Mixed-Initiative Assistant for Modeling Expert's Reasoning

Mihai Boicu, Gheorghe Tecuci, Dorin Marcu

George Mason University, Learning Agents Center,
MSN 4A5, 4400 University Dr., Fairfax, VA 22030,
{mboicu, tecuci, dmarcu}@gmu.edu

Abstract

This paper presents a mixed-initiative assistant that helps a subject matter expert to express the way she solves problems in the task reduction paradigm. It guides the expert to follow a predefined modeling methodology, supports the expert to express her reasoning by using natural language with references to the objects from the agent's ontology, and helps her in the process of specifying solutions to new problems by analogy with previously solved problems. The assistant, which is integrated into the Disciple system for agents development, has been successfully evaluated by subject matter experts at the US Army War College.

1 Instructable agents

For many years we have researched a general theory, a methodology, and a family of tools, called Disciple, for the rapid development of knowledge-based agents by subject matter experts, with limited assistance from knowledge engineers (Tecuci, 1988, 98; Boicu 2002). The short-term goal of this research is to overcome the knowledge acquisition bottleneck in the development of expert and decision-support systems (Buchanan and Wilkins, 1993). The long-term goal of this research is to develop the technology that will allow non-computer scientists to develop their own cognitive assistants that incorporate their subject matter expertise and can support them in their regular problem solving and decision-making activity.

The main idea of our approach to achieve these goals consists in developing an instructable (learning) agent that can be taught directly by a subject matter expert to become a knowledge-based assistant. The expert will teach the agent how to perform problem solving tasks in a way that is similar to how the expert would teach a person. That is, the expert will teach the agent by providing it examples on how to solve specific problems, helping it to understand the solutions, and supervising and correcting the problem solving behavior of the agent. The agent will learn from the expert by generalizing the examples and building its knowledge base. In essence, the goal is to create a synergism between the expert that has the knowledge to be formalized and the agent that knows how to formalize it.

This is achieved through:

- mixed-initiative problem solving, where the expert solves the more creative parts of the problem and the agent solves the more routine ones;
- integrated learning and teaching, where the expert helps the agent to learn (for instance, by providing examples, hints and explanations), and the agent helps the expert to teach it (for instance, by asking relevant questions);
- multistrategy learning (Michalski and Tecuci, 1994), where the agent integrates complementary strategies, such as learning from examples, learning from explanations, and learning by analogy, to learn general concepts and rules.

In order to teach an agent how to solve problems, the expert has first to be able to make explicit the way he or she reasons, in a way that is formal and precise enough for the agent to learn from it.

The process of informally expressing the expert's problem solving process for a given problem, using a problem-solving paradigm, and a given methodology, is called *modeling expert's reasoning process*.

Our experience shows that *modeling is the single most difficult agent training activity for the expert*. This is not surprising because it primary involves human creativity and often requires the extension of the agent domain language. Moreover, the description of the problem solving process has to be formal and precise enough, both to facilitate agent's learning, and to assure that the learned knowledge is applicable to other situations.

A Disciple agent uses task-reduction as the main problem solving paradigm. In this paradigm, a problem solving task is successively reduced to simpler tasks, the solutions of the simplest tasks are found, and these solutions are successively composed into the solution of the initial task. The knowledge base of the agent is structured into an object ontology that represents the objects from an application domain, and a set of task reduction rules and solution composition rules expressed with these objects.

To develop a Disciple agent for a specific application domain, one needs to define the ontology for that domain and then to teach the agent how to perform various tasks, in a way that resembles how one would teach a human apprentice. This requires the expert to consider specific problems and to show the agent how to solve them by following the task reduction paradigm.

As mentioned above, this modeling process is very complex and the question is how to develop an assistant that can help the expert to perform it. One idea is to define a simple modeling methodology and associated guidelines which the expert can easily follow to express her reasoning in the task reduction paradigm (Bowman, 2002). At the same time develop mixed-initiative methods to help the expert follow the methodology. Another idea is to allow the expert to express her reasoning in a language that combines natural language with references to the objects from the agent's ontology. This, in turn, requires the modeling assistant to help the expert in identifying the objects from the knowledge base she wants to refer to. Yet another idea is to help the expert in the process of specifying the solutions to new problems, by analogy with previously defined solutions. All these ideas are at the basis of the mixed-initiative modeling assistant integrated into the Disciple system, as described in the rest of this paper.

