The Benefits of Arguing in a Team

Milind Tambe Hyuckchul Jung
Information Sciences Institute, University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292
{tambe, jungh}@isi.edu

August 11, 1999

Abstract

In a complex, dynamic multi-agent setting, coherent team actions are often jeopardized by conflicts in agents' beliefs, plans and actions. Despite the considerable progress in teamwork research, the challenge of intra-team conflict resolution has remained largely unaddressed. This paper presents CONSA, a system we are developing to resolve conflicts using argumentation-based negotiations. CONSA is focused on exploiting the benefits of argumentation in a team setting. Thus, CONSA casts conflict resolution as a team problem, so that the recent advances in teamwork can be brought to bear during conflict resolution to improve argumentation flexibility. Furthermore, since teamwork conflicts sometimes involve past teamwork, teamwork models can be exploited to provide agents with reusable argumentation knowledge. Additionally, CONSA also includes argumentation strategies geared towards benefiting the team rather than the individual, and techniques to reduce argumentation overhead.

1 Introduction

Teamwork is a critical capability in a large number of multi-agent applications, such as virtual environments for education and training [Tambe1997], robotic teams [Kitano *et al.*1997] and teams on the Internet. In these applications, agents must act together despite the uncertainties of their complex dynamic environment. Considerable progress has indeed been made in teamwork research. For instance, recent advances in teamwork models [Jennings1995, Tambe1997], which explicitly outline agents' commitments and responsibilities in teamwork, have significantly improved flexibility in teamwork coordination and communication. However, this research has so far not addressed the challenge of resolving conflicts within a team.

Yet, as agent applications advance to meet the requirements of scale and autonomy, inter-agent conflicts become increasingly inevitable. For instance, while autonomously reacting to dynamic events, agents may unintentionally interfere in others' actions, or faulty sensors may provide them with conflicting information or lead them to conflicting inferences. While such conflict resolution is difficult in general, it is even more problematic in teams if intra-team conflicts are not anticipated.

To address the problem of conflict resolution in team settings, we are building a system called CONSA: COllaborative Negotiation System based on Argumentation. In argumentation, agents negotiate by providing arguments (which may be justifications or elaborations) in support of their proposals to one another. CONSA builds on past work in argumentation [Parsons & Jennings1996, Kraus, Sycara, & Evenchik1998, Chu-Carroll & Carberry1995], but our focus here is to exploit the benefits of argumentation in a team setting. Thus, given CONSA's roots in past teamwork research [Tambe1997], a key idea is to cast conflict resolution as an explicit common team goal. As a result, the recent advances in teamwork models are brought to bear during conflict resolution, improving flexibility of agent behaviors during negotiations. For instance, if a team member privately discovers an event that renders the current team conflict irrelevant, it will inform its team members — it will not just withdraw privately from negotiations. Additionally, with an explicit common team goal, novel argumentation strategies emerge, e.g., agents may attempt to improve the quality of teammates' arguments. Furthermore, since team conflicts may be rooted in past teamwork, CONSA enables agents to argue effectively about teamwork, by exploiting the teamwork models in a novel way, i.e., not only as a guide to agent behavior during conflict resolution, but as a source for reusable argumentation knowledge. Finally, CONSA is being built within existing agent teams in complex environments, and has focused on practical issues, such as minimizing the resources consumed in negotiations.

2 Domains and Motivations

The motivation for current research on negotiation is based on our previous work in complex, multi-agent domains such as real-world battlefield simulations [Tambe1997]. We have built different teams of synthetic pilot agents that participate in combat simulations in these environments. These pilot agent teams include companies of attack helicopter pilots and divisions of transport and escort helicopter pilots. A second domain for our work is Robocup [Kitano *et al.*1997] where we have twice successfully participated in the RoboCup tournaments. These agent teams have been developed based on a teamwork model called STEAM [Tambe1997]. STEAM is based on the joint intentions [Cohen & Levesque1991] and SharedPlans [Grosz1996] theories of teamwork, but with practical extensions for monitoring and replanning as well as decision-theoretic communication selectivity. STEAM has provided significant teamwork flexibility in all of these applications. Yet, STEAM does not address the problem of conflicts in agents' beliefs and relevant negotiations to resolve such conflicts, limiting teamwork flexibility in key instances.

