Computational Intelligence, Volume 21, Number 4, pp.462-479, November 2005

THE DISCIPLE-RKF LEARNING AND REASONING AGENT

Gheorghe Tecuci, Mihai Boicu, Cristina Boicu,

Dorin Marcu, Bogdan Stanescu, Marcel Barbulescu

MSN 4A5, Learning Agents Center and Computer Science Department,

George Mason University, 4400 University Drive, Fairfax, VA 22030, USA

{tecuci, mboicu, ccascava, dmarcu, bstanesc}@gmu.edu, mbarb@cs.gmu.edu

http://lac.gmu.edu, tel: 1 703 993-1722, fax: 1 703 993-1710

Abstract: Over the years we have developed the Disciple theory, methodology, and family of tools for building knowledge-based agents. This approach consists of developing an agent shell that can be taught directly by a subject matter expert in a way that resembles how the expert would teach a human apprentice when solving problems in cooperation. This paper presents the most recent version of the Disciple approach and its implementation in the Disciple-RKF system. Disciple-RKF is based on *mixed-initiative problem solving*, where the expert solves the more creative parts of the problem and the agent solves the more routine ones, *integrated teaching and learning*, where the agent helps the expert to teach it, by asking relevant questions, and the expert helps the agent to learn, by providing examples, hints and explanations, and *multistrategy learning*, where the agent integrates multiple learning strategies, such as learning from examples, learning from explanations, and learning by analogy, to learn from the expert how to solves problems. Disciple-RKF has been applied to build learning and reasoning agents for military center of gravity analysis, which are used in several courses at the US Army War College.

Key Words: multistrategy apprenticeship learning, task reduction, mixed-initiative reasoning, plausible version spaces, rule learning, ontology, military center of gravity analysis

1. INTRODUCTION

For almost 20 years we have performed research on developing a theory and the associated methodologies and tools for building agents that incorporate the knowledge of a subject matter expert (Tecuci 1988, 1998; Boicu 2002). The resulting approach to this problem, which we have called Disciple, consists of developing a problem solving and learning agent that can be taught directly by a subject matter expert to become a knowledge-based assistant. The expert should be able to teach the agent to perform problem solving tasks in a way that is similar to how the expert would teach a person. For instance, the expert may show the agent how to solve specific

problems, and may help it to understand the reasoning process. As the agent learns general problem solving rules from these problem solving examples and builds its knowledge base, the expert-agent interaction evolves from a teacher-student interaction toward an interaction where both collaborate in solving a problem. During this joint problem solving process, the agent learns not only from the contributions of the expert, but also from its own successful or unsuccessful problem solving attempts. Over the years we have continuously extended and improved the Disciple approach, which is reflected in a sequence of increasingly more powerful problem solving and learning agents from the Disciple family of agents. The goal of this paper is to provide an overview of the knowledge representation, problem solving and learning methods of the most recent member of this family, Disciple-RKF, which has been developed as part of the DARPA's Rapid Knowledge Formation (RKF) program (Tecuci et al., 2001, 2002).

2. APPLICATION DOMAIN: MILITARY CENTER OF GRAVITY ANALYSIS

Military center of gravity analysis was used as a challenge problem in the DARPA's RKF program to test the knowledge acquisition, learning, and problem solving methods of Disciple-RKF, and it will be used in this paper to illustrate these methods. The concept of center of gravity, introduced by Karl von Clausewitz (1832), is fundamental to military strategy, denoting the primary source of moral or physical strength, power or resistance of a force (Strange, 1996). The most important objective of a force (state, alliance, coalition, or group) in any type of conflict is to protect its own center of gravity while attacking the center of gravity of its enemy. Therefore, in the education of strategic leaders at all the U.S. senior military service colleges, there is great emphasis on the center of gravity analysis. This analysis requires a wide range of background knowledge not only from the military domain, but also from the political, psychosocial, economic, geographic, demographic, historic, international, and other domains

(Giles and Galvin 1996). In addition, the situation, the adversaries involved, their goals, and their capabilities can vary in important ways from one scenario to another. Therefore, this is a very good example of knowledge-intensive, expert problem solving that a Disciple agent should be able to learn.

Our approach to center of gravity analysis, based on the work of Strange (1996) and Giles and Galvin (1996), and developed with experts from the US Army War College, consists of two main phases: identification and testing. During the identification phase, center of gravity candidates from different elements of power of a force (such as government, military, people, economy) are identified. For instance, a strong leader is a center of gravity candidate with respect to the government of a force. Then, during the testing phase, each candidate is analyzed to determine whether it has all the critical capabilities that are necessary to be the center of gravity. For example, a leader needs to be protected, stay informed, communicate (with the government, the military, and the people), be influential, be a driving force, have support, and be irreplaceable. For each capability, one needs to determine the existence of the essential conditions, resources, and means that are required by that capability to be fully operative, and which of these, if any, represent critical vulnerabilities.

3. AGENT ARCHITECTURE

The architecture of Disciple-RKF includes the components from Figure 1, each implemented as a set of collaborative agents (Boicu et al., 2004). The core of the system is the learning agent shell, which consists of domain-independent components that will be part of any Disciple agent.

The three components in the right hand side of Figure 1 are the typical domain dependent components of a Disciple-RKF agent that was customized for a specific application, such as center of gravity analysis.

4. PROBLEM SOLVING

A Disciple-RKF agent performs problem solving tasks by using the task reduction paradigm (Nilsson 1971; Powell and Schmidt 1988). In this paradigm, a complex problem solving task is successively reduced to simpler tasks. Next the solutions of the simplest tasks are found and these solutions are successively combined into the solution of the initial task. In the Disciple approach, we have refined this general strategy so that it can be easily used both by the expert (when teaching the agent or when contributing to the joint problem solving process) and the agent (when solving a problem). We did this by introducing questions and answers that guide the task reduction process, as illustrated in Figure 2 and discussed in more detail in (Bowman 2002). In this refined task reduction approach, finding a solution to a problem solving task (e.g. "Determine a center of gravity for the Sicily 1943 scenario") becomes an iterative process where, at each step, the expert (or the agent, depending on who is doing the problem solving) looks for some relevant information for solving this task by asking a question (e.g. "Which is an opposing force in the Sicily 1943 scenario?"). The answer (Allied Forces 1943) identifies that piece of information and leads to the reduction of the current task to one or several simpler tasks (e.g. "Determine a center of gravity for Allied Forces 1943"). Alternative questions correspond to alternative problem solving strategies; multiple answers of a question correspond to multiple solutions. Solution composition is also guided by questions and answers.

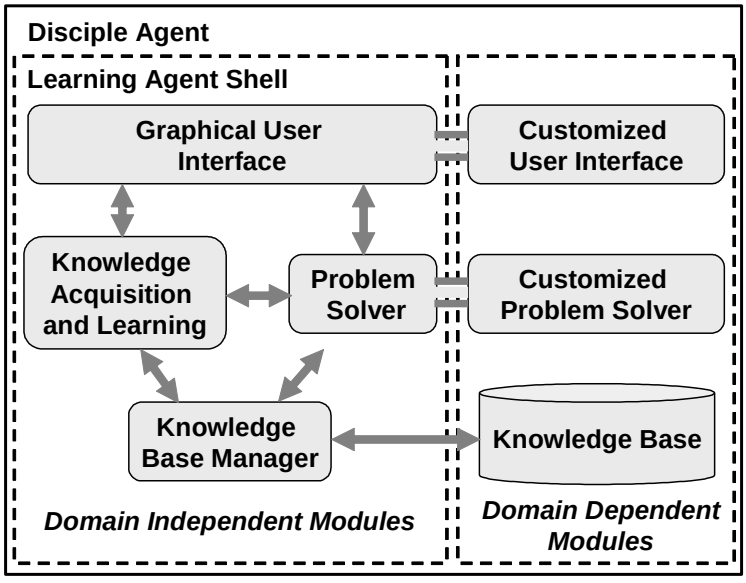

Figure 1: General architecture of a Disciple agent.

