Towards Flexible Multi-Agent Decision-Making Under Time Pressure

Sanguk Noh and Piotr J. Gmytrasiewicz
Department of Computer Science and Engineering
University of Texas at Arlington
Arlington, TX 76019, Box 19015
{noh, piotr}@cse.uta.edu

Abstract

To perform rational decision-making, autonomous agents need considerable computational resources. In multi-agent settings, when other agents are present in the environment, these demands are even more severe. We investigate ways in which the agent's knowledge and the results of deliberative decision-making can be compiled to reduce the complexity of decision-making procedures and to save time in urgent situations. We use machine learning algorithms to compile decision-theoretic deliberations into condition-action rules on how to coordinate in a multi-agent environment. Using different learning algorithms, we endow a resource-bounded agent with a tapestry of decision making tools, ranging from purely reactive to fully deliberative ones. The agent can then select a method depending on the time constraints of the particular situation. We also propose combining the decision-making tools, so that, for example, more reactive methods serve as a pre-processing stage to the more accurate but slower deliberative decision-making ones. We validate our framework with experimental results in simulated coordinated defense. The experiments show that compiling the results of decision-making saves deliberation time while offering good performance in our multi-agent domain.

1 Introduction

It is desirable that an autonomous agent, operating under uncertainty in complex environments, be able to make optimal decisions about which actions to execute. Rational decision-making under such circumstances using, for instance, the paradigm of expected utility maximization, is costly [Horvitz, 1988; Russell and Wefald, 1991; Russell and Subramanian, 1995; Zilberstein and Russell, 1996]. In our work, we consider additional complexities presented by multi-agent environments. In these settings, an agent has to make decisions as to the rational course of action considering not only the possibly complex and not fully known state of its environment, but also considering the beliefs, goals, intentions and actions of the other agents. Clearly, these demands may lead to its failure to decide an action within the time constraint.

To cope with time constraints imposed by various decision-making situations in complex and uncertain rnulti-agent settings, we endow an agent with a tapestry of decision-making procedures, from strictly reactive to purely deliberative. The reactive procedures are constructed by compiling the deliberative decision-theoretic reasoning into condition-action rules. The compilation process exploits the regularities of the decision-theoretic reasoning and avoids costly deliberations in urgent situations. The rules are obtained from machine learning algorithms, which, as inputs, use the results of full-blown decision-theoretic computations performed offline. Each of the compiled methods is assigned a performance measure that compares it to the full-blown decision-theoretic benchmark. The various compilations available, and their combinations with more deliberative methods, constitute a spectrum of approaches to making decisions under the constraints of available computational (and cognitive) resources, and under time pressure.

Given the various decision-making methods at its disposal, an agent should consider a number of factors to choose the appropriate decision-making mechanism for the situation at hand. The key factors include *the quality of the decision* provided by a method, *the method's running time,* and *the urgency of the situation at hand.* Intuitively, when a situation is not urgent, the agent can afford the luxury of full-blown decision-theoretic reasoning since it results in highest quality of the choice made. If the situation is very urgent, the agent should save as much time as possible by using a crude but fast reactive tool. If the situation is somewhat urgent, the agent should use methods that are somewhat sophisticated although not necessarily optimal.

Interestingly, the spectrum between the purely reactive and fully deliberative decision-making tools can be spanned by combining these two varieties of methods. For example, the agent can use fast reactive rules as a pre-processing stage to narrow down the set of viable alternatives. These alternatives can then be passed on to a deliberative decision-making method that uses all of the agent's detailed knowledge to compute the expected utility of these few courses of action.

In this paper, we develop a suite of decision-making procedures for agents operating in multi-agent environments, and we measure[1] their performance and running time. We use a particular multi-agent domain in which automated agents have to decide how to coordinate their attempts to intercept multiple incoming threats (as in anti-air defense), but we believe that lessons learned in this domain generalize to other multi-agent, domains.

2 Background and Related Work

Our prior work on deliberative decision-theoretic method includes the Recursive Modeling Method (R.MM) [Gmytrasiewiez, 1996; Gmytrasiewicz *at* al., 1998; Noh and Gmytrasiewicz, 1997; 1998]. We have implemented a full-blown version of RMM which allows an agent to compute its best action given what is known about the other agents and about their states of knowledge and capabilities. In the task of coordinating agents in a simulated

Figure 1: An example anti-air defense scenario.

anti-air defense domain (such as in Figure 1) the performance of RMM agents was comparable to or better than the human performance. Figure 2 presents these results in terms of the average total damage suffered by each of the coordinating defense teams. We show the performance of three different teams: RMM-RMM, RMM-Human, and Human-Human team. We experimented with all the teams in cases when communication was, and was not, available. When the communication was available, the performance achieved by three teams was improved, with the all-RMM team performing slightly

