocity mechanism is introduced to generate stable and cooperative behavior among a group of self-interested agents. In [30], authors identify procedures and environments under which self-interested agents may find it beneficial to help others and point out that sharing of experiences about other agents among reciprocative agents will limit the exploitative gains of selfish agents.

A large numbers of Multi-Agent Systems using learning techniques have been proposed in literature. Among these, we cite some significant proposals:

- In [25], a learning system, called COLLAGE, that endows the agents with the capability to learn how to choose the most appropriate coordination strategy from a set of available coordination strategy, is presented.
- Amalthaea [20], that is an evolving Multi-Agent Ecosystem for personalized filtering, discovery and monitoring of information sites.
- [8] presents an information retrieval system where a multi-agent learning approach to information retrieval on the Web is proposed, in which each agent learns its environment from the user’s relevance feedback by using a neural network mechanism.
- In [19] a system of collaborative agents is proposed, where the collaboration among agents assisting different users is introduced in order to improve the efficiency of the local learning.
- The system Challenger [7], consisting of agents which individually manage local resources and communicate with one another to share their resources in the attempt of utilizing them more efficiently, in order to obtain desirable global system objectives.
- [33] presents a multi-agent architecture applied to Cooperative System En-
gineering, useful in modeling activities and providing support to cooperative tasks.

- [27] aims to establish a mechanism that enables agents to cope with environments that contain both selfish and co-operative entities, where the mixture and the behavior of these entities is previously unknown to all agents. Authors achieve this by enabling agents to evaluate trust in others, based upon the observations they gather.

Such techniques open, on the one hand, the possibility of integrating individual agent knowledge for acquiring an enhanced knowledge of the environment. On the other hand, they consider the problem of determining which agents are promising candidates for suitable knowledge integration, but, differently from our approach, none of them proposes automatic techniques for solving such a problem.

We point out that in the above approaches the knowledge involved in the cooperative exchange is not stored in a complex data structure, but it generally consists of unstructured elementary information about both the environment and the various actions performed by the agents. But an important issue recently emerged in the MAS field deals with the necessity of organizing the available knowledge in *ontologies* [22,15,16,13], that are sophisticated content oriented data structures.

## 3 The Knowledge Bases

Throughout the paper we refer to a given set of agents \( \Lambda \) of cardinality \( n \) and we suppose that all agents in \( \Lambda \) can cooperate with each other. Thus we can see the set \( \Lambda \) as a undirected complete graph of agents whose arcs represent possible cooperation. W.l.o.g., we identify agents in \( \Lambda \) by the cardinal numbers \{1,...,n\}.

### 3.1 An Ontology for describing the domain of interest

Since we consider only homogeneous agents, we suppose that a unique environment is associated with our agent net. We represent such an environment in our model by a set of objects. For the rest of the section we consider a set of objects \( O \) as given.

A domain \( D \) on \( O \) is a set of classes of objects. We suppose that a suitable semantics underlying the classification provided in a domain is given. The notion of domain is formally defined next.
**Definition 1** A Domain on $O$, denoted by $D$, is a set $D \subseteq 2^O$ such that both: (1) $\forall o \in O, \{o\} \in D$, and (2) there exists an element $r$ of $D$, called the root such that, $\forall o \in O, o \in r$. Elements of $D$ are called classes.

In words, a domain is a set object classes containing, at least, a root class collecting all the objects of $O$ and, for each object $o$ of $O$, the singleton class $\{o\}$. Throughout the rest of the section, we assume a domain $D$ on $O$ as given.

Next we present an example of domain.

**Example 1** Consider a library of a Computer Science Group. The set of objects is the set of books belonging to the library. We assume that each book is identified by authors. The domain is that shown in Table 1. We call this domain CSL. Classes represent categories of books (e.g., Books of Theory for Computer Science I Course).

Set containment induces a partial ordering among elements of a domain $D$. A domain $D$ plus this ordering in called ontology.

**Definition 2** An ontology on $D$, denoted by $O_D$, is a partially ordered set $\langle D, \subseteq \rangle$. The ontology graph of $O_D$ is a directed graph $G(O_D) = \langle N, A \rangle$, where $N$ (set of nodes) coincides with $D$ and $A$ (set of arcs) is the binary relation obtained as transitive reduction of the relation $\subseteq$ of $O_D$\(^1\). The node $r$ of $G(O_D)$, where $r$ is the root of $D$, is called root of $G(O_D)$.

Note that, as a consequence of item (2) of Definition 1, each non-root node is reachable from the root $r$ of $G(O_D)$. Furthermore, by item (1) of Definition 1, nodes of $G(O_D)$ with out-degree 0 coincide with singleton classes of $D$.

**Example 2** For an example of ontology graph see Figure 1, reporting the ontology graph of the CSL domain of Example 1.

An ontology based on a generalization hierarchy is suitable for representing many real-world situations, like the topics in web engines and in web sites, the items in e-commerce, the staff hierarchy of an organization and so on. It is worth noting that this is not the only possible choice for representing the environment of agents. Indeed, in different contexts, as semi-structured data in web documents, other kinds of ontologies may be better suited (for example OEM-graphs [1], SDR-networks [23], etc.).

---

\(^1\)(A, B) is in the transitive-reduction of $\subseteq$ iff $A \subseteq B$ and $\not\exists C$ such $A \subseteq C$ and $C \subseteq B$. 

