An Argumentation-based Framework for Deliberation in Multi-Agent Systems

Santi Ontañón[1] and Enric Plaza[2]

[1] CCL, Cognitive Computing Lab,
Georgia Institute of Technology
Atlanta, GA 303322/0280.
santi@cc.gatech.edu
[2] IIIA, Artificial Intelligence Research Institute,
CSIC, Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra, Catalonia (Spain).
enric@iiia.csic.es

Abstract. This paper focuses of the group judgments obtained from a committee of agents that use deliberation. The deliberative process is realized by an argumentation framework called AMAL. The AMAL framework is completely based on learning from examples: the argument preference relation, the argument generation policy, and the counterargument generation policy are case-based techniques. For join deliberation, learning agents share their experience by forming a committee to decide upon some joint decision. We experimentally show that the deliberation in committees of agents improves the accuracy of group judgments. We also show that a voting scheme based on assessing the confidence of arguments improves the accuracy of group judgments than majority voting.

1 Introduction

Argumentation frameworks for multi-agent systems can be used for different purposes like joint deliberation, persuasion, negotiation, and conflict resolution. In this paper we focus on committees of agents that use deliberation to achieve more informed and accurate group judgments. Since most work on multi-agents systems is oriented towards bargain-based decision-making (like negotiation or persuasion) it is important to remark the following difference: while bargain-based decision-making assumes that individual preferences are "given" (i.e. preferences preexisting and/or fixed), deliberation-based decision-making preferences are formed [15].

Argumentation-based joint deliberation involves discussion over the outcome of a particular situation or the appropriate course of action for a particular situation. Learning agents are capable of learning from experience, in the sense that past examples (situations and their outcomes) are used to predict the outcome for the situation at hand. However, since individual agents experience may be limited, individual knowledge and prediction accuracy is also limited. Thus, learning agents that are capable of arguing their individual predictions with other agents may reach better prediction accuracy after such an argumentation process.

Most existing argumentation frameworks for multi-agent systems are based on deductive logic or some other deductive logic formalism specifically designed to support argumentation, such as default logic [3]. Usually, an argument is seen as a logical statement, while a counterargument is an argument offered in opposition to another argument [4, 14]; agents use a preference relation to resolve conflicting arguments. However, logic-based argumentation frameworks assume agents with preexisting knowledge and preference relations. This is similar to the difference in assumptions between bargain-based decision-making and deliberation-based decision-making: our interest is in an adaptive and dynamic approach for deliberation processes where agents are responsive to external arguments or factual statements and, by integrating them, being able changing their minds.

In this paper, we focus on an *Argumentation-based Multi-Agent Learning* (AMAL) framework where both knowledge and preference relation are learned from experience. Thus, we consider a scenario with agents that (1) work in the same domain using a shared ontology, (2) are capable of learning from examples, and (3) communicate using an argumentative framework. Having learning capabilities allows agents effectively use a specific form of counterargument, namely the use of *counterexamples*. Counterexamples offer the possibility of agents learning *during* the argumentation process. Moreover, learning agents allow techniques that use learnt experience to generate adequate arguments and counterarguments. Specifically, we will need to address two issues: (1) how to define a technique to generate arguments and counterarguments by generalizing from examples, and (2) how to define a preference relation over two conflicting arguments that have been generalized from examples.

This paper presents a case-based approach to address both issues. The agents use case-based reasoning (CBR) [1] to learn from past cases (where a case is a situation and its outcome) in order to predict the outcome of a new situation. We propose an argumentation protocol inside the AMAL framework at supports agents in reaching a joint prediction over a specific situation or problem — moreover, the reasoning needed to support the argumentation process will also be based on cases. In particular, we present two *case-based measures*, one for generating the arguments and counterarguments adequate to a particular situation and another for determining preference relation among arguments. Finally, we experimentally show that the deliberation in committees of agents improves the accuracy of group judgments compared to voting without deliberation. We also show that a voting scheme based on assessing the confidence of arguments improves the accuracy of group judgments compared to majority voting.

The paper is structured as follows. Section 2 discusses the relation among committees, deliberation and social choice. Then Section 3 introduces our multi-agent CBR framework and the notion of justified prediction. After that, Section 4 formally defines our argumentation framework. Sections 5 and 6 present our case-based preference relation and argument generation policies respectively. Later, Section 7 presents the argumentation protocol in our AMAL framework. After that, Section 8 presents an exemplification of the argumentation framework. Finally, Section 9 presents an empirical evaluation of our apparoach. The paper closes with related work and conclusions sections.

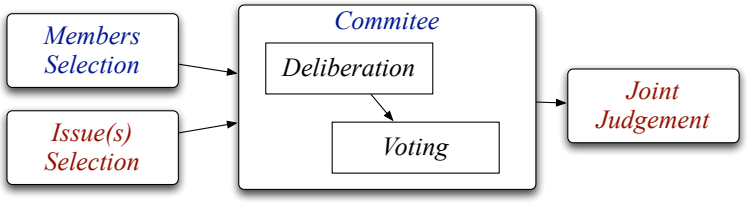

Fig. 1. The main aspects of a deliberative committees of agents..

2 Deliberation, Committees and Social Choice

While there is ample research work on multi-agent systems concerning teams (agents associated in some joint action) and coalitions (agents that temporarily combine their action for a specific purpose), this paper focuses on committees. A common definition of committee is "A group of people officially delegated (elected or appointed) to perform a function, such as investigating, considering, reporting, or acting on a matter." Considered as an institution, the committee is a widespread form of coordination, deliberation, and joint decision-making. Philip Pettit in *Republicanism* says that "the committee is the enzyme of the body politic" (page 239, [9]) because committees are ubiquitous and because of the importance of their proper functioning to sustain a reliable working of the whole body politic.

In our approach, a committee of agents is a form of electronic institution designed to perform *group judgments*. The issue of *group judgments*, following Cass Sunstein [16], is answering the following question: How can groups obtain or use the information that their members have? The author then studies three approaches: deliberation, statistical means, and information markets. In our previous work on committees of agents [11] we focused on statistical means, in the sense of using voting schemes as the aggregation function to achieve group judgments. These approaches are based on what is called the *ensemble effect* [11] in Machine Learning and the *Condorcet Jury Theorem* [16] in social choice theory — stating succinctly that the accuracy of the group judgement is higher than that of the best individual member when some properties are satisfied by the members and an adequate aggregation function (e.g. majority voting) is used.

Since human committees also employ deliberation, we focus on this paper on developing a framework for deliberative committees of agents. Figure 1 shows the main aspects of a committee of agents: a way to select the members of the committee, the selection of the issues to be addressed by that committee, a deliberation stage and (if a consensual agreement is not achieved) a voting stage. Notice that if the committee addresses not a single issue but several related issues the stages of deliberation and voting can be iterated. In this paper we focus on single-issue committees of agents and on the deliberation and voting stages. We offer no contribution to the problems of selecting relevant issues and member selection, focusing on the internal workings of a committee proposing an argumentation-based approach to deliberation and new confidence-based voting mechanism.