We describe here just a few key examples that outline some of the basic issues for collaborative negotiations:

- The firing position case: Individual pilots in a helicopter team typically attack the enemy from firing positions.

These positions are planned by a commander agent, who ensures that they do not conflict, i.e., the positions are planned to be at least one kilometer apart from each other. However, despite careful planning, individual pilots may autonomously react to unexpected enemy vehicles, and end up in conflicting positions (e.g., much less than 1 km apart).

- The proceed case: In planning the positions described above, the commander pilot plans one position(e.g. position to hide behind a small hill) per team member, and communicates it to the relevant team member via radio. In one run, a message was lost due to radio interference, i.e., the commander thought the position was communicated, but a team member M1 never received it. Thus, when the commander asked the team to proceed because it believed all of the positions were successfully communicated, there was a conflict with M1.
- The enemy position case: Two scout helicopter agents may have conflicting beliefs about the closest enemy unit seen. For instance, one scout may report completion of scouting and the closest enemy unit seen as part of this report, while the second scout may see an even closer enemy unit than the one reported.
- The ball position case: In our player team in RoboCup soccer simulation, defenders inform each other if the ball is close by and hence a threat. However, the players' belief of the ball's threat may differ, leading them to have conflicting beliefs about whether the ball is a threat.

These examples illustrate some of the key issues we are investigating in team negotiations. First, conflicts may arise between two team members' local actions, as in the firing position case, where an agent's local reaction has led to conflicts with another agent's local actions. In contrast, in the remaining three cases above, conflicts in agents' beliefs impact the team's joint action, to proceed, to report enemy location or to defend against the opponent team. Second, conflicts may or may not be related to past teamwork. Thus, while in the proceed case, conflicts are related to team members' past actions in teamwork, this is not true of the enemy position and ball position cases. Third, negotiations may need to be performed under real-time pressure, as in the ball position case, where negotiation delays are highly detrimental to team performance. These and other issues in negotiations are highlighted in further detail in the following sections. In addressing these issues, we aim to avoid any specialized solutions, and focus instead on a general approach that would be applicable to a wide variety of conflicts.

3 Teamwork Model: A Brief Overview

Before we discuss CONSA, it is useful to briefly overview teamwork models, particularly the STEAM [Tambe1997] model, since it is the basis of our team implementations. STEAM consists of two components, both currently realized in the Soar [Newell1990] architecture. The first is an enhanced agent architecture with explicit representation of team's joint intentions, mutual beliefs and team goals. Figure 1 shows an operator hierarchy (i.e., a reactive plan hierarchy) for a synthetic helicopter pilot developed using STEAM. Team operators (reactive team plans), which explicitly express a team's joint activities, are shown in [], such as [Engage]. At any time, one path through this hierarchy is active. This active hierarchy of operators is the team's joint intentions (team operators) and individual intentions (individual operators).

The second component of STEAM is the domain-independent teamwork knowledge to enable individual agents' flexible teamwork. Of particular importance here are two of the classes of domain-independent actions. The first is coherence-preserving actions, derived from the joint intention theory [Cohen & Levesque1991]. These require agents to jointly activate and terminate team operators, by establishing mutual beliefs in their initiation and termination; individual operators are executed without such mutual beliefs. The second class of domain-independent actions is maintenance and repair actions, for re-planning and team reorganization. These actions require an explicit specification of the dependency relationship of the joint intention on individual team members' activities, based on the notion of a role. A *role* constrains a team member Mi to some suboperator op_{Mi} of the team operator [OP]. Three primitive role-relationships (and their combinations) can currently be specified in STEAM. An AND-combination implies that the achievement

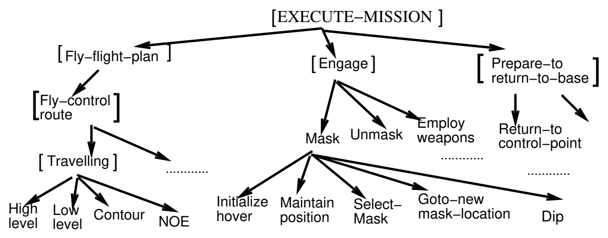

Figure 1: Portion of pilot operator hierarchy.

of team operator requires achievement of each one of the roles. An OR-combination requires success in at least one role for the team operator to be achieved. The role-dependency relationship states that an op$_{Mi}$ depends on op$_{Mj}$.

