Dynamics in the Normative Group Recognition Process

Daniel Villatoro and Jordi Sabater-Mir
Artificial Intelligence Research Institute (IIIA)
Spanish Scientific Research Council (CSIC)
Bellatera, Barcelona, Spain
email: {dvillatoro,jsabater}@iiia.csic.es

***Abstract*— This paper examines the decentralized recognition of groups within a multiagent normative society in dynamic environments. In our case, a social group is defined based on the set of social norms used by its members. These social norms regulate interactions under certain situations, and situations are determined by the environmental conditions. Environmental conditions might change unexpectedly, and so should the notion of social group for each agent. Consequently, agents need mechanisms to adjust their notion of group dynamically and accordingly the agents with whom it is socially related. In this work we analyze how different algorithms (*whitelisting*, *blacklisting*, *labelling*), that allow agents to recognize the others as members of a certain social group, behave in these dynamic environments. Simulation results are shown, confirming that the limited memory approach reacts better against environmental changes. Moreover we compare two approaches that regulate the adaptation of the relevance of norms and the notion of group: the unlimited normative memory and the limited memory.**

I. Introduction and Related Work

Social norms are part of our everyday life, and they have been of interest in several areas of research [1]. They help people self-organizing in many situations where having an authority representative is not feasible. On the contrary to institutional rules, the responsibility to enforce social norms is not the task of a central authority but a task of each member of the society. From the book of Bicchieri [2], the following definition of social norms is extracted: "The social norms I am talking about are not the formal, prescriptive or proscriptive rules designed, imposed, and enforced by an exogenous authority through the administration of selective incentives. I rather discuss informal norms that emerge through the decentralized interaction of agents within a collective and are not imposed or designed by an authority". Social norms are used in human societies as a mechanism to improve the behavior of the individuals in those societies without relying on a centralized and omnipresent authority. In recent years, the use of these kinds of norms has been considered also as a mechanism to regulate virtual societies and specifically societies formed by artificial agents ([3], [4], [5]).

The main objective of this research is to analyze the process of group recognition around a common set of social norms and which algorithms and mechanisms make this process more efficient in dynamic environments. Several researchers have already covered the problem of group recognition. One seminal article is the work of Hales presenting the SLACER algorithm [6]. In this algorithm given a network of agents, when an agent finds another agent in a better situation than itself, it copies its strategy and neighbours. This rewiring algorithm produces an emergent behavior similar to the one we try to obtain by forming groups. Another interesting work is the one presented in [7], where the authors try to answer the question "to whom should agents connect to?" by experimenting on different agents' networks structures, to solve optimization problems. Finally, and also a technique used by Hales [8], is the "tagging mechanism", initially presented by Holland [9]. Tags are markings or social cues attached to individuals (agents) and observable by others. Agents maintain and modify tags on themselves and a team is formed by only collaborating with agents with the same tag.

One possible application of this work is oriented towards virtual societies where the social component is crucial for their subsistence. The work presented here is the continuation of another one [10], where the algorithms were presented and their behaviour deeply analyzed on static environments. However, most of the examples of usage of multi-agent systems are done in dynamic environments, therefore the analysis of the algorithms in dynamic environments is necessary.

II. Statement of the Problem

Social norms provide multiagent societies with a decentralized control mechanism. By donating agents with a set of social norms to use in the environment where they are located, they can self-regulate it and control undesired behaviors. But the definition of social norm that we have used in previous sections does not expressly include one vital aspect of the social norm, that is, the *coordinated reaction against outsiders*. The coordinated reaction from the group is twofold: (1) a *coordinated punishment* from the group against the outsider who explicitly does not abide by the same social norms; and (2) an *integration mechanism* to force this outsider to change its actual behavior to the one accepted by the group.

Therefore, agents need to have a notion of social group. The notion of belonging to a social group will be determined by the agent's behaviour. The behaviour of an agent is determined at the same time by the social norms it uses and the environmental conditions (which determine the situations in which an agent behaves and uses its norms). In this way,

a social group will only be formed by agents that under a determined environmental condition share the most relevant norms.

Once agents have built their notion of social group they might be able to coordinate according to the situation in which they meet outsiders. However, environmental conditions might be dynamic. Environmental changes might vary the situations that an agent will interact in, and therefore, the relevance of the norms used. A change in the relevance of the norms might produce agents to change their notion of social group, and consequently, two agents that were previously socially related might not be socially related anymore with the new environmental conditions.

Agents need to be provided with mechanisms to self-organize against changes in the environment and be able to adapt dynamically their notion of group. To achieve such a task we will compare two different memory approaches: (1) an unlimited memory (where all the uses of each of the norms are saved), and (2) a limited memory (where only a limited number of uses of the norms will be saved). In this work our contribution is twofold:

1) We will *compare different algorithms for the group recognition process and contrast their efficiencies in dynamic environments*. These algorithms will affect the speed of convergence of the group recognition process.
2) We will *contrast the effect of limited and unlimited memory* under different environmental changes: permanent and sporadic.