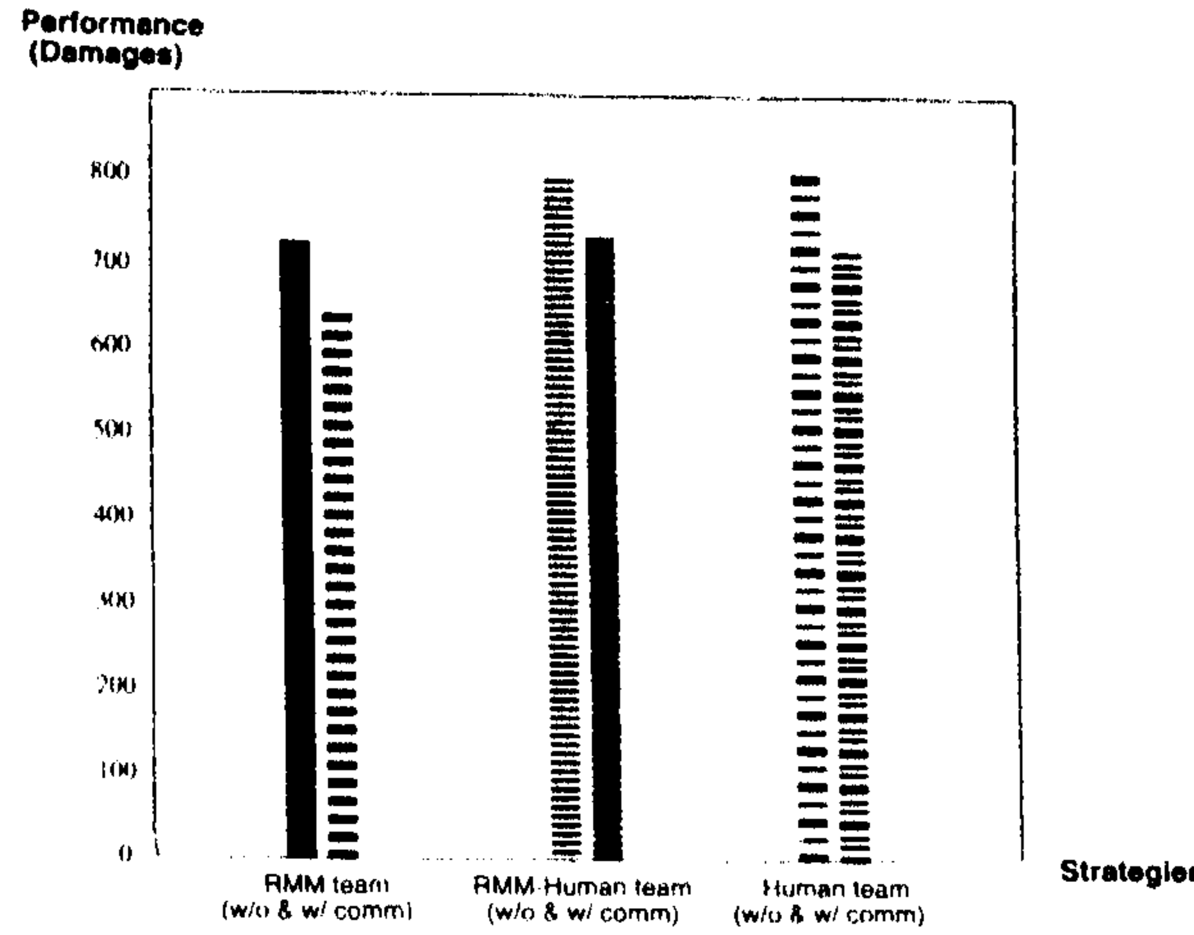


Figure 2: The performance of RMM, RMM-Human, and Human teams in anti-air defense, without and with communication, respectively.

better than the others. However, since the RMM decision procedure considers all of the combinations of the agents' alternative actions, it is not surprising that its complexity is, in the worst case, exponential in the number of agents present. As the complexity of the multi-agent situation increases, as in Figure 3, the running time of RMM grows to over one hour (our current implementation is in Lisp running on a P90 machine). It is clear that, the full-blown RMM needs to be supplemented by other, more reactive, methods for more[1] complex and time-critical scenarios.