5. LEARNABLE KNOWLEDGE REPRESENTATION

The knowledge base of Disciple-RKF is structured into an object ontology and a set of task reduction rules and solution composition rules. The object ontology is a hierarchical representation of the objects from the application domain. It represents the different kinds of objects, the properties of each object, and the relationships existing between objects. A fragment of this object ontology for the center of gravity domain is shown in Figure 3. In addition to the hierarchy of instances and concepts illustrated in Figure 3, the object ontology also includes a hierarchy of features. In this hierarchy the feature "has_as_head_of_government" is a subfeature of "has_as_political_leader," which is a subfeature of "has_as_controlling_leader." Each feature F is characterized by a domain and a range. The domain of F is a concept that represents all objects that may have the feature F. The range of F is a concept that represents all the possible values of F. The concepts from the object ontology are used to define more complex concepts. The basic representation unit (BRU) for such a concept has the form $\{?O_1, ?O_2, \ldots, ?O_n\}$, where each $?O_i$ has the structure indicated by [1].

$$\begin{array}{lll} ?O_i & \text{is} & \text{concept}_i \\ & \text{feature}_{i1} & ?O_{i1} \\ & \ldots & \\ & \text{feature}_{in} & ?O_{im} \end{array} \qquad [1]$$

Concept_i is an object concept from the object ontology, a numeric interval, or a list of strings, and $?O_{i1} \ldots ?O_{im}$ are distinct variables from the set $\{?O_1, ?O_2, \ldots, ?O_n\}$. In general, a concept may be a conjunctive expression of form [2], meaning that any instance of the concept satisfies BRU and does not satisfy BRU_1 and … and does not satisfy BRU_p.

$$\text{BRU \& Except When } BRU_1 \text{ \& } \ldots \text{ \& Except When } BRU_p \qquad [2]$$

For instance, the concept from [3] represents "the pair of entities $?O_1$ and $?O_2$, where $?O_1$ is an equal partner multi-state alliance that has, as one of its members, $?O_2$, which is single-state force,

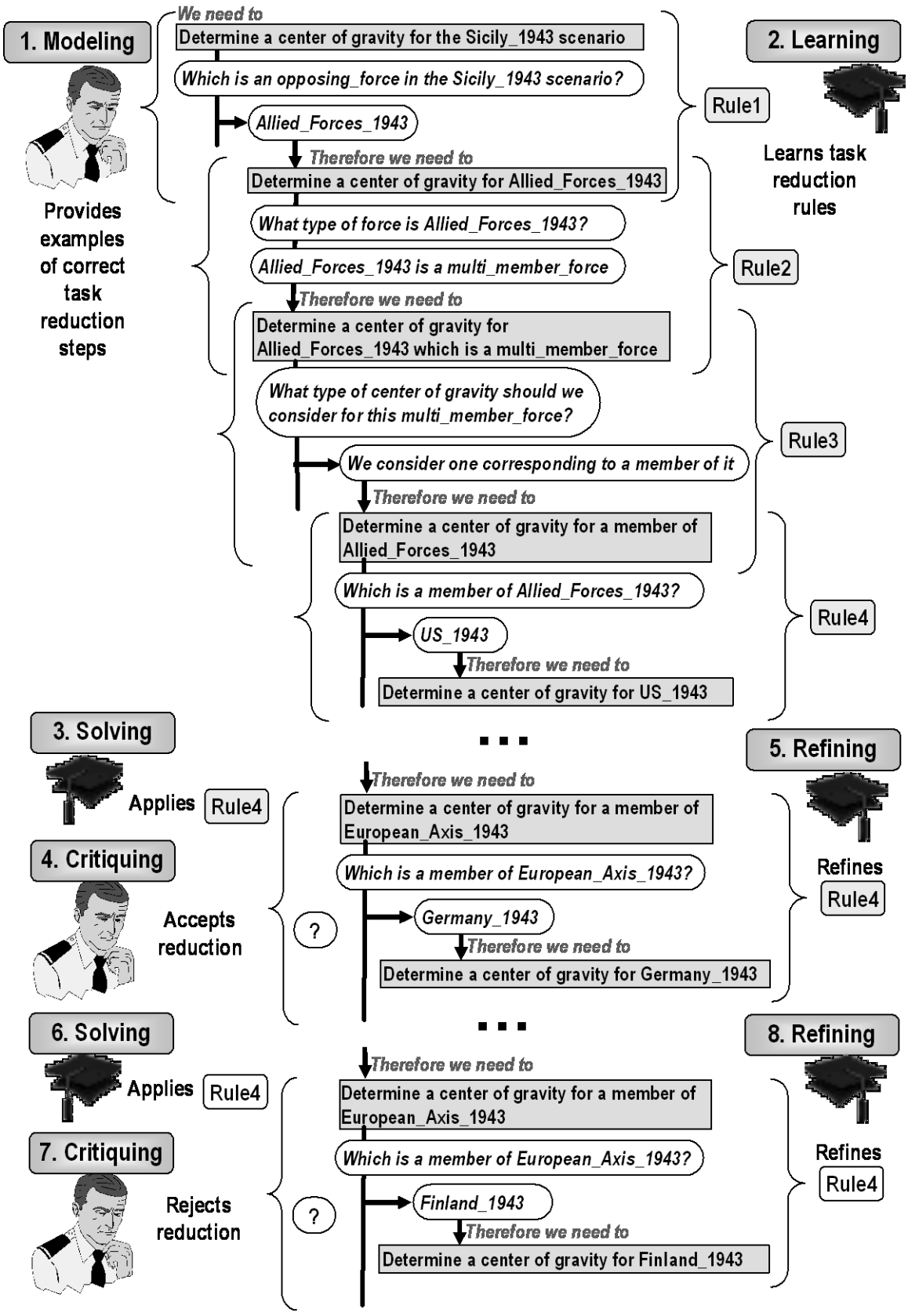

Figure 2: An illustration of mixed-initiative modeling, problem solving, and learning.

except when ?O_2 is a single-state force with a minor military contribution."

?O_1	is	equal_partners_multi_state_alliance	[3]
	has_as_member	?O_2	
?O_2	is	single_state_force	
Except When			
?O_2	is	single_state_force	
	has_as_military_contribution	?O_3	
?O_3	is	minor_military_contribution	

The object ontology is at the basis of the generalization language for learning. For instance, a concept such as [3] may be generalized by replacing an object concept from its description (e.g. "equal_partners_multi_state_alliance") with a more general concept from the ontology (e.g. "multi_state _alliance"). Other generalization or specialization rules consist of dropping or adding an object feature or an Except When condition, generalizing a number to an interval, or generalizing an interval to a larger interval (Tecuci, 1998). Partially learned concepts are represented as plausible version spaces (Tecuci, 1998), as illustrated in Figure 4. The plausible upper bound of this version space contains two concepts, one where ?O_1 is a multi member force,