III. Simulation Model

In order to prove the algorithms and both memory approaches, we have developed a multi-agent based simulation. In this simulation agents are distributed and initially do not know each other. Agents survive by consuming resources and these resources are obtained in two ways: assigned by the simulation following a uniform distribution, or, receiving them from other agents. When one agent decides to donate some of its resources it means that this agent is losing some of its own resources to give them to another agent. The set of social norms tell the agents under which situations (depending on both agents' states) they will donate energy to another agents and in which they will not. The task of the algorithms presented here is to detect the norms that each agent follows, determining in this way to which social group each agent belongs to.

The simulation algorithm is based on a discrete step timing model, where in each time step the algorithm observes the state and consequent actions of each agent before ticking another time step. Every time step, the simulation algorithm runs over every agent. The order in which the algorithm runs over the agents is randomly changed each time step.

Initially, in each time step, the algorithm evaluates (following a continuous uniform probability distibution) if every agent has to find resources by observing the agent *Resource Gathering Probability* ($P_{rg} \in [0,1]$). P_{rg} specifies the probability an agent has to find resources each time step. In case the algorithm evaluates that an agent has to find resources, the agent will receive a large amount of resources that can either be used for its own consumption or for donating.

After that, the algorithm evaluates if an agent has to meet another agent by observing the agent *Interaction Probability* ($P_{int} \in [0,1]$). P_{int} specifies the probability of an agent to meet another agent present in the simulation. The other partner is chosen following a uniform distribution from all the agents present in the simulation. As we will see, this process of mate selection will be slightly modified later in the presented algorithms.

Agents are initially loaded in the simulation platform with 100 resource units, and each time step, agents consume one resource unit as energy consumption. When an agent's resources are exhausted, the agent is not able to interact with any other agent and no agent can interact with it, remaining inactive until new resources are assigned again by the simulation algorithm according to a uniform distribution and the agent's *Resource Gathering Probability*.

The interactions among agents are done always in pairs, and both agents have to choose an action when interacting. This action is selected by following the *set of social norms* that each agent has internalized. The set of norms specifies if the agent has to give or not to give resources to the other agent, depending on both agent's internal resource levels. In order to formalize our concept of social norm, we first need to define several terms.

All agents can perceive a finite set of *observables* $ob \in \mathcal{O}$ and perform a finite set of *actions* $a \in A$.

Every agent can find itself in a finite set of different *situations* $sit \in \mathcal{S}$, where a *situation* is a combination of different observables.

We define a **social norm** SN_i as a tuple formed by a situation and an action: $SN_i = \{\langle sit,a\rangle \mid sit \in \mathcal{S}, a \in A\}$.

In our scenario, the set of observables is formed by the following propositional terms: $\mathcal{O} = \{$ *Plenty(Me)*, *Plenty(You)*, *Normal(Me)*, *Normal(You)*, *Starving(Me)*, *Starving(You)*$\}$, where: *Plenty(X)* indicates that *Agent's X* resource level is over 100 units; *Normal(X)* indicates that *Agent's X* resource level is between 25 and 100 units; and, *Starving(X)* indicates that *Agent's X* resource level is below 25 units. The values that X can take are *Me* and *You*, representing the acting agent and the partner agent in the interaction. When two agents meet, each agent is able to observe its own level of resources and its partner level. The whole list of possible situations (formed by two observables) is detailed in Table I. The set of possible actions is $A = \{\text{Give Resources}, \text{Do not Give Resources}\}$. The combination of all possible situations, each one associated to a concrete action, generates a **set of social norms**. The use of one set or another determines the behavior of an agent in the environment and its social identity.

IV. Environmental Memory

Agents are endowed with a memory, in which they can save the situations in which they have interacted with other agents. This is done in order to keep a record of how many

ID	Situation		Action
1	Starving(Me)	Starving(You)	To Give / Not To Give
2	Starving(Me)	Plenty(You)	To Give / Not To Give
3	Starving(Me)	Normal(You)	To Give / Not To Give
4	Plenty(Me)	Starving(You)	To Give / Not To Give
5	Plenty(Me)	Plenty(You)	To Give / Not To Give
6	Plenty(Me)	Normal(You)	To Give / Not To Give
7	Normal(Me)	Starving(You)	To Give / Not To Give
8	Normal(Me)	Plenty(You)	To Give / Not To Give
9	Normal(Me)	Normal(You)	To Give / Not To Give

TABLE I

Situations and Actions. Structure of a set of social norms.

times a situation happens. The main utility of this memory is to calculate the relevance (as it will be explained later) of each of the norms based on the situations that the norms regulate. However, there are two posible ways of maintaining this memory:

1) *Unlimited Memory*: Agents maintain a counter of the happenings of each of the 9 situations during the whole simulation. In this way, the relevance of a norm is calculated using all the past interactions.
2) *Limited Memory*: Agents maintain a limited number of records in its memory. With this approach, the relevance of a norm is calculated using only the information available in the memory of the agent. In case the memory is full and new records have to be added, in order to allow agents to notice the changes in the environment, its memory is modelled as a FIFO queue (the first records included are also the first records deleted).