The argumentation-based framework assumes agents capable of learning — in particular in agents capable of reasoning with (and learning from) cases. This approach gives an empirical grounding to several important issues of argumentation frameworks, like generation and selection of arguments and counterarguments. In our approach, the agents using case-based reasoning (CBR) will argue based on what they have learnt, and they will accept or reject counterarguments posited by other agents based on what they have learnt. Finally, since deliberation is only useful if agents are capable of changing their mind as a result of their argumentation with others, learning offers a basis from which individual changes in judgment are integrated with (and based on) the acquisition of new information from communicating with other agents. The next section introduces CBR agents and the requirements for sustaining an argumentation framework.

3 Multi-Agent CBR Systems

A *Multi-Agent Case Based Reasoning System* ($\mathcal{M}$AC) $\mathcal{M} = \{(A_1, C_1), ..., (A_n, C_n)\}$ is a multi-agent system composed of $\mathcal{A} = \{A_i, ..., A_n\}$, a set of CBR agents, where each agent $A_i \in \mathcal{A}$ possesses an individual case base C_i. Each individual agent A_i in a $\mathcal{M}$AC is completely autonomous and each agent A_i has access only to its individual and private case base C_i. A case base $C_i = \{c_1, ..., c_m\}$ is a collection of cases. Agents in a $\mathcal{M}$AC system are able to individually solve problems, but they can also collaborate with other agents to solve problems.

In this framework, we will restrict ourselves to analytical tasks, i.e. tasks like classification, where the solution of a problem is achieved by selecting a solution class from an enumerated set of solution classes. In the following we will note the set of all the solution classes by $\mathcal{S} = \{S_1, ..., S_K\}$. Therefore, a *case* $c = \langle P, S \rangle$ is a tuple containing a case description P and a solution class $S \in \mathcal{S}$. In the following, we will use the terms *problem* and *case description* indistinctly. Moreover, we will use the dot notation to refer to elements inside a tuple; e.g., to refer to the solution class of a case c, we will write $c.S$.

Therefore, we say a group of agents perform *joint deliberation*, when they collaborate to find a joint solution by means of an argumentation process. However, in order to do so, an agent has to be able to *justify* its prediction to the other agents (i.e. generate an argument for its predicted solution that can be examined and critiqued by the other agents). The next section addresses this issue.

3.1 Justified Predictions

Both expert systems and CBR systems may have an explanation component [17] in charge of justifying why the system has provided a specific answer to the user. The line of reasoning of the system can then be examined by a human expert, thus increasing the reliability of the system.

Most of the existing work on explanation generation focuses on generating explanations to be provided to the user. However, in our approach we use explanations (or justifications) as a tool for improving communication and coordination among agents.

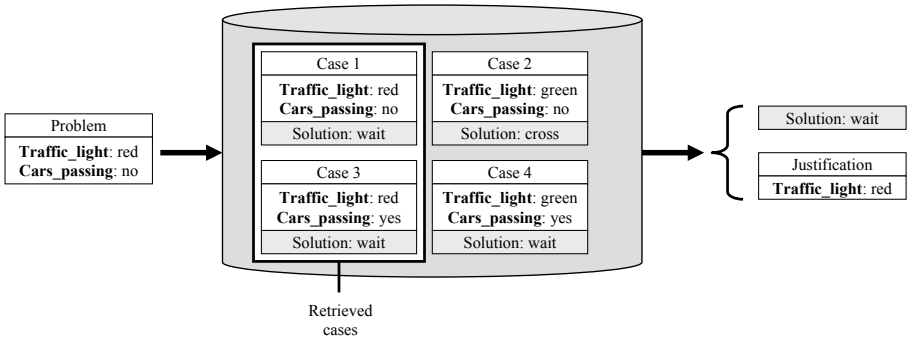

Fig. 2. An example of justification generation in a CBR system. Notice that, since the only relevant feature to decide is *Traffic_light* (the only one used to retrieve cases), it is the only one appearing in the justification.

We are interested in justifications since they can be used as arguments. For that purpose, we will benefit from the ability of some machine learning methods to provide justifications.

A *justification* built by a CBR method after determining that the solution of a particular problem P was S_k is a description that contains the relevant information from the problem P that the CBR method has considered to predict S_k as the solution of P. In particular, CBR methods work by retrieving similar cases to the problem at hand, and then reusing their solutions for the current problem, expecting that since the problem and the cases are similar, the solutions will also be similar. Thus, if a CBR method has retrieved a set of cases $C_1, ..., C_n$ to solve a particular problem P the justification built will contain the relevant information from the problem P that made the CBR system retrieve that particular set of cases, i.e. it will contain the relevant information that P and $C_1, ..., C_n$ have in common.

For example, Figure 2 shows a justification build by a CBR system for a toy problem (in the following sections we will show justifications for real problems). In the figure, a problem has two attributes (*Traffic_light*, and *Cars_passing*), the retrieval mechanism of the CBR system notices that by considering only the attribute *Traffic_light*, it can retrieve two cases that predict the same solution: *wait*. Thus, since only this attribute has been used, it is the only one appearing in the justification. The values of the rest of attributes are irrelevant, since whatever their value the solution class would have been the same.

In general, the meaning of a justification is that all (or most of) the cases in the case base of an agent that satisfy the justification (i.e. all the cases that are *subsumed* by the justification) belong to the predicted solution class. In the rest of the paper, we will use $\sqsubseteq$ to denote the subsumption relation. In our work, we use LID [2], a CBR method capable of building symbolic justifications such as the one exemplified in Figure 2. When an agent provides a justification for a prediction, the agent generates a *justified prediction*:

Definition 1. *A* Justified Prediction *is a tuple* $J = \langle A, P, S, D \rangle$ *where agent* A *considers* S *the correct solution for problem* P*, and that prediction is justified a symbolic description* D *such that* $J.D \sqsubseteq J.P$*.*

Justifications can have many uses for CBR systems [8, 10]. In this paper, we are going to use justifications as arguments, in order to allow learning agents to engage in argumentation processes.

4 Arguments and Counterarguments

For our purposes an *argument* α generated by an agent A is composed of a statement S and some evidence D supporting S as correct. In the remainder of this section we will see how this general definition of argument can be instantiated in specific kind of arguments that the agents can generate. In the context of $\mathcal{M}$AC systems, agents argue about predictions for new problems and can provide two kinds of information: a) specific cases $\langle P, S \rangle$, and b) justified predictions: $\langle A, P, S, D \rangle$. Using this information, we can define three types of arguments: justified predictions, counterarguments, and counterexamples.

A *justified prediction* α is generated by an agent A_i to argue that A_i believes that the correct solution for a given problem P is $\alpha.S$, and the evidence provided is the justification $\alpha.D$. In the example depicted in Figure 2, an agent A_i may generate the argument $\alpha = \langle A_i, P, \textit{Wait}, (\textit{Traffic\_light} = \textit{red}) \rangle$, meaning that the agent A_i believes that the correct solution for P is *Wait* because the attribute *Traffic_light* equals *red*.