5
<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
<th>Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year 1</td>
<td>year 1</td>
<td>Horstmann, Kamin, Hubbard, Colin</td>
</tr>
<tr>
<td>Year 2</td>
<td>year 2</td>
<td>Hubbard, Papadimitriou</td>
</tr>
<tr>
<td>Year 3</td>
<td>year 3</td>
<td>Lauch, Tanenbaum, Linux Guide</td>
</tr>
<tr>
<td>Year 4</td>
<td>year 4</td>
<td>Aho, Cormen, Elmasri</td>
</tr>
<tr>
<td>Year 5</td>
<td>year 5</td>
<td>Haykin, Soho</td>
</tr>
<tr>
<td>CS I</td>
<td>C. S. I Course</td>
<td>Horstmann, Kamin, Hubbard</td>
</tr>
<tr>
<td>Lab</td>
<td>C.S Lab. Course</td>
<td>Colin</td>
</tr>
<tr>
<td>CS II</td>
<td>C. S. II Course</td>
<td>Hubbard, Papadimitriou</td>
</tr>
<tr>
<td>O.S.</td>
<td>O.S. Course</td>
<td>Tanenbaum, Lauch, Linux Guide</td>
</tr>
<tr>
<td>Alg.</td>
<td>Algorithms</td>
<td>Aho, Cormen</td>
</tr>
<tr>
<td>D.B.</td>
<td>Data Bases</td>
<td>Elmasri</td>
</tr>
<tr>
<td>A.I.</td>
<td>AI Course</td>
<td>Haykin, Soho</td>
</tr>
<tr>
<td>CS I-Th</td>
<td>Theory I Course</td>
<td>Horstmann, Kamin</td>
</tr>
<tr>
<td>CS I-Th</td>
<td>Exercices I Course</td>
<td>Hubbard</td>
</tr>
<tr>
<td>CS II-Th</td>
<td>Theory II Course</td>
<td>Papadimitriou</td>
</tr>
<tr>
<td>CS II-Th</td>
<td>Exercices II Course</td>
<td>Hubbard</td>
</tr>
<tr>
<td>O.S.-Th</td>
<td>Theory for O. S.</td>
<td>Tanenbaum, Lauch</td>
</tr>
<tr>
<td>O.S.-Ex</td>
<td>Exercices for O. S.</td>
<td>Linux Guide</td>
</tr>
<tr>
<td>Horstmann</td>
<td>Java Book</td>
<td>Horstmann</td>
</tr>
<tr>
<td>Kamin</td>
<td>Java Book</td>
<td>Kamin</td>
</tr>
<tr>
<td>Hubbard</td>
<td>C++ Book</td>
<td>Hubbard</td>
</tr>
<tr>
<td>Colin</td>
<td>C++ Book</td>
<td>Colin</td>
</tr>
<tr>
<td>Papadimitriou</td>
<td>Complexity</td>
<td>Papadimitriou</td>
</tr>
<tr>
<td>Tanenbaum</td>
<td>O. S. Book</td>
<td>Tanenbaum</td>
</tr>
<tr>
<td>Lauch</td>
<td>O. S. Book</td>
<td>Lauch</td>
</tr>
<tr>
<td>Linux Guide</td>
<td>Linux Manual</td>
<td>Linux Guide</td>
</tr>
<tr>
<td>Aho and al.</td>
<td>Algorithms</td>
<td>Aho</td>
</tr>
<tr>
<td>Cormen</td>
<td>Algorithms</td>
<td>Cormen</td>
</tr>
<tr>
<td>Elmasri</td>
<td>Data Bases</td>
<td>Elmasri</td>
</tr>
<tr>
<td>Haykin</td>
<td>Neural Networks</td>
<td>Haykin</td>
</tr>
<tr>
<td>Soho</td>
<td>Artificial Vision</td>
<td>Soho</td>
</tr>
</tbody>
</table>

Table 1
The CSL Domain of Example 1.

3.2 The Local Knowledge Base

The ontology represents the common knowledge about the environment in which the agents work. However, each agent may have a partial view of the ontology representing the portion of the world the user monitored by the
agent selects by her/his activity. Inside this portion of the ontology, different priorities for the classes can be inferred by exploiting user behaviour. This is encoded in the notion of the Local Knowledge Base (LKB for short), defined next.

**Definition 3** Given an ontology $O_D$ on $D$ and an agent $a$, a **Local Knowledge Base** $LKB^a$ (of $a$ on $O_D$), is a pair $\langle G^a, \rho^a \rangle$, such that: (i) $G^a = \langle N^a, A^a \rangle$ is a sub-graph of $G(O_D)$ (i.e., $N^a \subseteq D$, $A^a \subseteq A$) containing the root $r$ of $G(O_D)$ and such that each $n \in N^a$ is reachable in $G^a$ from $r$, and (ii) $\rho$ is a function, called *priority function*, defining a real weight ranging from 0 to 1 associated to each arc $(i, j)$ of $G^a$ such that:

$$\rho(i, j) = \frac{c_{ij}}{\sum_{k \in Adj(i)} c_{ik}}$$

where $Adj(i)$ is the set of nodes adjacent to $i$, and for each $k \in Adj(i)$, $c_{ik}$ counts how many times the user of $a$ has selected an object (that is, a leaf node) through a path selection including the arc $(i, k)$. Note that coefficients $c_{ij}$ in a path $\langle r, i_1, \ldots, i_s \rangle$ are updated only when the leaf node $i_s$, corresponding to a single object of the domain, is selected. The root $r$ of $G(O_D)$ is also called the *root of $LKB^a$*.

A Local Knowledge Base, representing the local view of the agent, is then obtained by extracting a sub-graph from the ontology graph including all the classes accessed by the user (and thus at least the root node). Moreover, arcs of the so obtained graph are weighted for assigning highest priority to most
Example 3 In Figure 2, we show the LKBs of four agents $A_1, A_2, A_3, A_4$ on the domain CSL of Example 2. We have reported on each arc $(i, j)$ the value $c_{ij}$ and, in brackets, the value $\rho(i, j)$. We have also denoted, for layout reasons, as $B_1, B_2, ..., B_{13}$ the books contained in the CSL.

4 Extraction of the Semantic Properties

Besides his/her local agent, each user looks at the other agents of the net as a source of potentially interesting information in order to enrich the support to his/her activity. Interest in agents can be defined by considering some semantic properties. Such properties, useful for driving users’ choices are of two types: (i) local properties, taking into account information stored in the LKBs, and (ii) global properties, merging local properties with external knowledge extracted from the general context. An important feature of the model is that the merge performed in the construction of global properties is based on an adaptive learning technique involving some parameters by which the user behaviour is taken into account. In other words, global properties exploit an important kind of properties (encoded in a number of parameters) directly reflecting reactions of users to system advice. We call such additional properties reactive properties. Next we describe the set of properties used in the model.

4.1 Local properties: Similarity

The only local property we consider is the property we call similarity between two agents $i$ and $j$, representing a measure of the similarity of the two corresponding LKBs. Such a coefficient is a real value ranging from 0 to 1.

Definition 4 Let $i$ and $j$ be two agents. Let $G^i = \langle N^i, A^i \rangle$ and $G^j = \langle N^j, A^j \rangle$ be the two graphs of their LKBs. Let $\rho^i$ and $\rho^j$ be the corresponding priority functions. We define the similarity $S_{ij}$ between $i$ and $j$ as:

$$S_{ij} = 1 - \frac{1}{|A^i \cup A^j|} \sum_{(h,k) \in A^i \cup A^j} \gamma_{hk}$$

where

$$\gamma_{hk} = \begin{cases} |\rho^i(h,k) - \rho^j(h,k)| & \text{if } (h,k) \in A^i \cap A^j \\ 1 & \text{otherwise.} \end{cases}$$
Observe that the term $\frac{1}{|A_i \cup A_j|} \sum_{(h,k) \in A_i \cup A_j} \gamma_{hk}$ in the expression defining $S_{ij}$ (for two agents $i$ and $j$) represents a dissimilarity between agents $i$ and $j$. This is defined as a mean of a number of contributions $\gamma_{hk}$, each corresponding to an arc $(h, k)$ belonging to the set $A_i \cup A_j$. For common arcs of the two LKBs, that is, arcs belonging to the intersection between $A_i$ and $A_j$, $\gamma_{hk}$ is the difference (in absolute value) between the respective priority functions (note that such a difference is a real value ranging from 0 to 1). In words, common arcs can be viewed as “homologous” arcs, and their dissimilarity measures how much these arcs differ in terms of weight. For the remaining arcs $(h, k) \not\in A_i \cap A_j$, we assign the value 1 to the coefficient $\gamma_{hk}$. Indeed, an arc belonging to $A_i$ but not belonging to $A_j$ has not a “homologous” arc in the LKB graph of the agent $j$ (and vice versa), and thus this is the case of maximum dissimilarity, leading to a contribution (to the overall dissimilarity) saturated to the value 1.