V. Group Recognition

In our scenario agents are initially loaded with 9 social norms (in our scenario an instance of those contained in Table I) that will determine their behavior on the interactions during the simulation.

A *social group* is a set of agents who are *socially related*, being conscious that they are socially related, and that interact with one another.

Two agents will know if they are *socially related* following a *similarity evaluation function*.

A *similarity evaluation function* will determine the degree of similarity between two agents.

Before explaining how the similarity evaluation function works we will introduce another important concept. Agents are assigned with a *Friendship Factor*. This factor will determine the minimum degree of similarity that two agents need to reach (by interacting and discovering the others norms) in order to be considered *socially related*. As we said, agents are also provided with a social memory, where they can remember the norms that other agents have used. Then, after each interaction with another agent, each agent will save in its memory the other agent's norm and update the similarity with it calculated using the similarity evaluation function.

The *similarity evaluation function* will determine that **two agents will be *socially related* if the sum of the relevances of the norms that they share minus the relevance of those they do not share is above a certain threshold**. This function is designed to consider both the common and uncommon norms shared with other agents in order to calculate the similarity.

The evaluation of the norm relevance is built during the execution of the simulation. This evaluation is personal of each agent, as it is built based on its own interactions with other agents.

The relevance of each of the 9 norms they have is calculated following a *Frequency Based Norm Relevance* function: the most important norms are those used more frequently. There are other norm relevance function as the *Benefit Based*, where a reward is assigned to some norms, giving a pre-assigned level of relevance to those norms.

The relevance of a norm N is calculated dividing the number of times that the situation associated to norm N occurs, by the total number of interactions that such agent has had.

Now we will define the basic algorithm of group recognition, and the new algorithms designed to improve that process.

A. Basic Algorithm

The simplest mechanism of group recognition in the previously described scenario is to allow agents to interact among them and make themselves keeping record of how *socially related* they are. This basic algorithm is represented in Algorithm 1.

```
repeat
    foreach Agent i do
        Randomly Meet Agent j;
        Interact following the set of social norms;
        Save information of partner behavior;
    end
until Exhausting Timesteps ;
```

Algorithm 1: Basic Group Recognition Algorithm

This algorithm makes each agent interacting with the rest of the society without any preference or intelligence in the partner selection process. During the simulation and after each interaction, the agent observes the actions taken by its partners (that follow their own set of social norms) in different situations. This allows agents to determine if those partners can belong to its social group.

In the following sections we will define new algorithms and functionalities that improve the behavior of this basic algorithm. These new algorithms take advantage of the social information that agents are gathering during the process and that can share with other agents through a communication protocol.

B. Intelligent Partner Selection

All the algorithms that will be presented from now on share this function that provides agents with more intelligence during the group recognition process. The mechanism in charge of making a smarter partner selection for an interaction is shown in Algorithm 2. As we said, agents have an *Interaction Probability* assigned to them that determine the frequency they meet random agents present in the

simulation. This algorithm modifies slightly that parameter, making agents to interact more frequently with other agents that have a positive degree of similarity. In this way, agents will promote interactions with other agents socially closer to themselves.

```
Probability To Meet Random Agent = 1 - NumberofKnownAgents / TotalNumberofAgents ;
if Probability To Meet Random Agent > Random Number then
    Meet Random Agent;
end
else
    Meet Known Agent;
end
```

Algorithm 2: Intelligent Partner Selection.

C. Whitelisting Algorithm

The *Whitelisting Algorithm* is based on the idea of recommending known trusted partners to your friends. When an agent discovers a new trusted partner (that is, the *degree of similarity* with that agent has gone above the *friendship factor*), it will recommend to this new agent some of its other socially-related agents as possible partners to interact with. If this agent does not know any of the recommended agents, it will add them to their preference list of agents to interact with in order to confirm this similarity. The algorithm is shown in Algorithm 3.

```
repeat
    foreach Agent i do
        Receive Recommendations;
        if Any Recommendation then
            Agent j = Any of the Recommended;
        end
        else
            Agent j = Intelligent Partner Selection;
        end
        if Similarity with Agent j > 0 OR Agent Unknown then
            Interact following the set of social norms;
            Save information of partner behavior;
            Recalculate Similarity with Agent j;
            if Similarity with Agent j > FRIENDSHIP FACTOR
            then
                Select M Known Agents with Similarity >
                FRIENDSHIP FACTOR ;
                Inform Agent j about these M Known Agents ;
            end
        end
        else
            Inform about the action the norms dictate, but Do Not
            Give;
        end
    end
until Exhausting Timesteps ;
```