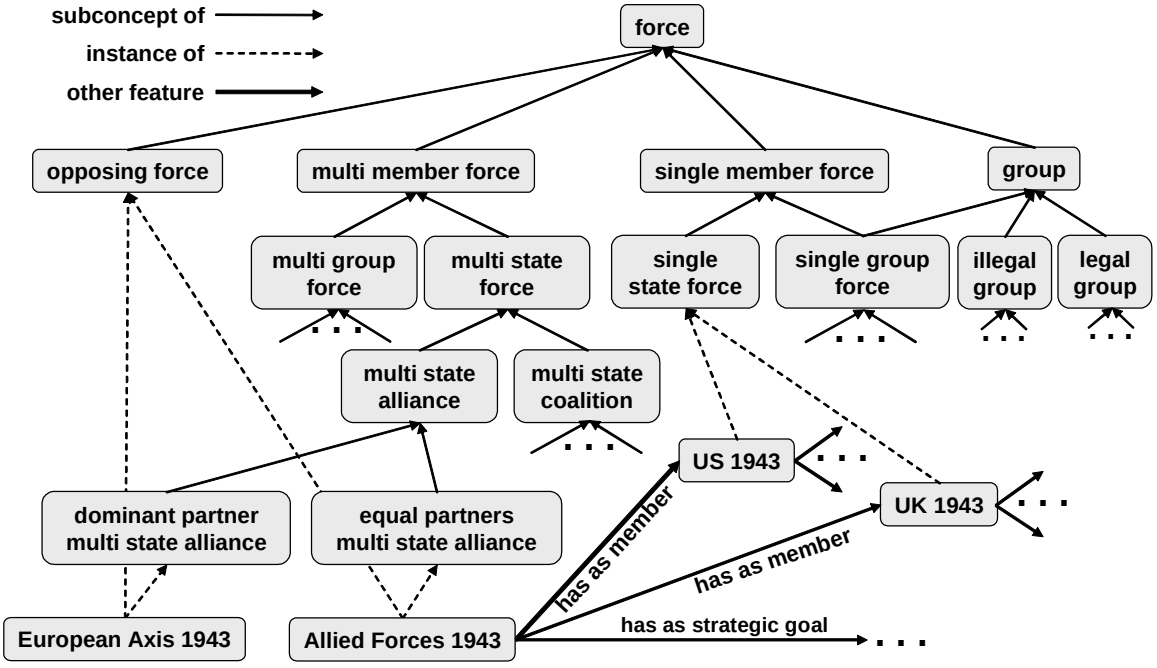

Figure 3: Fragment of the object ontology for the center of gravity domain.

and the other where $?O_1$ is an opposing force. Similarly, the plausible lower bound of this version space contains two concepts. In the current version of Disciple, the same features appear both in the upper bound and in the lower bound (such as "has_as_member" in Figure 4).

The concept E_h to be learned (see Figure 4) is, ***as an approximation***, less general than ***one*** of the concepts from the plausible upper bound. E_h is also, again, ***as an approximation***, more general than ***any*** of the concepts from the plausible lower bound. During learning, the two bounds converge toward one another through successive generalizations and specializations, approximating E_h better and better. This is different from the version spaces introduced by Mitchell (1978), where ***one*** of the concepts from the upper bound ***is always more general*** than the concept to be learned (and the upper bound is always specialized during learning), and ***any*** of the concepts from the lower bound ***is always less general*** than the concept to be learned (and the lower bound is always generalized during learning). The major difference is that the version spaces introduced by Mitchell (1978) are based on a complete representation space that includes the concept to be learned. On the contrary, the representation space for Disciple is based on an incomplete object ontology. Indeed, there are relevant concepts and instances from the application domain which are not represented. Moreover, the representation of a given concept or instance may be incomplete in the sense that it does not include all of its relevant properties and relationships. This object ontology will be extended by the agent during the problem solving

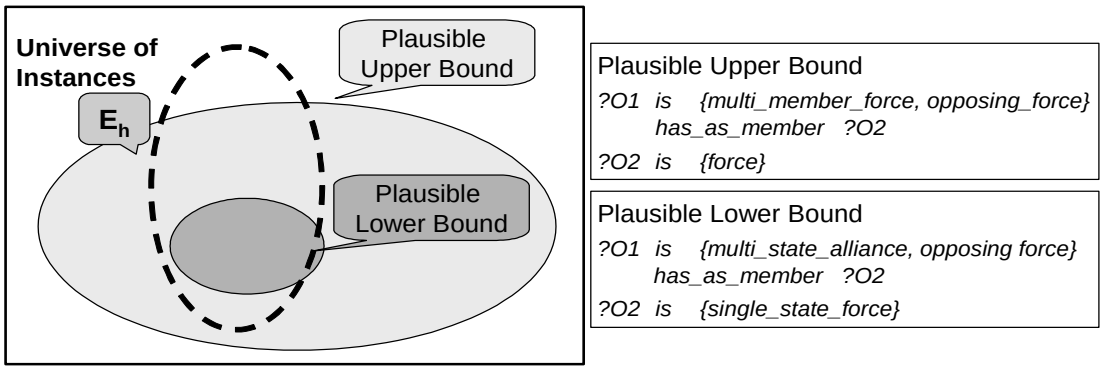

Figure 4: A plausible version space for a partially learned

and learning process (Boicu et al., 2003).

The notion of plausible version space is fundamental to the knowledge representation, problem solving, and learning methods of Disciple, as discussed below and in section 6. All the knowledge elements from the knowledge base are represented using this construct and are learned or refined by Disciple. For instance, Disciple-RKF learns general feature definitions from specific facts. The domains and the ranges of the partially learned features are represented as plausible version spaces. The knowledge base of Disciple-RKF also contains tasks reduction rules and solution composition rules. Figure 5 shows an example of a task reduction step and the task reduction rule learned from it. The rule is an IF-THEN structure that expresses under what condition a certain type of task may be reduced to a simpler subtask (or to several subtasks, in case of other rules). The rule in Figure 5 is interpreted as follows: If the task to be solved is ”Determine a center of gravity for a member of ?O1,” and the applicability condition of the rule is satisfied, then we can reduce the above task to “Determine a center of gravity for ?O2.” Because the rule shown in Figure 5 is only partially learned, its applicability condition (Main condition) is not a single condition, but a plausible version space for the exact condition to be learned. The rule in Figure 5 is a very simple one, with only a Main Condition. In general, however, in addition to a Main Condition, a learned rule may have several Except When Conditions (which should not be satisfied for the rule to be applicable), as well as positive and negative exceptions. Thus, in general, the condition of the rule is a concept of the form [2], meaning that the rule may be applied for any instance of the condition concept.

6. MIXED-INITIATIVE MODELING, LEARNING AND PROBLEM SOLVING

The Disciple approach covers all the phases of agent development and use. First, a knowledge engineer works with a subject matter expert to develop an ontology for the application domain.

They use the ontology import module (to extract relevant ontology elements from existing knowledge repositories) as well as the various ontology editors and browsers of Disciple-RKF. The result of this knowledge base development phase is an object ontology (see Figure 3), which is complete enough to be used as a generalization hierarchy for learning, allowing the expert to teach the Disciple agent how to solve problems, with limited assistance from a knowledge engineer. The teaching process is illustrated in Figure 2 and discussed in the following.

6.1. Rule Learning

The expert formulates an initial problem solving task and shows the agent how to solve it by using the task reduction paradigm described in section 4. Figure 2 shows a sequence of task reduction steps. Each such step consists of a task, a question, its answer, and a subtask. From each of these steps the agent learns a general task reduction rule. Table 1 and Table 2 present the rule learning problem and method of Disciple-RKF. To illustrate them let us consider the 4th step from the task reduction tree in Figure 2. This step is also shown on the left hand side of Figure 5. From this task reduction step, Disciple-RKF learned the task reduction rule shown in the right hand side of Figure 5.

