A Case-Based Reasoning approach for Norm adaptation

Jordi Campos[1], Maite López-Sánchez[1], and Marc Esteva[2]

[1] Universitat de Barcelona, 585 Gran Via, 08007 Barcelona, Spain
{jcampos,maite}@maia.ub.es
[2] IIIA - CSIC, Campus UAB, 08193 Bellaterra, Spain
marc@iiia.csic.es

Abstract. Existing organisational centred multi-agent systems regulate agents' activities. However, population/environmental changes may lead to a poor fulfilment of system's goals, and therefore, adapting the whole organisation becomes key. In this paper, we propose to use Case-Based Reasoning learning to adapt norms that regulate agents' behaviour. Moreover, we empirically evaluate this approach in a P2P scenario.

1 Introduction

Developing Multi Agent Systems (MAS) is a complex task due to the difficulties of having flexible and complex interactions among autonomous entities. Organising such systems to regulate agent interactions helps to predict/regulate the system's evolution within certain bounds. However, certain environmental or population changes may decrease its ability to achieve its organisational goals. Thus, adapting such an organisation is now becoming an important topic [10].

Concerning this adaptation, we propose to add an *Assistance layer* [6] in charge of it, instead of expecting agents to increase their behaviour complexity —it is relevant in open MAS, since there is no control over agent code. In particular, we use this layer to adapt norms that are part of system's organisation. Notice, that these norms regulate agents' activity by bounding their actions, but agents still keep its degree of freedom to choose their actual actions. Thus, this additional layer can adapt the organisation while preserving agents' autonomy. However, the relationship between these norms and system's outcomes makes the adaptation process very complex, since there is no direct mapping among them. Such a process can be coded by the system designer or learnt. Though, due to difficulties to define an optimal mechanism, we advocate by the learning approach. In this paper, we use a Case-Based Reasoning method, which faces new situations based on past experience [2]. As an illustration, we present a Peer-to-Peer sharing network (P2P) scenario where computers contact among them to share some data. In such a scenario, the relationship among computers' activity, the network traffic and the time required to share a datum is complex.

Next section 2 provides details on related previous work. Afterwards, our proposal is described in sections 3-4 and evaluated in section 5. Finally, our conclusions are presented in section 6.

2 Related work

Within MAS area, organisation-centred approaches regulate open systems by means of persistent organisations —e.g. EI [11]. 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 [10]. 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. In this sense, there are other works that also use the leadership of certain agents to create/spread norms —e.g. the role model based mechanism [9]. Besides, most works on norm emergence are agent-centred approaches that depend on participants' implementation and they rarely create/update persistent organisations —e.g. infection-based model [15].

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 some empowered agents that finally reach an agreement about norm updates. Currently, they use a voting scheme to agree on actual norms, but they could use some other agreement mechanisms present in the literature —e.g. argumentation protocols [3]. In particular, these agents take their local decisions using the Case-Based Reasoning (CBR) learning technique described in [2], which faces new situations based on past experience. The Autonomic EI [5] also use CBR to adapt their organisation, taking a centralised approach. On the contrary, we take a distributed approach both at the processing and knowledge levels as defined in [14].

Regarding our P2P scenario, there are network management perspective approaches that try to promote local communications but they cannot directly act on network consumption to balance net capacity and traffic —e.g. P4P [16]. From a MAS angle, there are works where agents adapt local norms using local information but they cannot reason/act at an organisational level —e.g. [12].

3 Assistance in P2P scenario

Our approach consists in providing support to the coordination of agents —see coordination support in [6]. In fact, we proposed a generic Two Level Assisted MAS Architecture (2-LAMA [8]) to help agents to participate in organisational-centred MAS. We use this generic architecture to develop systems that self-adapt their organisation depending on their evolution. In particular, we model our Peer-to-Peer sharing network case study (P2P) as a MAS with two level of organised agents —see Figure 1. Both levels share the same goal, which is that all participant agents obtain the data by consuming minimum time.

On the one hand, we model the set of computers that share some data as agents ($Ag_{DL} = \{P_1 \ldots P_n\}$) within a *domain-level* (DL). Its single role `peer`

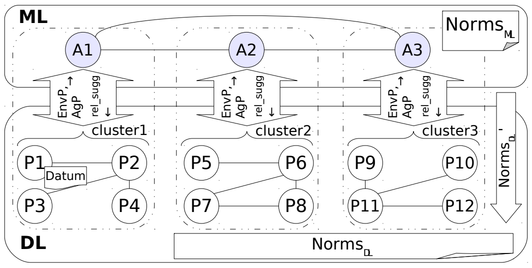

Fig. 1. 2-LAMA in P2P scenario.