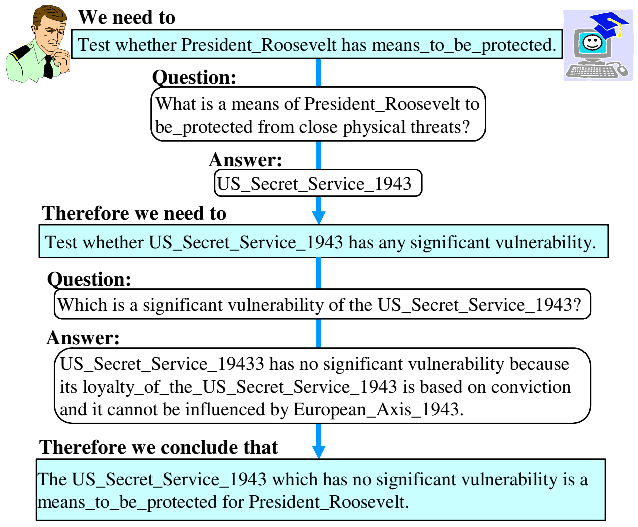
2 Modeling expert's reasoning process

We have developed a simple and intuitive modeling language in which the expert, with the help of the modeling assistant, expresses the way she is solving a specific problem, using natural language, as if the expert would think aloud, as illustrated in Figure 1 (Bowman, 2002).

We need to *"Assess whether President-Roosevelt has means to be protected."* In order to perform this assessment task, the expert and the agent will ask themselves a series of questions. The answer to each question will lead to the reduction of the current assessment task to simpler assessment tasks. The first question asked is: *"What is a means of President Roosevelt to be protected from close physical threats?"* The answer, *"US Secret Service 1943,"* leads to the reduction of the above task to the task *"Test whether US Secret Service 1943 has any significant vulnerability."*

In general, the question associated with a task considers some relevant piece of information for solving that task. The answer identifies that piece of information and leads to the reduction of the task to one or several simpler tasks. Alternative questions correspond to alternative approaches to solving the current problem solving task. Several answers to a question correspond to several potential solutions. The modeling language includes many helpful guidelines for the expert, such as: *Ask small, incremental questions that are likely to have a single category of answer (but not necessarily a single answer). This usually means ask who, or what, or where, or what kind of, or is this or that etc., not complex questions such as who and what, or what and where.*

The expert expresses her reasoning in natural language, but the modeling assistant provides her with helpful and non-disruptive mechanisms for automatic identification of the knowledge base elements in her phrases. In particular, the modeling assistant has an effective word completion capability. When the expert types a few characters of a phrase, such as, "means to be protected" (see Figure 1), it proposes all the partially matching names from the knowledge base, ordered by their plausibility, to be used in the current context, including "means_to_be_protected." The expert selects this name only because it is simpler than typing it. However, now the system also partially "understands" the English sentence entered by the expert, which will significantly facilitate their collaboration.


We need to

Test whether President_Roosevelt has means_to_be_protected.

Question:

What is a means of President_Roosevelt to be_protected from close physical threats?

Answer:

US_Secret_Service_1943

Therefore we need to

Test whether US_Secret_Service_1943 has any significant vulnerability.

Question:

Which is a significant vulnerability of the US_Secret_Service_1943?

Answer:

US_Secret_Service_19433 has no significant vulnerability because its loyalty_of_the_US_Secret_Service_1943 is based on conviction and it cannot be influenced by European_Axis_1943.

