Argumentation-based agent support for learning policies in a coalition mission

†Chukwuemeka David Emele, *Timothy J. Norman,
and ‡Frank Guerin
Computing Science Department
University of Aberdeen, Aberdeen
United Kingdom. AB24 3UE
{†c.emele, *t.j.norman, ‡f.guerin}@abdn.ac.uk

Simon Parsons
Department of Computer and Information Science,
Brooklyn College
City University of New York
2900 Bedford Avenue, Brooklyn. 11210 NY
parsons@sci.brooklyn.cuny.edu

***Abstract*—Developing and resourcing coalition plans require an understanding of the policy and resource availability constraints under which coalition members operate. Policies and resource availability information are not necessarily public knowledge within the coalition. Also, it is difficult to keep track of the policies and resource availabilities of others. What is required is agent support for keeping track of who might have and be willing to provide the resources required for enacting a plan and modeling the policies of others regarding resource use, information provision, etc. We propose a technique that combines machine learning and argumentation for identifying and modeling the policies of others and advising on how a plan may be resourced using this model. Also, we present the results of the initial evaluation of this model.**

I. Introduction

In today's critical mission scenarios such as emergency response, coalitions are often formed because stakeholders may not have (or be willing to solely provide) all the resources, capabilities and logistics necessary for the mission. The formation of these alliances are predicated on members agreeing to collaborate and perform joint activities in a mutually acceptable fashion. Arguably, coalition members representing different organisations, nations, etc., share in the common goal of the coalition. Needless to say, they possess individual interests and constraints which they seek to satisfy as well. These individual interests and constraints largely determine the way and manner in which coalition partners carry-out the tasks assigned to them during the mission.

In this paper, we focus on policy and resource availability constraints of coalition members, and define policy constraints as explicit obligations, permission and prohibitions (sometimes called norms in some other contexts [1] [2]) that members of the coalition are required to adhere to. These policy constraints may be coalition-wide or individual. Coalition policy guidelines are global and are expected to be public knowledge within the coalition. On the other hand, individual policies are often private to that individual member or subset of the coalition. In order to develop effective plans, an understanding of the policy and resource availability constraints of other members in the coalition is required. Tracking and reasoning about such information is non-trivial.

We introduce a system that uses agent technology to keep track of who might have and be willing to provide the resources required for enacting a plan. The system also models the policies of other coalition members regarding resource use, information provision, etc., and evidence acquired with respect to their policies. This knowledge is used to learn behavioural patterns of others and thereby enhance future decisions about which coalition member is most appropriate for the provision of a resource given other constraints and prevailing circumstances.

In this paper, we propose a technique that combines machine learning and argumentation for identifying and modeling the policies of others and advising on how a plan may be resourced using this model. We present our work in the area of agent support for coalition missions under policy constraints. In this framework, we employ argumentation as a mechanism for teasing out vital information from partners and using that to aid the learning of underlying constraints (e.g. policies). We describe an experimental framework and present initial results of our evaluation showing that argumentation-based mechanisms combined with a standard machine learning technique out-performs the machine learning technique on its own in the learning of policy constraints in a coalition.

The remainder of this paper is organised as follows: In Section II we introduce the problem domain. In Section III we briefly describe our simulation environment. We present the learning mechanism in Section IV and Section V discusses argumentation. In Section VI we present the initial results of the experiments and discuss related work in Section VII. We outline future directions in Section VIII and Section IX concludes.

II. Problem Domain

Problem solving activities often require the interplay of many units, logistics and expertise, and are rarely carried out in isolation. Many nations and organizations are motivated to form coalitions as the basis for future operations (be it peace-keeping, humanitarian relief, disaster response, warfare, or otherwise). Such alliances are heterogeneous in nature and introduce various constraints to the theatre of operations. Similar constraints are evident in many team problem solving activities. These constraints introduce complications in the process of planning and coordinating joint activities. One

such example in team-based problem solving contexts is constraints due to the policies of individual team members. Team members may also be constrained in terms of resource availability, but these constraints tend to be shorter term; policy constraints being medium to long term. Some examples of policy constraints are given below:

1) A coalition partner may be forbidden from releasing a specific type of resource to some other partner but may be permitted to release the same resource to a third if requested.
2) A coalition partner may be permitted to share intelligence with another but might be prohibited from revealing the source of that intelligence.
3) A coalition partner P may be forbidden from flying unmanned aerial vehicles (UAVs) in bad weather conditions. Therefore, if intelligence reveals that the weather is bad then P is not allowed to include UAVs as resources to be used in the plan.

The task of planning for joint action in coalition operations is a complex problem on its own and could be further complicated by the various constraints that partners in the coalition may have, the prevailing circumstances and the goals to be achieved. In such scenarios, human planners are faced with huge challenges and are usually over-loaded with various details, some of which are important and may be critical to the success or failure of the operation. In this paper, we focus on policy and resource availability constraints. Our conjecture is that machine learning techniques may be employed to aid human decision making. Although this is not a new claim (see [3]), it is novel to combine it with argumentation analysis.