4 Argument Representation & Evaluation

This section describes CONSA's underlying representation and algorithms to evaluate arguments, which are embedded in a larger CONSA process, discussed in the next section. CONSA's representation of arguments is based on Toulmin's [Toulmin1958] argumentation pattern (henceforth TAP), chosen for its generality. In a TAP, an argument consists of the following elements: (i) *claim:* a conclusion whose merit an agent seeks to establish. (ii) *data:* the facts that are a foundation for the claim. (iii) *warrants:* the authority (e.g., a rule) for taking the step from the data to the claim. (iv) *qualifications:* degree of force which conferred on the claim based on the data and warrant.

In CONSA, claims are agents' individual or mutual beliefs. During argumentation, these claims form the proposals, with the supporting TAP as the argument for the proposal. For example, in RoboCup soccer, a claim (proposal) may be that "the ball is a threat," supported by data that "the ball is 30 meters from own goal," and a warrant that "if the soccer ball is within 35 meters of own goal, then it is very likely a threat." In CONSA, the data may itself be another claim (belief), with its own supporting TAP, so that a recursive tree of TAP structure may emerge in support of a claim. Finally, in CONSA, the qualifications on claims determine the strengths of arguments. Currently, claims have qualitative strengths: high, medium and low. Thus, a strong warrant and data will lead to a "high" strength for the claim.

When an agent sends a proposal to its team, team members must determine if their own beliefs conflict with the proposal. Figure 2 presents a simplified version of CONSA's algorithm to make this determination. The input is a proposed TAP tree Θ, which forms the proposal (claim), with supporting arguments. The output is a set Ω of tuples ($\{$reject(claim$_i$) or accept(claim$_i$)$\}$, justification). Here, a reject tuple implies an agent's conflict with the claim$_i \in \Theta$, while an accept tuple implies an improved justification in support of the claim. The justifications consist of TAPs. If Ω is empty, then no conflicts or improvements are found.

In the algorithm, step 1-a checks the input TAP tree Θ for conflicts with the agent's own claims. If a conflict is found, strengths of the conflicting claims are compared and the other agent's claim is rejected if own claim is found stronger. Step 1-b now compares the input claims from Θ for coincidence or agreement. If coincidence is found, then the supports of coincident claims are compared, to determine the stronger support. If one is found, it is added to Ω. For expository purpose, two complicating factors addressed in CONSA are not shown here. First, CONSA can address the presence of multiple conflicts and coincidences. Second, when no coincidence or conflict is found in Θ itself, CONSA will not immediately accept Θ. Since leaf nodes in Θ may hold undesirable implications, CONSA derives implications from Θ. While in general checking undesirable implications is difficult, CONSA currently executes one iteration of such derivations, checking for conflict or coincidence and adding the result to Ω.

Evaluate-proposal(Input: TAP-tree Θ; Output: Ω)

1. In parallel, for all claims α_i in TAP-tree Θ do:
   (a) { Check α_i for conflict with own claims;
   If α_i conflicts with own claim β_j, **Compare-strengths**(α_i,β_j);
   If β_j is stronger, add (reject(α_i),β_j) to Ω; }
   (b) { Check α_i for coincidence with own beliefs; If coincidence with own claim β_i,
   { **Compare-strengths**(support(α_i), support(β_i));
   If support(β_i) is stronger, add (accept(α_i),support(β_i)) to Ω; } }
2. Output Ω; if Ω is empty, no conflicts or coincidence found.

Figure 2: A simplified version of CONSA's algorithm for evaluating proposal.

To determine the strengths of the claims in the *compare-strengths* procedure in Figure 2, CONSA relies on the supporting TAP structure. Given that the TAP structure can itself be recursive, claim strengths are evaluated recursively. For leaf-level claims, evidential rules are used. Here, CONSA exploits the benefits of argumentation in a team setting, by relying on the following rules of evidence: assertions from a team member regarding its own role and capability are judged to provide high-strength claims.