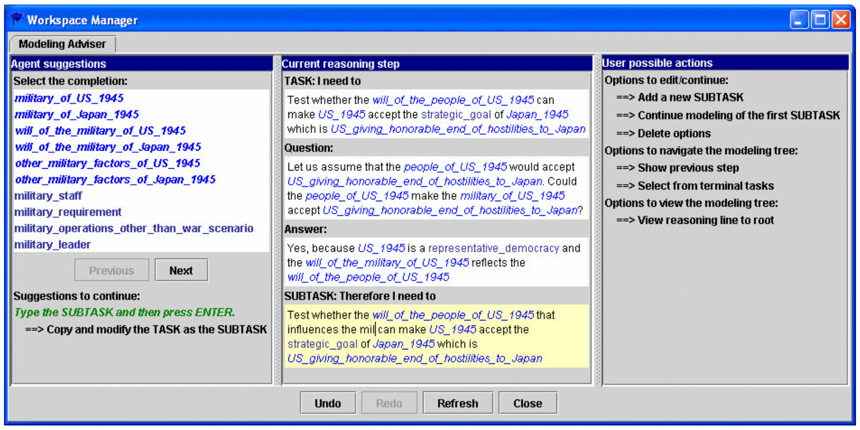
Therefore we conclude that

The US_Secret_Service_1943 which has no significant vulnerability is a means_to_be_protected for President_Roosevelt.

Figure 1: A sequence of two task reductions steps

3 Modeling Assistant interface

Figure 2 shows the interface of the modeling assistant. The middle part of the screen contains the current task reduction step that the expert is composing. At each state in this process, the right hand side of the screen shows all the actions that could be performed in that state, and the left hand side shows the action that the modeling assistant is actually recommending. For instance, to specify the current subtask, the advisor suggested the expert to copy



Agent suggestions

Select the completion:

military_of_US_1945
military_of_Japan_1945
will_of_the_military_of_US_1945
will_of_the_military_of_Japan_1945
other_military_factors_of_US_1945
other_military_factors_of_Japan_1945
military_staff
military_requirement
military_operations_other_than_war_scenario
military_leader

Suggestions to continue:

Type the SUBTASK and then press ENTER.

==> Copy and modify the TASK as the SUBTASK

Current reasoning step

TASK: I need to

Test whether the *will_of_the_people_of_US_1945* can make *US_1945* accept the strategic_goal of *Japan_1945* which is *US_giving_honorable_end_of_hostilities_to_Japan*

Question:

Let us assume that the *people_of_US_1945* would accept *US_giving_honorable_end_of_hostilities_to_Japan*. Could the *people_of_US_1945* make the *military_of_US_1945* accept *US_giving_honorable_end_of_hostilities_to_Japan*?

Answer:

Yes, because *US_1945* is a representative_democracy and the *will_of_the_military_of_US_1945* reflects the *will_of_the_people_of_US_1945*

SUBTASK: Therefore I need to

Test whether the *will_of_the_people_of_US_1945* that influences the mil can make *US_1945* accept the strategic_goal of *Japan_1945* which is *US_giving_honorable_end_of_hostilities_to_Japan*

User possible actions

Options to edit/continue:

==> Add a new SUBTASK
==> Continue modeling of the first SUBTASK
==> Delete options

Options to navigate the modeling tree:

==> Show previous step
==> Select from terminal tasks

Options to view the modeling tree:

==> View reasoning line to root

Figure 2: Modeling Adviser interface

and to modify the if-task. The modeling assistant may also suggest the question to be asked, or the answer of the question. As mentioned, the expert expresses her reasoning in English. However, each time she starts to type a word, the agent lists in the left hand side of the screen all the instances and concepts from the knowledge base that are consistent with the characters typed so far (see Figure 2).

4 Checking expert's modeling

The modeling assistant, through its Example Analyzer module, also checks whether a task reduction step specified by the expert is correct and suggests improvements to make it complete and consistent with the modeling that was already done. Figure 3 illustrates some of the improvements that might be suggested by the Example Analyzer.

There are two types of analysis done by the Example Analyzer. There is a global analysis that attempts to discover problems with the entire example or with its place in the overall model. There is also a local analysis that deals with the components of the example.