The question and its answer from the task reduction step represent the expert's reason (or explanation) for performing that reduction. Because they are in natural language, the expert has to help Disciple "understand" them in terms of the concepts and features from the object ontology. Consider [4], the question and the answer from the example in Figure 5. The meaning of [4] in the object ontology is expressed as in [5]. We call [5] the "explanation" of the example.

Which is a member of Allied_Forces_1943? US_1943 [4]

Allied_Forces_1943 has_as_member US_1943 [5]

While a subject matter expert can understand the meaning of the above formal expression, s/he cannot easily define it for the agent because s/he is not a knowledge engineer. For instance, s/he

Table 1: The Rule Learning Problem.

GIVEN: • An example of a task reduction step. • A knowledge base that includes an object ontology and a set of task reduction rules. • A subject matter expert that understands why the given example is correct and may answer the agent's questions. DETERMINE: • A plausible version space task reduction rule which is a generalization of the specific task reduction step. • An extended object ontology (if needed for rule learning).

would need to use the formal language of the agent. But this would not be enough, as the expert would also need to know the names of the potentially many thousands of concepts and features from the agent's ontology. Therefore, the agent will hypothesize plausible meanings of the question-answer pair by using simple natural language processing, analogical reasoning with previously learned rules, and general heuristics, and will express them as explanation fragments. In general, an explanation fragment identified by the agent, such as [5], is a relationship (or a relationship chain) involving instances, concepts, and constants from the task reduction step and from the knowledge base. The agent will then propose these explanation pieces to the expert, ordered by their plausibility, so that the expert can select the ones that express approximately the same meaning as the question-answer pair. The expert may also help the agent to propose the right explanation pieces by providing hints, such as pointing to a relevant object that should be part of the explanation (Boicu et al., 2000).

Using the example and its explanation, Disciple-RKF will generate the task reduction rule from the right hand side of Figure 5. First the agent will generate a variable for each instance, number, or string that appears in the example and its explanation. Then it will use these variables V to generalize the task reduction example E into an IF-THEN rule R, by replacing

Table 2: The Rule Learning Method.

1. Identify a formal explanation EX of why the example E is correct, through mixed-initiative interaction with the subject matter expert. The explanation is an approximation of the meaning of the question and answer, expressed with the objects and the features from the object ontology. During the explanation generation process, new objects and features may be elicited from the expert and added to the object ontology. 2. Generate a variable for each instance, number and string that appears in the example and its explanation. Then use these variables, the example, and the explanation, to create an instance IC of the concept representing the applicability condition of the rule to be learned. This is the concept to be learned as part of rule learning. 3. Generate the tasks, question, and answer of the rule by replacing each instance or constant from the example E with the corresponding variable generated in step 2. Then generate the plausible version space of the applicability condition of the rule. The concept represented by this condition is the set of instances and constants that produce correct instantiations of the rule. The plausible lower bound of this version space is the minimally general generalization of IC determined in step 2, generalization which does not contain any instance. The plausible upper bound of this version space is the set of the maximally general generalizations of IC. 5. If there is any variable from the THEN part of a rule which is not linked to some variable from the IF part of the rule, or if the rule has too many instances in the knowledge base, then interact with the expert to extend the explanation of the example and update the rule if new explanation pieces are found. Otherwise end the rule learning process.

each instance or concept with the corresponding variable.

The next step in the rule learning process is to determine which are the instantiations of the variables V that lead to correct task reduction steps. That is, we have to learn the concept that represents the set of instances of the rule's variables V for which the corresponding instantiation of the rule R is correct. We call this concept "the applicability condition of the rule R," and Disciple-RKF learns it by using a plausible version space approach. That is, it considers the set of all the applicability conditions that are consistent with the known examples and their explanations and it reduces this set as new examples and additional explanations are found. Moreover, as in the candidate elimination algorithm, this version space is represented by a plausible lower bound and by a plausible upper bound.

The initial plausible version space condition for the rule R is determined as follows. First one determines the instance of this condition, IC, corresponding to the initial example, as shown in the left hand side of Figure 5. Notice that this condition includes the feature "has_as_member" from the explanation of the example. This is an essential feature of the objects from this example

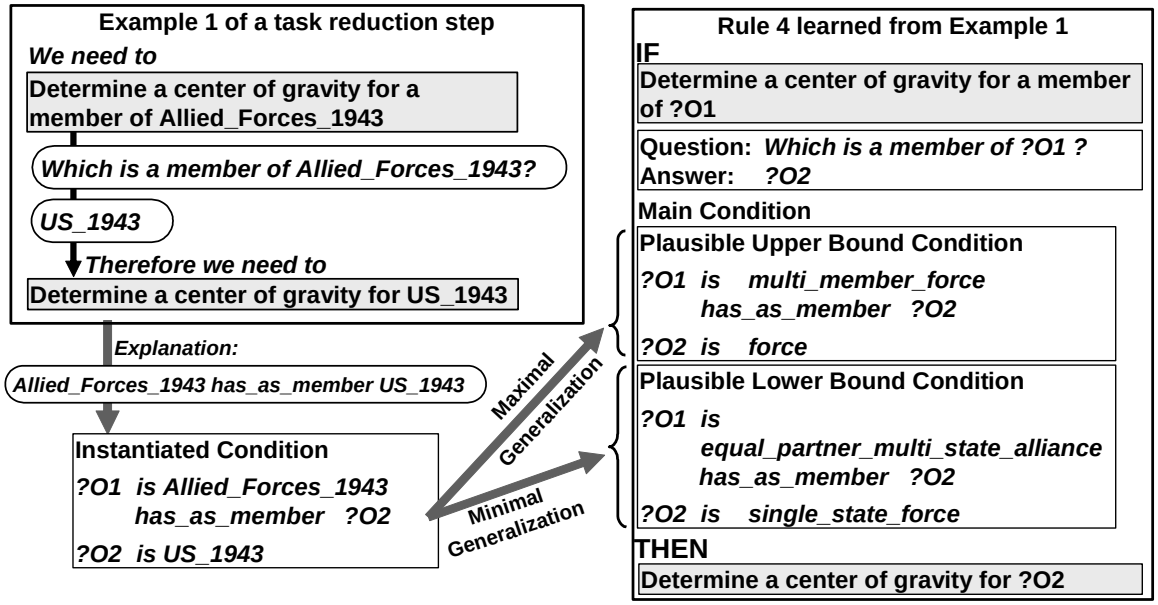

Figure 5: An example of a task reduction step and the rule learned from it.

and, for the same reason as in the case of explanation-based learning (Mitchell et al., 1986, DeJong and Mooney, 1986), it significantly reduce the number of examples needed for learning.

Then one generalizes IC in two different ways to generate the two bounds of the version space, shown in the right hand side of Figure 5, under Main Condition. The plausible lower bound is the set of the least general generalizations of IC which include no instance. The least general concepts from the object ontology that cover Allied_Forces_1943 are opposing_force and equal_partner_multi_state_alliance. However, Allied_Forces_1943 has the feature has_as_member and, therefore, any of its generalization should be in the domain of this feature, which happens to be multi_member_force. As a consequence, the set of the minimal generalizations of Allied_Forces_1943 is given by the following expression:

{opposing_force, equal_partner_multi_state_alliance} ∩ {multi_member_force} =

= {equal_partner_multi_state_alliance}

Similarly (but using the range of the has_as_member feature, which is force), one determines the set of the minimal generalizations of US_143 as {single_state_force}.

The reason the lower bound cannot contain any instance is that the learned rule will be used by Disciple in other scenarios (such as Afghanistan_2001_2002), where the instances from Sicily_1943 do not exist, and Disciple-RKF would not know how to generalize them. On the other hand, we also do not claim that the concept to be learned is more general than the lower bound, as discussed in section 5 and illustrated in Figure 4.