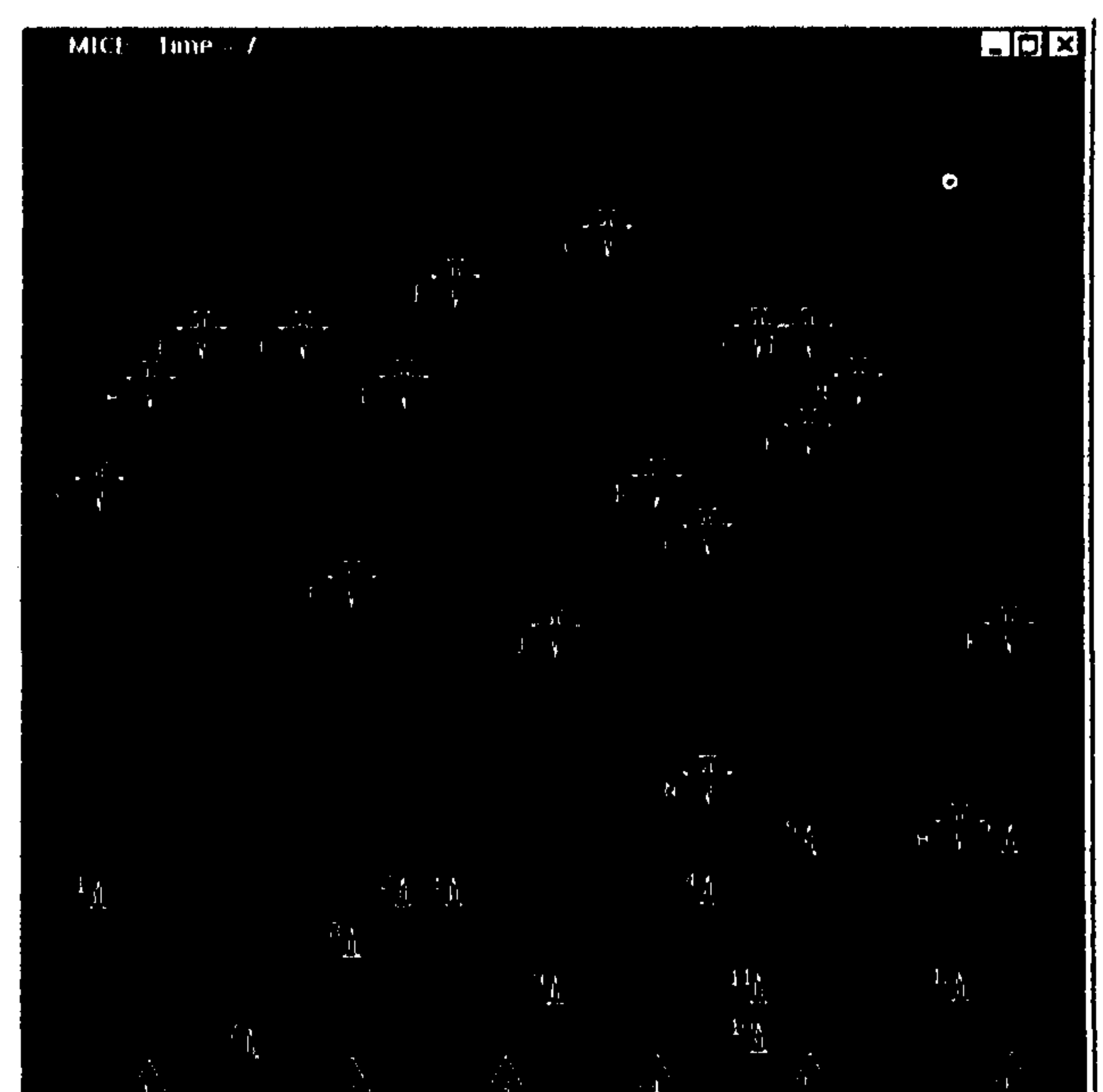


Figure 3: A complex anti-air defense scenario.

Our present work builds on a body of related research. Fox et al. [Fox and Krause, 1991] provides a theoretical framework for symbolic reasoning as reactive decision procedure. Other rigorous efforts to make decision-theoretic systems computationally tractable include work on the use of metareasoning procedures to control inference [Horvitz, 1988; Horvitz *et* al., 1989], and anytime algorithms [Dean and Boddy, 1988; Russell and Subramanian, 1995; Zilberstein and Russell, 1996]. In yet another approach Bratman [Bratman *et* al., 1988] describes an agent's architecture that includes both means-end reasoning and decision-theoretic reasoning. For a resource-bounded agent, the agent's beliefs, desires, and intentions (BDI) involve the formation, revision, and execution of plans to constrain the deliberation process. Rao et al. [Rao and Georgeff, 1995] explore the applicability of reactive and deliberative behavior provided by the BDI architecture, and use it for air-traffic management system (OASIS). Ephrati and collaborators [Ephrati *et* al., 1995J apply a filtering strategy to the multi-agent tile world system. The filtering is accomplished using rules provided by the system designer, and it only guides the agent's role allocation. Our condition-action rules, on the other hand, represent learning over the results of agent's rational decisions in sample scenarios obtained from a deliberative method. Our filtering strategy is to accumulate agents[1] knowledge and effectively use it to limit deliberation in urgent situations.

In the following sections of this paper we propose a compilation process that explores the regularities of a deliberative decision making, and show how an autonomous agent can use the compiled information, given performance metrics and time constraints. Then, we validate our framework empirically, and discuss the experimental results. In conclusion, we summarize our results and further research issues.

3 Formalism of Compilation and Filtering

To reduce the complexity and time needed for decision making in time constrained situation, we compile the results of deliberative decision-making into a set of reactive condition-action rules with numerous machine learning algorithms. An autonomous agent can use the compiled knowledge [Russell, 1989; Zilberstein, 1995] and either eliminate the deliberative decision-making all together, or constrain the number of alternative actions considered by excluding the ones that are likely to be suboptimal.

We propose an *adaptive and deliberative agent* (ADA) architecture, as consisting of compiled and deliberative decision procedures that allow the agent's bounded rationality to emerge from their combined usage. Let *N* be the set of agents, $A_i, i \in N$, be the set of actions of agent i, and $S_i, i \in N$, be the set of world states that the $i-th$ agent can discriminate among. For each action $a_i^k \in A_i$ we define *condition*(a_i^k, S_i) to be the abstraction of the world state that includes only the parameters relevant for this action. For example, if the action is to pick up a block (shoot at a given threat), then the corresponding abstraction specifies the location and other parameters of the block (threat). Finally, let $L_i, i \in N$, be the set of compilation methods (learning algorithms) that the agent i employs.

Given a learning method $l \in L_i$, a compiled decision-making procedure of an adaptive and deliberative agent implements a function $\rho_{i[l]}$: *coriditum*$(a_i^k, S_i) \mapsto \{Yes, No\}$, representing that the action a_i^k is (or is not) recommended in the state S_i. Thus, various machine learning algorithms compile decision-theoretic models into different functions $\rho_{i[l]}$. As we mentioned, we generated the training examples for these learning algorithms from deliberative reasoning performed by RMM.