A *counterargument* β is an argument offered in opposition to another argument α. In our framework, a counterargument consists of a justified prediction $\langle A_j, P, S', D' \rangle$ generated by an agent A_j with the intention to rebut an argument α generated by another agent A_i, that endorses a solution class S' different from that of $\alpha.S$ for the problem at hand and justifies this with a justification D'. In the example in Figure 2, if an agent generates the argument $\alpha = \langle A_i, P, \textit{Walk}, (\textit{Cars\_passing} = \textit{no}) \rangle$, an agent that thinks that the correct solution is *Wait* might answer with the counterargument $\beta = \langle A_j, P, \textit{Wait}, (\textit{Cars\_passing} = \textit{no} \wedge \textit{Traffic\_light} = \textit{red}) \rangle$, meaning that, although there are no cars passing, the traffic light is red, and the street cannot be crossed.

A *counterexample* c is a case that contradicts an argument α. Thus a counterexample is also a counterargument, one that states that a specific argument α is not always true, and the evidence provided is the case c. Specifically, for a case c to be a counterexample of an argument α, the following conditions have to be met: $\alpha.D \sqsubseteq c$ and $\alpha.S \neq c.S$, i.e. the case must satisfy the justification $\alpha.D$ and the solution of c must be different than the predicted by α.

By exchanging arguments and counterarguments (including counterexamples), agents can argue about the correct solution of a given problem, i.e. they can engage a joint deliberation process. However, in order to do so, they need a specific interaction protocol, a preference relation between contradicting arguments, and a decision policy to generate counterarguments (including counterexamples). In the following sections we will present these elements.

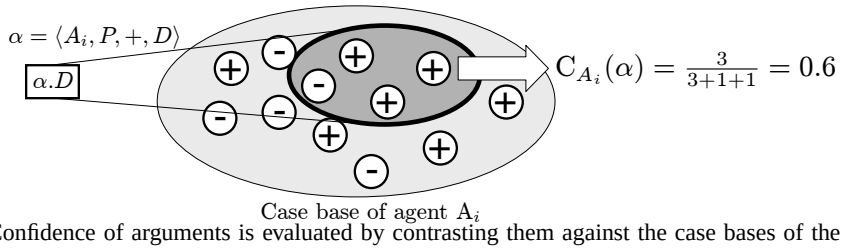

Fig. 3. Confidence of arguments is evaluated by contrasting them against the case bases of the agents.

5 Preference Relation

A specific argument provided by an agent might not be consistent with the information known to other agents (or even to some of the information known by the agent that has generated the justification due to noise in training data). For that reason, we are going to define a preference relation over contradicting justified predictions based on cases. Basically, we will define a *confidence* measure for each justified prediction (that takes into account the cases owned by each agent), and the justified prediction with the highest confidence will be the preferred one.

The idea behind case-based confidence is to count how many of the cases in an individual case base *endorse* a justified prediction, and how many of them are counterexamples of it. The more the endorsing cases, the higher the confidence; and the more the counterexamples, the lower the confidence. Specifically, to assess the confidence of a justified prediction α, an agent obtains the set of cases in its individual case base that are subsumed by $\alpha.D$. With them, an agent A_i obtains the Y (*aye*) and N (*nay*) values:

- $Y_\alpha^{A_i} = |\{c \in C_i |\ \alpha.D \sqsubseteq c.P \wedge \alpha.S = c.S\}|$ is the number of cases in the agent's case base *subsumed* by the justification $\alpha.D$ that belong to the solution class $\alpha.S$,
- $N_\alpha^{A_i} = |\{c \in C_i |\ \alpha.D \sqsubseteq c.P \wedge \alpha.S \neq c.S\}|$ is the number of cases in the agent's case base *subsumed* by justification $\alpha.D$ that do *not* belong to that solution class.

An agent estimates the confidence of an argument as:

$$\mathsf{C}_{A_i}(\alpha) = \frac{Y_\alpha^{A_i}}{1 + Y_\alpha^{A_i} + N_\alpha^{A_i}}$$

i.e. the confidence on a justified prediction is the number of endorsing cases divided by the number of endorsing cases plus counterexamples. Notice that we add 1 to the denominator, this is to avoid giving excessively high confidences to justified predictions whose confidence has been computed using a small number of cases. Notice that this correction follows the same idea than the Laplace correction to estimate probabilities. Figure 3 illustrates the individual evaluation of the confidence of an argument, in particular, three endorsing cases and one counterexample are found in the case base of agents A_i, giving an estimated confidence of 0.6

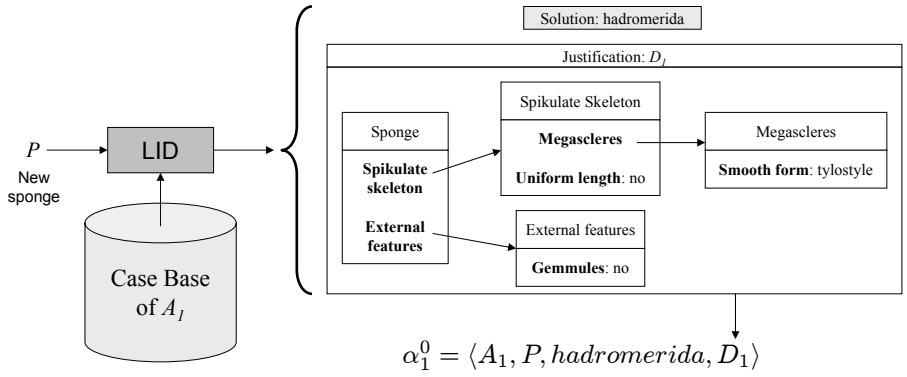

Fig. 4. Example of a real justification generated by LID in the marine Sponges data set.

Moreover, we can also define the *joint confidence* of an argument α as the confidence computed using the cases present in the case bases of all the agents in the group:

$$\mathsf{C}(\alpha) = \frac{\sum_i Y_\alpha^{A_i}}{1 + \sum_i \left(Y_\alpha^{A_i} + N_\alpha^{A_i} \right)}$$

Notice that, to collaboratively compute the joint confidence, the agents only have to make public the aye and nay values locally computed for a given argument.

In our framework, agents use this joint confidence as the preference relation: a justified prediction α is preferred over another one β if $C(\alpha) \geq C(\beta)$.

6 Generation of Arguments

In our framework, arguments are generated by the agents from cases, using learning methods. Any learning method able to provide a justified prediction can be used to generate arguments. For instance, decision trees and LID [2] are suitable learning methods. Specifically, in the experiments reported in this paper agents use LID. Thus, when an agent wants to generate an argument endorsing that a specific solution class is the correct solution for a problem P, it generates a justified prediction as explained in Section 3.1.