6.2. Rule Refinement

As Disciple-RKF learns new rules from the expert, the interaction between the expert and Disciple evolves from a teacher-student interaction toward an interaction where both collaborate in solving a problem. During this mixed-initiative problem solving phase, Disciple learns not only from the contributions of the expert, but also from its own successful or unsuccessful

Table 3: The Rule Refinement Problem.

GIVEN: • A plausible version space task reduction rule R. • A positive or a negative example E of the rule (i.e. a correct or an incorrect task reduction step that has the same IF and THEN tasks as R). • A knowledge base that includes an object ontology and a set of task reduction rules. • A subject matter expert that understands why the task reduction step is correct or incorrect and can answer the agent's questions. DETERMINE: • A refined rule that covers the example if it is positive, or does not cover the example if it is negative. • An extended object ontology (if needed for rule refinement).

problem solving attempts, which leads to the refinement of the learned rules. At the same time, Disciple may extend the object ontology with new objects and features.

The rule refinement problem and methods are presented in Tables 3, 4 and 5. The result of the rule learning process described in Table 2 and illustrated above is a rule with a main plausible version space condition. During rule refinement, however, the rule may accumulate several Except When plausible version space conditions. For that reason, Table 3 and Table 4 assume that the rule R to be refined, based on the example E, has both a partially learned main condition and a partially learned Except When condition. These methods are extended naturally when there are several such conditions. Figure 7 shows an abstract representation of the rule's conditions. The new (positive or negative) example of the rule (and of the rule's condition) may be situated in one of several relevant regions, as indicated in Figure 6. The way the rule is refined depends on the type of the example and the region in which it is situated, as described in Tables 3 and 4. In the following, we will briefly illustrate these methods.

As indicated in Figure 2, Disciple-RKF applied Rule 4 to reduce the task "Determine a center of gravity for a member of European_Axis_1943," generating an example that is covered

Table 4: Rule refinement with a positive example.

1. If the positive example E is covered by ML and is not covered by XU (case 1 in Figure 6), then the rule does not need to be refined because the example is correctly classified as positive by the current rule. 2. If E is covered by MU, but it is not covered by ML and XU (case 2 in Figure 6), then minimally generalize ML to cover E and remain less general than MU. Remove also from MU the elements that do not cover E. 3. If E is not covered by MU (cases 3, 4, and 5 in Figure 6), or if E is covered by XL (cases 5, 6, and 7 in Figure 6), then keep E as a positive exception of the rule. 4. If E is covered by ML and XU, but it is not covered by XL (case 8 in Figure 6), then interact with the expert to find an explanation of the form: "The task reduction step is correct because I_i is C_i," where C_i is a concept from the ontology. If such an explanation is found, then XU is minimally specialized to no longer cover C_i. Otherwise, E is kept as a positive exception. 5. If E is covered by MU and XU, but it is not covered by ML and XL (case 9 in Figure 6), then minimally generalize ML to cover E and remain less general than MU. Also remove from MU the elements that do not cover E. Then continue as in step 4.

by the plausible upper bound condition of the rule. This reduction was accepted by the expert as correct. Therefore, Disciple generalized the plausible lower bound condition to cover it. For instance, European_Axis_1943 is a multi_member_force, but it is not an equal_partner_multi_state_alliance. It is a dominant_partner_multi_state_alliance dominated by Germany_1943 (see also Figure 3). As a consequence, Disciple-RKF automatically generalizes the plausible lower bound condition of the rule to cover this example. The refined rule is shown in the left-hand side of Figure 7. This refined rule is then generating the task reduction from the bottom part of Figure 2. Although this example is covered by the plausible lower bound condition of the rule, the expert rejects the reduction as incorrect. This shows that the plausible lower bound condition is not more general than the concept to be learned (as it would have been the case in the Mitchell (1978) classical candidate elimination algorithm) and it would need to be

Table 5: Rule refinement with a negative example.

1. If the negative example E is covered by ML and it is not covered by XU (case 1 in Figure 6), then interact with the subject matter expert to find an explanation of why E is a wrong task reduction step. If an explanation EX is found, then generate a new Except When plausible version space condition and add it to the rule. Otherwise, keep E as a negative exception. 2. If E is covered by MU but it is not covered by ML and by XU (case 2 in Figure 6) then interact with the expert to find an explanation of why E is a wrong task reduction step. If an explanation EX is found and it has the form "I_i is not a C_i," where C_i is a concept covered by MU, then specialize MU to be covered by C_i. Otherwise, if another type of explanation EX is found then learn a new Except When condition based on it, and add this condition to the rule. 3. If E is not covered by MU (cases 3, 4, 5 in Figure 6), or it is covered by XL (cases 5, 6, 7 in Figure 6), then the rule does not need to be refined because the example is correctly classified as negative by the current rule. 4. If E is covered by ML and XU but it is not covered by XL (case 8 in Figure 6), or E is covered by MU and XU but it is not covered by ML and XL (case 9 in Figure 6), then minimally generalize XL to cover E and specialize XU to no longer include the concepts that do not cover E.

specialized. This rejection of the reduction proposed by Disciple-RKF initiates an explanation generation interaction during which the expert will have to help the agent understand why the reduction step is incorrect. The explanation of this failure is that Finland_1943 has only a minor military contribution to European_Axis_1943 and cannot, therefore, provide the center of gravity of this alliance. The actual failure explanation (expressed with the terms from the object ontology) has the form: "Finaland_1943 has_as_military_contribution military_contribution_of_Finaland_1943 is minor_military_contribution." Based on this failure explanation, Disciple_RKF generates a plausible version space Except When condition and adds it to the rule, as indicated in the right hand side of Figure 7. In the future, this rule will only apply to situations where the main condition is satisfied and the Except When condition is not satisfied.

Notice that the addition of the Except When condition specializes both bounds of the applicability condition. Other types of failure explanations may lead to different modifications of

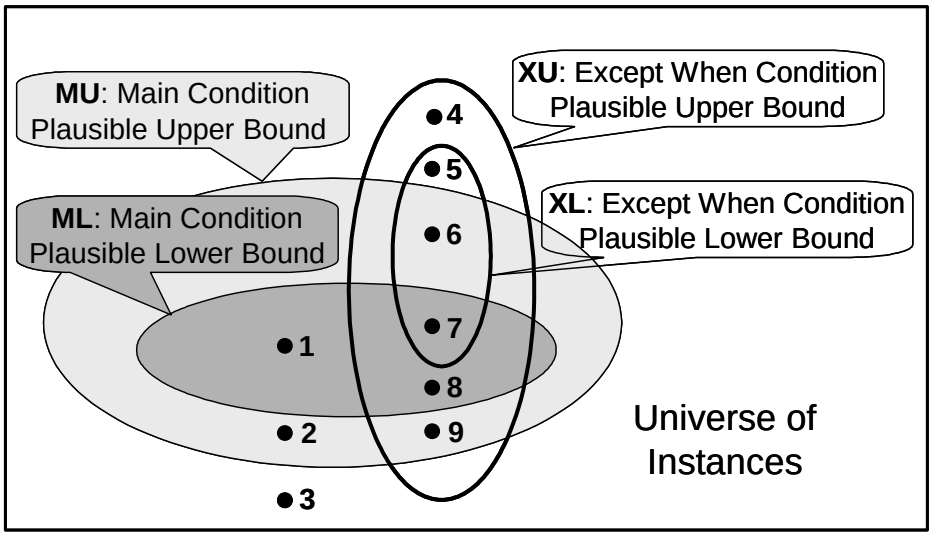

Figure 6: Possible regions for a new (positive or negative) example of a rule.