4.2 Global Properties: Interest and Attractiveness

Recall that global properties merge local properties with knowledge extracted from the context. In this section we introduce the notion of interest coefficient, representing just a measure of the global properties of a given agent as perceived by another one. Hence, for a pair of agents $i$ and $j$, the interest coefficient, besides the similarity between $i$ and $j$, must take into account also knowledge extracted from the context. But which kind of contextual knowledge has to be considered as meaningful? The choice we make in our model is the following: The knowledge extracted from the context, used by the agent $i$ for defining the interest coefficient $I_{ij}$ w.r.t. another agent $j$, is a measure of the global interest of all the other agents (different from $i$) w.r.t. the agent $j$, that is a measure of a sort of attractiveness of the agent $j$ as perceived by the agent $i$. Recalling that the interest, besides the contextual knowledge, must take into account also the local knowledge (that is, the similarity), the above definition of contextual knowledge leads to require that, for each $i \in \Lambda \setminus \{j\}$:

$$I_{ij} = \phi_{ij}(S_{ij}, \mu_{ij}(\{I_{kj} \mid k \neq i, j\})) \quad (1)$$

where $\mu_{ij}$ and $\phi_{ij}$ are suitable functions yielding real values from 0 to 1. In particular, $\mu_{ij}$ returns a measure of the attractiveness of the agent $j$ detected by the agent $i$ from the value of the interest coefficients of all the agents (different from $i$) w.r.t. $j$, while $\phi_{ij}$ combines such a measure with the similarity $S_{ij}$. Clearly, the function $\phi_{ij}$ plays also the role of weighing the importance for the agent $i$ of the local knowledge w.r.t. the contextual one.

For $\mu_{ij}$ and $\phi_{ij}$ (where $i$ and $j$ are two agents) we adopt in our model the following choices: (i) $\mu_{ij}$ is a function returning the average of the interest coefficients of all the other agents different from $j$, (ii) $\phi_{ij}$ is a function com-
puting a linear combination of the similarity coefficient between \(i\) and \(j\) and the attractiveness of \(j\) w.r.t \(i\). Applying the above definitions for \(\mu_{ij}\) and \(\phi_{ij}\), (1) becomes the following linear system:

\[
(\forall i \in \Lambda \setminus \{j\}) \quad \left( I_{ij} = \psi_{ij} \cdot (P_i \cdot S_{ij} + (1 - P_i) \cdot S_{ij} \cdot \frac{1}{n-2} \sum_{k \in \Lambda \setminus \{i,j\}} I_{kj}) \right) \quad (2)
\]

where \(\psi_{ij}\) and \(P_i\), for each \(i \in \Lambda \setminus \{j\}\), are adaptive parameters ranging from 0 to 1 representing a measure of reactive properties that we suppose to be learned from the user behaviour. \(\psi_{ij}\) plays the role of a reducing factor, filtering the advice of the system on the basis of the user behaviour, while \(P_i\) measures the importance that the user gives to the local knowledge (similarity) w.r.t. the contextual one. Note that both \(\psi_{ij}\) and \(P_i\) can be estimated once the reactive properties are defined. We deal with this issue in the next section. Thus, given an agent \(j\), any value assignment to the interest coefficients of all the other agents w.r.t. \(j\) must satisfy (2). The next theorem shows that, for every value of the parameters occurring in (2), there exists a unique solution of the linear system (2), that is a value assignment to the interest coefficients satisfying (2). Obviously, such a solution can be polynomially computed.

**Theorem 1** Given an agent \(j \in \Lambda\) and a set of 3-tuples of \([0, 1]\) real coefficients \(\{(P_i, \psi_{ij}, S_{ij}) \mid i \in \Lambda \setminus \{j\} \land (\forall h \in \Lambda \setminus \{j\}, P_h \neq 0 \lor \psi_{ij} \neq 1 \lor S_{ij} \neq 1)\}\), there exists a unique \((n - 1)\)-tuple of \([0, 1]\) real values \(S = \langle I_1j, \ldots, I_{(j-1)}j, I_{(j+1)}j, \ldots, I_{nj} \rangle\) satisfying (2).

**Proof.** The theorem is trivially true in case \(0 \leq n \leq 2\). Thus, we have to prove the claim for \(n > 2\).

**Existence and Uniqueness.** We first prove that there exists unique \((n - 1)\)-tuple \(S\) satisfying (2). To this aim, it is sufficient to show that the rank \(r\) of the coefficient matrix \(H\) of (2) is full, i.e., \(r = n - 1^2\). W.l.o.g., just for notation convenience, we suppose \(1 < j < n\). The coefficient matrix \(H\) is:

\[
H = \begin{pmatrix}
-1 & \psi_{1j} \cdot S_{1j} \cdot \frac{1-P_1}{n-2} & \psi_{1j} \cdot S_{1j} \cdot \frac{1-P_1}{n-2} & \cdots & \psi_{1j} \cdot S_{1j} \cdot \frac{1-P_1}{n-2} \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
\psi_{nj} \cdot S_{nj} \cdot \frac{1-P_n}{n-2} & \psi_{nj} \cdot S_{nj} \cdot \frac{1-P_n}{n-2} & \cdots & -1
\end{pmatrix}
\]

We proceed by contradiction supposing that \(r < n - 1\). In such a case, there exists a row \(i\) of \(H\) that can be expressed as linear combination of the other rows by means of \(n - 2\) coefficients, say \(a_h, h = 1..n, h \neq i, j\). In particular,

\[2\] Recall that the size of \(H\) is \(n - 1\)
for the diagonal element $H(i, i) = -1$ the following holds:

$$H(i, i) = \sum_{h=1..n, h \neq i, j} \psi_{hj} \cdot S_{hj} \cdot \frac{1 - P_h}{n-2} \cdot a_h = -1.$$  \hfill (3)

For the other elements $H(i, t)$ of the row $i$, where $t = 1..n$, $t \neq i, j$, we obtain:

$$H(i, t) = \psi_{tj} \cdot S_{tj} \cdot \frac{1 - P_t}{n-2} = -a_t + \sum_{h=1..n, h \neq i, j, t} \psi_{hj} \cdot S_{hj} \cdot \frac{1 - P_h}{n-2} \cdot a_h \quad (4)$$

that is:

$$\psi_{tj} \cdot S_{tj} \cdot \frac{1 - P_t}{n-2} = -a_t \left( 1 + \psi_{tj} \cdot S_{tj} \cdot \frac{1 - P_t}{n-2} \right) + \sum_{h=1..n, h \neq i, j, t} \psi_{hj} \cdot S_{hj} \cdot \frac{1 - P_h}{n-2} \cdot a_h. \quad (5)$$