For instance, Figure 4 shows a real justification generated by LID after solving a problem P in the domain of marine Sponges identification. In particular, Figure 4 shows how when an agent receives a new problem to solve (in this case, a new sponge to determine its order), the agent uses LID to generate an argument (consisting on a justified prediction) using the cases in the case base of the agent. The justification shown in Figure 4 can be interpreted saying that "the predicted solution is hadromerida because the smooth form of the megascleres of the spiculate skeleton of the sponge is of type tylostyle, the spikulate skeleton of the sponge has no uniform length, and there is no gemmules in the external features of the sponge". Thus, the argument generated will be $\alpha = \langle A_1, P, hadromerida, D_1 \rangle$.

6.1 Generation of Counterarguments

As previously stated, agents may try to rebut arguments by generating counterargument or by finding counterexamples. Let us explain how they can be generated.

An agent A_i wants to generate a counterargument β to rebut an argument α when α is in contradiction with the local case base of A_i. Moreover, while generating such counterargument β, A_i expects that β is preferred over α. For that purpose, we will present a specific policy to generate counterarguments based on the *specificity* criterion [12].

The specificity criterion is widely used in deductive frameworks for argumentation, and states that between two conflicting arguments, the most specific should be preferred since it is, in principle, more informed. Thus, counterarguments generated based on the specificity criterion are expected to be preferable (since they are more informed) to the arguments they try to rebut. However, there is no guarantee that such counterarguments will always win, since, as we have stated in Section 5, agents in our framework use a preference relation based on joint confidence. Moreover, one may think that it would be better that the agents generate counterarguments based on the joint confidence preference relation; however it is not obvious how to generate counterarguments based on joint confidence in an efficient way, since collaboration is required in order to evaluate joint confidence. Thus, the agent generating the counterargument should constantly communicate with the other agents at each step of the induction algorithm used to generate counterarguments (presently one of our future research lines).

Thus, in our framework, when an agent wants to generate a counterargument β to an argument α, β has to be more specific than α (i.e. $\alpha.D \sqsubset \beta.D$).

The generation of counterarguments using the specificity criterion imposes some restrictions over the learning method, although LID or ID3 can be easily adapted for this task. For instance, LID is an algorithm that generates a description starting from scratch and heuristically adding features to that term. Thus, at every step, the description is made more specific than in the previous step, and the number of cases that are subsumed by that description is reduced. When the description covers only (or almost only) cases of a single solution class LID terminates and predicts that solution class. To generate a counterargument to an argument α LID just has to use as starting point the description $\alpha.D$ instead of starting from scratch. In this way, the justification provided by LID will always be subsumed by $\alpha.D$, and thus the resulting counterargument will be more specific than α. However, notice that LID may sometimes not be able to generate counterarguments, since LID may not be able to specialize the description $\alpha.D$ any further, or because the agent A_i has no case inC_i that is subsumed by $\alpha.D$. Figure 5 shows how an agent A_2 that disagreed with the argument shown in Figure 4, generates a counterargument using LID. Moreover, Figure 5 shows the generation of a counterargument β_2^1 for the argument α_1^0 (in Figure 4) that is a specialization of α_1^0.

Specifically, in our experiments, when an agent A_i wants to rebut an argument α, uses the following policy:

1. Agent A_i uses LID to try to find a counterargument β more specific than α; if found, β is sent to the other agent as a counterargument of α.
2. If not found, then A_i searches for a counterexample $c \in C_i$ of α. If a case c is found, then c is sent to the other agent as a counterexample of α.

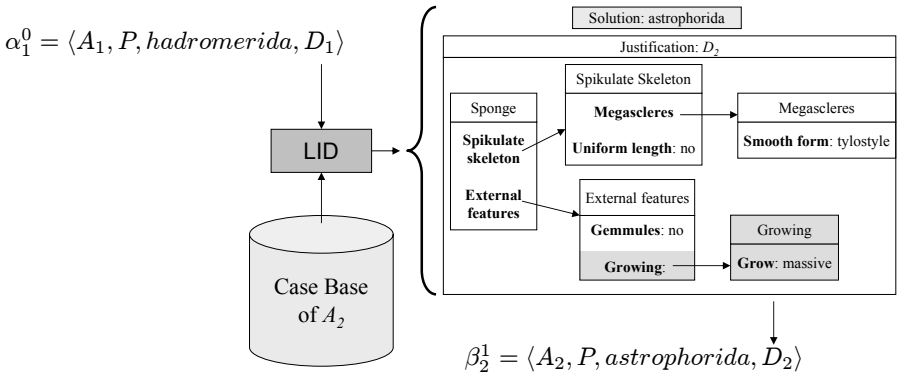

Fig. 5. Generation of a counterargument using LID in the Sponges data set.

3. If no counterexamples are found, then A_i cannot rebut the argument α.

7 Argumentation-based Multi-Agent Learning

The interaction protocol of AMAL allows a group of agents $A_1, ..., A_n$ to deliberate about the correct solution of a problem P by means of an argumentation process. If the argumentation process arrives to a consensual solution, the joint deliberation ends; otherwise a weighted vote is used to determine the joint solution. Moreover, AMAL also allows the agents to learn from the counterexamples received from other agents.

The AMAL protocol consists on a series of rounds. In the initial round, each agent states which is its individual prediction for P. Then, at each round an agent can try to rebut the prediction made by any of the other agents. The protocol uses a token passing mechanism so that agents (one at a time) can send counterarguments or counterexamples if they disagree with the prediction made by any other agent. Specifically, each agent is allowed to send one counterargument or counterexample each time he gets the token (notice that this restriction is just to simplify the protocol, and that it does not restrict the number of counterargument an agent can sent, since they can be delayed for subsequent rounds). When an agent receives a counterargument or counterexample, it informs the other agents if it accepts the counterargument (and changes its prediction) or not. Moreover, agents have also the opportunity to answer to counterarguments when they receive the token, by trying to generate a counterargument to the counterargument.

When all the agents have had the token once, the token returns to the first agent, and so on. If at any time in the protocol, all the agents agree or during the last n rounds no agent has generated any counterargument, the protocol ends. Moreover, if at the end of the argumentation the agents have not reached an agreement, then a voting mechanism that uses the confidence of each prediction as weights is used to decide the final solution (Thus, AMAL follows the same mechanism as human committees, first each individual member of a committee exposes his arguments and discuses those of the other mem-