Table 1. Results in P2P scenario.

	time	*cNet*	*h*	*data*	*cML*
BT	941.2	205344.1	3.4	11.0	-
2L.a	834.9	293526.7	2.9	35.9	5133.3
2L.b	741.5	292357.7	3.0	33.8	4694.1

and the relationships among them (i.e. the overlay network) conform the social structure of their organisation. This organisation also has its own social conventions: a sharing protocol derived from standard BitTorrent [8] and two norms ($Norm_{DL}$). First norm limits agents' network usage in percentage of its nominal bandwidth: $normBW_{DL}$ =“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. Notice that a massive network use, may saturate it, and therefore, it may delay all communications. Second norm limits the number of peers to whom a peer can send the data: $normFR_{DL}$ =“a peer cannot simultaneously send the data to $> \texttt{max}_{\texttt{FR}}$ peers”.

On the other hand, in order to support the coordination of previous agents, we add an *Assistance layer* to the described MAS. Currently, this support consists in adapting domain-level's organisation to changing circumstances. More precisely, it consists in adapting two DL's organisational components: norms –see section 4– and part of the social structure —by suggesting social relationships among pairs of DL agents (rel_sugg), see [8]. These adaptations are performed by a *meta-level* (ML) set of agents ($Ag_{ML} = \{A_1 \dots A_m\}$) that play the role `assistant`. Each assistant is in charge of a disjoint subset of Ag_{DL} (cluster).

In fact, assistants use an *interface* among both levels to collect local information about connection bandwidths and communication latencies (i.e. environment observable properties, $EnvP$) and about who has the datum (i.e. agent observable properties, AgP). In particular, each assistant counts on both detailed information from its cluster and aggregated information from other clusters supplied by other assistants. They weight them to combine the information before starting its own decision process —in current tests, each assistant gives the same importance to its local information than to remote one.

4 Learning norm adaptation

As we mentioned, in P2P scenario, adapting norms to obtain desired system outcomes is a complex task. Mainly, because there is no direct mapping between norms and system's behaviour. For instance, when updating a norm (e.g. increasing $\texttt{max}_{\texttt{FR}}$), it is difficult to foresee the effect of organisational changes (e.g. how many data messages will be transmitted), and it is even more complex to

Algorithm 1 Retrieve(left&top-right) & Reuse(bottom-right) phases.

```
01 def retrieve( newCase ):
02  retrCases = {} ; bestS = 0
03  foreach prevCase in caseBase:
04    s = $\sigma$ ( prevCase.prb, newCase.prb )
05    if ( s > MIN_SIM ):
06      case ( s > bestS ):
07        retrCases = { prevCase }
08        bestS     = s
09      case ( s $\simeq$ bestS ):
10        retrCases=retrCases$\cup${prevCase}
11  if ( retrCases is empty ):
12    heuCase = Heuristic.solve(newCase)
13    retrCases  = { heuCase }
14  return retrCases

01 def reuse( retrCases, newCase ):
02  if ( $\delta$(retrCases) > MAX_DIV )
03    heuCase = Heuristic.solve(newCase)
04    retrCases  = { heuCase }
05  sol = adapt( retrCases, newCase )
06  return Case( newCase.prb, sol )
```

anticipate system's outcomes (e.g. the total time required to spread data). In order to face this complexity, the meta-level uses a learning technique to decide how to adapt domain-level norms depending on current system status. In particular, we apply a CBR [2] learning approach, to suggest norm updates (solution) to a new system status (problem) based on similar previous situations (previous cases). Our CBR approach is based on a heuristic that tries to align the amount of serving/receiving capacity —see [7]. In fact, the heuristic itself is used by our CBR to suggest a solution when no similar cases are found.