To allow for further flexibility in the ADA agents, we allow the procedures to be combined depending on circumstances at hand. Clearly, when the agents have enough time, they should try to make a deliberatively rational decision that maximizes their expected utility. In a time-critical situation, howʳever, agent's decision-making is bounded by the available computation time. For an adaptive and deliberative agent, therefore, we use the set of compiled rules to remove from consideration the likely unprofitable actions, and to reduce the deliberation cost. This strategy is represented by the agent i's filtering criterion $\delta_{i[l]}$: $S_i \times A_i \mapsto A_i'$, where $A_i' \subseteq A_i$. Intuitively, the value of δ is the set of plausible actions the agent should consider in situation S_i. The filtering criterion $\delta_{i[l]}$ results from applying the rules in the function $\rho_{i[l]}$ to the current state to obtain the plausible alternatives. For example, if $\delta_{i[l]}(s_i^1, \{a_i^j, a_i^k, a_i^m\}) = \{a_i^k, a_i^m\}$, then a_i^k and a_i^m are plausible, and a_i^j is not, in situation s_i^1.

Given the set of plausible actions, our agent maximizes the expected utility among them:

$$\arg\max_{a_i^j \in A_i'} EU(a_i^j) \quad \text{(i)}$$

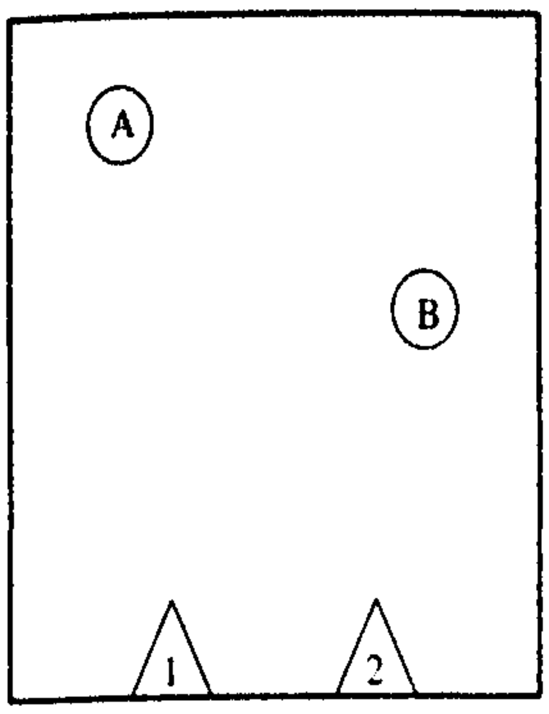
We now apply the above formalism to agents making coordinated decisions in the anti-air defense domain.

4 Deliberation About Action in Anti-Air Defense Domain

Our specific domain, the anti-air domain, consists of a number of attacking targets, labelled A and B in Figure 4, and a number of defending units, labelled 1 and 2.[1] The mission of the defense units is to attempt to intercept attacking targets so as to *minimize damages* to the defended ground area. Let us note that this situation makes coordination necessary. The defense batteries do not want to miscoordinate and attempt to intercept the same threat, both due to the wasted ammunition and due to the increase in likelihood that the remaining threat will reach its destination and cause damage proportional to its warhead size.

Given these factors, the expected benefit of shooting at a threat can be quantified as a product of the size of

[1] In the figure, the left top corner of the screen is (0,0), x is pointing right, and y is pointing down.



Missile	Position	Warhead
A	(6,3)	100
B	(14,9)	120

Battery	Position	$P(H_{iA})$	$P(H_{iB})$
1	(7,20)	0.94	0.56
2	(13,20)	0.58	0.94

Figure 4: An example of a simple anti-air defense scenario.

the threat and the interception probability. The interception probability[2], $P(H_{ij})$, is dependent on the angle d_{jj} between the target j's direction of motion and the battery i's line of sight, the distance d_{ij} between the battery and the target, and the speed of target s_j (see, for example, [Macfadzean, 1992]), as follows:

$$P(H_{ij}) = e^{-\mu a_{ij} d_{ij} s_j} \tag{2}$$

where u is an interceptor-specific constant (assumed here to be 0.001).

For example, in Figure 4, the combined benefit of Battery l's shooting at the threat A and Battery2 s shooting at the threat D amounts to 206.8(= 100 x 0.94 + 120 x 0.94). This value is entered in the payoff matrix, such as one on top in Figure 5. In this payoff matrix the rows represent Batteryl's alternative actions of shooting at .4, B, and not shooting at all (5), respectively, and the columns represent the alternative actions of Battery2.

