Norm Adaptation using a Two-Level Multi-Agent System Architecture in a Peer-to-Peer Scenario

Jordi Campos
Universitat de Barcelona
585 Gran Via
08007 Barcelona, Spain
jcampos@maia.ub.es

Maite López-Sánchez
Universitat de Barcelona
585 Gran Via
08007 Barcelona, Spain
maite@maia.ub.es

Marc Esteva
IIIA - CSIC
Campus UAB
08193 Bellaterra, Spain
marc@iiia.csic.es

ABSTRACT

Existing organisational centred multi-agent systems (MAS) regulate agents' activities. Nevertheless, population and/or environmental changes may lead to a poor fulfilment of the system's purposes, and therefore, adapting the whole organisation becomes key. This is even more needed in open MAS, where participants are unknown beforehand, they may change over time, and there are no guarantees about their behaviours nor capabilities. Hence, in this paper we focus on endowing an organisation with self-adaptation capabilities instead of expecting agents to increase their behaviour complexity. We regard this organisational adaptation as an assisting service provided by what we call the *Assistance Layer*. Our abstract Two Level Assisted MAS Architecture (2-LAMA) incorporates such a layer. We empirically evaluate our adaptation mechanism in a P2P scenario by comparing it with the standard BitTorrent protocol. Results provide a performance improvement and show that the cost of introducing an additional layer in charge of system's adaptation is lower than its benefits.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems, Coherence and coordination*

General Terms

Design, Experimentation, Performance

Keywords

Adaptation, Norms, Coordination, Organisation, MAS

1. INTRODUCTION

Developing Multi Agent Systems (MAS) entails the problems of designing a distributed concurrent system plus the difficulties of having flexible and complex interactions among autonomous entities [1]. Organising such systems to regulate agent interactions is a practise that helps to face their complexity [2]. Specially in open MAS, since agents are developed by third-parties, so they may enter or leave the system at any moment and there are no guarantees about their behaviour. To face the derived complexity, some approaches [3, 4] use organisation entities as regulative structures. Such an organisation helps designers to predict/regulate the system evolution within certain bounds. The fact that these structures persist with independence of their participants reinforces their role as first-order entities. Moreover, these approaches usually provide an infrastructure to support the enactment of these entities —to create them, to store their specifications, to check if participants fulfil them, etc. In fact, these approaches provide an organisational framework to agents, which minimises the number of possibilities they have to face. This is because agents can construe other participant's behaviour under a certain context.

As we previously mentioned, an organisational structure helps to regulate MAS. However, certain environmental or population changes may decrease its performance to achieve goals. Thus, adapting such an organisation is an important topic [5, 6, 7, 8], since it can help to obtain the expected outcomes under changing circumstances. This is motivated by the computational organisational theory, which claims that the best organisation designs are domain and context dependent [9]. Adaptation can be seen as a reconfiguration aspect of autonomic computing, where a MAS is able to reconfigure itself [10].

Concerning such an adaptation, we propose to add a *meta-level* in charge of adapting system's organisation instead of expecting agents to increase their behaviour complexity. This is specially relevant when dealing with open MAS, since there is no control over participant's implementation. Hence, we cannot expect agents to be endowed with the necessary mechanisms to adapt the organisation when it is not achieving its goals. We regard this adaptation –together with other possible meta-level functionalities– as an assistance to agents that can be provided by MAS infrastructure. Thus, we call our approach Two Level Assisted MAS Architecture (2-LAMA). In order to avoid centralisation limitations such as fault-tolerance or global information unavailability, we propose a distributed *meta-level* composed of several agents. This paper is focused on 2-LAMA's organisational adaptation capabilities. In particular, it focuses on *norm adaptation* —we assume norms are an organisational regulative structure.

Our approach requires domains with organisations that can be dynamically changed. Besides, it is able to deal with highly dynamic environments and even with domains where there is no direct mapping between goals and the tasks required to achieve them —i.e. it is not possible to derive a set of tasks that achieve a certain goal. As an illustration, we present a Peer-to-Peer sharing network (P2P) as a representative case study. In such a network, computers interact to share some data. Furthermore, their relationships change over time depending on network status and participants. We use this scenario to perform an empiric evaluation and compare our approach with existing BitTorrent protocol [11].

Our general model and its application are described in sections 2 and 3. Further, the adaptation process is detailed in section 4. Next, it is compared with BitTorrent in section 5 and with related work in section 6. Finally, section 7 presents the derived conclusions.

2. GENERAL MODEL

Previous section identifies organisations as useful entities to regulate agents' behaviours and facilitate their coordination. In particular, these entities provide a framework that is useful for agent coordination. Besides, there are MAS infrastructures that provide some organisational-related features as domain independent services. Thus, we regard them as *Coordination Support* services [12] that alleviate agent development. These services also include basic coordination elements such as elemental connectivity or agent communication languages. In brief, all these services are devoted to enact agent coordination. In addition to that, we propose an extra set of services that provides an added value by assisting coordination. We propose to add an *Assistance Layer* on top of a regular system in order to provide such coordination assistance services The main contribution of this paper is the proposal of a distributed pro-active service at the Assistance Layer that adapts organisations depending on the system's evolution.