What is argumentation? We define argumentation as the process whereby arguments are exchanged and evaluated in the light of their interactions with other arguments. By arguments, we refer to explanations offered in support of an action. Consider the following snippet of dialogue that may occur between two agents i and j:

Example 1:
i: Can I have R1?
j: No.

What can be inferred from the interaction? Why did agent j say no to agent i's request?

1) Could it be that there exist some policy X that forbids agent j from providing R1 to agent i?
2) Could it be that R1 is not available at the moment?

There is very little that we can learn from the dialogue. On the other hand, suppose we have an argumentation framework that allows agents to ask for and provide explanations as in examples 2 and 3 below then agent i can gather more evidence regarding why agent j did not provide R1.

Example 2:	Example 3:
i: Can I have R1?	i: Can I have R1?
j: No	j: No.
i: Why?	i: Why?
j: I'm not permitted to release R1.	j: R1 is not available.

From the above snippets, we see that example 1 is not very helpful in terms of learning the underlying constraints of agent j. However, employing argumentation mechanisms (see examples 2 and 3), agent i could disambiguate the reason for the refusal. This is a very simple example but in richer dialogical contexts further evidence may be acquired through, for example, suggestions of alternative resources to use for the same goal.

Integrating learning techniques into the agent support framework will provide a level of support to human planners. However, can the use of argumentation mechanisms improve the effectiveness and accuracy of the information learned about the policy constraints of others? We hypothesize that argumentation can be used to disambiguate between policy and resource availability constraints.

Generally, we envisage a framework whereby software agents aid human decision makers, possibly distributed in space, to communicate, collaborate and coordinate their activities during joint operations. This system of support for human teams is not only interesting but realistic and can be investigated in real world team-based activities like warfare. One area of agent support that has been identified as important in this context is guidance in making policy-compliant decisions [4]. This prior research focuses on giving guidance to humans regarding their own policies. An important and open question, however, is how can agents support human decision makers in developing models of others' policies and using these in guiding the decision maker?

In this paper, we present a system where agents learn from practical dialogue by automatically extracting useful information from the dialogue and using these to model the policies, preferences and priorities of others in order to adapt their behaviour in the future. We, therefore, propose a technique that combines machine learning and argumentation for identifying and modeling the policies of others and advising on how a plan may be resourced taking resource availability constraints into consideration. We describe an experimental framework and present initial results of our evaluation which shows that an argumentation-based mechanism combined with a standard machine learning technique out-performs the machine learning technique on its own.

Fig. 1 presents an overview of the proposed system.

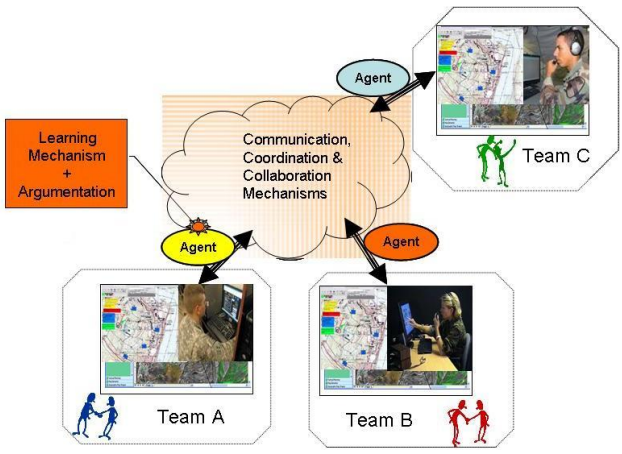

Fig. 1. General overview of the framework.