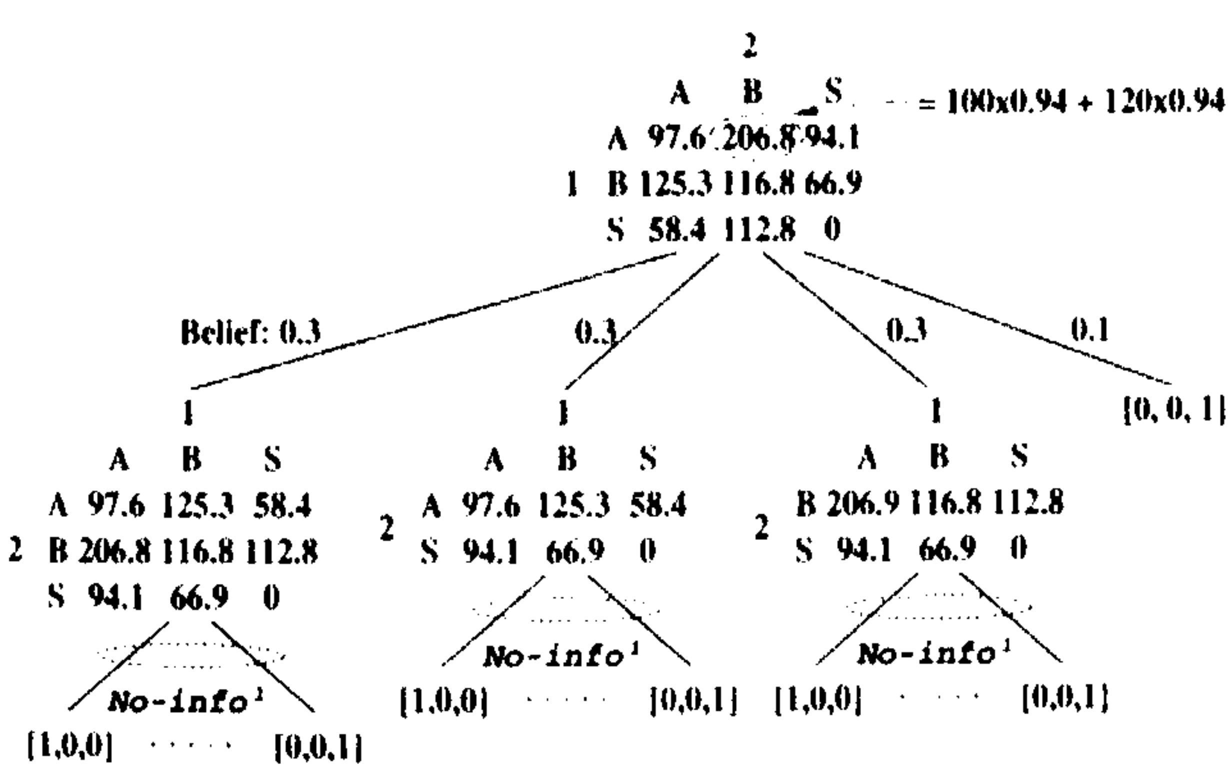


Figure 5: RMM's recursive model structure for Battery l's decision making.

The actual behavior of Battery2, however, depends on a number of factors that Battery 1 may be uncertain about. For example, if Battery2 has been hit and incapacitated, it will not be able to launch any interceptors. If it is not incapacitated then its own decision-making situation can be represented as another payoff matrix. Further, Battery2 may have run out of long-range or short-range interceptors. If Battery2 has only long-range interceptors, it would be unable to attack target B, and can only attempt to shoot down target A. If Battery2 has only short-range interceptors, it can only attempt to shoot at target B. These four models, of Battery2 being fully operational and having long- and short-range interceptors, operational with only long-range interceptors, having only short-range interceptors, and incapacitated, are depicted as the second level models in Figure 5, with their associated probabilities, in this example case 0.3, 0.3, 0.3, and 0.1, respectively.

The Recursive Modeling Method uses dynamic programming [Gmytrasiewicz, 199G; Noh and Gmytrasiewiez, 1997] to process model structures as in Figure 5 and determine the rational choice of coordinated action. In this case, Battery 1 computes that if Battery2 is fully operational then the probability distribution over Battery2's actions A, B, and 5 is [0.03,0.97,0.0]. If Battery 2 has only long-range interceptors it will choose to shoot at target .4, i.e., the probability distribution over Battery2's actions becomes [1,0,0]. If Battery 2 has only short-range interceptors it, will intercept target B. These probability distributions are combined with the model of Battery2 being incapacitated:

$$0.3 \times [0.03, 0.97, 0] + 0.3 \times [1, 0, 0] + 0.3 \times [0, 1, 0] + 0.1 \times [0, 0, 1] = [0.31, 0.59, 0.10].$$

The resulting distribution is Batteryl's overall expectation of Battery2's actions, given all of the remaining uncertainties. Propagating these results to Level 1, the combined probability distribution describing Battery2's actions is used to compute the expected utilities of Battery l's alternative actions. We have:

$$EU(A) = 0.31 \times 97.6 + 0.59 \times 206.9 + 0.10 \times 94.1 = 161.74$$
$$EU(B) = 0.31 \times 125.3 + 0.59 \times 116.8 + 0.10 \times 66.9 = 114.45$$
$$EU(S) = 0.31 \times 58.4 + 0.59 \times 112.8 + 0.10 \times 0 = 84.66$$

Thus, given the uncertainties about Battery2, Batteryl's rational coordinated choice is to intercept target A.

5 Compilation of Deliberative Decisions in Air-Defense

To construct compiled rules for our agents in the coordinated defense domain, we used four machine learning algorithms: Hayes Classifier, C4.5 [Quinlan, 1993], CN2 [Clark and Niblett, 1989], and FOIL [Cameron-Jones and Quinlan, 1994].[3] The input data for the learning

[2] In the cooperative weapon-target allocation problem, air defense units calculate probability of kill to evaluate target, survivability depending upon situation-specific characteristics. Here, the interception probability is a probability of kill, and is based on our heuristic factors.