Exploiting (3), (5) becomes:

$$\psi_{tj} \cdot S_{tj} \cdot \frac{1 - P_t}{n-2} = -a_t \left( 1 + \psi_{tj} \cdot S_{tj} \cdot \frac{1 - P_t}{n-2} \right) - 1$$  \hfill (6)

from which we derive that $a_t = -1$. For symmetry, $a_h = -1$ for each $h = 1..n$, $h \neq i, j$. As a consequence, (3) becomes:

$$\sum_{h=1..n, h \neq i, j} \psi_{hj} \cdot S_{hj} \cdot \frac{1 - P_h}{n-2} = 1. \quad (7)$$

Hence, we have reached a contradiction, since by hypothesis for each $h$, either $P_h > 0$ or $\psi_{hj} < 1$ or $S_{hj} < 1$. Thus, (7) is false since $\sum_{h=1..n, h \neq i, j} \psi_{hj} \cdot S_{hj} \cdot \frac{1 - P_h}{n-2} < 1$.

We have thus proven that the system (2) admits a unique solution $S = \langle I_{1j}, \ldots, I_{(j-1)j}, I_{(j+1)j}, \ldots I_{nj} \rangle$. Now, it remains to prove that $I_{hj}$, for each $h \in \Lambda \setminus \{j\}$, belongs to the interval $[0, 1]$.

$I_{hj} \geq 0$, for each $h \in \Lambda \setminus \{j\}$. We start by proving that $I_{hj} \geq 0$, for each $h \in \Lambda \setminus \{j\}$. In particular, we show that the set $V_j = \{ h \in \Lambda \setminus \{j\} \mid I_{hj} < 0 \}$ is empty. We proceed by contradiction, supposing that $V_j \neq \emptyset$. Let $W_j = \{ h \in \Lambda \setminus \{j\} \mid I_{hj} \geq 0 \}$.

(2) can be rewritten as follows:

$$I_{ij} = \psi_{ij} \cdot S_{ij} \cdot \left( P_i + \frac{1 - P_i}{n-2} \left[ \sum_{r \in V_j} I_{rj} + \sum_{r \in W_j} I_{rj} - I_{ij} \right] \right).$$
Thus:
\[ I_{ij} \cdot \left(1 + \frac{\psi_{ij} \cdot S_{ij} \cdot (1 - P_i)}{n - 2}\right) = \psi_{ij} \cdot S_{ij} \cdot \left(P_i + \frac{1 - P_i}{n - 2} \cdot \left(\sum_{r \in V_j} I_{rj} + \sum_{r \in W_j} I_{rj}\right)\right). \]

Now, posing \( a_{ij} = 1 + \frac{\psi_{ij} \cdot S_{ij} \cdot (1 - P_i)}{n - 2} \) and applying the summation for each \( i \in V_j \), we obtain:
\[ \sum_{i \in V_j} I_{ij} = \sum_{i \in V_j} \frac{\psi_{ij} \cdot S_{ij}}{a_{ij}} \cdot \left(P_i + \frac{1 - P_i}{n - 2} \cdot \left(\sum_{r \in V_j} I_{rj} + \sum_{r \in W_j} I_{rj}\right)\right) \]
from which we derive:
\[ \sum_{r \in V_j} I_{rj} \cdot \left(1 - \sum_{i \in V_j} \frac{\psi_{ij} \cdot S_{ij} \cdot (1 - P_i)}{a_{ij} \cdot (n - 2)}\right) = \sum_{i \in V_j} \frac{\psi_{ij} \cdot S_{ij}}{a_{ij}} \cdot \left(P_i + \frac{1 - P_i}{n - 2} \cdot \sum_{r \in W_j} I_{rj}\right). \]

Posing \( b_j = \sum_{i \in V_j} \frac{\psi_{ij} \cdot S_{ij} \cdot (1 - P_i)}{a_{ij} \cdot (n - 2)} = \sum_{i \in V_j} \frac{1}{1 + \psi_{ij} \cdot s_{ij} \cdot (1 - r_{ij})} \), we obtain:
\[ \sum_{r \in V_j} I_{rj} = \frac{1}{1 - b_j} \cdot \sum_{i \in V_j} \frac{\psi_{ij} \cdot S_{ij}}{a_{ij}} \cdot \left(P_i + \frac{1 - P_i}{n - 2} \cdot \sum_{r \in W_j} I_{rj}\right). \]

Since \( V_j \) is not empty by hypothesis, \( \sum_{r \in V_j} I_{rj} < 0 \). As a consequence, \( (8) \) is false, since its right-hand term is greater than or equal to 0. Indeed, as it can be easily verified, \( b_j < 1 \). We have thus reached a contradiction and hence the set \( V_j \) must be empty. It remains to prove that \( I_{hj} \leq 1 \), for each \( h \in \Lambda \setminus \{j\} \).

**I_{hj} \leq 1, for each h \in \Lambda \setminus \{j\}**. We shall demonstrate that the set \( M_j = \{h \in \Lambda \setminus \{j\} \mid I_{hj} > 1\} \) is empty. We proceed by contradiction, supposing that \( M_j \neq \emptyset \). Let \( N_j = \{h \in \Lambda \setminus \{j\} \mid I_{hj} \leq 1\} \).

First observe that \( M_j \) cannot be a singleton. Indeed, if in the a tuple \( S = (I_{1j}, \ldots, I_{(j-1)j}, I_{(j+1)j}, \ldots, I_{nj}) \) just one element, say it \( I_{hk} \), is greater than 1, then \( (2) \) is not satisfied, as \( \psi_{hk} \cdot S_{hk} \cdot (P_h + (1 - P_h) \cdot \frac{1}{n - 2} \sum_{r \in \Lambda \setminus \{h,k\}} I_{rk}) \leq 1 \).

Thus, we have to consider only the case \( |M_j| \geq 2 \). Denote by \( s \) the cardinality of \( M_j \).