Case description. The description of a problem and its solution conforms a *case* that can be stored as a previous case in a *case base*. The former (*Prob*) is described by a set of attributes (*Attribs*) derived from measures perceived through the interface —as they derive from observable measures, there are not unknown attributes. In particular, we use the following discretised attributes to describe a problem: `srvCap`, it indicates if there is enough serving capacity to serve all receiving peers by comparing the bandwidth of all serving peers with the bandwidth of all receiving peers (`rcvBW`); `netSat`, it estimates the network saturation by comparing the actual receiving bandwidth of receiving peers and their expected bandwidth ($\texttt{rcvExpBW} = \texttt{rcvBW} \cdot \texttt{max}_{\texttt{BW}}$, since $\texttt{max}_{\texttt{BW}}$ limits the data injected towards receiving peers); `wait`, it reflects the amount of peers that lack the datum and are not receiving it currently; `sRatio`, it indicates sources' maximum ratio to spread the datum, thus it is derived from current friends' norm; `bwUsg`, it indicates the bandwidth used by peers in their communications, thus it is derived from current bandwidth limit norm. Besides, a *solution* is described by two discrete attributes: `vFR`, it indicates how to update $\texttt{max}_{\texttt{FR}}$ by increasing one unit, decreasing one unit, keeping the same value or avoiding influencing it (i.e. a *blank ballot-paper*); `vBW`, it defines how to adapt $\texttt{max}_{\texttt{BW}}$ by setting it to 100%, keeping its value or dividing it by two.

CBR Cycle. There are four main phases [2]: retrieve, reuse, revise and retain. The first phase (*retrieve*) fetches the most similar cases (`retrCases`) from the case base (`caseBase`) as illustrated in left side of Algorithm 1. It starts with an empty list of cases and a minimum reference similarity (`bestS`) —see line 2. Then, it traverses the case base –line 3– computing the similarity (σ, see below) of each previous case's problem description (`prevCase.prb`) with the new problem (`newCase.prb`) —line 4. In case this similarity is greater than a *mini-*

mum trusted similarity (`MIN_SIM`) the case is a candidate to be retrieved —line 5. In particular, if this similarity is greater than any previous one –line 6– then the previous case is the one to be retrieved —line 7. Alternatively, if the similarity is equal to previous greatest one –line 9– then current previous case is collected with the rest of similar ones —line 10. In other words, it tries to return the most similar previous case, although it can return more cases when they have nearly the same similarity. However, if no previous case has the minimum trusted similarity to consider it is representative enough to adapt its solution to the new problem –line 11–, the algorithm executes the heuristic –line 12– to solve this unknown problem.Finally, in both cases the cases are returned —line 14. The *case similarity function* (σ) among two problems ($p_x, p_y \in Prob$) consists on computing the *attribute similarity function* (ς_i) among corresponding values of the same attribute ($a_i^{p_x}, a_i^{p_y} \in Attrib_i$) to aggregate them in a weighted manner: $\sigma(p_x, p_y) = \sum_{i \in Attribs} \left(w_i^\sigma \cdot \varsigma_i(a_i^{p_x}, a_i^{p_y})\right)$, $\sum w_i^\sigma = 1$. In order to compute this attribute similarity function (ς_i), we define a *label distance function* (λ_i) that provides a numeric distance among two discrete labels (e.g. $\lambda_{\texttt{waiting}}(\texttt{NONE}, \texttt{NONE}) = 0$, $\lambda_{\texttt{waiting}}(\texttt{NONE}, \texttt{FEW}) = 1$, $\lambda_{\texttt{waiting}}(\texttt{NONE}, \texttt{A\_LOT}) = 2$). In fact, we regard discrete labels as an ordered set of equidistant values. Then, we define ς_i as an *inverse mapping* from labels's distance $[0..\lambda_i^{MAX}]$ to the $[0..1]$ interval: $\varsigma_i(a_i^{p_x}, a_i^{p_y}) = 1 - \frac{\lambda_i(a_i^{p_x}, a_i^{p_y})}{\lambda_i^{MAX}}$. In sum, in both similarity functions (σ, ς_i), a 0 means no coincidence at all and a 1 means that the items are equal.

From retrieved cases, the second CBR phase (*reuse*) employs their solutions to build a new one for the current case. In case there is more than one similar case, we count on a *divergence function* (δ) to compute the divergence among them —it is the standard deviation of `vFR` discrete values converted into integers, since in our experiments `vBW` was correlated with it. Thus, reuse phase starts by checking if the divergence of retrieved cases is greater than a *maximum trusted divergence* (`MAX_DIV`) —see line 2 in right side of Algorithm 1. In such a case, it considers that previous cases' solutions are too contradictory to provide a good single solution. Hence, the heuristic is used –lines 3-4. Once there is a set of slightly divergent previous cases –notice that a single previous case has no divergence– it adapts their solution to the current problem —line 5. This task can take into account (i) all retrieved solutions but also (ii) the differences between the retrieved problems and the current one. In current implementation, our *adapt* function uses only the former (i). In particular, it returns a solution composed by the most frequent `vFR` and the most frequent `vBW`. In case there is a tie, the less conservative actions (i.e. change values) have priority over the more conservative ones (i.e. keep the same values) —since they may make the system evolve in a different way and avoid a tie in a subsequent adaptation process.