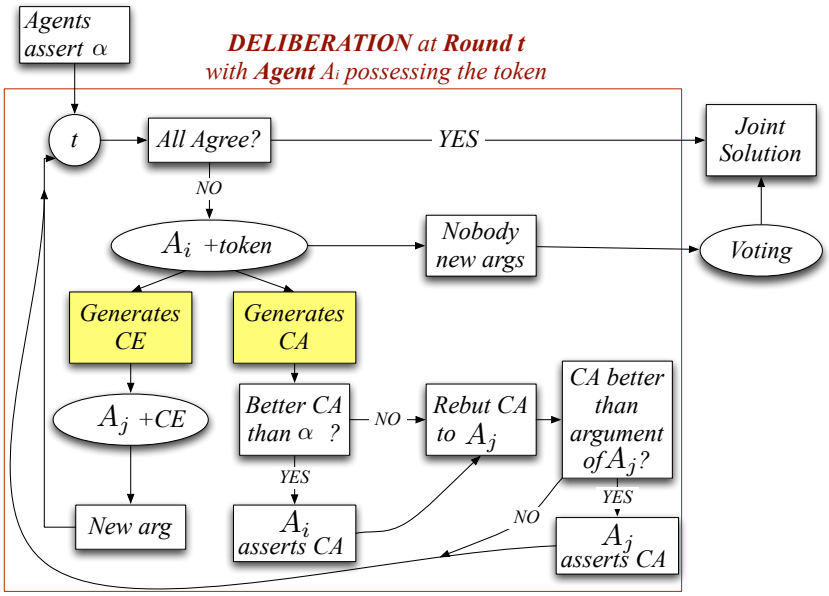

Fig. 6. Graphical representation of the argumentation protocol at round t with the token in possession of agent A_i.

bers (joint deliberation), and if no consensus is reached, then a voting mechanism is required).

At each iteration, agents can use the following performatives:

- $assert(\alpha)$: the justified prediction held during the next round will be α. An agent can only hold a single prediction at each round, thus is multiple asserts are send, only the last one is considered as the currently held prediction.
- $rebut(\beta, \alpha)$: the agent has found a counterargument β to the prediction α.

We will define $H_t = \langle \alpha_1^t, ..., \alpha_n^t \rangle$ as the predictions that each of the n agents hold at a round t. Moreover, we will also define $contradict(\alpha_i^t) = \{\alpha \in H_t | \alpha.S \neq \alpha_i^t.S\}$ as the set of contradicting arguments for an agent A_i in a round t, i.e. the set of arguments at round t that support a different solution class than α_i^t.

The protocol is initiated because one of the agents receives a problem P to be solved. After that, the agent informs all the other agents about the problem P to solve, and the protocol starts:

1. At round $t = 0$, each one of the agents individually solves P, and builds a justified prediction using its own CBR method. Then, each agent A_i sends the performative $assert(\alpha_i^0)$ to the other agents. Thus, the agents know $H_0 = \langle \alpha_i^0, ..., \alpha_n^0 \rangle$. Once all the predictions have been sent the token is given to the first agent A_1.
2. At each round t (other than 0), the agents check whether their arguments in H_t agree. If they do, the protocol moves to step 5. Moreover, if during the last n rounds

no agent has sent any counterexample or counterargument, the protocol also moves to step 5. Otherwise, the agent A_i owner of the token tries to generate a counterargument for each of the opposing arguments in $contradict(\alpha_i^t) \subseteq H_t$ (see Section 6.1). Then, the counterargument β_i^t against the prediction α_j^t with the lowest confidence $C(\alpha_j^t)$ is selected (since α_j^t is the prediction more likely to be successfully rebutted).

- If β_i^t is a counterargument, then, A_i locally compares α_i^t with β_i^t by assessing their confidence against its individual case base C_i (see Section 5) (notice that A_i is comparing its previous argument with the counterargument that A_i itself has just generated and that is about to send to A_j). If $C_{A_i}(\beta_i^t) > C_{A_i}(\alpha_i^t)$, then A_i considers that β_i^t is stronger than its previous argument, changes its argument to β_i^t by sending $assert(\beta_i^t)$ to the rest of the agents (the intuition behind this is that since a counterargument is also an argument, A_i checks if the newly counterargument is a better argument than the one he was previously holding) and $rebut(\beta_i^t, \alpha_j^t)$ to A_j. Otherwise (i.e. $C_{A_i}(\beta_i^t) \leq C_{A_i}(\alpha_i^t)$), A_i will send only $rebut(\beta_i^t, \alpha_j^t)$ to A_j. In any of the two situations the protocol moves to step 3.
- If β_i^t is a counterexample c, then A_i sends $rebut(c, \alpha_j^t)$ to A_j. The protocol moves to step 4.
- If A_i cannot generate any counterargument or counterexample, the token is sent to the next agent, a new round $t+1$ starts, and the protocol moves to state 2.

3. The agent A_j that has received the counterargument β_i^t, locally compares it against its own argument, α_j^t, by locally assessing their confidence. If $C_{A_j}(\beta_i^t) > C_{A_j}(\alpha_j^t)$, then A_j will accept the counterargument as stronger than its own argument, and it will send $assert(\beta_i^t)$ to the other agents. Otherwise (i.e. $C_{A_j}(\beta_i^t) \leq C_{A_j}(\alpha_j^t)$), A_j will not accept the counterargument, and will inform the other agents accordingly. Any of the two situations start a new round $t+1$, A_i sends the token to the next agent, and the protocol moves back to state 2.
4. The agent A_j that has received the counterexample c retains it into its case base and generates a new argument α_j^{t+1} that takes into account c, and informs the rest of the agents by sending $assert(\alpha_j^{t+1})$ to all of them. Then, A_i sends the token to the next agent, a new round $t+1$ starts, and the protocol moves back to step 2.
5. The protocol ends yielding a joint prediction, as follows: if the arguments in H_t agree then their prediction is the joint prediction, otherwise a voting mechanism is used to decide the joint prediction. The voting mechanism uses the joint confidence measure as the voting weights, as follows:

$$S = \arg\max_{S_k \in \mathcal{S}} \sum_{\alpha_i \in H_t | \alpha_i.S = S_k} C(\alpha_i)$$

Moreover, in order to avoid infinite iterations, if an agent sends twice the same argument or counterargument to the same agent, the message is not considered.

Figure 6 graphically illustrates this process. Where the greyed area is the loop formed by steps 2, 3, and 4.

8 Exemplification

Let us consider a system composed of three agents A_1, A_2 and A_3. One of the agents, A_1 receives a problem P to solve, and decides to use AMAL to solve it. For that reason, invites A_2 and A_3 to take part in the argumentation process. They accept the invitation, and the argumentation protocol starts.

Initially, each agent generates its individual prediction for P, and broadcasts it to the other agents. Thus, all of them can compute $H_0 = \langle \alpha_1^0, \alpha_2^0, \alpha_3^0 \rangle$. In particular, in this example:

- $\alpha_1^0 = \langle A_1, P, hadromerida, D_1 \rangle$
- $\alpha_2^0 = \langle A_2, P, astrophorida, D_2 \rangle$
- $\alpha_3^0 = \langle A_3, P, axinellida, D_3 \rangle$

A_1 starts (Round 0) owning the token and tries to generate counterarguments for α_2^0 and α_3^0, but does not succeed, however it has one counterexample c_{13} for α_3^0. Thus, A_1 sends the the message $rebut(c_{13}, \alpha_3^0)$ to A_3. A_3 incorporates c_{13} into its case base and tries to solve the problem P again, now taking c_{13} into consideration. A_3 comes up with the justified prediction $\alpha_3^1 = \langle A_3, P, hadromerida, D_4 \rangle$, and broadcasts it to the rest of the agents with the message $assert(\alpha_3^1)$. Thus, all of them know the new $H_1 = \langle \alpha_1^0, \alpha_2^0, \alpha_3^1 \rangle$.