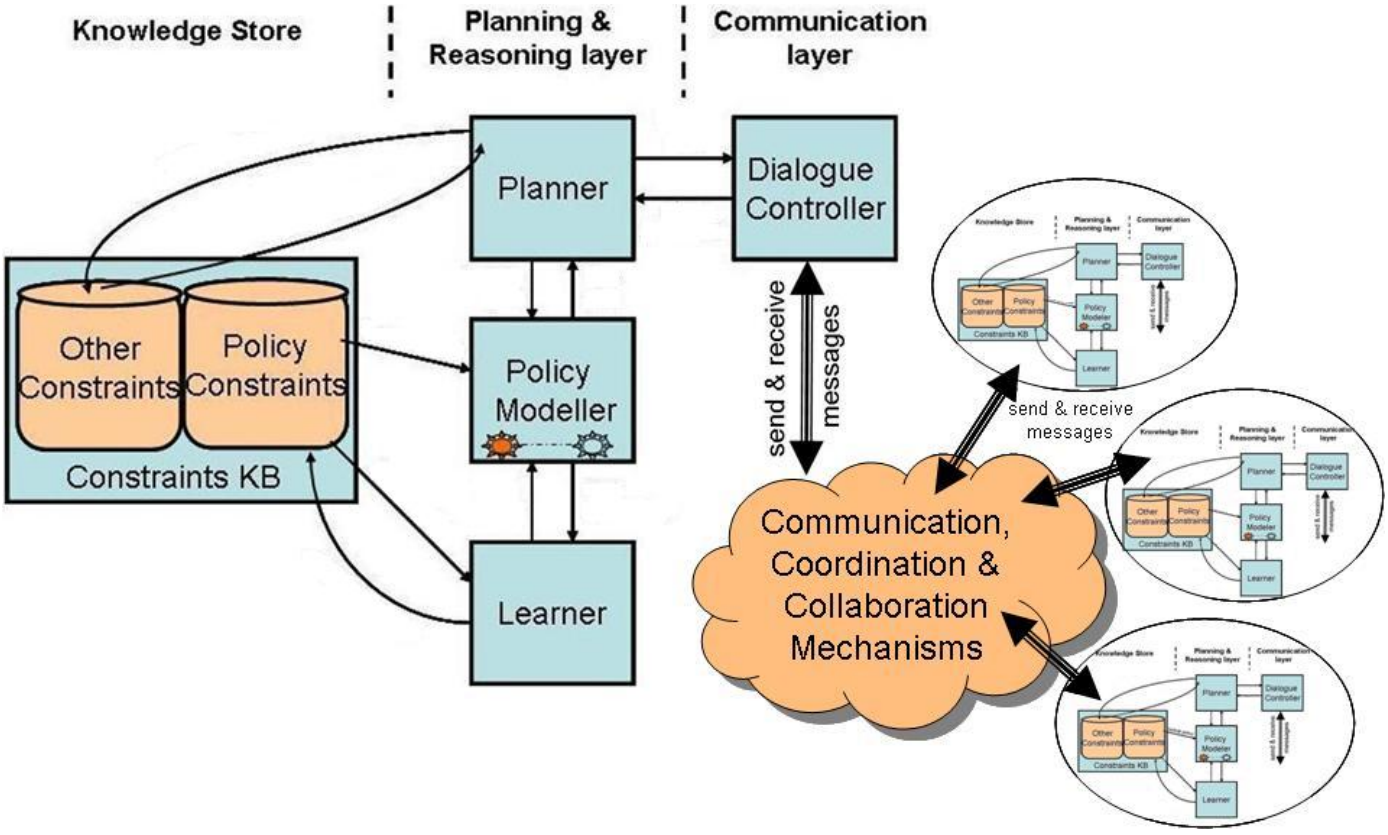

Fig. 2. Architecture of the framework for learning policy constraints in team-based activities using dialogue.

III. Simulation Environment

Each agent has two main layers, the communication layer and the planning and reasoning layer (See Fig. 2). The communication layer embodies the dialogue controller, which handles the communication with other agents in the domain. The planning and reasoning layer consists of three modules: the planner, the policy modeller, and the learner. The planning module of the agent uses heuristics (not covered in this paper) to generate a plan that is consistent with the individual policies of the agent (we assume this plan is a sub-plan of the joint plan constructed collaboratively by the coalition). This assumption is reasonable because planning is often done in hierarchies, therefore, higher level plans need to be refined (or replanned) at the lower level. For the sake of brevity, the term plan will be used to mean a complete plan, partial plan, and/or a plan step. Once a plan of action is constructed, the agent is ready to communicate with other agents in the domain to identify and gain commitments for the resources required to execute the plan. The dialogue controller module sends (and receives) messages to (and from) other agents and reasons over the dialogue. The policy modeller looks up policy constraints from the knowledge-base and generates appropriate utterance (or action) for the agent. Policy constraints are stored in the policy constraint knowledge-base while other (non-policy) constraints (e.g. resource constraints) are captured in the "other constraints" knowledge-base. The learner uses decision trees to learn policies based on the perceived actions of other agents (See Section IV for details). The knowledge from this experience is used to update the constraint knowledge base so that future planning will take that knowledge into account.

To test our hypothesis, we have developed a simulation environment for agent support in coalition missions and integrated learning mechanism and argumentation into the framework. The environment is implemented in Java [5]. The policies were captured as rules and implemented in a production engine (Jess [6]). The application programming interface provided in Weka [7] was used to integrate a standard machine learning algorithm into the framework. We note that although decision trees were used, the simulation environment is configured such that other machine learning algorithms can be plugged in.

Agents in this domain could play the role of a seeker, provider, or both in different interactions. For simplicity, we consider a setup with one seeker and a number of providers. The seeking agent simulates an agent that has come up with some plans to execute a task (the planning mechanism of the agent is beyond the scope of this paper) and needs to find the best way (or best partner to collaborate with in order) to resource these plans. The plans are resourced by convincing a providing agent to release some resources from its resource pool. The seeking agent identifies the resources required for a task and sends a request message to the providing agent. The providing agent evaluates the request and responds accordingly. If the resource is available for use and doing so does not conflict with policies then the providing agent agrees, otherwise it refuses. By availability we mean, the resource is not committed to another task (or agent) at the time requested

for and also the resource is in a usable state.