Algorithm 3: Whitelisting Group Recognition Algorithm

D. Blacklisting Algorithm

In this case, unlike the whitelisting algorithm, the idea is to inform about unsatisfactory interactions to the rest of agents in the social group. Once an agent is detected as a non-member of the social group, this information is transmitted to the other members of the group so they can avoid that agent in the future. The algorithm is shown in Algorithm 4.

```
repeat
    foreach Agent i do
        Agent j = Intelligent Partner Selection;
        if Similarity with Agent j > 0 OR Agent Unknown then
            Interact following the set of social norms;
            Save information of partner behavior;
            Recalculate Similarity with Agent j;
            if Similarity with Agent j == −1 then
                Select M Known Agents with Similarity >
                FRIENDSHIP FACTOR ;
                Inform these M Known Agents about Agent j;
            end
        end
        else
            Inform about the action the norms dictate, but Do Not
            Give;
        end
    end
until Exhausting Timesteps ;
```

Algorithm 4: Blacklisting Group Recognition Algorithm

E. Labelling Algorithm

The last algorithm implemented to solve the problem of group recognition is the *Labelling Algorithm*. The main idea here is that agents are able to assign "labels" to other agents that can be accessed by a partner when two agents interact. Each agent carries the labels that others assign to it. The content of these labels is: (1) the identity of the agent with whom it interacted, (2) the situation in which they interacted and (3) the result of the interaction (*true* in case the action taken in that situation was the same for both agents, and *false* otherwise). It is straightforward to see that the power of this algorithm is based on publishing and making accesible to all agents previous interactions with different agents. Thus, when one agent A interacts with another agent B, A does not only access the information it obtained from direct experiences with B or communicated experiences from members of its group, but also indirectly (and through the labels) from all the agents that have interacted with B. Consequently, after evaluating the result of the interaction, agent A can also check other agents with similar experiences with agent B, and use that information. The algorithm is shown in Algorithm 5.

VI. Experiments

The purpose of the following experiments is to analyse how the different algorithms behave in dynamic environments, and how a partial knowledge of the history of the environment helps in the process of adaptation. Each experiment will vary in (1) the changes that happen in the environment, and (2) the type of memory that agents have (limited vs. unlimited).

To measure the efficiency of the algorithms during the process of group recognition and adaptation, we will use the average number of social relations[1] of each predefined group with each of the other predefined groups.

[1]Two agents will be *socially related* if the sum of the relevances of the norms they share minus the relevance of those they do not share is above the *Friendship Factor*

```
repeat
    foreach Agent i do
        Agent j = Intelligent Partner Selection;
        if Similarity with Agent j > 0 OR Agent Unknown then
            Interact following the set of social norms;
            Save information of partner behavior;
            if Action Agent j == Action Agent i would have taken
            in Agent j Situation then
                Add label(Agent i ID, Agent j Situation, true) to
                Agent j ;
                IDs of Agents to Veto = Obtain IDs from Agent's
                j Labels with (Agent j Situation, false);
            end
            if Action Agent j != Action Agent i would have taken
            in Agent j Situation then
                Add label(Agent i ID, Agent j Situation, false) to
                Agent j ;
                IDs of Agents to Veto = Obtain IDs from Agent's
                j Labels with (Agent j Situation, true);
            end
            foreach Agents to Veto do
                Save information from the label;
                Recalculate Similarity with another Agent;
            end
        end
        else
            Inform about the action the norms dictate, but Do Not
            Give;
        end
    end
until Exhausting Timesteps ;
```