Before provinding an insight into this organisational adaptation service, we detail how we model an organisational structure itself. Usually, organisation-centred MAS provide services that range from establishing the basis for agent communication through individual messages to providing organisational structures. We denote one of those organisations as: $Org = \langle SocStr, SocConv, Goals \rangle$, its compoments are detailed next. It has a social structure ($SocStr$) consisting of a set of roles (Rol) and their relationships (Rel). In addition, it has some social conventions ($SocConv$) that agents should conform and expect others to conform. They are expressed as interaction protocols ($Prot$) and/or norms ($Norms$). In more detail, protocols define legitimate sequences of actions performed by agents playing certain roles. Whereas norms delimit agent actions by expressing related permissions, prohibitions or obligations. Notice, that in our case study, the only possible actions are message physical exchanges among agents. Finally, it has some goals ($Goals$) that describe the organisation design purpose —they may differ from participant's individual ones. These goals are expressed as a function over the system's observable properties —it may include the reference values they should approach. This way, system performance can be evaluated by using these goals to determine in which degree the system is fulfilling its design objectives.

2.1 Assistance Layer

The Assistance Layer we propose, provides an assistance that may facilitate the enrolment of third-party agents and/or adapt their organisation. This layer provides two main types of services [12]: assisting individual agents to achieve their goals following current social conventions (Agent Assistance); and adapting social conventions to varying circumstances (Organisational Assistance). The former includes services to inform agents about useful information to participate in the MAS (Information service), to provide justifications about the consequences of their actions (Justification service), to suggest alternative plans that conform social conventions

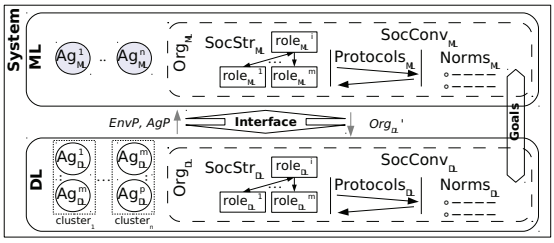

Figure 1: Two Level Assisted MAS Architecture (2-LAMA).

(Advice service) and to estimate the possible consequences of certain actions due to current conventions (Estimation service). The latter, the Organisational Assistance, consists on adapting existing organisations to improve system's performance under varying circumstances. To provide such an adaptation, we propose goal fulfilment as its driving force within the context of a rational world assumption. Hence, the Assistance Layer requires some way (i) to observe system evolution, (ii) to compare it with the organisational goals and (iii) to adapt the organisation trying to improve goal fulfilment. See [12] for further details about all enumerated services.

In order to provide Assistance Layer services, we proposed a Two Level Assisted MAS Architecture (2-LAMA, [13]). The bottom level, we call it domain-level (DL), is composed by agents carrying out domain activities regulated by an organisational structure. On top of it, there is a distributed meta-level (ML) also composed by agents and an organisational structure targeted to provide assistance services to domain-level agents. In between, there is an interface (Int) that communicates both levels as shown in Figure 1. Thus, the whole system can be expressed as: $2LAMA = \langle ML, DL, Int \rangle$[1]. Each level has an organised set of agents so they are respectively defined as $ML = \langle Ag_{ML}, Org_{ML} \rangle$ and $DL = \langle Ag_{DL}, Org_{DL} \rangle$. Using the interface, the meta-level can perceive environment observable properties ($EnvP$, e.g. date or temperature) and agents observable properties (AgP, e.g. colour or position). Specifically, we assume each meta-level agent ($a_{ML} \in Ag_{ML}$) has partial information about them, so it only perceives a subset of $EnvP$ and AgP —in many scenarios global information is not available. In fact, a a_{ML} has partial information about the subset of domain-level agents it assists. We call this subset of agents a *cluster*, which would be grouped according to a domain criterion —e.g. they could be grouped because interactions among them have lower costs than with other agents. However, an assistant can share part of this information with other meta-level agents in order to provide better assistance services.

3. 2-LAMA IN A P2P SCENARIO

Our case study is a Peer-to-Peer sharing network (P2P), where a set of computers connected to the Internet (peers) share some data. We apply our model to this scenario because it is a highly dynamic environment due to the very

[1]In fact, it is possible to nest subsequent meta-levels that update previous level's organisation.

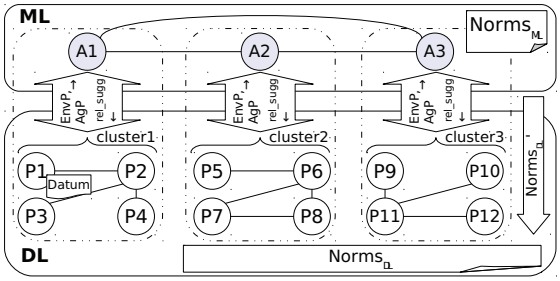

Figure 2: 2-LAMA in the P2P scenario.

Phase	Level	Protocol Messages
initial	Int	`join<hasDatum>`
latency	Int	`get_lat<peers>, lat<peer><measure>`
	DL	`lat_req, lat_rpl`
soc.struct.	Int	`contact<peers>`
handshake	DL	`bitfield<hasDatum>`
share data	DL	`rqst,data,cancel,have,choke,unchoke`
	Int	`complete, has_datum<peer>`
	ML	`all_complete, complete_peer<peer>`
norms	ML	`norm_bw<value>, norm_friends<value>`
	Int	`norm_updated<norm_id><new_def>`

Table 1: Protocol messages grouped into subsequent phases and involved levels —only domain-level (DL), only meta-level (ML) or both (Int).

nature of the Internet communications. We regard the *overlay network*[2] of current contacted peers as its organisational social structure, which is dynamically updated. Finally, this scenario allows the addition of some *norms* to regulate communications. Overall, it lets us apply our organisational and adaptive autonomic approach.

The performance in this scenario is evaluated in terms of time and network consumptions during the sharing process. Thus, we can define as global goals the minimisation of such measures so that the faster the data is obtained and the less network is consumed, the better for the users. Notice, though, that there is a trade-off between time and network usage. Therefore, although a peer can potentially contact any other peer, it usually contacts just a subset in order to consume less network resources —i.e. overlay network.