Round 1 starts and A_2 gets the token. A_2 tries to generate counterarguments for α_1^0 and α_3^1 and only succeeds to generate a counterargument $\beta_2^1 = \langle A_2, P, astrophorida, D_5 \rangle$ against α_3^1. The counterargument is sent to A_3 with the message $rebut(\beta_2^1, \alpha_3^1)$. Agent A_3 receives the counterargument and assesses its local confidence. The result is that the individual confidence of the counterargument β_2^1 is lower than the local confidence of α_3^1. Therefore, A_3 does not accept the counterargument, and thus $H_2 = \langle \alpha_1^0, \alpha_2^0, \alpha_3^1 \rangle$.

Round 2 starts and A_3 gets the token. A_3 generates a counterargument $\beta_3^2 = \langle A_3, P, hadromerida, D_6 \rangle$ for α_2^0 and sends it to A_2 with the message $rebut(\beta_3^2, \alpha_2^0)$. Agent A_2 receives the counterargument and assesses its local confidence. The result is that the local confidence of the counterargument β_3^2 is higher than the local confidence of α_2^0. Therefore, A_2 accepts the counterargument and informs the rest of the agents with the message $assert(\beta_3^2)$. After that, $H_3 = \langle \alpha_1^0, \beta_3^2, \alpha_3^1 \rangle$.

At Round 3, since all the agents agree (all the justified predictions in H_3 predict $hadromerida$ as the solution class) The protocol ends, and A_1 (the agent that received the problem) considers $hadromerida$ as the joint solution for the problem P.

9 Experimental Evaluation

In this section we empirically evaluate the AMAL argumentation framework for deliberative committees. We have made experiments in two different data sets: *Soybean* (from the UCI machine learning repository) and *Sponges* (a relational data set). The Soybean data set has 307 examples and 19 solution classes, while the Sponges data set has 280 examples and 3 solution classes. In an experimental run, the data set is divided in 2 sets: the training set and the test set. The training set examples are distributed among

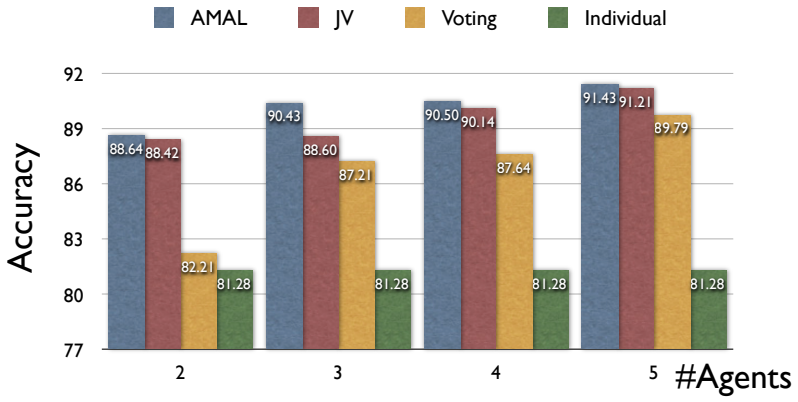

Fig. 7. Accuracy in Sponges data set for committees of 2 to 5 agents where predictions are achieved individually, by majority voting, by justification-based voting, and by the full AMAL argumentation framework.

5 different agents without replication, i.e. there is no example shared by two agents. In the testing stage, problems in the test set arrive randomly to one of the agents, and their goal is to predict the correct solution.

The experiments are designed to test the hypothesis that argumentation-based deliberation is useful for group judgment and improves over other typical methods such as majority voting. Moreover, we also expect that the improvement achieved from argumentation will increase as the number of agents participating in the argumentation increases (since more information will be taken into account). For this purpose, we ran four experiments, using committees of 2, 3, 4, and 5 agents respectively (in all experiments each agent has a 20% of the training data, since the training is always distributed among 5 agents).

Figures 7 and 8 show the result of those experiments in the Sponges and Soybean data sets. Classification accuracy is plotted in the vertical axis, and in the horizontal axis the number of agents that took part in the argumentation processes is shown. For each number of agents, four bars are shown: *Individual*, *Voting*, *JV*, and AMAL. The individual bar shows the average accuracy of individual agents predictions; the Voting bar shows the average accuracy of the agents using a majority voting system to aggregate their predictions (i.e. without deliberation); the last AMAL bar shows the average accuracy of the joint prediction using argumentation and (if need be) the confidence-based voting explained in the step 5 of the protocol. Therefore, since the AMAL framework has in fact to phases, namely deliberation and voting, then it is fair to ask how much contributes each phase to the final result. For this purpose, we have included the JV bar in Figures 7 and 8 that correspond to an experiment performed where the deliberation phase is skipped. More specifically, the agents simply present their justified predictions once (i.e. H_0 is generated) and then a confidence-based voting is performed immediately (i.e. without sending any counterargument or counterexample). The results shown are the average of 5 10-fold cross validation runs.

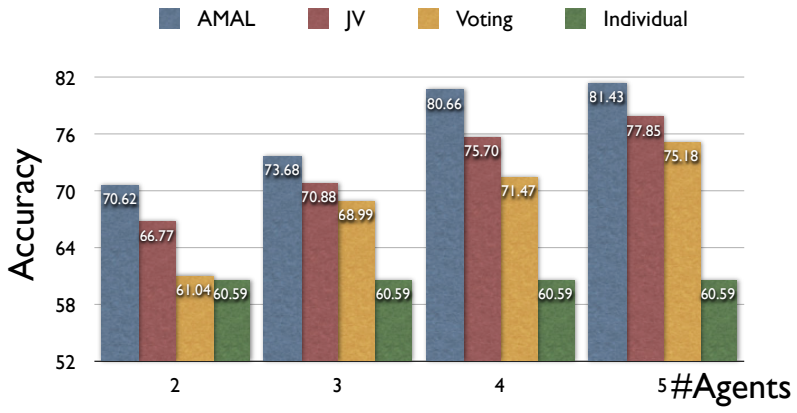

Fig. 8. Accuracy in Soybean data set for committees of 2 to 5 agents where predictions are achieved individually, by majority voting, by justification-based voting, and by the full AMAL argumentation framework.

Figures 7 and 8 show that collaboration (Voting, JV, and AMAL) outperforms individual problem solving. Moreover, as we expected, the accuracy improves as more agents collaborate, since more information is taken into account. Since AMAL always outperforms *Majority Voting*, it is clear that having deliberation is better than not having it. Moreover, AMAL always outperforms JV, indicating that having a confidence-based voting stage *after* the deliberation stage is better than skipping deliberation and use a confidence-based voting stage *before*. Thus, we conclude that joint predictions are based on better information that has been provided by the deliberation stage.