Algorithm 5: Labelling Group Recognition Algorithm

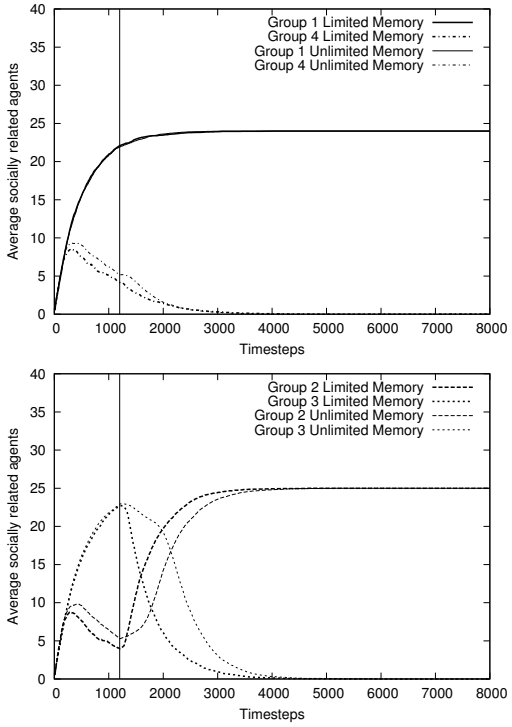


Fig. 1. Simulation results. Experiment 1

A. Experiment Design

We load into the simulation a society with the following characteristics: 100 agents where each agent can interact with all of the rest; all agents have the same Interaction Probability that will be fixed depending on the experiment. All the agents have the same Resource Gathering Probability fixed to $P_{RG}(Agent_i) = 0.0025$, except when we introduce an environmental change, that is refixed to $P_{RG}(Agent_i) = 0.1$. These environmental changes pretend to simulate an environment "poor" of resources when $P_{RG}(Agent_i)$ is fixed to 0.0025, and, a "rich" environment when $P_{RG}(Agent_i)$ is fixed to 0.1. The amount of resources found is also fixed to 250 units, although agents are not allowed to carry more than 1000 resource units. With these values we simulate an environment where resources are difficult to find although when they are found, they appear in a huge amount. Agents are not able to change their set of social norms. The *Friendship Factor* has been fixed to 0.3. Several values has been tried obtaining equivalent results. The society is initially loaded with 25 agents of 4 different types: (1) selfish in *Starving* and *Plenty* situations (from now on group 1, and Cyan colour in Fig. 2(l)); (2) altruists in *Starving* situations but selfish in *Plenty* situations (group 2, and Green colour); (3) selfish in *Starving* situations but altruists in *Plenty* situations (group 3, and Red colour); and, (4) altruists in *Starving* and *Plenty* situations (group 4, Blue colour). In the rest of the situations (the *Normal* ones) all the agents behave in an altruistic way.

When the environmental conditions are the ones with $P_{RG}(Agent_i) = 0.0025$, the *Starving* situations will be the most frequent; and when an environmental change happens (by refixing the Resource Gathering Probability of all agents to $P_{RG}(Agent_i) = 0.1$), the most frequent norms will be the *Plenty* norms.

Each simulation is run 10 times, during 10000 steps, and then the results are averaged. The results between simulations are very similar, and they have not shown any extreme behaviour that could affect the average of the results.

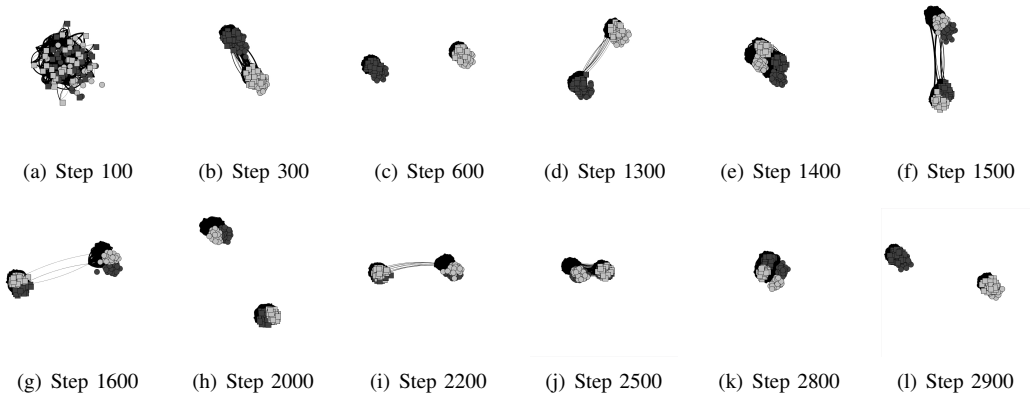
B. Experiment 1: One Permanent Change

The intention of this first experiment is to observe the improvements in speed of adaptation of the limited memory approach with respect to the unlimited memory approach. In order to do this, we will set the environment to a first configuration where agents self-adapt, and once they have reached an stable configuration of groups, the environment will radically change its state. This change will affect agents' notion of group, changing it.

In order to simulate this process, we have set the environment to remain "poor" of resources ($P_{RG}(Agent_i) = 0.0025$) until the timestep 1200. After that timestep, the environment will turn highly populated of resources ($P_{RG}(Agent_i) = 0.1$) until the end of the simulation.