Real P2P networks are highly complex, so we try to reduce complexity by assuming some simplifications about the protocol and the underlying network. Specially, we assume information is composed of a single piece of data — accordingly, we say a peer is *complete* when it has that single piece. The rest of this section provides the details of the actual scenario and our 2-LAMA approach applied to it.

3.1 Architecture in P2P

We model the P2P scenario as a MAS where computers sharing data are participant agents within the domain-level (Ag_{DL}= $P1 \dots P12$). All of them play a single role Rol_{DL} = {*peer*} within the domain-level organisation (Org_{DL}) — see Figure 2. In addition, we define a type of relationship called *contact* between two agents playing the role peer. Thus, as all agents in domain-level play the role peer, they can establish contact relationships at run-time. These actual relationships form the overlay network mentioned previously. In our model, the meta-level can suggest changes in this net of relationships (*rel_sugg*) taking into account the system's status. Regarding social conventions, peers use the sharing protocol ($Prot_{DL}$) specified below and two norms $Norm_{DL} = \{normBW_{DL}, normFR_{DL}\}$. First *norm* ($normBW_{DL}$) limits agents' network usage in percentage of its nominal *bandwidth*[3]. This norm can be expressed as: $normBW_{DL}$ ="a peer cannot use more than $\texttt{max}_{\texttt{BW}}$ bandwidth percentage to share data". This way, it prevents peers from massively using their bandwidth to send/receive data to/from all other peers. Second norm ($normFR_{DL}$) limits the number of peers to whom a peer can simultaneously send the data. Analogously to previous norm, we define $normFR_{DL}$ ="a peer cannot simultaneously send the data to more than $\texttt{max}_{\texttt{FR}}$ peers". The last compoment of domain-level's organisation is its goal (*Goals*). This is that all peers –i.e. all computers sharing data– have the data as soon as possible using the minimal network resources. Thus, given some time cost (c_t) and network cost (c_n) metrics, we can define a global goal function that minimises a weighted combination of them: $Goals = min(w_t \cdot c_t + w_n \cdot c_n)$, where ($w_t$,$w_n$) are the corresponding weights that represent the relative importance of each measure.

In order to provide assistance to the domain-level, we add the meta-level on top of it. This *meta-level* also has a single role $Rol_{ML} = \{assistant\}$. Each agent in Ag_{ML}= $A1 \dots A3$ assists a disjoint subset of domain-level agents (cluster$\subset$ Ag_{DL}). It does it so by collecting information about them –about agents or their environment– and adapting their local organisation. Its decisions are based on local information about its associated cluster, aggregated information about other clusters –provided by other assistants— and the norms at their level ($Norm_{ML}$). Some examples of local information are latencies ($EnvP$) or which agents have the data (AgP). Information about other clusters come from other assistants —notice that meta-level agents have their own social structure too. Regarding meta-level norms, we consider one that limits the number of domain-level agents to inform about another domain-level agent having the data. More precisely, when an assistant receives the information that one agent in another cluster has become complete, the number of domain-level agents in its cluster it can inform to is limited. In particular, the norm is expressed as $normHas_{ML}$ ="upon reception of a complete agent (agent $\notin$ cluster) message, inform no more than $\texttt{max}_{\texttt{Has}}$ agents $\in$ cluster". Finally, we assume assistants are located at Internet Service Providers (ISP) and thus related communications are fast.

3.2 Protocol

Our proposed protocol is a simplified version of the widely used BitTorrent [11] protocol. Table 1 lists all its messages,

[2]An overlay network is a network build on top of another one. In the P2P scenario, the base network that connects all peers is the Internet. Then, the network of peers that are really interacting among them is an overlay network on top of the Internet.

[3]The *bandwidth* is the capacity to transfer data over user's network connection. It is expressed as the number of data units that can traverse a communication channel in a time unit. The less is used by the peer, the more is left for other purposes.

which follow the sequence detailed next. At the beginning, a domain-level agent (peer) initiates a handshake phase with another one by sending it a `bitfield <hasDatum>` message. `<hasDatum> = [1/0]` indicates if it has (1) or has not (0) the data —i.e. it is a *complete* or *incomplete* agent. Notice that in current implementation, the data has only a single piece. In turn, the other agent finishes this handshake phase by replaying with another `bitfield <hasDatum>` message to indicate its status. In case one of these agents have the datum and the other lacks it, the later sends a `rqst` (request) message to the former. Then, the former replies with a message containing the datum. On the contrary, if none of the agents have the datum they will not exchange further messages. However, as soon as one agent receives the datum, it will send a `have` message to these other contacted agents to let them know that its status has changed. In such cases, if they still lack of the datum, they will request it. Additionally, an agent may reply to a request with a `choke` message if it is already serving $\texttt{max}_{\texttt{FR}}$ agents —it means this agent is going to ignore any further message. Later on, when a transmission ends, it sends `unchoke` messages to all choked agents, so they can request the datum again. On the other hand, a requester agent is allowed to get data from two sources simultaneously. This is done –for a short time– in order to compare their effective bandwidth so to choose the fastest source (the other one is discarded with a `cancel` message).

Previous messages are related to communication at domain-level. However, there are other messages related to communication at *meta-level* and among levels. Initially, a new domain-level agent sends its `join <hasDatum>` message to the closest assistant —a domain-level agent measures it latency to all assistants and chooses the one having the smallest latency. Then, the assistant asks the agent to measure its latencies with all other agents in its cluster by sending a `get_lat <peers>` message. The agent measures latencies by exchanging `lat_req`/`lat_rpl` messages, and informs back the assistant with a `lat <measure>` message. Once an assistant has all latencies among its domain-level agents ($EnvP$) and knows which ones have the datum, it estimates which would be the best social structure —see [13]. Then it suggests the agent relationships by sending `contact <peers>` messages to all the agents in its cluster.