5 CONSA Approach

Figure 3 presents the overall CONSA negotiation process. Step 1 is a proposal generated by a team member. Steps 2, 3 and 4 are the *opening*, *argumentation* and *termination stages* of CONSA's negotiation. In the opening stage, agents agree to jointly resolve the current conflict. In the argumentation stage, they cycle through proposals and counter-proposals, terminating arguments in the termination phase.

1. A team member Mi generates a proposal α.
2. *Opening stage:*
   (a) A team member Mj detects a conflict with α.
   (b) If Mj believes joint action not beneficial to resolving conflict, terminate, return;
   (c) Else Mj communicates with team members to establish team operator to resolve current conflict.
3. *Argumentation stage*
   (a) Any member Mk in the current team operator may generate proposal to resolve conflict;
   (b) Other team members evaluate-proposal (see Figure 2).
   (c) If no conflict/coincidence found, accept the proposal and go to step 4;
   (d) Else if proposal found to conflict/coincide; continue argument if cost-benefit-wise useful, else accept the proposal and goto step 4;
4. *Closing stage*
   (a) If suggested proposal accepted, then terminate conflict-resolution team operator;
   (b) Else if the conflict resolution found unachievable or irrelevant, terminate conflict-resolution team operator;

Figure 3: Three stages of argumentation in CONSA.

Opening and Closing Stages: In CONSA's opening stage, the conflict detection step (2-a) requires it to address two different types of conflicts. In particular, based on the description of the teamwork model (Section 3), conflicts can be of two types: (1) Team members may have conflicting beliefs about jointly initiating or terminating a team operator, e.g., one agent believes the team operator must be terminated, while the other believes it cannot be terminated; or (2) Agents executing individual operators may unintentionally conflict with each other's role performance. Thus, in the examples from Section 2, the "firing position case" is a type 2 conflict, but the rest are type 1 conflicts. To detect a type 1 conflict, an agent must evaluate proposals sent by their teammates to jointly initiate or terminate team activities, detected via the **Evaluate-proposal** algorithm in Figure 2. In contrast, to detect type 2 conflicts, CONSA uses *role constraints*, that explicitly specify the maintenance goals for the successful performance of the role. For instance, in the firing position case, the *lateral-range* (distance) between Mj (the agent performing this role) and any other teammate must be at least one kilometer.

Having detected a conflict in Step 2-a, we temporarily skip over step 2-b to focus on step 2-c. Here, a team member Mj, who has detected a conflict, initiates establishment of a team operator to resolve the current conflict. If the conflict is of type 1, Mj initiates the establishment of *resolve-joint-conflict* as a team operator, involving the entire team from the original joint activity. If the conflict is of type 2, Mj initiates the establishment of *resolve-role-conflict* as a team operator, but the involved team here is only Mj and the agent that caused a conflict for Mj's role. For instance, in the firing position case, *resolve-role-conflict* is established as a team operator between Mj and Mk(the agent that caused the role conflict).

By casting conflict-resolution itself as a team operator, all of STEAM's flexible teamwork capabilities are brought to bear, to guide agents' behavior during conflict resolution. For instance, agents jointly establish the conflict-resolution team operators, using protocols that ensure synchronization and agreement among team members. In particular, teammates may disagree about the existence of the conflict, or they may be unable to negotiate if they are performing another higher priority task. However, by using a team operator for conflict resolution, an agent Mj begins negotiations only after ensuring its teammates agree to and are able to engage in negotiations. Furthermore, STEAM's reasoning about commitments leads team members to behave responsibly towards each other. If a dynamic event causes a team member to privately discover that their conflict is resolved or unresolvable or irrelevant, it will be committed to make this mutually believed in the team. A team member cannot just on its own drop out from participation in the conflict resolution. The utility of such flexibility can be seen in the firing position case. If a team member sees enemy vehicles approaching, it will terminate the current on-going negotiations, but do so responsibly while informing teammates of the situation.

Argumentation Stage: The argumentation stage involves an agent (sender) making a proposal to the agent-team (receiver) with an attached justification (argument). The receivers evaluate the proposal taking the justification into account, and either accept or refute it. If refuting the proposal, a receiver may send back a counter-proposal to the team, who may continue this cycle of proposals and counter-proposals. Refutation may be done via rebutting or undercutting [Parsons & Jennings1996]. Briefly, rebutting refutes the teammate's claim (proposal) directly, with some justifications. In contrast, undercutting attacks the justification provided with the proposal, rather than the proposal itself.