Simulation results are shown in Fig 1. A vertical line at the timestep 1200 indicates when the change in the environment is introduced. The figures show the number of agents than the average agent of group 1[2] is *socially related* with, when using the basic algorithm. We have analyzed the results for

[2]The average of the results of each of the members of group 1



(a) Step 100 (b) Step 300 (c) Step 600 (d) Step 1300 (e) Step 1400 (f) Step 1500

(g) Step 1600 (h) Step 2000 (i) Step 2200 (j) Step 2500 (k) Step 2800 (l) Step 2900

Fig. 2. Group Recognition Simulation: Experiment 2 with Limited Memory Approach

the average agent of each of the groups, and the dynamics observed are equivalent for all of them. Given that, we will only show the results for the average agent from group 1, although the results can be generalized for all the groups.

In the top of the figure we can observe the number of agents *socially related* to the average agent in group 1 with other agents in group 1. Thus, we can observe how group 1 coheres. Moreover, we can observe how the number of agents *socially related* from group 4 decrease to zero. Agents from group 1 and group 4 are different in the norms that will be more relevant before and after the environmental change (that are, as explained in VI-A, those that regulates the *Starving* and *Plenty* situations respectively), therefore, rarely they will be *socially related*.

In the bottom of the figure we can observe the number of agents *socially related* from group 2 and group 3. Before the environmental change, the most relevant norms are the ones containing the *Starving* particle, and consequently agents from group 1 form a group with agents in group 3, as they share the same norms for the *Starving* situations. After the environmental change, it is easy to observe that agents can self-organize using both types of memory, although, adaptation is faster when using the limited memory than when using the unlimited memory. The unlimited memory approach needs at least the same number of interactions in the new environmental state as the ones that had already occured at the moment of the environmental change to consider the norms regulating the new environmental situations as relevant. However, the limited memory approach only needs a number of interactions equals to the memory size (in these experiments fixed to 50[3]) as the old memory records will be substituted by the new ones, creating the new normative relevance.

Furthermore, in both figures there is an initial increase of the number of agents *socially related* from all groups during the first timesteps. This effect is produced during a transition period produced for a initialization effect: agents are initially loaded with 100 resource units that set them in *Plenty* situations for at least one timestep. After the first timestep, and if agents do not find resources, agents interact in *Normal* situations for a period of time sufficient to consider the norms regulating those interactions as the relevant ones. As all agents share the same norms for those situations, this initial increase is observed. After consuming those initial resources and reach an stability within the environment, the social groups also stabilize.

We can observe in Fig. 3 the effects of the different algorithms on the group recognition process in dynamic environments. In Fig. 3 we can observe the number of agents from group 2 (top of the figure), and from group 3 (bottom of the figure), that are *socially related* with the average agent of group 1, under the effects of the different algorithms. It is easy to observe than the Labelling algorithm outperforms all the other three in speed of convergence, although, blacklisting and whitelisting outperform the basic one. The basic algorithm slightly improves in the number of agents correctly socially related. This is due to the effect of the intelligent partner selection that the intelligent algorithms (whitelisting, blacklisting and labelling) have. The intelligent partner selection makes agents interact more frequently with know agents, reducing the chances of meeting new agents. As the basic algorithm does not use this intelligent partner selection, it will make agents meet more agents and interact with different agents than the known ones.

On the other hand, the success of the labelling algorithm is based on the effects of making information about the norms and interactions available to other agents. As agents carry with them the results of interactions with other agents, it is easy to gather information in a faster way when environmental changes happen.

C. Experiment 2: One Sporadic Change

After observing how both memory approaches allow self-adaptation when a permanent change happens in the environment, we want to observe now the reaction of both approaches when sporadic changes occur. As we did in the previous experiment, we will set the environment to a

[3]The size of the memory affects to the speed of convergence to the new notion of group. Several values were tried and this one showed good performance