Additionally, when a domain-level agent receives the datum, it also informs its assistant with a `complete` message. Then, at meta-level this assistant informs other assistants with a `complete_peer <peer>` message. For instance, in Figure 2, when P2 receives the datum, it informs A1, which will inform A2 and A3. Next, contacted assistants spread this information towards their domain-level agents –limited by $\texttt{max}_{\texttt{Has}}$– with a `has_datum <peer>` message —e.g. A2 may inform P6 and P8 that P2 has the datum, if $\texttt{max}_{\texttt{Has}} = 2$. In that moment, informed agents measure their latencies to the new agent and request it, if it is better than any previous source. Finally, when an assistant detects that all domain-level agents in its cluster are complete, it sends an `all_complete` message to other assistants to avoid receiving more `complete_peer` notifications.

Last, the norm adaptation process requires some more messages —see section 4. When an assistant wants to update $normBW_{DL}$, it sends a `norm_bw <value>` message to the rest of assistants. Analogously, it would send a `norm_friends <value>` in case of a $normFR_{DL}$ update. Then, when a new value is finally agreed, each *assistant* informs its the domain-level agents in its cluster with a `norm_updated <norm_id> <new_def>` message.

4. ORGANISATIONAL ADAPTATION

Within our 2-LAMA architecture, the meta-level is able to adapt domain-level's organisation. In particular, we are working on social structure and norm adaptation. The former consists in the meta-level updating domain-level's overlay network as detailed in [13]. The latter is the focus of this paper, and it is described in this section. In brief, norm adaptation proceeds as follows. Initially, assistants collect status information from their cluster domain-level agents but also from other assistants —in a summarised form. Afterwards, they aggregate all this information. Next, they compute their desired values of norm parameters depending on this aggregated information. Finally, they use a voting scheme as a group decision mechanism to choose the actual norm updates before notifying their agents.

The underlying rationale of the norm adaptation process is to align the amount of served data with the amount of received data. Thus, the information collected by each assistant consists of some measures about the agents serving the datum and the ones that lack it. Specifically, they collect the following information:

- `srvBW`: the sum of the nominal bandwidths of the individual channels of the agents that are serving data.
- `rcvBW`: the sum of the nominal bandwidth of the individual channels of the agents that are receiving data.
- `rcvEffBW`: the sum of the effective receiving bandwidth of the agents that are receiving data. It can be smaller than `rcvBW` when only a few data is served or there is network saturation that delays message transport.
- `rcvExpBW`: the expected receiving bandwidth. It is estimated using the nominal one (`rcvBW`) re-scaled by current bandwidth limit ($\texttt{max}_{\texttt{BW}}$). It is computed to be compared with `rcvEffBW`. If effective serving bandwidth is limited by a $\texttt{max}_{\texttt{BW}} < 100$, the reference receiving bandwidth may be lesser than the nominal one (`rcvBW`) —since less data is being injected towards receiving agents.
- `waiting`: the number of agents that do not have the datum and are neither receiving it.

Such information could be collected by each assistant from its agents or by accessing network information. In the former case, assistants would query agents about such information. Thus, this method would require that domain-level agents would report true values —which would be difficult to guarantee in an open MAS. In contrast, we use the latter case, which does not require collaborative agents. In this method, assistants inspect domain-level agent communications to obtain such information by themselves —this requires assistants to have privileges to access network resources, which is acceptable if they are related to ISPs.

Depending on the cost of collecting such information, it may be retrieved continuously or at certain intervals. Also, depending on the cost of applying norm changes, the norm adaptation process may be performed at given intervals. In the current implementation, this process is performed at a

Algorithm 1 Adaptation algorithm used by *assistants*.

```
00 def adapt( srvBW, rcvBW, rcvEffBW,
01              waiting, maxFR, maxBW ):
02   τ = 0.1 ; ϵ = 0.2
03   rcvExpBW = rcvBW * (maxBW / 100)
04
05   // Adapt maxFR -------------
06   case (srvBW<(1-τ)*rcvBW)  : vFR=decr
07
08   case (srvBW>(1+τ)*rcvBW
09                && waiting>ϵ): vFR=incr
10
11   case (srvBW>(1+τ)*rcvBW
12                && waiting<ϵ): vFR=blnk
13
14
15   other /*srvBW ≈  rcvBW */: vFR=same
16
17
18   if(rcvEffBW<(1-τ)*rcvExpBW):vFR=decr
19
20   // Adapt maxBW ------------
21   case (vFR==decr ∧ maxFR==1  : vBW=maxBW/2
22   case (vFR==incr ∧ maxBW<100): vBW=100
23   other                      : vBW=maxBW
24
25   return [ vFR, vBW ]
```

fixed time interval ($\texttt{adapt}_{\texttt{interv}}$) with an average of these measures along it.