Next, the third phase (*revise*) requires a way to evaluate the solution, but current performance measure (total time) is unknown until the end of execution —i.e. there is a credit assignment problem [13]. As we are working on this topic in the P2P scenario, current implementation has only the fourth phase (*retain*). It consists on storing only the new previous cases returned by the heuristic. This way, the case base grows every time the heuristic is used —when we implement the third phase, the system will revise its adapted solutions and retain them if they are representative enough. After each assistant computes a convenient update for norm parameters using CBR, all of them agree on their actual modification using a voting approach —in case there is a tie, parameters are not modified. Finally, each assistant sends to its domain-level agents the norms if they have been modified —in current implementation peers do not violate norms but they adapt their behaviour when receiving a new norm specification. As applying norm changes has an associated cost –e.g. cancelling some started data transmissions–, the norm adaptation process is performed at an empirically tested time interval ($\texttt{adapt}_{\texttt{interv}}$) specified in next section.

5 Empirical evaluation

In order to test our approach, we have implemented a P2P MAS simulator. This simulator is implemented in Repast Simphony [1] 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 initial conditions. Thus, we have performed several tests on BitTorrent and 2-LAMA approaches to empirically evaluate our proposal's performance. The evaluated approaches in this work are: a single-piece version of the standard BitTorrent protocol (BT, it is detailed in [8]), our architecture using always an heuristic to adapt norms (2L.a) and our approach using learning techniques (2L.b). In order to make a fair comparison –see [8]– among BT and 2-LAMA, we have used the following initial norm parameters: $\texttt{max}_{\texttt{BW}} = 100\%$, $\texttt{max}_{\texttt{FR}} = 3$. These norms are adapted at intervals of $\texttt{adapt}_{\texttt{interv}} = 50$ time steps. The learning approach, 2L.b, uses 0.8 as the minimum similarity threshold (i.e. MIN_SIM=0.8) and 1 as the maximum divergence threshold (i.e. MAX_DIV=1) —both values come from an empirical study. Notice that in this approach, assistants start with an empty case base and use the heuristic to generate an initial case. Later, if a problem is similar to previous ones, they reuse their knowledge instead of using the heuristic.

We have tested all three methods by varying the peer that initially has the datum —we call *round* to a single execution with the data in a certain initial position. In subsequent rounds, 2L.b's assistants already know some previous cases since case base is kept when sharing more data among the same agent community. This process is repeated until the data has been initially in all peers (*multiple-round*). Due to the random nature of the BT –some served peers are selected haphazardly–, the results show the average of executing a multiple-round 50 times (i.e. $12 \times 50 = 600$ rounds, where the 12 corresponds to all possible initial data positions in a round, and the 50 corresponds to repeat the unique multiple-round). In contrast, a multiple-round does not need to be repeated when using 2-LAMA methods, because they do not present random issues. However, as assistants in 2L.b learn at each round, the order of initial data positions influences this approach. Thus, 2L.b results show the average of executing this alternative on 50 random multiple-rounds (i.e. $12 \times 50 = 600$ rounds, where the 50 corresponds to different multiple-rounds with distinct order of 12 initial data positions). Table 1 shows the average per round of the following metrics: *time* as the total time required to spread the datum among all peers; *cNet* which is the network cost consumed by all messages (each message cost is computed as its length times the number of links it traverses); h as the average number of links traversed by each message (hops); *data* as the total number of sent data messages; and *cML* that is the cost of all messages related with the meta-level —i.e. all messages sent to or by assistants. Notice that the *data* metric refers to all data messages, although some of them may not be totally transmitted if: (i) a destination peer sends a cancel message to its source peer because it found a better source or (ii) a peer stops sending data to fulfil an updated $normFR_{DL}$.