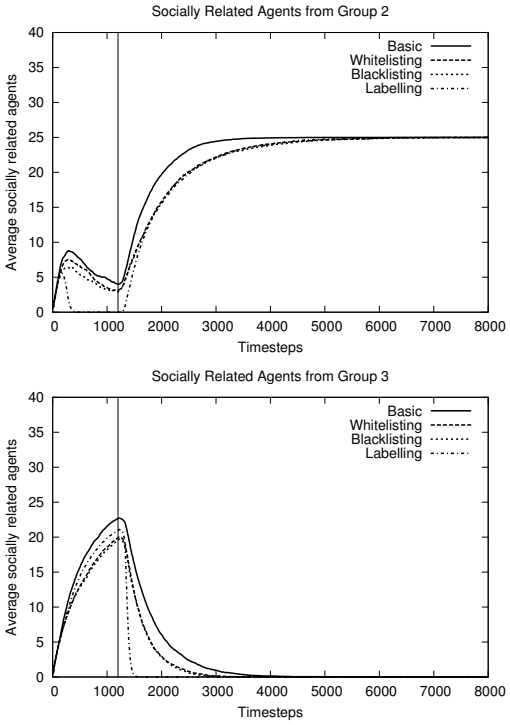


Fig. 3. Simulation results: Comparative Analysis of the Algorithms

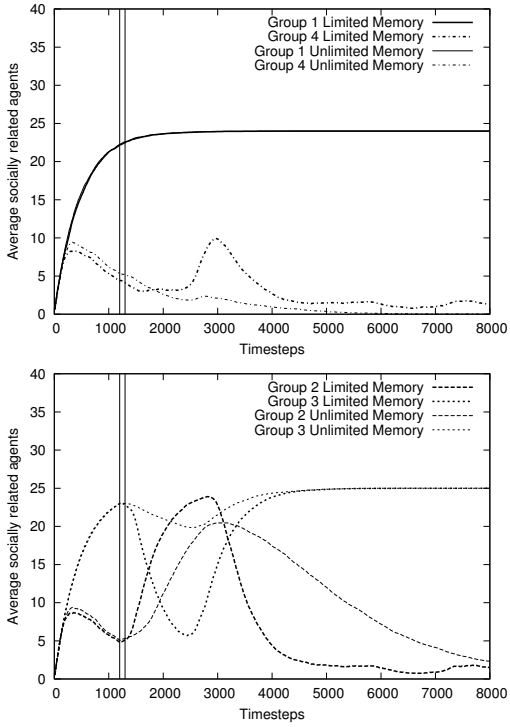


Fig. 4. Simulation results Experiment 2

configuration in which agents will adapt themselves; after that, a radical environmental change will be introduced, although for a limited period of time. Meanwhile the effects of the environmental change are active, agents should be able to adapt to those changes; notwithstanding, once the effects of this change are over, agents have to recover the initial configuration.

In order to simulate that effect, our environment will remain "poor" of resources ($P_{RG}(Agent_i) = 0.0025$) until the timestep 1200. After that timestep, the environment will turn highly populated of resources ($P_{RG}(Agent_i) = 0.1$) until the timestep 1300. Then, the environment will return to "poor" ($P_{RG}(Agent_i) = 0.0025$). The period where the environmental change is introduced is showed in the figures with two vertical lines, representing the beginning and the end of the change.

In the same way as we saw in the previous experiment, in Fig 4, in the top of the figure we can observe the number of agents from group 1 and 4 *socially related* with the average agent from group 1. The results obtained are equivalent to those obtained in the first experiment. However, there is a small detail that deserves to be explained. It is easy to observe that when using the limited memory approach the number of agents from group 4 *socially related* with the average agents from group 1 increases around the timestep 3000. This effect is produced in a transition period (as the one that happened at the begining of the simulation and have been previously explained): agents, from timestep 1200 to 1300, gather resources (up to the limit of 1000 resource units) that set them in *Plenty* situations. Therefore, after having consumed those 1000 resource units (around the timestep 2300), agents interact in *Normal* situations for a period of time sufficient to make the limited memory approach to consider the norms regulating those interactions as the relevant ones. We have to remember that in those situations all agents behave in the same way.

On the bottom part of the figure we observe the number of agents from group 2 and 3 *socially related* with the average agent from group 1. We can observe how agents recognize their groups before the environmental change: we can see in Fig 2(a) how all agents, without initially knowing each other, they all interact together, then, they start forming groups (Fig. 2(b)), until they form the groups as it can be seen in Fig. 2(c) (light grey agents and dark grey agents) . When the change occurs, agents using the limited memory, readapt the notion of group (and the agents they are *socially related* with) while they are under the effects of the change (agents gather resources during those timesteps and use them): when the change occurs, as we can see in Fig. 2(d), agents start reforming the groups, passing through a transition period where they interact in the situations where they share norms (Fig. 2(e)), and finally forming the new groups (Fig. 2(f), Fig. 2(g) and Fig. 2(h), circle agents and square agents). Once the effects of the environmental change are over, we observe how agents readapt to the initial configuration of groups (Fig. 2(i), Fig. 2(j), Fig. 2(k) and Fig. 2(l), light grey agents and dark grey agents). On the other hand, agents with the unlimited memory are not able to detect those changes because, as we explained in the previous experiment, agents would need a much higher number of interactions in the new