Rule 4. For instance, the failure explanation may have had the form "Finland_1943 is not a major_ally," if "major_ally" would have been part of the object ontology (which it is not). In such a case, Disciple-RKF would not add an Except When condition but it would specialize both bounds of the main condition to cover only "major_ally." Yet another possibility is to find an additional feature of the positive examples of the rule which is not a feature of the current negative example. This feature would then be added to the corresponding object from the main condition (both in the upper bound and in the lower bound). Additional negative examples may lead to additional Except When conditions and specializations of the main condition.

It may be the case that the actual explanation of an example contains elements that are not part of the object ontology. In such situations, Disciple-RKF elicits these elements from the expert and adds them to the ontology as new concepts or new features. Thus the learning process is performed in an evolving representation space in which the object ontology (which is used as the generalization hierarchy for learning) may be modified at any time. If the ontology is modified, the learned rules may no longer be correct. Therefore, the rules need to be relearned. This can be automatically done if the system keeps the examples and the explanations from

which the rules were learned. However, different examples and explanations of a rule contain instances that existed in different scenarios (e.g. World War II or Afghanistan 2001-2002). Therefore, Disciple keeps minimal generalizations of the examples and explanations that do not contain any instance and use these minimally generalized examples and explanations to regenerate the rules when the object ontology changes.

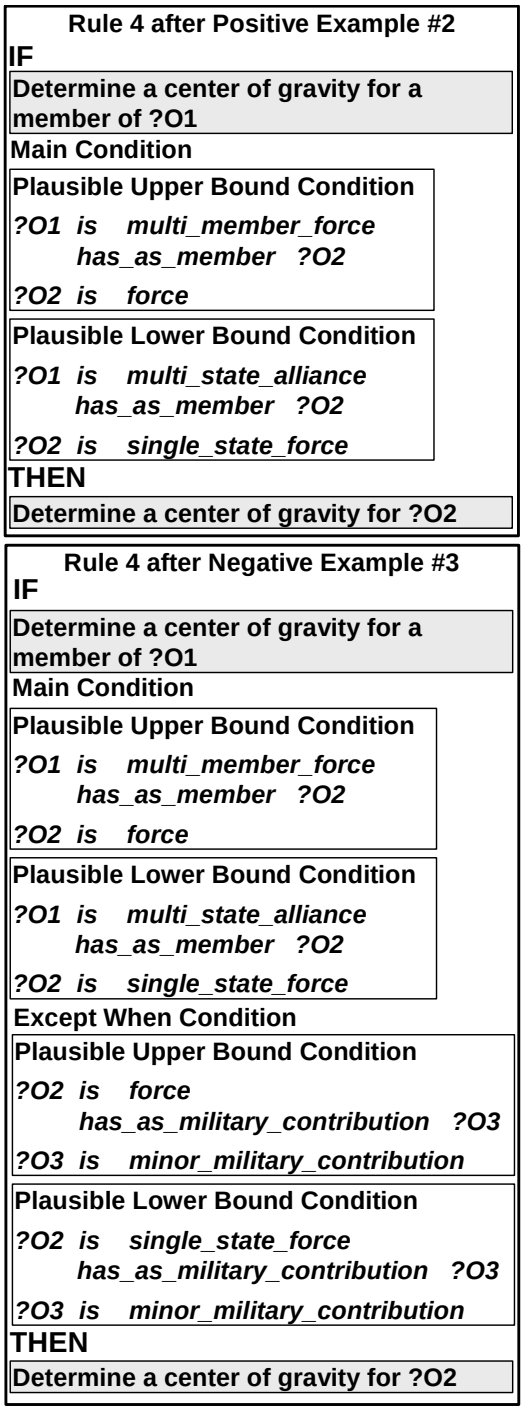

Figure 7: Refinements of Rule 4

7. EVALUATION OF DISCIPLE-RKF

DISCIPLE-RKF is a complex development environment for knowledge-based agents that incorporate the subject matter expertise of human experts. Evaluating such systems is very difficult both because it takes a significant amount of time and effort to build a knowledge-based agent, and because the critical evaluation resource consists of subject matter experts who are very expensive even for large research projects such as DARPA's RKF. To deal with the prohibitive cost of the evaluating experts we have developed a systematic approach to the evaluation of Disciple-RKF that involved its use by military experts (lieutenant colonels and colonels from different military services) while they are students at the US Army War College. The main idea was to define an evaluation process that

also constitutes a useful educational experience for the experts and can, therefore, take place during their courses. The two courses in which we have evaluated Disciple-RKF are Case Studies in Center of Gravity Analysis (COG) course, and Military Applications of Artificial Intelligence (MAAI) course.

The main goal of the evaluation of Disciple-RKF as part of the COG course was to determine whether the Disciple-RKF system can be used to develop knowledge-based agents that are considered useful by their users. To this purpose, we worked with the course's instructor, Dr. Jerome Comello (COL retired), to develop the object ontology of a Disciple-RKF agent and to teach it how to identify and test center of gravity candidates for war scenarios. The resulting knowledge base consisted of 355 object concepts, 193 feature definitions, 368 reduction rules, and 269 composition rules. The object ontology was developed using the ontology browsers and editors of Disciple. The reduction rules were entirely learned and refined using the algorithms described in this paper, and the composition rules were defined using a rule composition editor.

Each of the 8 COG students used a copy of this trained Disciple-RKF agent as an intelligent assistant that helped him to develop a center of gravity analysis of a war scenario. Each student interacted with the scenario elicitation module of Disciple-RKF that guided him to describe the relevant aspects of the analyzed scenario. Then he invoked the autonomous problem solver (which used the rules learned by Disciple-RKF) and the report generator, obtaining a center of gravity analysis report. This report contains the center of gravity candidates found by Disciple-RKF, together with the justifications for their identification as candidates, and the justifications for the results of their testing (i.e. their elimination or their preservation as likely centers of gravity). These justifications are generated based on the rules learned by Disciple-RKF, and are intended to help the students learn how to identify and test the center of gravity candidates for war scenarios. The students were asked to study and evaluate the justifications

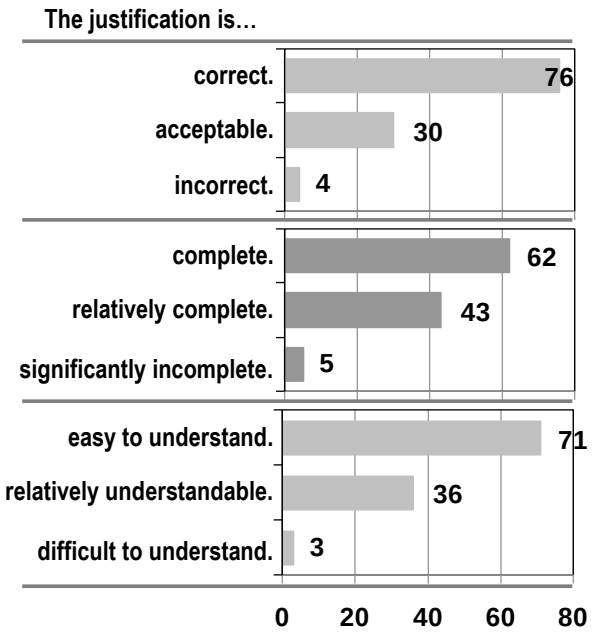

Figure 8: Subjective evaluation of the justifications generated by Disciple.

generated by Disciple and to finalize the report. Figure 8 summarizes the results of their evaluations. For instance, out of the 110 justifications generated for all the analyzed scenarios, 76 were considered correct, 30 acceptable, and only 4 incorrect. Moreover, most of the time the students have found these justifications to be complete and easy to understand. The use of Disciple extended over four 3-hour sessions. At the end the students were asked to evaluate a wide rage of aspects related to the usability and utility of the three Disciple modules used, and of Disciple-RKF as a whole. The students were presented with statements on various aspects of Disciple-RKF and were asked to express their level of agreement with these statements by using a 5-point scale (strongly disagree, disagree, neutral, agree, strongly agree). Figure 9 includes some of the global evaluation results, showing that the Disciple approach allowed the development of an agent that has been found to be useful for a complex military domain.