In this argumentation stage, the teamwork setting provides two key novel ideas. First, it enables and requires a third strategy in addition to rebutting and undercutting, which we call "improve support." In particular, an agent receiving a proposal from its team member may accept the proposal, but may have a better justification for the proposal than the one offered by the sender. For instance, in the "enemy position" case from Section 2, the second scout detected a closer enemy unit. The second scout agrees with the top-level claim that the scouting is completed, but it offers a higher quality solution about the closer enemy unit, which allows the helicopter team's performance to improve. It is to enable this "improve-support" strategy that the **Evaluate-proposal** algorithm (Fig 2) checks for claim coincidence.

Second, teamwork models provide reusable argumentation knowledge. In particular, team conflicts are sometimes rooted in past teamwork, as for instance in the proceed case. To argue effectively about teamwork, agents must be knowledgeable about teamwork. Here, STEAM provides general, reusable warrants for constructing TAPs. For instance, the warrants shown below, extracted from STEAM's role relationships, are employed in CONSA. Here, warrant $\omega 1$ states that if a team operator τ is an AND-combination, and all of its roles are not achieved, then the team operator is not achieved. $\omega 2$ is a variation for an OR-combination and $\omega 3$ is that for an AND-combination.

- $\omega 1$: Team-Operator(τ) $\wedge$ AND-combination(τ) $\wedge$ $\neg$ All-roles-fulfilled(τ) $\rightarrow$ $\neg$ achieved(τ)
- $\omega 2$: Team-Operator(τ) $\wedge$ OR-combination(τ) $\wedge$ $\neg$ All-roles-unachievable(τ) $\rightarrow$ $\neg$ unachievable(τ)
- $\omega 3$: Team-Operator(τ) $\wedge$ AND-combination(τ) $\wedge$ All-roles-fulfilled(τ) $\rightarrow$ achieved(τ)

Real-time, Efficient Argumentation: There are three techniques used in CONSA to reduce resources utilized in argumentation and enhance its real-time performance (shown in steps 2-b and 3-d of Figure 3). One technique is decision-theoretic reasoning of the cost-benefit analysis of argumentation. For instance, in the "ball position case" in Section 2, the cost of arguing may outweigh the benefits (e.g., the ball may be shot into the goal by the time the defenders complete their negotiations). Therefore, an agent will not negotiate with teammates even though it detects a conflict in the teammates' proposal. The second technique is ordering of arguments. If there are multiple arguments applicable, CONSA will communicate the strongest first, in order to speed up the argumentation process. CONSA also uses pruning (see below) to avoid communication of commonly held warrants.

Detailed Example of CONSA application: For a detailed example of CONSA's application, we take the simple "proceed case" from Section 2. Figure 4 shows the initial warrants and claims that are mutually known by the pilot agent team (of five agents). τ is the current team operator, an AND-combination. The initial proposal is generated by the commander agent (Step 1 of Figure 3) to suggest termination of the team operator τ. This proposal is $\alpha 3 \Leftarrow \alpha 2$, where $\alpha 3$ is the claim "achieved(τ)" and $\Leftarrow$ stands for a justification.

$\Rightarrow$Mutually believed warrants: $\omega\mathbf{1}$, $\omega\mathbf{2}$, $\omega\mathbf{3}$ and $\omega\mathbf{4}$: $\neg$Role-fulfilled(self) $\rightarrow$ $\neg$All-roles-fulfilled(τ)
$\Rightarrow$Commander pilot agent's initial claims: claim $\alpha\mathbf{2}$: All-roles-fulfilled(τ), claim $\gamma\mathbf{1}$: AND-combination(τ)
$\Rightarrow$Pilot agent **M1**'s initial claims: claim $\beta\mathbf{4}$: $\neg$ Role-fulfilled (self), claim $\gamma\mathbf{1}$: AND-combination(τ)

Figure 4: Initial state: Commander believes all-roles-fulfilled, M1 believes own role not fulfilled.