Suppose a seeker *A* sends a request for resource *R* to provider *P*, then we can represent the decision function of the provider generally as follows:

IF	(is_available(*R*) ∧ NOT (forbidden(release(*R, A*)))
THEN	agree(release(*R, A*))
ELSE	refuse(release(*R, A*))

Fig. 3. A simple decision function

We note that in reality there are a number of factors, other than policies and the availability of resources, that might be vital in determining whether a resource is made available or not. However, as a first step towards investigating these factors we concentrate on policies and resource availabilities, and hope to explore other factors like the cost of lending a resource, the risk of losing the resource, etc., subsequently.

IV. Learning Mechanism

Despite the recent progress in advanced induction algorithms such as Support Vector Machines [8], decision trees are still considered attractive for many real-life applications. Perhaps, the attraction stems from the fact that Decision Tree Classifiers have the ability to break down a complex decision-making process into a collection of simpler decisions, thus providing a solution which is often easier to interpret [9]. In terms of accuracy, decision trees have been shown to be competitive with other classifiers for several learning tasks [10]. By Occam's Razor [11], smaller trees should generally have better predictive power. The C4.5 algorithm uses gain ratio as a heuristic for predicting which attribute will yield a smaller tree. However, in general, the problem of finding the smallest consistent tree[1] is known to be NP-complete [12], [13].

Our multi-agent learning framework is based on the decision tree learner, which applies the C4.5 algorithm [14] on a given set of examples and generates a decision tree model. C4.5 builds decision trees from a set of training data, using the concept of information entropy [15] (beyond the scope of this paper). The training data is a set $S = s_1, s_2, ..., s_n$ of already classified samples. Each sample $s_i = x_1, x_2, ..., x_m$ is a vector where $x_1, x_2, ..., x_m$ represent attributes of the sample. The training data is augmented with a vector $C = c_1, c_2, ..., c_n$ where $c_1, c_2, ..., c_n$ represent the class to which each sample belongs. Agent policies are represented as a vector of attributes (e.g. resource, purpose, location, etc.) and the C4.5 algorithm is used to classify each policy instance into a class.

The C4.5 algorithm has three base cases.

- All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class.
- None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using the expected value of the class.
- Instance of previously-unseen class encountered. Again, C4.5 creates a decision node higher up the tree using the expected value.

In pseudocode the C4.5 algorithm is presented in Fig. 4:

Step 1.	Check for base cases
Step 2.	For each attribute *D,* Find the normalized information gain from splitting on *D*
Step 3.	Let *D_best* be the attribute with the highest normalized information gain
Step 4.	Create a decision node that splits on *D_best*
Step 5.	Recurse on the sublists obtained by splitting on *D_best* , and add those nodes as children of node

Fig. 4. The C4.5 algorithm for building decision trees [16].

In other words, at each node of the tree, the C4.5 algorithm chooses one attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. Its criterion is the normalized information gain (that is, difference in entropy [15]) that results from choosing an attribute for splitting the data. The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurses on the smaller sublists.

V. Argumentation

Resource-bounded reasoning situations, such as planning, are complex to model and this become even more complicated when the reasoning mechanism of the other agent is partially known, or in some cases not known at all. Agent reasoning mechanisms are usually private, therefore, it is non-trivial to model. One way that one could attempt to model it is by obtaining additional evidence (through explanations) to indicate what constraints others may be operating with. We adopt this approach and utilise the power of arguments (in this case, explanations/justifications) in aiding the learning process. In circumstances where knowledge is incomplete or imperfect, argumentation has proven to be effective in reaching some goals that would have otherwise, been unreachable [17], [18]. From the results, we can show that embedding an argumentation layer into the framework made a significant improvement on the performance of our agents in learning the policies of others. (See Section VI for experimental results)

Many argumentation-based studies have involved the use of some form of game built using language[19], rules [20] and conventions [21], [22] such that the actions of the players were guided and informed by structures often referred to as a protocol. Dialogue games have proven extremely useful for modeling various forms of reasoning in many domains, for example, in the legal domain [23], [24], medicine [25], [26], and many more. In this work we develop a dialogue game that involves two players only. The players take turns to play and a move in the game involves making an utterance (e.g request for some resource) and could also be modified to include an action (e.g allocate a resource to an agent).

In order to specify the game protocol, we need to define some terms. Let $\mathcal{A}$ be the set of agents in the domain such

[1]A consistent decision tree is a tree that correctly classifies all training examples.

that $i, j \in \mathcal{A}$. Assume agent i has a plan (a subset of the joint plan) requiring the use of a set of resources $\mathcal{R}$ in order to achieve a goal G.

Definition: ***A resource allocation****, denoted as Λ_i^t is a collection of resources that an agent i has at its disposal at time t, where t denotes the time step in the dialogue.*

- *$\Lambda_i^t \subseteq \mathcal{R}$, and t = {0, 1, 2, ...n}*

The game starts with an agent, say i sending a request to another agent, say j for the use of some resources needed to fulfill a plan. The other agent can then respond with an agree or refuse (based on certain issues, e.g policy constraints). The requesting agent could ask for reasons and explanations, and so on until the game ends.