In order to compute the desired norms, an assistant weighs the information it has collected from its cluster with the information provided by other assistants. This way it can give more importance to local information. For instance, $\texttt{srvBW} = \texttt{w}_\texttt{L} \cdot \texttt{srvBW}_\texttt{L} + \sum (\texttt{w}_{\texttt{R}_\texttt{i}} \cdot \texttt{srvBW}_{\texttt{R}_\texttt{i}})$, where $\texttt{srvBW}_\texttt{L}$ stands for the local cluster's measure, $\texttt{srvBW}_{\texttt{R}_\texttt{i}}$ stands for the remote ones, $\texttt{w}_\texttt{L}$ stands for the weight of local information, and $\texttt{w}_{\texttt{R}_\texttt{i}}$ stands for the weight of remote one. Moreover, $\texttt{w}_\texttt{L} + \sum \texttt{w}_{\texttt{R}_\texttt{i}} = 1$ and $\nexists \texttt{w}_{\texttt{R}_\texttt{i}}, \texttt{w}_{\texttt{R}_\texttt{i}} > \texttt{w}_\texttt{L}$.

If the local weight is the maximum ($\texttt{w}_\texttt{L} = \texttt{1}$), then each assistant computes desired norms taking into account only its cluster status. On the contrary, if this weight is the minimum ($\forall_i \texttt{w}_{\texttt{R}_\texttt{i}} = \texttt{w}_\texttt{L}$), then each assistant gives the same importance to local information as to remote one —this is the case in the current implementation. The mid-point is a local weight greater than any remote one ($\forall_i \texttt{w}_{\texttt{R}_\texttt{i}} < \texttt{w}_\texttt{L}$) such as an assistant takes its decisions giving more importance to its local cluster, but taking into account the rest of the system.

With this aggregated information each assistant computes its vote for $\texttt{max}_{\texttt{BW}}$ (`vBW`) and $\texttt{max}_{\texttt{FR}}$(`vFR`). In the case of `vBW`, the vote is the numeric desired value for $\texttt{max}_{\texttt{BW}}$. Whereas in `vFR`, the vote is an action among incrementing $\texttt{max}_{\texttt{FR}}$ by one (`incr`), decrementing it by one (`decr`), keeping the same value (`same`) or abstaining with a *blank ballot-paper* (`blnk`) to avoid influencing in new $\texttt{max}_{\texttt{FR}}$ value. They use the process schematised in Algorithm 1 to compute both votes. This algorithm receives the measures we described plus current norm parameter values. Next, in line 2, some constants are initialised to be used as thresholds in comparisons (their values were empirically tested). Then, the expected receiving bandwidth is computed from the nominal one re-scaled by current bandwidth limit (line 3).

The main decision to choose a $normFR_{DL}$ is related to compare the available bandwidth used to serve (`srvBW`) to the available bandwidth used to receive (`rcvBW`). If there is a lack of serving bandwidth (line 6), the suggestion is to decrease the number of friends. This way, server *agents* will be simultaneously serving data to fewer agents, and these transmissions will finish sooner. Afterwards, once these other agents get the datum, there will be more data sources in the system and it will take less time to finish the datum distribution. On the other hand, if there is an excess of serving bandwidth and there are still agents waiting for data (lines 8-9) then, the assistant can increase the number of friends in order to serve more agents. There is another situation in which there is also an excess of serving bandwidth but there are no agents waiting for data (lines 11-12). This does not necessarily mean all agents have the datum, but at least the ones lacking it are receiving it from some source. In this case, the assistant uses a blank-ballot paper to let other assistants push for their own interests[4].

Finally, if none of the previous cases is true, it means that the serving bandwidth is similar to the receiving one (line 15) then, the vote is for keeping the same norm. This is because if there is no excess of serving bandwidth, the assistant prefers to vote for the same norm instead of just leaving the decision to the rest of assistants.

Despite previous cases, if there is network saturation in the intermediate channels, it is always better to decrease the number of friends. This will reduce the number of data transmissions. Hence, it will cut back network traffic and hopefully network saturation. In order to estimate if there is network saturation, the assistant checks if the effective receiving bandwidth (`rcvEffBW`) is smaller than the expected one (`rcvExpBW`). This is a sign that data packets are delayed by the intermediate network because it is saturated. Consequently, as a solution to saturation, the assistant votes for decreasing $\texttt{max}_{\texttt{FR}}$ (line 18).

Regarding the $normBW_{DL}$, it is only decreased in case it is not possible to further diminish the network usage by decreasing the number of friends —since $\texttt{max}_{\texttt{FR}}$ is already 1. In such a case, the assistant votes for dividing $\texttt{max}_{\texttt{BW}}$ by 2 (line 21). This way, server agents will use less bandwidth, which can help to diminish the network saturation. On the contrary, if the bandwidth is previously limited but there is no network saturation –since the assistant chose to increase $\texttt{max}_{\texttt{FR}}$–, then the bandwidth limit can be established again back to 100% (line 22). For the remaining cases, $\texttt{max}_{\texttt{BW}}$ keeps its value (line 23).

After choosing a convenient value for each norm parameter, an assistant sends its votes (`vFR`, `vBW`) to the rest of assistants —see `norm_bw` and `norm_friends` messages. Then, when assistants receive all the votes, they compute the actual norm parameters. To conclude, they send to their domain-level agents the new norms using the `norm_updated` message. Notice that the average may provide the same norm parameters values as before, thus no changes would be performed —in practise, it means no update message would be sent. This situation may occur when opposite options are interesting for the same amount of clusters.

[4]Notice, though, that the weighting method applied to measures may bring an assistant to this case when no agents in its cluster are waiting for data, but there are still waiting agents in other clusters. In such a case, if there is enough serving bandwidth, it is better to let other assistants choose by themselves the norm parameter values.

Regarding norm updates application, once a domain-level agent receives new norms, it tries to fulfil them. Thus, when an agent receives a $normBW_{DL}$, it adapts its sending ratio and when it receives a $normFR_{DL}$ it also tries to fulfil it. This means that if an agent is serving to less friends than the new $\texttt{max}_{\texttt{FR}}$, it will send `unchoke` messages to those agents it has previously choked. This may result in new data requests that it will be able to serve. On the contrary, if it was serving to more friends than the new $\texttt{max}_{\texttt{FR}}$, it will cancel some of those data transmissions and send a `choke` message[5].