Moreover, Figures 7 and 8 show that the magnitude of the improvement obtained due to the argumentation process depends on the data set. For instance, the joint accuracy for 2 agents in the Sponges data set is of 88.64% for AMAL, 88.42% for JV, and 82.21% for majority voting (while individual accuracy is just 81.28%). Moreover, the improvement achieved by AMAL over voting is larger in the Soybean data set. The reason is that the Soybean data set is more "difficult" (in the sense that agents need a higher percentage of the data set to achieve a reasonably good accuracy level). These experimental results show that AMAL effectively exploits the opportunity for improvement: the accuracy is higher only because more agents have changed their prediction during argumentation (otherwise they would achieve the same result as Voting). For instance, the joint accuracy for 2 agents in the Soybean data set is of 70.62% for AMAL, 66.77% for JV, and 61.04% for majority voting (while individual accuracy is just 60.59%)

Figure 9 shows the frequency in which the agent committee was able to reach consensus or needed a final voting stage for committees with 2, 3, 4, and 5 agents in the Sponges and Soybean data set. The first bar (*Unanimity*) shows the percentage in which the agents predictions on Round 0 of the protocol are equal (and no deliberation in needed), the second bar (*Consensus*) shows the percentage in which all the agents agree on a joint prediction after deliberation, and the third bar (*Voting*) shows the remaining

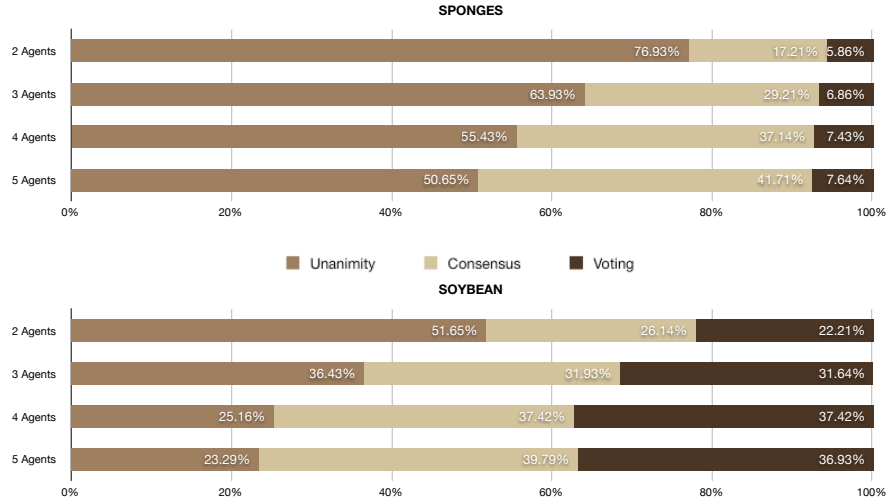

Fig. 9. Percentage of times that agents agree before the deliberation, after deliberation, and times when final voting is needed.

percentage in which the agents vote to determine the joint prediction. A first difference is between data sets: Soybean, being more "difficult" (average error is higher than in Sponges) has as expected higher disagreement and the percentage of times a vote is needed is higher than in the Sponges data set. We can also observe that larger committees have less unanimity (as expected), but since smaller committees also have larger errors, the additional deliberation and voting needed is also to be expected. Concerning deliberation, we see the committees can use the information exchanged using AMAL to reach a consensual solution in a fairly large number of occasions (more often in Sponges, since in Soybean the higher error rate makes consensus more difficult).

Table 1 shows the average number rounds of argumentation performed (and also the maximum number rounds in one deliberation) for committees of 2, 3, 4, and 5 agents. The difficulty of the Soybean data set is reflected in the higher number of argumentation rounds performed compared to the Sponges data set, as well as in the higher maximum number of rounds some deliberation stages achieve..

Let us analyze in more detail the difference in accuracy obtained by the AMAL argumentation process versus *Voting* and *JV* in the scenario with 5 agents. This improvement is only possible if agents change their mind about the correct prediction during

	Sponges				Soybean			
	2 Agents	*3 Agents*	*4 Agents*	*5 Agents*	*2 Agents*	*3 Agents*	*4 Agents*	*5 Agents*
Average rounds	*1.32*	*1.68*	*2.07*	*2.51*	*1.76*	*2.80*	*4.19*	*5.27*
Maximum rounds	*5*	*15*	*25*	*20*	*16*	*16*	*179*	*141*

Table 1. Average and maximum rounds of argumentation.

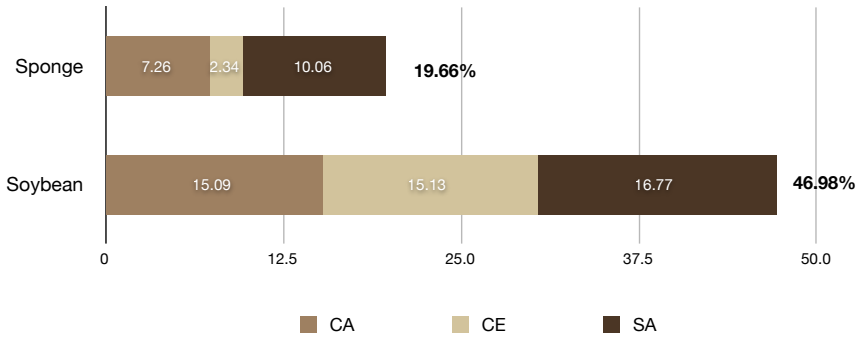

Fig. 10. Percentage of prediction change due to a counterargument (CA), a counterexample (CE), or to the finding a better argument (SA).

the deliberation phase due to the argumentation process. Figure 10 shows as percentage the average number of times that an agent changes its prediction during the deliberation phase, according too three possible mechanisms:

Counterargument (CA) when a counterargument received by the agent is accepted (since it has higher individual confidence than the previously held argument)

Counterexample (CE) when a counterexample is received and since added to the individual case base the agent finds that now another argument has higher individual confidence (this may be due to this single counterexample or to a number of previously received counterexamples in addition to this one)

Self-Argument (SA) when an agent changes its mind because, while trying to generate a counterargument for another agent, it explores a different region of the hypothesis space and finds an argument with higher individual confidence than the one currently holding.

In the Sponges data set, an agent changes its mind 19.66% of the times due to the argumentation process: 7.26% of the times due to the reception of a counterargument, 2.34% of the times due to the reception of a counterexample, and the remaining 10.06% is due to a self-argument. If we look at the same numbers in the Soybean data set, agents change their minds a 46.98% of the times due to argumentation: 15.09% of the times due to counterarguments, 15.13% of the times due to counterexamples, and the remaining 16.77% is due to a self-argument. Clearly, we can establish a relation between the number of times an agent changes its mind with the increase in classification accuracy. In the Sponges data set, agents change their minds less times, and thus the increase in accuracy (from Voting to AMAL) is lower, while in the Soybean data set they change their minds more often, and thus the increase (from Voting to AMAL) of accuracy is higher.