Fig. 5 captures the protocol for the dialogue game developed in this work and Fig. 6 shows an example of the kind of dialogue that may occur between two agents, i and j using the protocol. However, we note that although this is presented as a dialogue between two agents, in reality the initiator (agent i, the agent that wishes to resource its plan) may engage in multiple instances of this dialogue with other agents. The dialogue in Fig. 6 is an instantiation of example 3 given earlier in Section II.

VI. Experiments and Results

A variety of experiments were conducted to test the performance and behavior of our framework. In this section, we describe our experimental scenario and present the results.

A. Experimental scenario

We present an illustrative scenario that will serve as a vehicle for testing our hypotheses. The scenario involves two software agents collaborating to complete a mission in the same region over a period of three days. The region of the mission is divided into five zones. There are five resource types and five purposes that a resource could be used to fulfill. A task involves the seeker identifying resource needs for a plan and collaborating with the provider to see how that plan can be resourced.

For the purpose of the experiment, the seeker simulates an agent that has a plan and needs to collaborate with the provider to resource it. The seeker predicts (based on the model of the provider) whether the provider has a policy that forbids/permits the provision of such a resource in that context. The seeker requests the resource from the provider and the provider uses a simple decision function (described earlier) to decide whether to grant or deny the request. The dialogue follows the protocol specified in Fig. 5 and at the end of the interaction the outcome is learned by the seeker and the model of the provider is updated accordingly.

1) Policies: The providing agents operate under set of policies which govern how resources are deployed to others (See Fig. 7 for an example). In other words, the providing agent can make resources available to other agents if the resources are available and there is no policy forbidding that course of action. The policies in this framework are based on a number of factors enumerated as follows:

- Organisation - refers the country/organisation of the requesting agent. In this framework, an agent is associated

Assume agent i has a plan (a subset of the joint plan) requiring the use of a set of resources $\mathcal{R}$ in order to achieve a goal G.

Agent i starts with initial allocation Λ_i^0 at time $t = 0$.
At time $t > 0$:
1. **request**(i, j, r): agent i requests agent j to provide resource r such that $\Lambda_i^t = \Lambda_i^{t-1} \cup \{r\}$
where $r \in \mathcal{R}$ and has not been requested of j before.
2. At the next time step ($t' = t + 1$), agent j either:
   (a). **agree**(j, i, r): agrees, and resource r is allocated to i.
   (b). **refuse**(j, i, r): refuses, and r is not allocated to i,
   in which case $\Lambda_i^{t'} = \Lambda_i^{t'-1}$.
3. At the next time step ($t'' = t' + 1$),
   if last received locution was **agree**(j, i, r) **then** agent i records the evidence in its knowledgebase, and moves to step 6.
   otherwise (switches to argumentation-based dialogue.)
   why(i, j, refuse(r)): i asks j for underlying interests or reasons why it has refused to provide resource r.
4. At the next time step ($t^{\diamond} = t'' + 1$),
   if last received locution was **why**(i, j, r) **then**
   agent j either:
   (a). **inform**(j, i, r, reason(x)): gives the reason for refusing to allocate r to i; or
   (b). **inform**(j, i, r, cant-tell): gives no reason for refusing to allocate r to i.
   otherwise inform(j, i, r, not-allowed): informs i that the message is invalid in this context.
5. At the next time step ($t^{\bullet} = t^{\diamond} + 1$),
   if last received locution was **inform**(j, i, r, reason(x)) **then** agent i records evidence in its knowledgebase, and moves to step 6.
   otherwise moves to step 6.
6. At the next time step ($t^{\star} = t^{\bullet} + 1$),
   if there are resources in $\mathcal{R}$ that are yet to be requested **then** move to step 1 with the current allocation, $\Lambda^{t^{\star}}$.
   otherwise
   close-dialogue(i, j): terminates the dialogue.

Fig. 5. Dialogue Game Protocol

i:	**request**(i, j, Jeep)
j:	**refuse**(j, i, Jeep)
i:	**why**(i, j, refuse (Jeep))
j:	**inform**(j, i, reason(resource-unavailable))
i:	**close-dialogue**(i, j)

Fig. 6. Simple dialogue between agents i and j

with the organisation it represents. Therefore, the policies that relate to that organisation are enacted whenever that agent makes a request for resources from the providing agent.

- Resources - generally denote physical equipment, capabilities or information that are required to carry out a task. The providing agent may be prohibited from deploying a missile or lending a UAV.
- Purpose - indicates the purpose for which the resource is being requested. For example, the providing agent may be obliged to release any resource to a member of the coalition if the resource is required for reconnaissance.
- Location - denotes the particular location or zone where the resource is to be deployed.
- Day - refers to the day the resource is to be deployed. This could be Day1, Day2 or Day3.

You are permitted to release resource R to team member X if his affiliation is O and the resource is to be deployed at location L for purpose P on day D.