First, rewrite \( (2) \) as follows:
\[ I_{ij} = \psi_{ij} \cdot S_{ij} \cdot \left(P_i + \frac{1 - P_i}{n - 2} \cdot \left(\sum_{r \in M_j} I_{rj} + \sum_{r \in N_j} I_{rj} - I_{ij}\right)\right) \]
thus:

\[ I_{ij} \cdot \left( 1 + \psi_{ij} \cdot S_{ij} \cdot \frac{(1 - P_i)}{n - 2} \right) = \psi_{ij} \cdot S_{ij} \cdot \left( P_i + \frac{1 - P_i}{n - 2} \cdot \left( \sum_{r \in M_j} I_{rj} + \sum_{r \in N_j} I_{rj} \right) \right) \]

Now, posing \( a'_{ij} = 1 + \psi_{ij} \cdot S_{ij} \cdot \frac{(1 - P_i)}{n - 2} \) and applying the summation for all \( i \in M_j \), we obtain:

\[
\sum_{i \in M_j} I_{ij} = \sum_{i \in M_j} \frac{\psi_{ij} \cdot S_{ij}}{a'_{ij}} \cdot \left( P_i + \frac{1 - P_i}{n - 2} \cdot \left( \sum_{r \in M_j} I_{rj} + \sum_{r \in N_j} I_{rj} \right) \right)
\]

from which we derive:

\[
\sum_{i \in M_j} I_{ij} \cdot \left( 1 - \sum_{i \in M_j} \frac{\psi_{ij} \cdot S_{ij} \cdot (1 - P_i)}{a'_{ij} \cdot (n - 2)} \right) = \sum_{i \in M_j} \frac{\psi_{ij} \cdot S_{ij} \cdot (1 - P_i)}{a'_{ij}} \cdot \left( P_i + \frac{1 - P_i}{n - 2} \cdot \sum_{r \in N_j} I_{rj} \right)
\]

Posing \( b'_j = \sum_{i \in M_j} \frac{\psi_{ij} \cdot S_{ij} \cdot (1 - P_i)}{a'_{ij} \cdot (n - 2)} = \sum_{i \in M_j} \frac{1}{1 + \frac{1 - P_i}{n - 2} \cdot \psi_{ij} \cdot S_{ij} \cdot (1 - P_i)} \), we have:

\[
\sum_{r \in M_j} I_{rj} = \frac{1}{1 - b'_j} \cdot \sum_{i \in M_j} \frac{\psi_{ij} \cdot S_{ij}}{a'_{ij}} \cdot \left( P_i + \frac{1 - P_i}{n - 2} \cdot \sum_{r \in N_j} I_{rj} \right)
\]

The right-hand term of (9) is upper-bounded by

\[
\sum_{i \in M_j} \frac{1}{1 + \frac{1 - P_i}{n - 2}} \cdot \left( P_i + \frac{1 - P_i}{n - 2} \cdot \sum_{r \in N_j} I_{rj} \right)
\]

In turn, the latter is upper-bounded by:

\[
\sum_{i \in M_j} \frac{1}{1 + \frac{1 - P_i}{n - 2}}
\]

since \( \sum_{r \in N_j} I_{rj} \leq n - s \leq n - 2 \).

But, \( \frac{1}{1 + \frac{1 - P_i}{n - 2}} < 1 \), for each \( i \in M_j \) and, thus, the right-hand term of (9) is less than \( s \). As a consequence, (9) is false, since \( \sum_{r \in M_j} I_{rj} > s \) and thus we have reached a contradiction. This concludes the proof. \( \square \)

The above result allows us to define the interest coefficients list of an agent \( j \) as the unique solution of (2).
**Definition 5** Given an agent \( j \in \Lambda \), the *interest coefficient list of \( j \)* is the unique \((n-1)\)-tuple of real values \( \langle I_{1j}, \ldots, I_{(j-1)j}, I_{(j+1)j}, \ldots I_{nj} \rangle \) satisfying (2). Given an agent \( i \neq j \), the *interest coefficient of \( i \) w.r.t. \( j \)* is the value \( I_{ij} \) occurring in the interest coefficient list of \( j \).

Besides the interest property, from the knowledge of the interest coefficients lists, agents can exploit a second type of property. Indeed, an agent can compare different agents on the basis of their *attractiveness coefficient*, representing the component of the interest capturing only the contextual knowledge.

**Definition 6** Given a pair of agents \( i, j \in \Lambda \), the *attractiveness of \( j \) perceived by \( i \)* is the real coefficient \( A_{ij} \) (ranging from 0 to 1) defined as: \( A_{ij} = \frac{1}{n-2} \sum_{k \in \Lambda \backslash \{i,j\}} I_{kj} \), where \( \langle I_{1j}, \ldots, I_{(j-1)j}, I_{(j+1)j}, \ldots I_{nj} \rangle \) is the interest coefficients list of the agent \( j \).

### 4.3 Choice Lists

Suppose the user of an agent \( i \) has the intention of contacting other agents in order to establish a cooperation. Suppose the similarities between \( i \) and every other agent is known as well as both the interest coefficient of \( i \) w.r.t. every other agent and the attractiveness of all the agents perceived by \( i \). As previously discussed, such values can be effectively computed once a number of parameter are set (actually, they can be suitably initialized and updated by learning from the behaviour of the user, as we shall explain in the sequel). Thus, three agent lists can be presented to the user \( i \) associated to the agent \( i \), each associated with one of the properties similarity, interest and attractiveness. We denote these lists \( L_S(i) \), \( L_I(i) \), and \( L_A(i) \). \( L_S(i) \) (\( L_I(i) \), \( L_A(i) \), resp.) is the list of the \( n-1 \) agents \( j \) (different from \( i \)) ordered by decreasing similarity (interest, attractiveness, resp.) coefficient \( S_{ij} \) (\( I_{ij} \), \( A_{ij} \), resp.). When the user \( i \) chooses an agent \( j \) from the list \( L_S(i) \) (\( L_I(i) \), \( L_A(i) \), resp.), it means that she/he perceived only the property of similarity (interest, attractiveness, resp.) about the agent \( j \). From the choices of the users, useful knowledge can be thus drawn, which is potentially usable as feedback for correcting advice given to them. This issue is discussed in the next section.

### 4.4 Reactive Properties

For reactive properties we mean properties describing reactions of users to the suggestions received from the system at a given time, that must be taken into account for adapting future responses of the system. We implement such adaptation of the system to the user behaviour by including into the interest coefficient definition (see Section 4.2) some specific coefficients that are au-
atomatically updated during system running. In this section we describe both the role of such coefficients and the rules defining their adaptation to user behaviour. Recall that, given a pair of agents $i$ and $j$, for defining the interest coefficient $I_{ij}$, two parameters $P_i$ and $\psi_{ij}$ must be set. They are real parameters ranging from 0 to 1. $P_i$ encodes the preference property and is called preference coefficient of the agent $i$, while $\psi_{ij}$ is split into the product of two others parameters $B_{ij} \cdot C_{ij}$ called benevolence coefficient and consent coefficient, of $i$ w.r.t. $j$, resp., in order to isolate two different reactive properties, which we describe in the following. Recall that, given an agent $i$, by $L_S(i)$, $L_I(i)$, and $L_A(i)$, we denote the three choice lists presented to the user of agent $i$ by the system.