A main goal of the Disciple approach is to allow a subject matter expert to teach a Disciple agent in a natural way, through examples and explanations. Therefore, the MAAI course was organized to perform a systematic evaluation of Disciple-RKF as an agent development environment for subject matter experts, with limited assistance from knowledge engineers. In addition, we also wanted to test the ability of teams of subject matter experts to develop a Disciple agent that integrates their problem solving expertise, since this was a main objective of the DARPA's RKF program. MAAI was a 10 week, 3 hours/week course, attended by 13

colonels and lieutenant colonels from different military services. The students, who were military experts with no prior knowledge engineering experience, were introduced to the Disciple approach, and used Disciple-RKF to jointly develop an agent that incorporated some of their expertise in centers of gravity determination. During each class, the experts were introduced to the Disciple theory and tools corresponding to a particular agent training activity: first, scenario specification; second, modeling expert's reasoning; third, agent teaching and rule learning; and forth, mixed-initiative problem solving and rule refinement. Immediately after a tool was demonstrated by the course's instructor, it was used by the subject matter experts. The experts were supervised by knowledge engineers who were asked not to offer help unless it was requested and were not allowed to do experts' work. Figure 10 summarizes the experiment.

Before starting the experiment, a Disciple-RKF agent was trained to identify leaders as center of gravity candidates. The knowledge base of this agent contained the definitions of 432 concepts and features and 18 task reduction rules. However, the agent had no knowledge of how to test the identified candidates. We then performed a joint domain analysis and ontology development with all the experts. For this, we have considered the case of testing whether Saddam Hussein, in the Iraq 2003 scenario, had all the required critical capabilities to be the center of gravity for Iraq. Based on this domain analysis, we have extended the ontology of

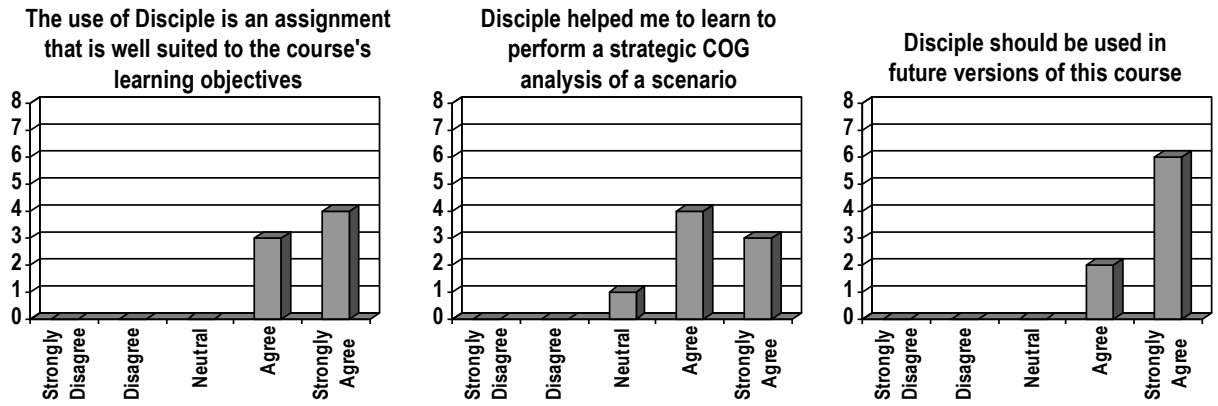

Figure 9: Global evaluation results from the COG class experiment.

Disciple-RKF with the definition of 37 new concepts and features elicited from the experts.

The 13 subject matter experts from the class were then grouped into five teams (of 2 or 3 experts each), and each team was given a copy of the extended Disciple-RKF agent. Next, each team trained its agent to test whether a leader has one or two critical capabilities, as indicated in Figure 10. For instance, Team 1 trained its agent how to test whether a leader had the critical capabilities of staying informed and being irreplaceable. The training was done based on three scenarios (Iraq 2003, Arab-Israeli 1973, and War on Terror 2003), the experts teaching Disciple-RKF how to test each strategic leader from these scenarios. As a result of the training performed by the experts, the knowledge bases of the five Disciple-RKF agents were extended with new object features and rules, as indicated in the middle of Figure 10. For instance, the knowledge base of the agent trained by Team 1 was extended with 5 features and 10 task reduction rules.

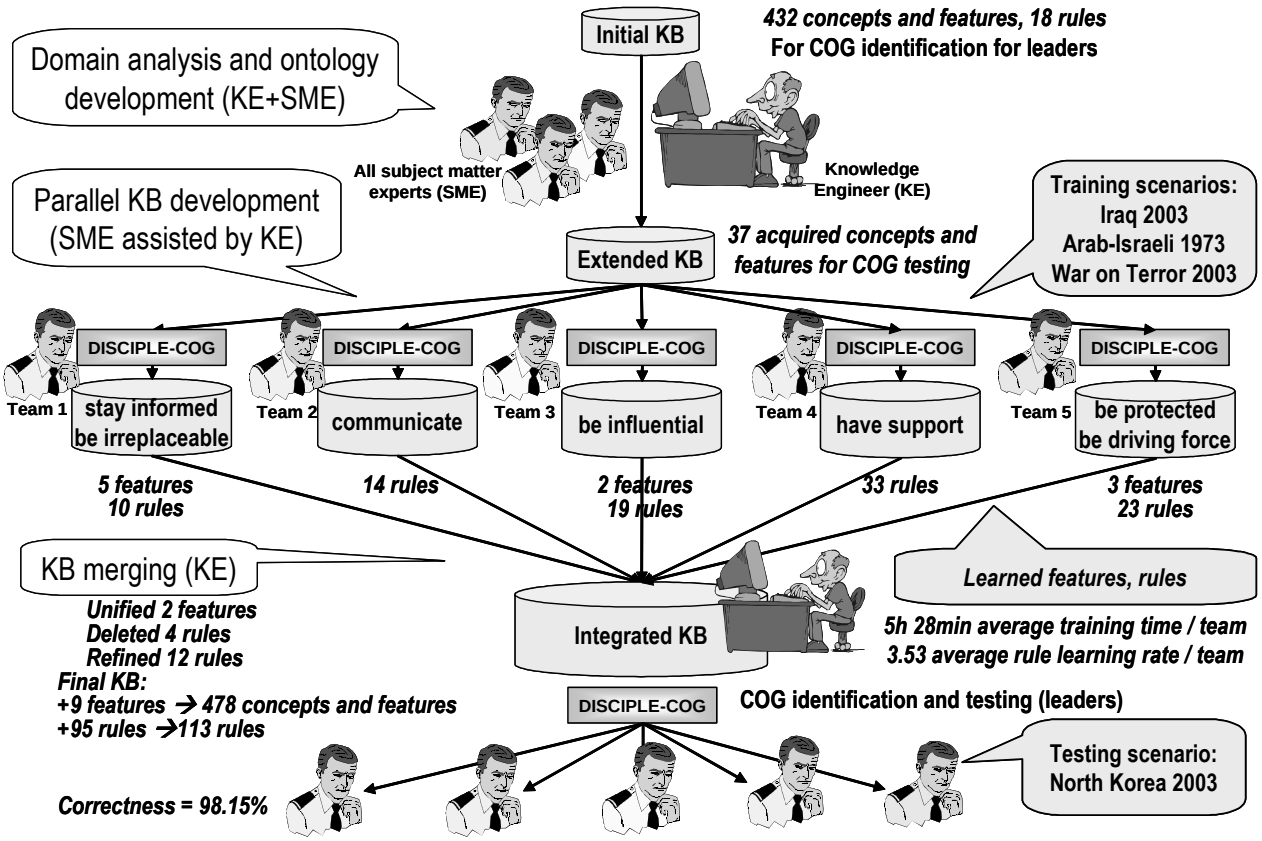

Figure 10: Experiment of rapid knowledge base development by subject matter experts.