Fig. 7. An example of a policy

2) Setup: In the experiment, resource seekers attempt to acquire resource types needed to fulfill the tasks assigned to them during the joint mission. The resource providers operate under strict policies, which are randomly generated at the beginning of each round, and these guide how resources may be made available to seeking agents upon request. Three agent support configurations were tested and the performance of the seeking agent was evaluated. The configurations include:

i. Random Selection (RS): Here, the seeker does not employ any machine learning nor argumentation technique, rather it randomizes its choice of attributing the refusal to policy or resource availability constraints.
ii. Learning without Argumentation (LOA): In this setup, the seeker applies the C4.5 decision tree learner to learn the provider's policy. This setup does not use argumentation in any way.
iii. Learning with Argumentation (LWA): Here, argumentation is employed as a mechanism to augment the C4.5 learner in learning the policy of the provider. In other words, dialogue is used to gather additional evidence that serves to improve the quality of the models learned by disambiguating between underlying constraints that may have similar observable actions.

Ten rounds of the experiment was conducted, each consisting of six runs of one hundred tasks executed by the agents. Each task involves the seeking agent identifying resource needs for a plan (or sub-plan) and collaborating with the provider to see how that plan can be resourced.

B. Results

This section presents the results of the experiments carried out to evaluate this work.

Fig. 8 illustrates the effectiveness of learning policies in the three configurations outlined earlier. These include: (1) random selection (RS), (2) standard machine learning approach only (LOA), and (3) combining machine learning with argumentation (LWA). It shows the percentage of the policies that the seeker predicted correctly in each configuration. The graph also shows that the argumentation-based approach enabled the agent to learn and build a more accurate model of the other agent's policies and thereby increased the accuracy of predictions. It is easy to see that the argumentation-based approach constantly out-performs the standard learning approach.

TABLE I
AVERAGE PERCENTAGE OF POLICIES CLASSIFIED CORRECTLY AT THE END OF THE 10 ROUNDS OF 6 RUNS EACH

Tasks	RS	LOA	LWA
1000	50.0 ± 1.8	57.3 ± 11.7	57.3 ± 11.7
2000	50.0 ± 1.4	68.0 ± 9.4	70.5 ± 10.3
3000	49.9 ± 0.8	73.7 ± 4.8	77.4 ± 6.1
4000	50.1 ± 1.7	74.5 ± 4.9	80.6 ± 5.0
5000	50.0 ± 1.6	73.9 ± 7.0	84.3 ± 3.5
6000	49.9 ± 2.0	67.9 ± 5.0	84.3 ± 4.7

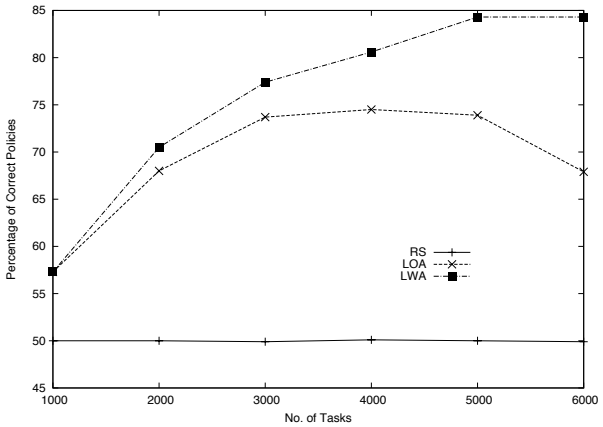

Fig. 8. Graph showing the effectiveness of learning policies using the three configurations (RS, LOA & LWA).

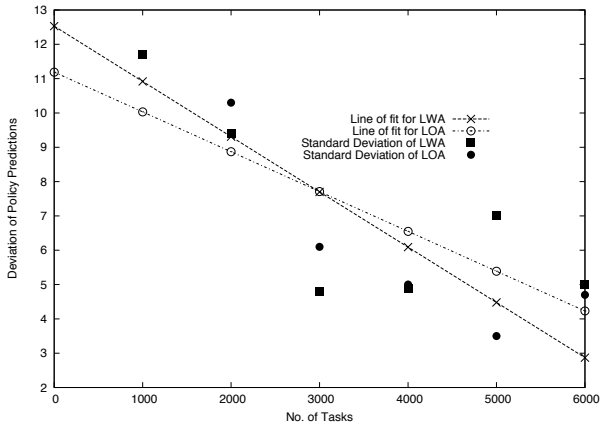

Fig. 9. Graph showing the convergence of the policy predictions using LOA & LWA respectively.

The standard deviations of the results were plotted and

the trend line (using linear regression) shows that as the number of tasks increases, the argumentation-based approach ($y = 12.5333 - 0.0016x$) consistently converges at 95% confidence interval, with a F value of 18.9133 and significance $p = 0.0122$. (See Fig. 9). On the other hand, with a significance $p = 0.0808$, there is no statistical significance as to whether the standard machine learning approach ($y = 11.1933 - 0.0012x$) converges or not.