The Preference Property. It is described by a real coefficient ranging from 0 to 1, denoted by $P_i$, and called preference coefficient. The property measures how much for an agent $i$ the similarity is more important than the attractiveness property for defining global properties. It is easily recognizable that in the definition of interest given in Section 4.2 the coefficient $P_i$ plays just this role. Now we define how the coefficient $P_i$ is updated. Suppose that at a given time the user of the agent $i$ makes a selection of agents. Denote by $SI_i$ ($SS_i$, $SA_i$, resp.) the set of the agents that the user has selected from the list $L_I(i)$ ($L_S(i)$, $L_A(i)$, resp.). Such choices are interpreted in order to define how to update the coefficient $P_i$. The reasoning we adopt is the following: the ratio between the number of agents selected according to the similarity property and the total number of selected agents, provides us a perception of the importance the user gives to the similarity versus the attractiveness. Thus, such a ratio could be used for evaluating the new value of $P_i$. How to infer the number of agents selected for their similarity? Certainly all the agents of $SS_i$ are chosen for their similarity. On the contrary, it is reasonably to assume that agents of $SA_i$ do not give any contribution to the above number, since they has been chosen only on the basis of the attractiveness property. What about agents in $SI_i$? Here, the choice was done on the basis of the interest property, which mixes similarity and attractiveness. But we can use the old value of $P_i$ for inferring which portion of $SI_i$ has been chosen for the similarity. And this is coherent with the semantic we have given to the preference coefficient. Thus, the total number of agents chosen on the basis of similarity can be assumed to be $P_i \cdot |SI_i| + |SS_i|$. Taking into account the above observations, updating $P_i$ after a selection step is defined as:

$$P_i = \frac{1}{2} \cdot \left( \frac{|SI_i| \cdot P_i + |SS_i|}{|SI_i| + |SS_i| + |SA_i|} + P_i \right).$$

where $|SI_i| + |SS_i| + |SA_i|$ is the total number of selected agents. This updating is obtained by computing the average between the new contribution with the old value of $P_i$ in order to keep memory of the past and avoiding
The Benevolence Property. This property measures a sort of availability of the agent \( j \) to which a user \( i \) requires to share knowledge. Such a property is used in order to weight the interest of \( i \) w.r.t. \( j \). For instance, an agent \( j \) that recently, and for several times, has denied collaboration in favor of \( i \) should become of little interest for \( i \). The parameter encoding such a knowledge is called benevolence coefficient, denoted by \( B_{ij} \), and takes real values ranging from 0 to 1. \( B_{ij} = 0 \) (resp., \( B_{ij} = 1 \)) means the agent \( j \) is completely unavailable (resp., available) to fulfill the requests of \( i \). The response of \( j \) to requests of \( i \) updates the value of \( B_{ij} \) according to the following rules:

\[
B_{ij} = \begin{cases} 
\min(1, B_{ij} + \delta) & \text{if } j \text{ grants the request of } i \\
\max(0, B_{ij} - \delta) & \text{if } j \text{ denies the request of } i 
\end{cases}
\]

where \( \delta \) is a (reasonably small) positive real value.

The Consent Property. This property describes how much the user of an agent \( i \) trusts suggestions of the system regarding another agent \( j \) done on the basis of the interest property. The coefficient associated with this property is denoted by \( C_{ij} \) and is called consent coefficient. The updating rules defining how to adapt the coefficients \( C_{ij} \) after a user selection step take into account only the portion of the selection performed on the list \( L_I(i) \). Indeed, from this portion of the user selection, we can draw information about the opinion of the user regarding the suggestions provided by the system. For instance, if the user of \( i \) completely trusts the system capability of providing the best suited agent for cooperation by providing the list \( L_I(i) \) she/he will choose exactly only the first \( k \) agents appearing in \( L_I(i) \), where \( k \) is the size of the portion of her/his selection extracted from \( L_I(i) \). This is not in general the case, that is, some of the \( k \) agents chosen from \( L_I(i) \) do not occur in the set of the first \( k \) agents of \( L_I(i) \). We defined updating rules by taking into account the above observations according to the following idea: every agent \( h \) chosen by the user from \( L_I(i) \) produces a gain of the consent coefficient \( C_{ih} \) if \( h \) is a candidate from the system to be selected, produces an attenuation of \( C_{ih} \) otherwise. More formally, given an agent \( i \) and a selection \( S_i \) (set of agents) extracted by the user of \( i \) from \( L_I(i) \), for each \( h \in S_i \):

\[
C_{ih} = \begin{cases} 
\min(1, C_{ih} + \delta) & \text{if } h \text{ appears among the first } |S_i| \text{ elements of } L_I(i) \\
\max(0, C_{ih} - \delta) & \text{otherwise} 
\end{cases}
\]

where \( \delta \) is a (reasonably small) positive real value.
Remark. As a final remark we point out that having three different lists in place of a unique synthetic one, allows us to exploit user choices as a feedback in a selective way (w.r.t. the semantic properties), paying of course a little cost in terms of system user friendliness.

5 Integration of Interesting Knowledge Bases

Cooperation between two agents is implemented in our model by the integration of their LKBs. Thus, the user of an agent \( i \) which has selected an agent \( j \) from one of the three choice lists can exploit the cooperation of \( j \) by consulting the Integrated Knowledge Base \( LKB^{ij} \), obtained by integrating \( LKB^i \) with \( LKB^j \). We show next how \( LKB^{ij} \) is defined. Once the \( LKB^{ij} \) has been computed, the integration of the knowledge of the agent \( j \) with that of the client agent \( i \) is simply implemented by replacing its LKB with the new \( LKB^{ij} \).

Definition 7 Let \( i \) be an agent. Let \( L \in \{ L_S(i), L_I(i), L_A(i) \} \) be one of the choice lists of \( i \) and \( j \) be an agent selected by \( i \) from the list \( L \). Let \( G^i = \langle N^i, A^i \rangle \) and \( G^j = \langle N^j, A^j \rangle \) be the two graphs of their LKBs.

The Integrated Knowledge Base of \( j \) in \( i \), denoted by \( IKB^{ij} \), is the pair \( \langle G^{ij}, \rho^{ij} \rangle \), where \( G^{ij} = \langle N^i \cup N^j, A^i \cup A^j \rangle \) and \( \rho^{ij}(h, k) \) (i.e., the priority function) is computed on the basis of the coefficients \( c_{hk} \) of the arcs outgoing \( h \) in \( G^{ij} \) as defined by Definition 3. Such coefficients, according to the semantics we have given to the priority function, are defined as follows:

\[
c_{hk} = \begin{cases} 
c^i_{hk} & \text{if } (h, k) \in A^i \setminus A^j \\
c^j_{hk} & \text{if } (h, k) \in A^j \setminus A^i \\
c^i_{hk} + c^j_{hk} & \text{if } (h, k) \in A^i \cap A^j
\end{cases}
\]

where we denote by a superscript the source ontology the coefficients refer to.

In words, the coefficient of an arc in the integrated LKB is obtained by copying the corresponding coefficient from the source LKB, say it \( i \), in case the arc belongs only to \( i \), by summing up the corresponding coefficients, in case the arc appears in both LKBs.

The integration process can be thus completed. Finally, once the \( LKB^{ij} \) has been computed, we update the LKB of \( i \) to be \( LKB^i = LKB^{ij} \).