Moreover, we can further analyze these numbers. In the Sponges data set, agents receive a counterexample to their arguments a 35.23% of the times, but they only change their minds due to them a 2.34% of the times due to them (i.e. only 1 out of 15 counterexamples makes an agent change its mind). That means that the extra information

that an agent receives with one counterexample is small, and may need several counterexamples before effectively making the agent change its prediction. However, in the Soybean data set, agents receive a counterexample to their arguments a 31.49% of the times, and they change their minds due to them a 15.13% of the times (i.e. 1 out of 2 counterexamples makes an agent change its mind). Therefore, we can see that the amount of information that an agent receives with one counterexample is larger in the Soybean data set (in the sense that 1 or 2 examples may suffice to change an agent's prediction).

Finally, notice that an agent can change its mind due to only two different reasons: due to learning new information (i.e. learning new cases), or due to seeing the information it already had from a different point of view. When an agent changes its mind due to a counterexample, it is changing its mind due to learning new information. When an agent changes its mind due to a counterargument or due to a self-argument (searching through a new area in the generalizations space while trying to find a counterargument), the reason is that the agent sees its cases from a new point of view. The agents in our experiments use LID as their method to generate predictions that uses a heuristic method to explore the *generalizations space*. The heuristic approach avoids exploring the whole space of generalizations but does not assure that the part of the space effectively explored is always the one with the best generalization. Thus, from a generalizations space point of view, the argumentation process is useful for the agents in two respects: it provides new information (by means of counterexamples) an it provides new points of view to analyze data (i.e. it forces the agents to explore parts of the generalizations space that they have not explored following their heuristics).

10 Related Work

Our work on multi-agent case-based learning started in 1999 [6]; later Mc Ginty and Smyth [7] presented a multi-agent collaborative CBR approach (CCBR) for planning. Finally, another interesting approach is *multi-case-base reasoning* (MCBR) [5], that deals with distributed systems where there are several case bases available for the same task and addresses the problems of cross-case base adaptation. The main difference is that our $\mathcal{M}$AC approach is a way to distribute the *Reuse* process of CBR (using a voting system) while *Retrieve* is performed individually by each agent; the other multi-agent CBR approaches, however, focus on distributing the *Retrieve* process.

Research on MAS argumentation focus on several issues like a) logics, protocols and languages that support argumentation, b) argument selection and c) argument interpretation. Approaches for logic and languages that support argumentation include defeasible logic [4] and BDI models [14]. Although argument selection is a key aspect of automated argumentation (see [13] and [14]), most research has been focused on preference relations among arguments. In our framework we have addressed both argument selection and preference relations using a case-based approach.

11 Conclusions and Future Work

In this paper we have presented an argumentation-based framework for multiagent deliberation. Specifically, we have presented AMAL, a framework that allows a committee of agents to argue about the solution of a given problem and we have shown how the learning capabilities can be used to generate arguments and counterarguments. The experimental evaluation shows that the increased amount of information provided to the agents during the deliberation stage increases the predictive accuracy of group judgments, and specially when an adequate number of agents take part in the argumentation.

The main contributions of this work are: a) an argumentation framework for learning agents; b) a case-based preference relation over arguments, based on computing an overall confidence estimation of arguments; c) a case-based policy to generate counterarguments and select counterexamples, and d) a voting scheme based on assessing the confidence of arguments (instead of assessing the trust on agents). Although we introduced justification-based voting (JV) in a previous paper [8], the full capability of this approach has not been established until now. JV is as good as the arguments provided, and we show here that the arguments sustained by the agents are refined and improved during deliberation; thus JV is better used not as a technique per se (as proposed in [8]) but as the later stage of a deliberation process where the arguments have been challenged and improved.

Future work will focus on extending this approach from single-issue to multiple-issue deliberation and group judgment. Social choice theory calls this the task of aggregating sets of judgments, and there is an impossibility theorem similar to Arrow's. The problems arise from the fact that the interdependencies between different judgments may cause logical paradoxes (e.g. logical contradiction between the votes on the premises and the votes on the conclusion). However, relaxing some properties of aggregation strategies may be feasible for specific application purposes.

Acknowledgments. This research is partially supported by the MID-CBR (TIN2006-15140-C03-01) project and the Agreement Technologies (CONSOLIDER CSD2007-0022, INGENIO 2010) project.

References

1. A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communications*, 7(1):39–59, 1994.
2. E. Armengol and E. Plaza. Lazy induction of descriptions for relational case-based learning. In *ECML'2001*, pages 13–24, 2001.
3. Gerhard Brewka. Dynamic argument systems: A formal model of argumentation processes based on situation calculus. *Journal of Logic and Computation*, 11(2):257–282, 2001.
4. Carlos I. Chesñevar and Guillermo R. Simari. Formalizing Defeasible Argumentation using Labelled Deductive Systems. *Journal of Computer Science & Technology*, 1(4):18–33, 2000.
5. D. Leake and R. Sooriamurthi. Automatically selecting strategies for multi-case-base reasoning. In *ECCBR'2002*, pages 204–219. Springer Verlag, 2002.
6. Francisco J. Martín, Enric Plaza, and Josep-Lluis Arcos. Knowledge and experience reuse through communications among competent (peer) agents. *International Journal of Software Engineering and Knowledge Engineering*, 9(3):319–341, 1999.

7. Lorraine McGinty and Barry Smyth. Collaborative case-based reasoning: Applications in personalized route planning. In I. Watson and Q. Yang, editors, *ICCBR*, number 2080 in LNAI, pages 362–376. Springer-Verlag, 2001.
8. Santi Ontañón and Enric Plaza. Justification-based multiagent learning. In *ICML'2003*, pages 576–583. Morgan Kaufmann, 2003.
9. Philip Pettit. *Republicanism*. Oxford University Press, 1997.
10. Enric Plaza, Eva Armengol, and Santiago Ontañón. The explanatory power of symbolic similarity in case-based reasoning. *Artificial Intelligence Review*, 24(2):145–161, 2005.
11. Enric Plaza and Santiago Ontañón. Learning collaboration strategies for committees of learning agents. *Journal of Autonomous Agents and Multi-Agent Systems*, 13:429–461, 2006.
12. David Poole. On the comparison of theories: Preferring the most specific explanation. In *IJCAI-85*, pages 144–147, 1985.
13. K. Sycara S. Kraus and A. Evenchik. Reaching agreements through argumentation: a logical model and implementation. *Artificial Intelligence Journal*, 104:1–69, 1998.
14. N. R. Jennings S. Parsons, C. Sierra. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8:261–292, 1998.
15. Cass R. Sunstein. *The partial Constitution*. Harvard University Press, 1993.
16. Cass R. Sunstein. Group judgments: Deliberation, statistical means, and information markets. *New York University Law Review*, 80:962–1049, 2005.
17. Bruce A. Wooley. Explanation component of software systems. *ACM CrossRoads*, 5.1, 1998.