[3] We implemented the anti-air defense domain with Common LISP on top of the MICE simulator [Durfee and Montgomery, 1989] on a LINUX machine, and also implemented a simple Bayesian classifier described in [Clark and Niblett, 1989].

algorithms were obtained from the Recursive Modeling Method (RMM), as described above. For the Bayesian classifier, the results are represented as rules specifying the probability of occurrence of each attribute value given a class [Clark and Niblett, 1989], in our case "Yes"[1] (also called *Select. Target* below) and "No" *{Don't Select Target* below). C4.5 represents its output as a decision-tree, and the output of CN2 is an ordered set of if-then rules. The trained results of FOIL are the relations of attributes as function-free Horn clauses. We now describe the agent's compiled knowledge by using the above learning algorithms, and compare their decision capabilities in the anti-air defense environment.

5.1 Learned Condition-Action Rules

In our experiments we considered agents that vary in their capacities, and they have limited knowledge about other agents. They decide on which behavior to execute based upon their sensory input and the limited information they have about the other agents.

The attributes of situations that the agents can sense in an anti-air defense environment are summarized in Table 1. They include the size, speed and angle of the attacking targets, the agent's own intercepting capabilities (i.e., its possessing both long- and short-range interceptors, only long-range interceptors, only short-range interceptors, and its being incapacitated and unable to shoot), and the probabilities associated with the capabilities of the other defense agent[4] (the other agent's possessing both long- and short-range interceptors, only long-range interceptors, only short-range interceptors, and its being incapacitated and unable to shoot). During the experiments the values of the attributes were randomly generated within the ranges of values.

Table 1: *condition*(a_i^k, S), describing the relevant attributes of targets in the anti-air defense.

Attribute	Type	Value
Target Size	numeric	100 - 300
Target Speed	nominal	slow, mid, fast
Target Angle	numeric	0 - 90
Distance	numeric	0 - 40
Capacity	nominal	both, long, short, incap.
P of both amino	numeric	0.0 - 1.0
P of only long	numeric	0.0 - 1.0
P of only short	numeric	0.0 - 1.0
P of incap.	numeric	0.0- 1.0

Based on the attributes in Table 1, the targets the defense agent considers for interception can be classified into two classes: $\{Select\ Target, Don't\ Select\ Target\}$. As an example, a decision tree obtained using C4.5 for these attributes is depicted in Figure 6.

Table 1 describes parameters for two agents and the generalization to a set of agents requires an additional set of probability parameters.

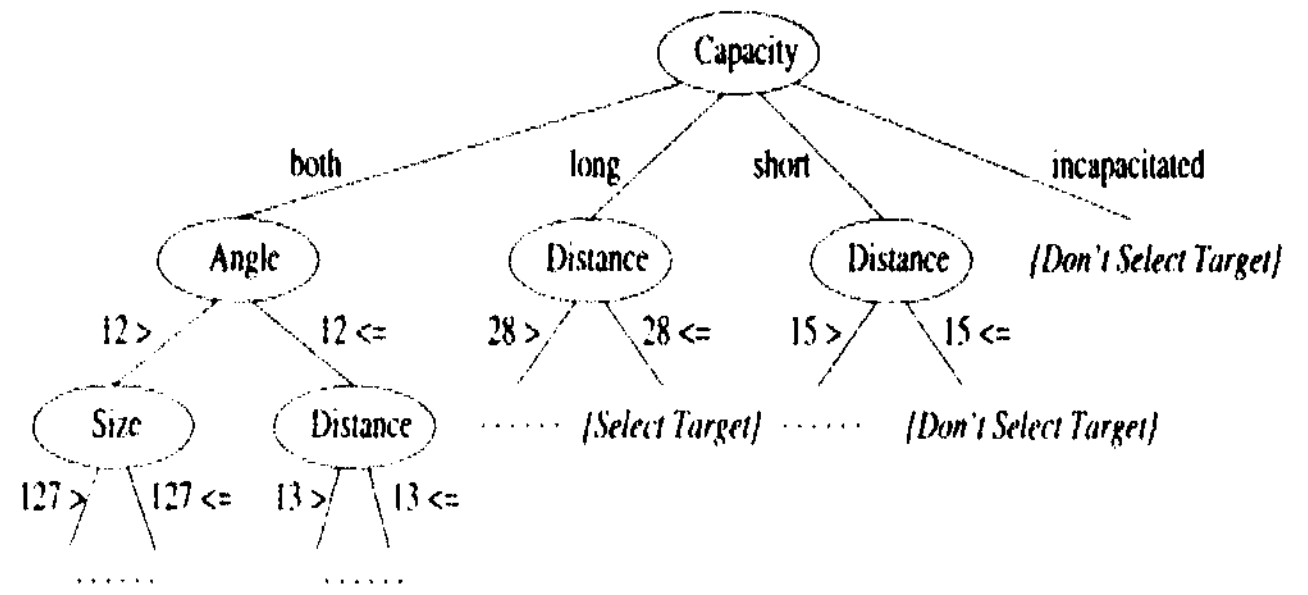


Figure 6: The decision tree obtained by C4.5.