Remark. Through integration an agent \( i \) enriches its knowledge by the knowledge embedded into the LKB of another agent \( j \). Clearly this may represent
a initial yet crucial step for implementing a cooperation between \(i\) and \(j\). Indeed, the aim of the cooperation request submitted by \(i\) to \(j\) is finding in the LKB of \(j\) useful knowledge not currently occurring in its LKB. Of course, the success probability of such an attempt is strongly related with the semantic “closeness” of the two LKBs, but also with the capability of \(j\) of satisfying cooperation requests, witnessed by its attractiveness. Our approach, based on the notion of interest, takes into account both the above issues. In this sense, selecting agents for cooperation by using suggestions given by the three choice lists (see the previous section), means trying to selecting, among all possible cooperation actions, the most promising ones. It is worth noting that, pruning the search space in a possible large agent network (consider for example peer to peer systems), is a hard but important problem whose solution may make more efficient, sometimes feasible, a large class of approaches based on knowledge sharing. Indeed, it is not in general admissible that cooperation requests are submitted in a broadcast fashion, both for efficiency reasons and for avoiding band consumption and, finally, for limiting noise carried by the answers (in our case noise might correspond to include undesired concepts in the LKB of the requesting agent).

The next section, through experiments, provides a validation of our framework, showing, among other issues, the role played by integration in the agent cooperation, as discussed in the above remark.

6 Experiments and validation

In the previous sections, we have presented a formal framework for representing cooperation among agents in a multi-agent environment. The model is based on the extraction of some semantic properties capturing both local and contextual knowledge about agents. Such properties, encoded by suitable coefficients, drive users towards selecting from the agent net the most promising candidate agents for fruitful cooperation. User choices are exploited as feedback for adapting coefficients in such a way that a trade-off among similarity and attractiveness, on the one hand, agent congestion and user dissatisfaction, on the other hand, is obtained. To better understand the above claim about features of the model, we next show some examples considering, for the sake of presentation, just small sets of agents. System evolution is traced by reporting the values of both reactive and global coefficients (i.e., \(B_{ij}\), \(C_{ij}\), \(P_i\), and \(S_{ij}\), \(A_{ij}\), \(I_{ij}\), respectively). Through a final example, we discuss how integration may implement cooperation between two agents.

Example 4 Consider four agents whose ontologies are depicted in Figure 2. In this example, we analyze how the system evolves, from a given (admissible) state, because of user actions. In particular, we consider some possible
<table>
<thead>
<tr>
<th>step = 0</th>
<th>j</th>
</tr>
</thead>
<tbody>
<tr>
<td>i = 1</td>
<td>1</td>
</tr>
<tr>
<td>$C_{ij}$</td>
<td>–</td>
</tr>
<tr>
<td>$B_{ij}$</td>
<td>–</td>
</tr>
<tr>
<td>$P_i$</td>
<td>0.50000</td>
</tr>
<tr>
<td>$S_{ij}$</td>
<td>–</td>
</tr>
<tr>
<td>$A_{ij}$</td>
<td>–</td>
</tr>
<tr>
<td>$I_{ij}$</td>
<td>–</td>
</tr>
</tbody>
</table>

Table 2
Example 4: initial state.

<table>
<thead>
<tr>
<th>step = 1</th>
<th>j</th>
</tr>
</thead>
<tbody>
<tr>
<td>i = 1</td>
<td>1</td>
</tr>
<tr>
<td>$C_{ij}$</td>
<td>–</td>
</tr>
<tr>
<td>$B_{ij}$</td>
<td>–</td>
</tr>
<tr>
<td>$P_i$</td>
<td>0.50000</td>
</tr>
<tr>
<td>$S_{ij}$</td>
<td>–</td>
</tr>
<tr>
<td>$A_{ij}$</td>
<td>–</td>
</tr>
<tr>
<td>$I_{ij}$</td>
<td>–</td>
</tr>
</tbody>
</table>

Table 3
Example 4: final state (i)

(reasonable) actions of the user supported by agent 1.

Table 2 reports values corresponding to an admissible (intermediate) system state. By analyzing such values, we may observe that the agent 1 perceives the agent 3 as having high similarity (i.e., 0.41237) and low attractiveness (i.e., 0.12657), and, further, the agent 2 as having a lower similarity (i.e., 0.37563), but higher attractiveness (i.e., 0.28465). In this case, despite the high similarity between agents 1 and 3, the interest of 1 w.r.t. 3 (i.e., 0.26411), is about 20% higher than the interest of 1 w.r.t. 2, and this is due to high attractiveness of 2 w.r.t. 1.

We analyze the following situations:

- (i) The user of agent 1 decides to contact the agent 2, on the basis of the interest coefficient. Thus we model the case the user currently does not fully rely on the similarity property, whereas considers more relevant the capability of other (sufficiently similar) agents of “attracting” the attention
of the community (this is just what is embedded in the concept of interest). We show in Table 3 the state reached after such a transition. Observe that, after integration of LKBs of agents 1 and 2, the new LKB of agent 1 (see Figure 3) becomes more similar to that of agent 2, and thus the similarity between 1 and 2 increases (as can be verified in Table 3). In turn, the similarity between 1 and 3 decreases, and this is coherent with dissatisfaction of the user of agent 1 about the previous value of similarity. Thus, the behaviour of the user has produced a feedback determining an alignment of the system to her/his expectancy.

• (ii) As a second possibility consider the case the user of agent 1 decides to contact 3 (more similar but less interesting than 2), since she/he guesses that high interest (and thus attractiveness) probably means long waiting time for obtaining answers. The resulting LKB for the agent 1 is reported in Figure 4, and Table 4 summarizes the values describing the state reached after this transition. Therein, it can be verified that the value of the interest of 1 w.r.t. 2 decreases (of about 20%), and this, once again, is what can be expected according to the user behavior.

Note also that the preference coefficient of the agent 1 is increased, due to the fact that, with its choice, it has decided to give more importance to the preference property w.r.t. the attractiveness.

<table>
<thead>
<tr>
<th>step = 1</th>
<th>j</th>
</tr>
</thead>
<tbody>
<tr>
<td>i = 1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>j</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.75000</td>
<td>0.85000</td>
<td>0.80000</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>3</td>
<td>0.75000</td>
<td>0.30907</td>
<td>0.64325</td>
<td>0.13992</td>
</tr>
<tr>
<td>4</td>
<td>0.27114</td>
<td>0.15988</td>
<td>0.15699</td>
<td>0.11877</td>
</tr>
</tbody>
</table>

Table 4
Example 4: final state (ii)

Let us show now another example.

**Example 5** Consider the set of four agents reported in Figure 5. Note that agents 3 and 4 are the same as Example 4, while agents 1 and 2 are new.

We analyze the following situations:

• (i) We suppose the user performing actions is associated to agent 4 and starting state is that reported in Table 5. Note that agent 4 perceives agent 3 as having high interest (i.e., 0.35085) and no similarity (i.e., the similarity coefficient is null). Moreover, agent 1 is perceived as more similar (i.e.,
Table 5
Example 5: initial state (i).