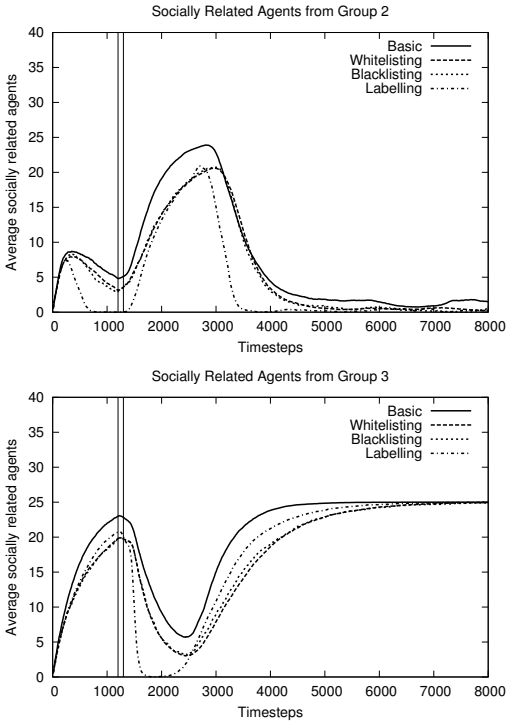


Fig. 5. Simulation results: Comparative Analysis of the Algorithms

situations (at least the same number of interactions in the new environmental state as the ones that had already occured at the moment of the environmental change) to consider the norms regulating the new environmental situations as relevant.

Lastly, in Fig. 5 we can observe the effects of the different algorithms in this experiment. In the same way as we did in the previous experiment, we can observe the number of agents from group 2, and from group 3 respectively, that are *socially related* with the average agent of group 1. The experimental results obtained show similar results than in the previous experiment: whitelisting and blacklisting outperform the basic algorithm, and the labelling outperforms the other three. The power of having more information available that the Labelling algorihtm provides is the key of its success in obtaining the faster results in the convergence of the groups.

VII. Conclusions and Future Work

In this work we have presented a self-adapting group recognition mechanism, that allows agents to change the notion of group and agents socially related to them depending on the environmental conditions. Simulation results have shown us that the unlimited memory approach works in the task of self-adaptation but in a much slower fashion than the limited memory approach, and only when the environmental changes are permanent or its effects are very longlasting.

The limited memory approach reacts against sporadic changes and allow agents to change in a faster way the notion of group. Besides we have compared the efficiency of the different algorithms in such environments, showing a good performance. Special attention should be paid to the Labelling algorithm which ourperforms all the others, taking into account the clear advantage it has due to its functioning (making information about other interactions accesible).

As part of the future work, we will study self-adating mechanisms for agents to adapt against different kinds of environmental changes, for example, gradual or periodic. Moreover, now that agents have a clear notion of group and are able to self-adapt against changes, we would like to start working on coordinated reaction mechanisms. This coordinated reaction mechanisms will allow agents to take decisions about other agents wanting to join their group or exploiting the advantages of the group without contributing into it. Consequently, we will center on punishment mechanisms and reasoning about when an agent will decide to start punishing another agent.

Acknowledgments

This work was supported by the European Community under the FP6 programme [eRep project CIT5-028575]; the Spanish Education and Science Ministry [AEI project TIN2006-15662-C02-01, AT project CONSOLIDER CSD2007-0022, INGENIO 2010]; Proyecto Intramural de Frontera MacNorms [PIFCOO-08-00017] and the Generalitat de Catalunya [2005-SGR-00093]. This work was partially supported by the Santa Fe Institute. Daniel Villatoro is supported by a CSIC predoctoral fellowship under JAE program.

References

[1] J. Elster, "Social norms and economic theory," *Journal of Economic Perspectives 3*, vol. 4, pp. 99–117, 1989.

[2] C. Bicchieri, *The Grammar of Society: The nature and Dynamics of Social Norms*. Cambridge University Press, 2006.

[3] N. J. Saam and A. Harrer, "Simulating norms, social inequality, and functional change in artificial societies," *Journal of Artificial Societies and Social Simulation*, vol. 2, no. 1, 1999.

[4] Y. Shoham and M. Tenneholtz, "On the synthesis of useful social laws for artificial agent societies (preliminary report)," in *Proceedings of the AAAI Conference*, 1992, pp. 276–281.

[5] A. Grizard, L. Vercouter, T. Stratulat, and G. Muller, "A peer-to-peer normative system to achieve social order," in *Workshop on COIN @ AAMAS' 06*, 2006.

[6] D. Hales and S. Arteconi, "Slacer: A self-organizing protocol for coordination in p2p networks," *IEEE Intelligent Systems*, vol. 21, pp. 29,35, 2006.

[7] R. Araujo and L. Lamb, "Memetic networks: Analyzing the effects of network propoerties in multi-agent performance," in *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, 2008.

[8] D. Hales and B. Edmonds, "Applying a socially-inspired technique (tags) to improve cooperation in p2p networks," *EEE Transactions in Systems, Man and Cybernetics - Part A: Systems and Humans, 35*, vol. 3, pp. 385–395, 2005.

[9] J. Holland, "The effects of labels (tags) on social interactions," *Working Paper Santa Fe Institute*, vol. 93-10-064, 1993.

[10] D. Villatoro and J. Sabater-Mir, "Group recognition through social norms," in *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, 2009.