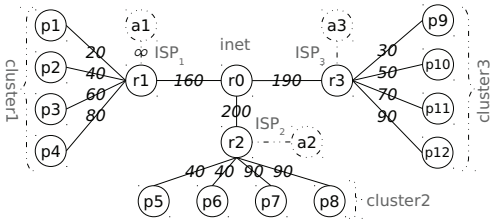

Figure 3: Network topology.

5. EMPIRICAL EVALUATION

In order to test our approach, we have implemented a P2P MAS simulator. This simulator is implemented in Repast Simphony [14] and provides different facilities to execute tests and analyse results. As it simulates both agents and network components, it allows to execute different sharing methods with identical populations and environmental conditions. Thus, we have performed several tests on BitTorrent and 2-LAMA to empirically evaluate the performance of our proposal.

5.1 Sharing methods

In this work, we compare three different approaches. A single-piece version of the BitTorrent protocol (BT) described in [15]. A 2-LAMA approach with social structure adaptation (2L.a) in which assistants update the actual contact relationships among domain-level agents as described in [13]. And a 2-LAMA approach with social structure and the norm adaptation (2L.b) described in this paper.

The BitTorrent implemented protocol (BT) among domain-level agents is very similar to 2-LAMA's since it inspired our approach. In order to make a fair comparison, we adapted BitTorrent to work with a single-piece datum —see [15] for further information. However, it does not have a distributed meta-level but a single agent (*Tracker*) that informs about connected agents. Consequently, agents do not receive any further assistance to share the datum. Instead, they use the algorithms described in [11]. In brief, the main algorithm of an agent having the datum consists in sending `choke` messages to all agents that are interested in it. Then, at certain intervals (`unchoke_interval`), the source agent sends `unchoke` messages to four of the previously choked agents. Next, these agents can request the datum and all of them are served. The selected agents to unchoke are those that were choked most recently. In case two of them were choked at the same time, the one having a larger network bandwidth (`upload_bw`[6]) is selected. In fact, if an agent's interest is older than a defined interval (`aging_period`), its age is ignored and only its agent's `upload_bw` is compared. In addition, in two out of three `unchoke_interval` selection processes, the fourth agent is randomly selected.

Regarding the configuration of our experiments, BitTorrent (BT) uses an `unchoke_interval` of 250 time units (ticks). It is approximately the time required to send four data messages along an average agent link in current topology. Thus, it is the average time that a server agent can invest sending data to four unchoked agents. This is the number of agents that BitTorrent protocol determines that an agent unchokes in an unchoke interval. Accordingly, they use an `aging_period` of 130 ticks to keep the ratio defined by the official protocol. On the other hand, the 2-LAMA experiments (2L.a, 2L.b) have been performed with the following initial norm parameters: $\texttt{maxHas} = \infty$, $\texttt{maxBW} = 100\%$, $\texttt{maxFR} = 3$. These norms lead 2-LAMA approach to a similar initial behaviour as BitTorrent because: $\texttt{maxHas} = \infty$ does not restrict communications among clusters, $\texttt{maxBW} = 100\%$ does not limit agent communication and $\texttt{maxFR} = 3$ is equivalent to the three non-random unchoked agents. This is specially the case because in our current implementation, domain-level agents always fulfil norms[7]. Additionally, for those tests including norm adaptation (2L.b), it has been done at an interval of $\texttt{adapt}_{\texttt{interv}} = 50$ time steps.

5.2 Results

In our experiments, we use a packet switching network model to simulate the transport of messages among agents. Figure 3 shows the network topology we use in our simulations. Notice that, as we are interested in having a different communication capacity for each domain-level agent, we place an *individual link* between each agent ($p1..p12$) and its corresponding Internet Service Provider ($ISP_1..ISP_3$ represented by routers $r1..r3$). In 2-LAMA experiments, each ISP has an associated assistant[8] ($a1..a3$) in charge of its connected domain-level agents. In addition, as we want to model simultaneous network usage by different agents, we place an *aggregated link* among each group of agents –i.e. a cluster, those connected to the same ISP– and the Internet ($r0$). In fact, in BitTorrent experiments, there are no assistants at all but a single tracker linked to this $r0$. Notice that the network topology influences the time required to transmit a message from one agent to another. In particular, this time depends on: message's length, the bandwidths of the traversed links, and the number of simultaneous messages traversing the same links —a link's bandwidth is divided among the messages that traverse it simultaneously. Regarding the former issue, we have used

[5]In the current implementation, an agent does not need to cancel a *friend* if it has already sent more than 75% of the datum to it. This behaviour avoids cancelling data transmissions that will finish really soon.

[6]In a multi-piece scenario, this measure is estimated from previous piece interchanges. However, since in a single-piece implementation no estimation can be performed, its value is taken from the network topology.

[7]Otherwise, we could assume there is an infrastructure mechanism at ISPs that detects and filters out messages that exceed the bandwidth limit (`maxBW`), or the simultaneous data messages limit ($\texttt{maxFR} = 3$).