If we compare the performance of both BT and 2-LAMA approaches, we see that our proposals require less *time* to share the datum. This means that it takes longer when there is no assistance despite the additional communication with the meta-level required by the assisted approach. In contrast, the *network cost* (*cNet*) is larger in 2-LAMA. This means that, in our approaches network is intensively used along the whole execution without achieving saturation —otherwise, time would increase. Our proposal requires more communication because: (i) it has extra communications due to the meta-level, (ii) it sends more data messages, and (iii) it initially measures latencies to adapt DL's social structure. Having a meta-level (i) implies that coordination messages are exchanged among domain-level agents and assistants and also between assistants. However, the derived network overload (*cML*) is small since these control messages are very small compared with data messages. On the contrary, (ii) having more data messages (*data*) consume a significant amount of network resources. These extra data messages are created because 2-LAMA peers compare data sources by retrieving some data from them —they replace their current data source whenever they find a faster one. Thus, we expect to minimise this network consumption when dealing with more than one piece of data, since peers could compare sources depending on previous retrieved pieces. Besides, latency measurements (iii) represent up to a 20% of the network cost increment. Notice, though, that these measures are used to improve system-wide data-paths —by suggesting certain neighbours to each domain-level agent. Regarding the number of links traversed by messages (h), our approaches have more local communications than BT. This is convenient because local messages have lower latencies and costs.

Overall, results show that our learning approach (2L.b) outperforms our heuristic approach (2L.a) and the BT one, since it requires less time. This means that our heuristic performs a good estimation of the mapping between system status, norms and outcomes, but it can be enhanced. In fact, our current CBR implementation is already improving this estimation.

6 Conclusions

Our vision is to endow the system with adaptation capabilities instead of expecting its agents to increase their behaviour complexity. Thus, we propose to add a meta-level that adapts a MAS organisation as a type of assistance to the coordination of its participants. Particularly, this paper applies CBR to perform such a task. It illustrates this approach in a P2P scenario, providing in-depth details about the adaptation of norms in this scenario. Moreover, it empirically compares this approach to the BitTorrent protocol —widely used in this scenario. As future work, we plan to go further in CBR methodology (e.g. evaluating solutions in a revise phase) and open MAS issues (e.g. entering/leaving agents).

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.

References

1. Repast Simphony. http://repast.sourceforge.net
2. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Commun. 7(1), 39–59 (1994)
3. Artikis, A., Kaponis, D., Pitt, J.: MAS: Semantics and Dynamics of Organisational Models, chap. Dynamic Specifications of Norm-Governed Systems (2009)
4. Boissier, O., Gâteau, B.: Normative multi-agent organizations: Modeling, support and control. In: Normative Multi-agent Systems (2007)
5. Bou, E., López-Sánchez, M., Rodríguez, J.A.: Autonomic Electronic Institutions' Self-Adaptation in Heterogeneous Agent Societies, vol. 5368 LNAI (2009)
6. Campos, J., López-Sánchez, M., Esteva, M.: Assistance layer, a step forward in Multi-Agent Systems Coordination Support. In: Autonomous Agents and Multiagent Systems. pp. 1301–1302 (2009)
7. Campos, J., López-Sánchez, M., Esteva, M.: Norm Adaptation using a Two-Level Multi-Agent System Architecture in a Peer-to-Peer Scenario. In: Proceedings of COIN at AAMAS'10 (to appear 2010)
8. Campos, J., López-Sánchez, M., Esteva, M., Novo, A., Morales, J.: 2-LAMA Architecture vs. BitTorrent Protocol in a Peer-to-Peer Scenario. In: Artificial Intelligence Research and Development - CCIA09. pp. 197–206. No. 202, IOS Press (2009)
9. Cranefield, B.S.S., Purvis, M., Purvis, M.: Role model based mechanism for norm emergence in artificial agent societies. LNCS 4870, 203 (2008)
10. Deloach, Oyenan, Matson: A capabilities-based model for adaptive organizations. Autonomous Agents and Multi-Agent Systems 16(1), 13–56 (2008)
11. Esteva, M.: Electronic Institutions: from specification to development. IIIA PhD, Vol. 19 (2003)
12. Grizard, A., Vercouter, L., Stratulat, T., Muller, G.: A peer-to-peer normative system to achieve social order. LNCS 4386, 274 (2007)
13. Jones, J., Goel, A.: Revisiting the Credit Assignment Problem. In: Challenges of Game AI: Proceedings of the AAAI. vol. 4, pp. 04–04 (2004)
14. Plaza, E., McGinty, L.: Distributed case-based reasoning. The Knowledge engineering review 20(03), 261–265 (2006)
15. Salazar-Ramirez, N., Rodríguez-Aguilar, J.A., Arcos, J.L.: An infection-based mechanism for self-adaptation in multi-agent complex networks. pp. 161–170 (2008)
16. Xie, H., Yang, Y.R., Krishnamurthy, A., Liu, Y., Silberschatz, A.: P4P: provider portal for applications (2008)