5.2 Experiments and Performance Results

To evaluate the quality of various rule sets generated by different learning algorithms the performance obtained was expressed in terms of the total expected damage to friendly forces after launching interceptors. The total expected damage is defined as a sum of the residual warhead sizes of the attacking targets. Thus, if a target was aimed for interception, then it contributed ((1 − *interception_probability)* × *warhead_size)* to the total damage. If a target was not intercepted, it contributed all of its warhead size value to the damage.

To find a meaningful size of the training set which could guarantee the soundness of the learning hypothesis, we generated several sets of training examples. As the number of the examples increased, the resulting performances improved drastically up to a certain point, after which performance did not improve. In anti-air defense scenarios that included two batteries and two targets the sufficient number of training instances we found was 250 examples. By using the compiled condition-action rules obtained by different learning methods, we tested the performances of the methods on a new set of 250 cases. The results of performance (damage's) and runtime (sec.) are described in Table 2.

Table 2: Performance and runtime of algorithms in the two units and two targets setting.

Methods	Perf(Damages)	Time
Deliberative		
RMM	199.4 ± 97.9	0.12 ±0.09
Reactive		
Daves	229.5 ± 105.9	0.07 ± 0.07
C4.5	241.6 ± 102.2	0.03 ± 0.01
FOIL	244.1 ± 104.8	0.08 ± 0.08
' CN2	245.7 ± 100.5	0.04 ± 0.04
ANOVA	8.935	162.173

We analyzed the performance results in Table 2 using the standard analysis of variance (ANOVA) method. Since the computed value of / — 8.94 in ANOVA exceeds $3.32 (= f_{.01,4,\infty})$, we know that the five teams were not all equally effective at the 0.01 level of significance, i.e., the differences in their performance were not due to chance with probability of 0.99. As we expected, the pure deliberative procedure RMM showed the best performance. The Bayesian classifier computes the probabilities for two classes, and enables the defense agents to select the target which has the highest probability of being in the class of targets to be intercepted. The Bayesian classifier showed reasonable performance and runtime. When the pure reactive procedures, C4.5, CN2, and FOIL, were used, they could not uniquely decide the target in some cases, if the sensory input values of attributes were similar. If the defense agents still were ambiguous in target interception after applying condition-action rules, they randomly selected the target. The agent's performance by using C4.5 was better than those of FOIL and CN2 while it took less runtime.

As expected, RMM required the longest runtime, and C4.5 needed the shortest runtime among the five decision procedures. A NOVA revealed that the differences in running time were not due to chance with probability 0.99 again. When making a decision, the defense agents compared the current sensory input with one of the condition-action rules to classify a new example. The learning results obtained from CN2 and FOIL were represented by the sequential comparisons while C4.5 used the decision tree. Due to this difference CN2 and FOIL took longer to run than C4.5. The advantage of decision tree was in that it reduced the number of matches by disregarding nodes in the tree unrelated to the input.

To measure the efficiency of an adaptive and deliberative agent architecture which uses the reactive rules to filter the alternatives considered by a deliberative procedure, we experimented with the scaled-up setting depicted in Figure 3. In this setting, there were six batteries and 18 targets. The RMM agents we implemented for these experiments modeled only their closest neighbors for coordinated decision-making to reduce their deliberation time. Since, as shown in Tabic[1] 2, the Bayes classifier and C4.5 performed best among the reactive rule sets, we used these two as filtering criteria for ADA agents. As training data for the two learning algorithms, we generated 13800 tuples, which consist of 10200 for *Don't Select Target* class, and 3600 for *Select Target* class. The results of performance (damages) and runtime (sec.) in the scaled-up setting are described in Table 3.

Table 3: Performance and runtime of algorithms in the scaled-up setting.

Methods	Perf(Damages)	Time
RMM	1141.5 ± 261.8	88.69 ± 1.30
Bayes-RMM	1210.9 ± 259.7	9.23 ± 1.04
C4.5-RMM	1238.9 ± 233.2	9.31 ± 0.63
Bayes	1435.1 ± 267.4	1.91 ± 0.08
C4.5	1714.5 ± 236.3	0.87 ± 0.06

Table 3 presents the average total expected damage after 200 trials. We focus on the performances of three different agents: RMM, Bayes-RMM (Bayesian rules used to filter alternatives for deliberations using RMM), and C4.5-RMM. The performance of the RMM agent was, again, the best. The performances of Bayes-RMM and C4.5-RMM agent were 94.3% and 92.1% of the RMM agent's performance, respectively. Further, the runtimes of adaptive and deliberative agents were drastically reduced. The size of the set of filtered plausible alternative actions were 7, and 9 out of 18 targets for Bayesian classifier and C4.5, respectively. The total runtime for Bayes-RMM agent was 9.23 on the average, and the C4.5-RMM agent needs 9.31 seconds for its target selection in the simulated anti-air defense environment. This result indicates that the combination of reactive and deliberative decision-making procedures saves the agent's deliberation time while offering good performance, compared with pure deliberative procedure. Also, among the reactive procedures, the performance of Bayesian classifier was better than that of C4.5 since it included the additional probabilistic information. Since the defense batteries controlled by C4.5 alone randomly selected the targets after filtering out unprofitable targets, its performance was the worst.

6 Conclusions

We investigated a set of condition-action rule sets achieved by compiling decision-theoretic reasoning implemented in RMM method using various learning algorithms. We found that the compiled rules reduce the complexity and running time in complex multi-agent scenarios. This approach enables an adaptive and deliberative agent to reach a decision within a reasonable time period.