[8]Our network model includes a quality of service (QoS) feature that gives more priority to messages among assistants or between assistants and domain-level agents. Thus, communications at meta-level and among levels are faster than communications at domain-level.

	time	*cNet*	*nHops*	*nData*	*cLat*	*cML*
BT	933.3	206182	3.4	11	0	0
2L.a	849.7	345060	3.2	40.1	21600	3749.9
2L.b	811.1	316190	3.0	30.7	21600	6596.0

Table 2: Results from BitTorrent (BT), 2-LAMA without norm adaptation (2L.a) and 2-LAMA with norm adaptation (2L.b).

the following message lengths: `piece` messages have 5000 data units, `lat_req` / `lat_rpl` have 150 data units and all the other control messages have a single data unit. Regarding the bandwidths links, Figure 3 shows them as numbers over the edges —we assume upload/download channels are symmetric. Finally, the latter issue, related to simultaneous link usage, is highly dynamic and depends on system's evolution.

We have tested all approaches in the described network topology by varying the agent that initially has the datum. Table 2 shows the results of different evaluation metrics in both approaches: BitTorrent (BT), 2-LAMA with social structure adaptation but no norm adaptation (2L.a) and 2-LAMA with social structure and norm adaptation (2L.b). Figures correspond to the average results for twelve different settings (so that they cover all possible initial datum positions in a single agent).

The evaluation metrics in Table 2 are the following: (1) *time* corresponds to the total time required to spread the datum among all agents; (2) *cNet* is the network cost consumed by all messages —each message cost is computed as its length times the number of links it traverses; (3) *nHops* is the average number of links traversed by each message; (4) *nData* is the total number of sent data messages —they may not be totally transmitted if: a destination agent sends a cancel message to its source or a source stops sending data to fulfil an updated $normFR_{DL}$; (5) *cLat* is the cost of all `lat_req/lat_rpl` messages; (6) *cML* is the cost of all messages related with the meta-level —i.e. all messages sent to or by *assistants*.

If we compare the performance of both approaches (BT and 2-LAMA), we see that our proposal requires less time to share the datum. Notice also, that 2-LAMA with social structure and norm adaptation (2L.b) presents shorter times that the version without norm adaptation (2L.a). In general, having better times in 2-LAMA means that the time invested in communicating with meta-level is less than the benefits of having such an additional level. Even more, we expect larger differences in performance when repeating the data sharing among the same P2P agent community since the information collected by our meta-level –e.g. measured latencies– will be used more than once. In fact, in our current 2-LAMA experiments, from 33 up to 56 ticks –depending on the cluster of agents– are invested in measuring latencies.

In contrast, the network cost (*cNet*) is larger in 2-LAMA, although norm adaptation (2L.b) provides the best performance again. Our proposal requires more communication because it initially measures latencies (*cLat*), it has extra communications due to the meta-level (*cML*), and it sends more data messages (*nData*). Specifically, latency measurements (*cLat*) represent up to a 20% of the network cost increment. This measurements are an initialisation phase that could be omitted in subsequent executions. On the other hand, 2-LAMA agents compare data sources by retrieving some data from them. This increases the number of data messages (*nData*) although most of them are cancelled. We expect to minimise this network consumption when dealing with more than one piece of data, since agents could compare sources depending on previous retrieved pieces. Regarding the number of links traversed by messages (*nHops*), our 2-LAMA approach has more local communications –i.e. intra-cluster– than BT. This is convenient because local messages have lower latencies and costs, since they are usually performed in the same cluster.

Overall, norm adaptation (2L.b) provides the best results despite requiring more assistant communication (*cML*). This stresses the idea that having a meta-level and exploiting its capabilities provides more benefits than the costs it causes.

6. RELATED WORK

Within MAS area, organisation-centred approaches regulate open systems by means of persistent organisations —e.g. Electronic Institutions [3]. Even more, several of these approaches offer mechanisms to update their organisational structures at run-time —e.g. Moise+ [4]. However, most work on adaptation maps organisational goals to tasks and look for agents with capabilities to perform them —e.g. OMACS [5]. Consequently, these approaches cannot deal with scenarios that lack of this goal/task mapping, like our case study. In order to deal with this sort of scenarios, our approach uses norms to influence agent behaviour, instead of delegating tasks. Specifically, our approach uses a norm adaptation mechanism based on social power —see norm taxonomy [16]. In this sense, there are other works that also use the leadership of certain agents (like our assistants) to create/spread norms —e.g. the role model based mechanism [17]. Besides, the most of norm emergence works are agent-centred approaches that depend on participants' implementation and they rarely create/update persistent organisations —e.g. infection-based model [18].

Relating norms and overall system behaviour, is a complex issue that increases its intricacy when there is no control over participant's implementation. In our approach, this task is distributed among a assistant agents which finally reach an agreement about norm updates. Currently, assistants use a voting scheme to agree on actual norms, but they could use some of the other agreement mechanisms present in literature —e.g. using an argumentation protocol [19]. Moreover, currently assistants use an heuristic to take their local decisions, but we are planning to use learning techniques in future work —like in AEI [20].

Regarding our P2P case study, there are some network management perspective approaches that also try to promote local communications but they cannot directly act on network consumption to balance net capacity and traffic —e.g. P4P [21] or ONO [22]. From a MAS angle, there are some works where agents adapt local norms using local information but they cannot reason/act at an organisational level —e.g. P2P normative system [23].

7. CONCLUSIONS

This work proposes an abstract MAS architecture (2-LAMA) to provide *assistance* to its participants. Particularly, this paper regards adapting a MAS organisation to varying cir-

cumstances as a type of assistance. It illustrates this approach in a P2P sharing network scenario, providing in-depth details about the adaptation process.

We endow the system with adaptation capabilities instead of expecting the agents to increase their behaviour complexity. Consequently, we propose to add a distributed *Assistance Layer* to improve system's performance by providing new support services to agents. In particular, in our architecture *meta-level* agents perceive information about MAS participants and environment, and are able to adapt the system's organisation.