0.15720) even if less interesting (i.e., 0.32323) than agent 3. We suppose the user of agent 4 is not satisfied by system suggestions based on interest property and thus decides to contact agent 1. The LKB of agent 4 after integration with agent 1 is reported in Figure 6 while the new state is described in Table 6.

The interesting observation arising from the analysis of Table 6 is that the consent coefficient of the agent 4 w.r.t. 3 decreases (coherently with the semantics we have given to such a coefficient) and this gives contribution to the decreasing of the interest of the agent 4 w.r.t. 3.

(ii) Now we suppose the user performing actions is associated to the agent 3 whose starting state is reported in Table 7. The user sees the highest interest coefficient is associated with the agent 1, that is also the most similar. However, the user notes that the highest attractiveness is associated with the agent 4, that is less similar than the agent 1. The user decides to give more importance to the attractiveness, in this case, since it would like to follow the suggestions of the other agents (that are synthetically represented by the attractiveness coefficient) and thus it chooses to contact the agent 4. The resulting LKB for the agent 3 is reported in Figure 7. We note (see Table 8) that the preference coefficient of the agent 1 is decreased w.r.t. the initial state and that the attractiveness of the agent 4 is increased.

As a final example, we consider the integration between two LKBs for showing how integration may support fruitful cooperation.

Example 6 Consider the case of the two agents 1 and 2 of Figure 2 and assume the state of 1 is that reported in Table 2. The user of the agent 1 is interested in books concerning Operating Systems (O.S.), as described in its ontology, and it knows only theoretical books, namely $B_6$ and $B_7$. Agent 2 appears the most interesting from the agent 1’s viewpoint (the interest of 1 for 2 is 0.26411, according to Table 2), and it is also similar enough to 1; thus, the user of 1 decides to merge its knowledge base with that of 2. The
merge, as shown in Figure 3, has the effect of enriching the 1’s ontology with knowledge concerning a book of S.O. exercises, namely $B_8$, that may be useful for the 1’s user. Moreover, thanks to the merging two more book categories,
namely fourth year’s books and fifth’s year books, has been discovered by 1. Reasonably, since these book categories are considered useful by the user of 2 whose interests are similar interests to those of the 1’s user, this latter may consider them interesting too. Note that this positive effect has been produced since the agent chosen for the merging has been suitably selected among the available agents, according to a semantic property (in this case, the interest).

7 Conclusions and Computational Complexity Issues

In this paper a framework for representing and managing cooperation among agents in a Multi-Agent community is provided. The core of the proposal is the definition of a formal model based on several semantic properties and on a linear system involving some coefficients associated to such properties. The solution of such a system allows the user to find the best agents for cooperation in the net, that is those agents from which the most fruitful cooperation can be reasonably expected. Cooperation between two agents is implemented by the integration of the respective knowledge bases.

The aim of the paper is to study some semantic aspects related to the general problem of collaboration among agents and modeling them through a quantitative approach. Moreover, we show how the obtained model may be used for defining an effective strategy for supporting cooperation. Under this perspective, implementation issues do not play a central role in the paper, and, as a consequence, we sketch only here a brief description of a system prototype.

It is based on a client-server architecture. Cooperation of “client” agents is coordinated and managed from the server side. For understanding how the system work, consider now the case the system is used for helping the users in retrieving information. The user is provided with a set of recommendations that are generated both by her/his agent and by agents the system has detected as promising for cooperation.

The server maintains a consistent copy of the LKB of each agent in the network. The agent provides its user with the three ordered lists described in Section 4.3 (that is, similarity, attractiveness and interest lists). Recall that such lists contain candidate agents for cooperation. The user selects from one or more lists those agents chosen for helping her/him in her/his activity. List selection is done according to the importance the user assigns to similarity, attractiveness and interest, respectively.

After having performed the choice, each selected agent is contacted and receives from the user a set of keywords, combined, as usually, by means of logical connectives. The answer to the request above is provided by the contacted
agents. Furthermore, the user may require the integration between her/his local knowledge base and the set of local knowledge bases of the contacted agents, in order to enrich its local knowledge with an external knowledge she/he considers useful.

The request generated by the Client Agent is handled by its Knowledge Base Management System (KBMS). KBMS manages the whole agent knowledge base (KBS) and communicates with the Agency for both retrieving information about the other agents of the community and updating the copy of its LKB stored in the Agency. The agent knowledge base is composed of the following three components, corresponding to the definitions given in the Section 3:

- the local copy of the Ontology Domain OD, shared by all the agents;
- the Local Knowledge Base LKB;
- the Network Knowledge Base, which stores preference, similarity, attractiveness, interest, consent and benevolence coefficients. Preference and consent are automatically updated by the agent each time the user carries out a new choice from the choice lists. The other properties are computed by the Agency server, on the basis of the copies of the LKBs and by following the formulation described in Section 4.

We give now a final discussion about the feasibility of the approach by analyzing its computation complexity. Of course, from the computational point of view, the only relevant task to analyze is the solution of a linear system like (2). This is a classic well studied problem, and many algorithms have been proposed for making feasible its solution also for large system dimensions. These results are of high practical interest, since large systems of linear equations occur in many applications such as finite element analysis, power system analysis, circuit simulation for VLSI CAD, and so on. In the general case, the cost for finding an exact solution is \( O(n^\omega) \), coinciding with the cost of executing a \( n \times n \)-matrix product. The currently best known \( \omega \) is 2.376... [9], while a practical bound is 2.81 [?]. We observe that the coefficient matrix of our linear system is sparse in many practical situations. This is the case of agent communities characterized by clusters of similar agents (in terms of ontology closeness) weakly connected each other. This leads to have many \( S_{ij} \) coefficients equal to null and, thus, a sparse coefficient matrix. Note that for sparse linear systems, more efficient solutions may be found. In [26] it is shown that the cost is \( O(n + s(n)^{\omega}) \), where \( s(n) \) is a function measuring the sparsity of the system’s coefficient matrix (note that \( s(n) = O(n) \), in the general case). Beside exact (direct) methods considered above, there exists a wide variety of iterative methods, whose effectiveness strongly depends on the matrix sparsity. In [26] an evaluation of the upper bound of the number of iterations needed for the convergence of an \( \epsilon \)-solution (for relative error \( \epsilon \)) is provided. Each iteration has in general the cost \( O(n^\omega) \) since it requires a
matrix product. Multigrid methods can be applied [?] in order to decrease the cost per iteration until $O(n^\omega)$, even though such methods are not applicable in general and may suffer of instability problems).

References


25


Fig. 2. The set of agent ontologies of Example 4
Fig. 3. The updated LKB of the agent 1 in case (i) of Example 4

Fig. 4. The updated LKB of the agent 1 in case (ii) of Example 4
Fig. 5. The set of agent ontologies of Example 5
Fig. 6. The updated LKB of agent 4 of Example 5 (i)

Fig. 7. The updated LKB of agent 3 of Example 5 (ii)