M1 evaluates the proposal from the commander agent to detect conflicts (step 2-a of Figure 3). During this evaluation, using the **Evaluate-proposal** algorithm from Figure 2, no direct conflict or coincidence is found. However, deriving implication of $\alpha 2$ leads to "Role-fulfilled(self)", which conflicts with $\beta 4$, M1's own belief. However, $\beta 4$ is evaluated to be stronger, as M1 is an expert in its own role. M1 next uses $\omega 4$ and $\omega 1$ to construct an argument: $\neg\alpha 3 \Leftarrow \neg\alpha 2 \Leftarrow \beta 4$. (Warrants $\omega 1$ and $\omega 4$ are pruned.) Essentially, M1 informs the commander agent that it disagrees that the team operator is achieved, since its own role is not fulfilled. Since this is a type 1 conflict, the argument from M1 is communicated to the entire team of pilot agents. This causes all members (including M1) to establish a team operator (resolve-joint-conflict); the team has thus entered the argumentation stage of CONSA. In this case, since $\beta 4$ is in the area of expertise of M1, the commander (and other team members) evaluate $\beta 4$ to have a high strength and accept it. They subsequently also accept $\neg\alpha 3$ and $\neg\alpha 2$ based on the support offered by $\beta 4$. Thus, the proceed case is resolved by the commander accepting M1's assertion, and it communicates this acceptance to teammates.

6 Applying CONSA

CONSA is currently realized in the Soar architecture in 109 rules. In the following we attempt a preliminary qualitative evaluation of CONSA. Our implementation has enabled agents to negotiate to resolve conflicts in the cases from Section 2, in the following manner:

- *Firing position case*: An agent detects a conflict in its firing position due to its role-constraint violation (one kilometer lateral range). It then establishes a team operator (with the teammate that violates the role constraint) to resolve role conflict. It generates a proposal to suggest an equidistant move by each agent (500 meters) to meet the lateral range role constraint. This proposal is accepted by the second agent. (However, if the second agent can not move, it rejects this proposal, causing the first agent to move 1 km on its own.)
- *Proceed case:* As discussed previously, M1 persuades teammates that the current team activity is not achieved.
- *Enemy position case:* The second scout finds an "improve-support" argument to inform the team that it has better support (i.e., a higher quality solution), in the form of closer-range enemy that it spotted.
- *Ball position case:* As the cost of negotiation exceeds the likely benefits, agents avoid negotiations, and act based on own (divergent) beliefs.

We also attempted a preliminary test CONSA's flexibility by creating some surprise variations of the above cases.

- proceed-1: The role relationship for the team operator τ was changed from AND-combination to OR-combination. Here, despite team member M1's role not being fulfilled, M1 did not detect a conflict, and no arguments were generated. This is correct, since an OR-combination does not require all roles to be fulfilled.

- proceed-2: We gave one pilot agent (M1), two arguments to attack the commander's proposal, one based on own role, and one based on another teammate M3's role. Here, M1 correctly selected the stronger argument based on own role to communicate first to the team.

- firing-position-1: When the pilots established the *resolve-role-conflict* team operator to resolve firing position conflicts, enemy vehicles were suddenly placed close by to them. The pilot who noticed these vehicles first, terminated the conflict-resolution team operator as it was irrelevant, and informed its teammate.

- firing-position-2: In a similar situation as above, we had one helicopter destroyed. The second terminated the negotiation, as this team operator had become unachievable.

7 Related Work

Previous work in argumentation-based negotiation has often assumed non-cooperative agents. For instance, [Kraus, Sycara, & Evenchik1998] uses several argument types borrowed from human argumentation in non-cooperative situations, e.g., threat, promise of a future reward, and appeal to self interest. An example from [Kraus, Sycara, & Evenchik1998] is negotiation among two robots on Mars. Here, to persuade a robot R2, a robot R1 threatens it (R2) that R1 will break R2's camera lens or antenna, if R2 does not comply. Such arguments appear inappropriate in team settings, e.g., if R1 and R2 are a team, and if R1 carries out its threat, then it will have a teammate (R2) without a lens or antenna. Other explicitly non-collaborative argumentation work appears in the legal domain, e.g., DART [Freeman & Farley1993], which is also based on Toulmin's representation schema. In contrast, [Parsons & Jennings1996] does not explicitly assume collaborativeness or non-collaborativeness in agents.