VII. Discussion and Related Work

Learning techniques have been applied in many fields, ranging from artificial intelligence, multi-agent system, medicine, etc., and argumentation has received much attention in the recent past. However, to the best of our knowledge no other work has attempted to combine machine learning and argumentation in the way this work does. This work is novel in that it is a first attempt at using argumentation enriched approaches to learn underlying social characteristics (e.g policies) without assuming that those underlying features are public knowledge. Having said that, there are several related works that warrant discussion here. We discuss some recent works in argumentation and machine learning that impact on this work.

Rahwan et al. [18] present a formal framework for analysing the outcomes of interest-based negotiation (IBN) dialogues and established that providing further information (especially about underlying interests) improves the likelihood and quality of an outcome. Policy constraints can be captured as underlying goals that agents are hoping to achieve (by adhering to them) and so argumentation can be used to tease out information regarding those constraints. In circumstances where knowledge is incomplete or imperfect, argumentation has proven to be effective in reaching some goals that would have otherwise been unreachable [17], [18]. It is worth noting that our work differs from Rahwan et al. [18] in that while the authors are interested in gathering meta-information and using it to support interest-based negotiation, we are interested in learning the policies that other agents are operating with and using this knowledge to guide how a plan is resourced. Another major difference stems from the fact that the framework proposed in [18] was demonstrated via examples rather than vigorous evaluation. We, however, present empirical evaluation of our framework with experimental results. Our framework neatly combines machine learning and argumentation in predicting what the other's policies are. Furthermore, our work is aimed at supporting human decision making in team-based activities.

Možina et al. [27] combined machine learning with concepts of argumentation to produce a new machine learning technique called Argumentation-Based Machine Learning (ABML). With this framework, an expert can provide arguments for some learning examples and thereby enhancing the predicting power of the learner. The work implemented an argument-based extension of CN2 rule learning (ABCN2) and was able to show that ABCN2 out-performed CN2 in most tasks. However, the framework is another kind of learning algorithm (even though arguments could be added to it) and will struggle to disambiguate between subtle constraints that may produce similar outcome/effect and that is the main issue we are addressing in our work.

Atkinson and Bench-Capon [28] treated reasoning about what action an agent should select as presumptive argumentation. The framework captured situations where the effect of an action is partially dependent upon the choices of another agent. In other words, an agent chooses a move, proposes presumptive reasons for the action and subjects it to critiquing in order to establish suitability or otherwise. This kind of framework is useful in our work as we argue that policy constraints impact on the behaviour (or action) of an agent and that, in turn, could be learned and used to infer what the policies of that agent are.

VIII. Future Directions

In our future work, we plan to develop strategies for advising human decision makers on how a plan may be resourced and who to talk to on the basis of policy and resource availability constraints learned from previous interactions [29]. Parsons et al. [30], [31] investigated the properties of argumentation-based dialogues and examined how different classes of protocols can have different outcomes. We plan to explore ideas from this work to see which class of protocol will yield the "best" result in this kind of task. We are hoping that some of these ideas will drive the work on developing strategies for choosing who to talk to (and also which class of protocol to present first, and so on). Furthermore, we plan to incorporate the ability to suggest alternative resources based on the preferences of team members and to see what effect this will have on the learning of policies.

IX. Conclusions

In this paper, we have presented an approach that combines machine learning and argumentation for learning policies in a coalition mission. Individual policies are private and local to the individual agent and are not necessarily public, but using argumentation we have been able to tease out certain information that can improve the performance in learning the policies of other agents. In our approach, an argumentation layer was built over a traditional learning mechanism (decision trees) such that arguments were passed back and forth and these enabled our agents to disambiguate between resource and policy constraints, thereby fine-tuning the policies learned. We have also shown that integrating argumentation capability into systems, potentially, empowers agents with incomplete or imperfect knowledge to perform better than they would have without argumentation in learning.

Acknowledgements

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and

REFERENCES

[1] W. Vasconcelos, M. J. Kollingbaum, and T. J. Norman, "Resolving conflict and inconsistency in norm-regulated virtual organizations," in *Proc. of the 6th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007)*, Hawaii, USA, 2007.

[2] D. Gaertner, A. Garcia-Camino, P. Noriega, J. A. Rodriguez-Aguilar, and W. Vasconcelos, "Distributed norm management in regulated multi-agent systems," in *Proc. of the 6th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007)*, Hawaii, USA, 2007.

[3] A. Kelemen, Y. Liang, and S. Franklin, "A comparative study of different machine learning approaches for decision making," in *Recent Advances in Simulation, Computational Methods and Soft Computing*, N. E. Mastorakis, Ed. Piraeus, Greece: WSEAS Press, 2002, pp. 181–186.

[4] G. Sukthankar and K. Sycara, "Analyzing team decision-making in tactical scenarios," *The Computer Journal*, p. bxp038, 2009. [Online]. Available: http://comjnl.oxfordjournals.org/cgi/content/abstract/bxp038v1

[5] Sun Microsystems Inc., "Java," uRL: http://www.java.sun.com/.