We experimented with the anti-air defense domain to assess the quality of the flexible decision-making procedure. Anti-air defense is certainly a real-time task, and any overhead or loss of timely response can result in additional damages. The combination of reactive and deliberative decision-making methods avoided catastrophic failure, and provided good-quality decisions in the time-constrained anti-air defense.

In our future research, we will consider how an autonomous agent can decide on its coordinated action in an any-time fashion [Ilorvitz, 1988; Horvitz *et* al., 1989; Dean and Boddv, 1988]. Our framework will provide the rational action under uncertain deadline by calculating the computational gain, representing the tradeoff between the costs and the benefits of computation.

Acknowledgments

We are grateful to Dr. Lawrence B. Holder and the anonymous referees for providing useful comments relating to this work. This research has been sponsored by the Office of Naval Research Artificial Intelligence Program under contract N00014-95-1-0775 and by the National Science Foundation CAREER award IRI-9702132.

References

[Bratman *et al,* 1988] M. E. Bratman, D. J. Israel, and M. E. Pollack. Plans and resource-bounded practical reasoning. *Journal of Computational Intelligence,* 4:349 355, 1988.

[Cameron-Jones and Quinlan, 1994] R. M. Cameron-Jones and J. R. Quinlan. Efficient top-down induction of logic programs. *SIGART Bulletin,* 5(1):33 42, January 1994.

[Clark and Niblett, 1989] P. Clark and T. Niblett, The CN2 induction algorithm. *Machine Learning Journal,* 3(4):261-283, 1989.

[Dean and Boddy, 1988] T. Dean and M. Boddy. An analysis of time-dependent planning. In *Proceedings of the 7th National Conference on Artificial Intelligence,* pages 49 54, St. Paul, Minnesota, August 1988.

[Durfee and Montgomery, 1989] Edmund H. Durfee and Thomas A. Montgomery. MICE: A flexible testbed for intelligent coordination experiments. In *Proceedings of the 1989 Distributed AI Workshop,* pages 25-40, September 1989.

[Ephrati *et ai,* 1995] E. Ephrati, M. E. Pollack, and S. Ur. Deriving multi-agent coordination through filtering strategies. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence,* pages 679-685, Montreal, Canada, August 1995.

[Fox and Krause, 1991] J. Fox and P. Krause. Symbolic decision theory and autonomous systems. In *Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence,* pages 103-110, UCLA, CA,'July 1991.

[Gmytrasiewicz *et ai,* 1998] P. J. Gmytrasiewicz, S. Noli, and T. Kellogg. Bayesian update of reeusive agent models. *An International Journal of User Modeling and User-Adapted Interaction,* 8(l/2):49-69, 1998.

[Gmytrasiewicz, 1996] P. J. Gmytrasiewicz. On reasoning about other agents. In *Intelligent Agents II: Agent Theories, Architectures, and Languages,* pages 143 155, Berlin, 1996. Springer.

[Horvitz *et* al., 1989] E. J. Horvitz, G. F. Cooper, and D. E. Heckerman. Reflection and action under scarce resources: Theoretical principles and theoretical study. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence,* pages 1121 1127, Detroit, Michigan, August 1989.

[Horvitz, 1988] E. J. Horvitz. Reasoning under varying and uncertain resource constraints. In *Proceedings of the 7th National Conference on Artificial Intelligence,* pages 111-116, St. Paul, Minnesota, August 1988.

[Macfadzean, 1992] R, H. M. Macfadzean. *Surface-Based Air Defense System Analysis.* Artech House, 1992.

[Noh and Gmytrasiewicz, 1997] S. Noh and P. J. Gmytrasiewicz. Agent modeling in antiair defense. In *Proceedings of the 6th International Conference on User Modeling,* pages 389-400, Sardinia, Italy, June 1997.

[Noh and Gmytrasiewicz, 1998] S. Noh and P. J. Gmytrasiewicz. Rational communicative behavior in antiair defense. In *Proceedings of the 3rd International Conference on Multi Agent Systems,* pages 214-221, Paris, France, July 1998.

[Quinlan, 1993] J. R. Quinlan. *C4-5: Programs for Machine Learning.* Morgan Kaufmann Publishers, 1993.

[Rao and Georgeff, 1995] A. S. Rao and M. P. Georgeff. BDI agents: From theory to practice. In *Proceedings of the 1st International Conference on Multiagent Systems,* pages 312-319, July 1995.

[Russell and Subramanian, 1995] S. J. Russell and D. Subramanian. Provably bounded-optimal agents. *Journal of Artificial Intelligence Research,* 2:575-609, 1995.

[Russell and Wefald, 1991] S. J. Russell and E. H. Wefald. Principles of metareasoning. *Artificial Intelligence,* 49:361 395, 1991.

[Russell, 1989] S. J. Russell. Execution architectures and compilation. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence,* pages 15 20, Detroit, Michigan, August 1989,

[Zilberstein and Russell, 1996] S. Zilberstein and S. J. Russell. Optimal composition of real-time systems. *Artificial Intelligence,* 82(1):181 213, 1996.

[Zilberstein, 1995] S. Zilberstein. Models of bounded rationality. In *AAAI Fall Symposium on Rational Agency,* Cambridge, Massachusetts, November 1995.