Figure 10 shows the number of features and rules learned by each team. Notice that the average training time per team was 5 hours and 28 minutes and the average rule learning rate per team was 3.53 rules/hour. This included the time spent in all the agent training activities (i.e., specifying the three training scenarios, modeling expert's reasoning, rule learning, mixed-initiative problem solving, and rule refinement).

After the training of the 5 Disciple-RKF agents, their knowledge bases were merged by a knowledge engineer, who used the knowledge base merging tool of Disciple-RKF. The knowledge engineer also performed a general testing of the integrated knowledge base, in which he included the 10 features and 99 rules learned. During this process, two semantically equivalent features were unified, 4 rules were deleted, and 12 other rules were refined by the knowledge engineer. The other 8 features and 83 rules learned were not changed. Most of the modifications were done to remove rule redundancies or to specialize overly general rules. Next, each team tested the integrated agent on a new scenario (North Korea 2003) and was asked to judge the correctness of each reasoning step performed by the agent, but only for the capabilities for which that team performed the training of the agent. We computed an overall correctness of 98.15% for all rules learned from the subject matter experts as the total number of correct reasoning steps divided by the total number of generated reasoning steps (for North Korea 2003). This was an unexpectedly high result, caused by the fact that the leader of North-Korea had, in the judgment of the experts, many features in common with leaders from the training scenarios.

As in the case of the COG course, the students were asked to evaluate a wide variety of aspects of each of the used modules of Disciple-RKF. The evaluations showed that the experts have found these modules to be reasonably easy to use. In particular, 7 of the 13 experts strongly agreed, 4 agreed, 1 was neutral, and 1 disagreed with the statement "I think that a subject matter expert can use Disciple to build an agent, with limited assistance from a knowledge engineer".

These results are important because they support the claim that Disciple-RKF is a good approach to direct knowledge acquisition from subject matter experts.

8. CONCLUSIONS

Disciple-RKF, as a whole, is quite unique both in the field of Machine Learning and in the field of Knowledge Acquisition, and we are not aware of any other system that is similar in terms of capabilities and methods to achieve them.

There are, however, many ways in which Disciple-RKF can still be considerably improved. For instance, one needs to develop more powerful methods for helping the expert to express his or her reasoning process using the task reduction paradigm, along the path opened by the Modeling Advisor described in (Boicu, 2002). Our agent training experiments have also revealed that the mixed-initiative learning methods of Disciple-RKF could be significantly empowered by developing the natural language processing capabilities of the system. More powerful ontology learning methods are also needed (Stanescu et al., 2003). Finally, because the expert who teaches Disciple-RKF has no formal training in knowledge engineering, the knowledge pieces learned by the agent and the knowledge base itself will not be optimally represented, and will require periodic revisions by the knowledge engineer. Examples of encountered problems with the knowledge base are semantic inconsistencies within a rule, and the violation of certain knowledge engineering principles. It is therefore necessary to develop mixed-initiative knowledge base reformulation and optimization methods to identify and correct such problems in the knowledge base.

ACKNOWLEDGEMENTS

This research was done in the Learning Agents Center of George Mason University and was sponsored by several US Government agencies, including the Defense Advanced Research

Projects Agency, the Air Force Research Laboratory, the Air Force Office of Scientific Research, and the US Army War College.

REFERENCES

Boicu, C., Tecuci, G., Boicu, M., and Marcu, D. 2003. Improving the Representation Space through Exception-Based Learning. In Proc. of the Sixteenth International FLAIRS Conference. Menlo Park, CA: AAAI Press. pp. 336-340.

Boicu M., Tecuci G., Marcu D., Bowman M., Shyr P., Ciucu F., and Levcovici C. 2000. Disciple-COA: From Agent Programming to Agent Teaching, Proc. of the Seventeenth International Conference on Machine Learning. Stanford, CA: Morgan Kaufmann.

Boicu, M. 2002. Modeling and Learning with Incomplete Knowledge, PhD dissertation. Fairfax, VA: George Mason University.

Boicu M., Tecuci G., Stanescu B., Marcu D., Barbulescu M., Boicu C. 2004. Design Principles for Learning Agents, in Proc. of AAAI-2004 Workshop on Intelligent Agent Architectures. Menlo Park, CA: AAAI Press.

Bowman, M. 2002. A Methodology for Modeling Expert Knowledge that Supports Teaching Based Development of Agents, PhD dissertation. Fairfax, VA: George Mason University.

Clausewitz, C.V. 1832. On War, translated and edited by M. Howard and P. Paret. Princeton, NJ: Princeton University Press, 1976.

DeJong, G. and Mooney, R.J. 1986. "Explanation-Based Learning: An Alternative View," Machine Learning, Vol. 1, pp. 145-176.

Giles, P.K., and Galvin, T.P. 1996. Center of Gravity: Determination, Analysis and Application. Carlisle Barracks, PA: CSL, U.S. Army War College.

Mitchell, T.M. 1978. "Version Spaces: an Approach to Concept Learning," Doctoral

Dissertation. Stanford, CA: Stanford University.

Mitchell, T.M., Keller, T. and Kedar-Cabelli, S. 1986. "Explanation-Based Generalization: A Unifying View," Machine Learning, Vol. 1, pp. 47-80.

Nilsson, N.J. 1971. Problem Solving Methods in Artificial Intelligence, NY: McGraw-Hill.

Powell G.M. and Schmidt C.F. 1988. A First-order Computational Model of Human Operational Planning, CECOM-TR-01-8, US Army CECOM, Fort Monmouth, New Jersey, August.

Stanescu B., Boicu C., Balan G., Barbulescu M., Boicu M., Tecuci G. 2003. Ontologies for Learning Agents: Problems, Solutions and Directions. In Proc. of the IJCAI-03 Workshop on Workshop on Ontologies and Distributed Systems, 75-82. Menlo Park, CA: AAAI Press.

Strange, J. 1996. Centers of Gravity & Critical Vulnerabilities: Building on the Clausewitzian Foundation So That We Can All Speak the Same Language. Quantico, VA: Marine Corps University.

Tecuci, G. 1988. Disciple: A Theory, Methodology and System for Learning Expert Knowledge, Thèse de Docteur en Science, University of Paris-South.

Tecuci, G. 1998. Building intelligent agents: an apprenticeship multistrategy learning theory, methodology, tool and case studies. London, UK: Academic Press.

Tecuci G., Boicu M., Bowman M., and Marcu D., with a commentary by Burke M. 2001. An Innovative Application from the DARPA Knowledge Bases Programs: Rapid Development of a High Performance Knowledge Base for Course of Action Critiquing. AI Magazine, 22, 2. Menlo Park, CA: AAAI Press. pp. 43-61.

Tecuci G., Boicu, M., Marcu, D., Stanescu, B., Boicu, C., and Comello, J. 2002. Training and Using Disciple Agents: A Case Study in the Military Center of Gravity Analysis Domain. AI Magazine 23(4) 51–68.