[6] E. Friedman-Hill, *Jess in Action*. Manning, 2003.

[7] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*, 2nd ed. San Francisco: Morgan Kaufmann, 2005.

[8] D. Meyer, F. Leisch, and K. Hornik, "The support vector machine under test," *Neurocomputing*, vol. 55, no. 1-2, pp. 169–186, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/B6V10-49CRCBP-1/2/346ddc665b1b67be089a7d5d46edca07

[9] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag, 2001.

[10] S. Esmeir and S. Markovitch, "Anytime learning of decision trees," *J. Mach. Learn. Res.*, vol. 8, pp. 891–933, 2007.

[11] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth, "Occam's razor," *Information Processing Letters*, vol. 24, no. 6, pp. 377–380, 1987.

[12] L. Hyafil and R. L. Rivest, "Constructing optimal binary decision trees is np-complete," *Information Processing Letters*, vol. 5, no. 1, pp. 15–17, 1976.

[13] O. J. Murphy and R. L. McCraw, "Designing storage efficient decision trees," *IEEE Transactions onComputers*, vol. 40, no. 3, pp. 315–320, 1991.

[14] J. R. Quinlan, *C4.5: programs for machine learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[15] T. M. Mitchell, *Machine Learning*. McGraw Hill, 1997.

[16] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques," *Informatica*, vol. 31, no. 3, pp. 249–268, 2007.

[17] S. Parsons and N. R. Jennings, "Negotiation through argumentation-A preliminary report," in *Proc. of the Second International Conference Multi-Agent Systems (ICMAS'96)*, Kyoto, Japan, 1996, pp. 267–274. [Online]. Available: citeseer.ist.psu.edu/parsons96negotiation.html

[18] I. Rahwan, P. Pasquier, L. Sonenberg, and F. Dignum, "On the benefits of exploiting underlying goals in argument-based negotiation," in *Proc. of the 22nd International Conference on Artificial Intelligence (AAAI)*. California, USA: AAAI Press, 2007.

[19] L. Wittgenstein, *Philosophical Investigations*. Oxford: Blackwell, 1957.

[20] C. L. Hamblin, *Fallacies*. London, UK: Methuen, 1970.

[21] J. Levin and J. Moore, "Dialogue-games: meta communication structure for natural language interaction," *Cognitive Science*, vol. 1, no. 4, pp. 395–420, 1980.

[22] W. C. Mann, "Dialogue games: conventions of human interaction," *Argumentation*, vol. 2, no. 4, pp. 511–532, 1988.

[23] T. J. M. Bench-Capon, J. B. Freeman, H. Hohmann, and H. Prakken, "Computational models, argumentation theories and legal practice," in *Argumentation Machines. New Frontiers in Argument and Computation*, C. Reed and T. J. Norman, Eds. Dordrecht, The Netherlands: Kluwer Academic Publishers, 2003, pp. 85–120.

[24] P. Dijkstra, F. J. Bex, H. Prakken, and C. N. J. De Vey Mestdagh, "Towards a multi-agent system for regulated information exchange in crime investigations," *Artificial Intelligence and Law*, vol. 13, pp. 133–151, 2005.

[25] L. Perrussel, S. Doutre, J. Thevenin, and P. McBurney, "A persuasion dialog for gaining access to information," in *Proc. of the AAMAS International Workshop on Argumentation in Multi-Agent Systems (ArgMAS 2007)*, Hawaii, USA, 2007.

[26] P. Tolchinsky, P. McBurney, S. Modgil, and U. Cortés, *Agents deliberating over action proposals using the ProCLAIM Model*, ser. Lecture Notes in Artificial Intelligence (LNAI). Berlin: Springer, 2007, vol. 4696, pp. 32–41.

[27] M. Možina, J. Žabkar, and I. Bratko, "Argument based machine learning," *Artif. Intell.*, vol. 171, no. 10-15, pp. 922–937, 2007.

[28] K. Atkinson and T. Bench-Capon, "Action-based alternating transition systems for arguments about action," in *Proc. of the 22nd Conference on Artificial Intelligence (AAAI 2007)*. Vancouver, Canada: AAAI Press, 2007, pp. 24–29.

[29] N. Oren, T. J. Norman, and A. Preece, "Loose lips sink ships: A heuristic for argumentation," in *In Proceedings of the Third International Workshop on Argumentation in Multi-Agent Systems (ArgMAS 2006*, 2006, pp. 121–134.

[30] S. Parsons, M. Wooldridge, and L. Amgoud, "Properties and complexities of some formal inter-agent dialogues," *Journal of Logic and Computation*, vol. 13, no. 3, pp. 347–376, 2003.

[31] S. Parsons, P. McBurney, and M. Wooldridge, "The mechanics of some formal inter-agent dialogues," in *Advances in Agent Communication*, ser. LNCS, F. Dignum, Ed., vol. 2922. Springer-Verlag, 2004, pp. 329–348.