CONSA differs from this work in its explicit exploitation of the team setting in argumentation. As seen earlier, it exploits teamwork models: (i) to guide flexible agent behavior in negotiation and (ii) as a source of reusable argumentation knowledge. It also adds argumentation strategies so agents can collaboratively improve each other's arguments. Also, CONSA includes techniques to avoid high overheads of negotiations.

Chu-Carroll and Carberry's work in argumentation does assume collaborativeness on part of the participating agents [Chu-Carroll & Carberry1995]. While they use SharedPlans [Grosz1996] in negotiations, they appear to treat SharedPlans as a data-structure, rather than a teamwork model. Thus, unlike CONSA, they do not use SharedPlans either for prescribing agents' behaviors in negotiations, or as source of reusable argumentation knowledge.

8 Summary and Future Work

Multi-agent teamwork in diverse applications ranging from planning, design, education and training, faces the problems of conflicts in agents' beliefs, plans and actions. Collaborative negotiation is thus a fundamental component of teamwork. We have begun to address this problem via an implemented system called CONSA for collaborative negotiation via argumentation. While CONSA continues to build on previous work in argumentation, it exploits the benefits of a team setting via the following key ideas: (i) CONSA casts conflict resolution as a team problem, bringing to bear some of the recent advances in flexible teamwork to improve the flexibility of agent behavior in conflict resolution; (ii) Since team conflicts are often about past teamwork, CONSA exploits teamwork models to provide agents with reusable argumentation knowledge; (iii) CONSA focuses on collaborative argumentation strategies such as improve-support; (iv) As an implemented system in a dynamic environment, CONSA uses a decision theoretic approach, argument ordering and pruning to reduce the cost of negotiation. Areas of future work include understanding CONSA's implications for argumentation in self-interested agents.

Acknowledgements

This research was sponsored in part by AFOSR contract no. F49620-97-1-0501, and in part by a subcontract from the Boeing Corp. We thank Zhun Qiu who implemented portions of the CONSA system described in this paper.

References

[Chu-Carroll & Carberry1995] Chu-Carroll, J., and Carberry, S. 1995. Generating Information Sharing Subdialogues in Expert-User Consultation. In *Proceedings of International Joint Conference on Artificial Intelligence*, 1243–1250. Menlo Park, Calif.: International Joint Conference on Artificial Intelligence.

[Cohen & Levesque1991] Cohen, P. R., and Levesque, H. J. 1991. Teamwork. *Nous* 25(4):487–512.

[Freeman & Farley1993] Freeman, K., and Farley, A. 1993. Towards Formalizing Dialectical Argumentation. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*, 440–445. Saline, MI: Cognitive Science Society.

[Grosz1996] Grosz, B. 1996. Collaborating Systems. *AI Magazine* 17(2):67–85.

[Jennings1995] Jennings, N. 1995. Controlling Cooperative Problem Solving in Industrial Multi-Agent Systems using Joint Intentions. *Artificial Intelligence* 75(2):195–240.

[Kitano *et al.*1997] Kitano, H.; Asada, M.; Kuniyoshi, Y.; Noda, I.; and Osawa, E. 1997. RoboCup: The Robot World Cup Initiative. In *Proceedings of the First International Conference on Autonomous Agents*, 340–347. New York, N.Y.: The Association for Computing Machinery.

[Kraus, Sycara, & Evenchik1998] Kraus, S.; Sycara, K.; and Evenchik, A. 1998. Reaching Agreements through Argumentation: a Logical Model and Implementation. *Artificial Intelligence* 104:1–70.

[Newell1990] Newell, A. 1990. *Unified Theories of Cognition*. Cambridge, Mass.: Harvard Univ. Press.

[Parsons & Jennings1996] Parsons, S., and Jennings, N. R. 1996. Negotiation through Argumentation — a Preliminary Report. In *Proceedings of the International Conference on Multi-Agent Systems*, 267–274. Washington, D.C.: IEEE Computer Society.

[Tambe1997] Tambe, M. 1997. Towards Flexible Teamwork. *Journal of Artificial Intelligence Research (JAIR)* 7:83–124.

[Toulmin1958] Toulmin, S. 1958. *The Uses of Argument*. London: Cambridge Univ Press.