Our 2-LAMA approach can be applied to domains with highly dynamic environments and no mapping between tasks and goals. It only requires that an organisation-centred MAS with an alterable organisation can be deployed. Such an organisation may include norms in its regulative structures. Moreover, the MAS can be open to third-party agents. As an illustration of all these issues, we introduce a representative case study based on a Peer-to-Peer sharing network. Additionally, to prove 2-LAMA's feasibility empirically, we have performed some experiments which show that the cost of adding our proposed Assistance Layer is lower than the obtained benefit. Specifically, 2-LAMA approach required less time than the original BitTorrent protocol. Even more, our approach results improved when increasing meta-level adaptation capabilities —i.e. when updating norms in addition to social structure adaptations.

As future work, we plan to confront further issues in open MAS such as how the system should react to agents joining or leaving the MAS anytime, or transgressing its organisational restrictions. In fact, we already have preliminary results about norm violations that show how system re-adapts to counter violation side effects. Besides, we are improving meta-level agents to use learning techniques in order to perform the adaptation process.

Acknowledgements: This work is partially funded by IEA (TIN2006-15662-C02-01), EVE (TIN2009-14702-C02-01 / TIN2009-14702-C02-02) and AT (CONSOLIDER CSD2007-0022) projects, EU-FEDER funds, the Catalan Gov. (Grant 2005-SGR-00093) and M. Esteva's Ramon y Cajal contract.

8. REFERENCES

[1] N. Jennings, K. Sycara, and M. Wooldridge, "A roadmap of agent research and development," *Autonomous Agents and Multi-Agent Systems*, vol. 1, no. 1, pp. 7–38, 1998.

[2] B. Horling and V. Lesser, "A survey of multi-agent organizational paradigms," *Knowl. Eng. Rev.*, vol. 19, no. 4, pp. 281–316, 2004.

[3] M. Esteva, *Electronic Institutions: from specification to development*, ser. IIIA PhD. Vol. 19, 2003.

[4] O. Boissier and B. Gâteau, "Normative multi-agent organizations: Modeling, support and control," in *Normative Multi-agent Systems*, 2007.

[5] S. A. Deloach, W. H. Oyenan, and E. T. Matson, "A capabilities-based model for adaptive organizations," *Autonomous Agents and Multi-Agent Systems*, vol. 16, no. 1, pp. 13–56, 2008.

[6] R. Kota, N. Gibbins, and N. Jennings, "Decentralised structural adaptation in agent organisations," in *AAMAS Workshop on Organised Adaptation*, 2008.

[7] M. Sims, D. Corkill, and V. Lesser, "Automated Organization Design for Multi-agent Systems," *Autonomous Agents and Multi-Agent Systems*, vol. 16, no. 2, pp. 151–185, 2008.

[8] C. Zhang, S. Abdallah, and V. Lesser, "MASPA: Multi-Agent Automated Supervisory Policy Adaptation," Tech. Rep. 03, 2008.

[9] K. Carley, "Computational and mathematical organization theory: Perspective and directions," *Computational & Mathematical Organization Theory*, vol. 1, no. 1, pp. 39–56, 1995.

[10] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *IEEE Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[11] BitTorrentInc., "BitTorrent protocol specification," 2001, http://www.bittorrent.org/beps/bep_0003.html.

[12] J. Campos, M. López-Sánchez, and M. Esteva, "Assistance layer, a step forward in Multi-Agent Systems Coordination Support," in *Autonomous Agents and Multiagent Systems*, 2009, pp. 1301–1302.

[13] Jordi Campos and Maite López-Sánchez and Marc Esteva, "Multi-Agent System adaptation in a Peer-to-Peer scenario," in *ACM SAC09 - Agreement Technologies*, 2009, pp. 735–739.

[14] M. North, T. Howe, N. Collier, and J. Vos, "Repast Simphony Runtime System," in *Agent Conference on Generative Social Processes, Models, and Mechanisms*, 2005.

[15] J. Campos, M. López-Sánchez, M. Esteva, A. Novo, and J. Morales, "2-LAMA Architecture vs. BitTorrent Protocol in a Peer-to-Peer Scenario," in *Artificial Intelligence Research and Development - CCIA09*, no. 202. IOS Press, 2009, pp. 197–206.

[16] B. S. S. Cranefield, "A categorization of simulation works on norms," 2009.

[17] B. S. S. Cranefield, M. Purvis, and M. Purvis, "Role model based mechanism for norm emergence in artificial agent societies," *Lecture Notes in Computer Science*, vol. 4870, p. 203, 2008.

[18] N. Salazar-Ramirez, J. A. Rodríguez-Aguilar, and J. L. Arcos, "An infection-based mechanism for self-adaptation in multi-agent complex networks," S. Brueckner, P. Robertson, and U. Bellur, Eds., 2008, pp. 161–170.

[19] A. Artikis, D. Kaponis, and J. Pitt, *Multi-Agent Systems: Semantics and Dynamics of Organisational Models*, 2009, ch. Dynamic Specifications of Norm-Governed Systems.

[20] E. Bou, M. López-Sánchez, and J. A. Rodríguez-Aguilar, *Autonomic Electronic Institutions' Self-Adaptation in Heterogeneous Agent Societies*. Springer, 2009, vol. 5368, pp. 18–35.

[21] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz, "P4P: provider portal for applications," 2008.

[22] D. Choffnes and F. Bustamante, "Taming the torrent: a practical approach to reducing cross-ISP traffic in peer-to-peer systems," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 363–374, 2008.

[23] A. Grizard, L. Vercouter, T. Stratulat, and G. Muller, "A peer-to-peer normative system to achieve social order," *LNCS*, vol. 4386, p. 274, 2007.