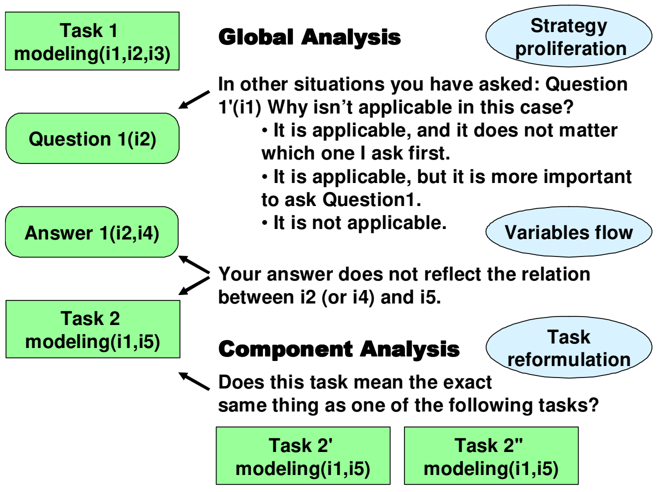

Task 1 modeling(i1,i2,i3)
Global Analysis
Strategy proliferation
In other situations you have asked: Question 1'(i1) Why isn't applicable in this case?
• It is applicable, and it does not matter which one I ask first.
• It is applicable, but it is more important to ask Question1.
• It is not applicable.
Question 1(i2)
Answer 1(i2,i4)
Variables flow
Your answer does not reflect the relation between i2 (or i4) and i5.
Task 2 modeling(i1,i5)
Component Analysis
Task reformulation
Does this task mean the exact same thing as one of the following tasks?
Task 2' modeling(i1,i5)
Task 2" modeling(i1,i5)

Figure 3: Example Analyzer strategies

One strategy for global analysis is to look at the variable flow. Our modeling methodology recommends that the subtasks should only refer to the variables from the if-task, question and answer. During learning all the new variables introduced by the question, the answer and the subtasks must be somehow linked to the input variables from the if-task. Therefore, another method is to check that there is a path between each new variable and one of the input variables. If this rule is not followed it is much more difficult to correctly generalize the example.

Another important aspect of the modeling methodology is the minimization of the number of strategies for solving a task (a strategy being represented by the question from the task reduction step). If the expert uses more than one strategy then it is important to make sure that this is a reasonable thing to do (by requesting a confirmation). If the expert changes her mind some correction is proposed. For instance, if the expert selects the first justification from Figure 3, (i.e. "It is applicable, and does not matter which one I ask"), the agent will replace Question1 (i2) with Question 1'(i1).

5 Assisting the expert to compose an example

The most challenging part is how to help the expert when he models a new problem solving step, as the ones presented in the Figure 1. Even though the system is not by itself creative it might still help the expert in this process, enhancing the expert's capabilities.

The modeling methodology follows a top-down definition of the examples. Therefore, the if-task of an example will be either the initial problem to be solved or it has already been defined from the previous problem solving episode. It is difficult for the agent to guess the right continuation, but it may suggest several plausible continuations, letting the expert to select and edit the most appropriate one.

There are three main goals in developing such methods:

1) to facilitate the continuation of a partial problem solving example;

2) to help the expert define the current example in a form which is consistent with previously entered knowledge;

3) to guide the expert follow the modeling methodology.

The heuristic methods developed fall under three categories that will be discussed in the next sections:

1) methods that suggest continuations of the current example based on different levels of similarity between the current example and previously learned rules and/or previously defined (but not yet formalized) examples;

2) methods that suggest questions and answers based on plausible explanations and interpretations of the example;

3) methods based on the modeling methodology and on knowledge engineering guidelines.

5.1 Suggesting questions to continue an example

In this section we consider the case when the user is expressing the question, as part of a problem solving step. We present heuristic methods that can be used to suggest this question. The questions and their answers have been introduced into the task-reduction paradigm in order to capture the rationality of performing the corresponding reductions. The question by itself may be considered an abstraction of the strategy used to reduce the current task. Also, the question together with its answer offers an informal justification of why a particular reduction is made.

In the general modeling methodology for task reduction we have the following guideline related to the question: *"For a task attempt to use the same question in all problem solving episodes."* But there are situations when the modeling requires asking a different question for the same task. However, we need to be very conservative with the proliferations of the questions for the same task. Therefore, the top level heuristic in suggesting a question is based on this methodological guideline.

This heuristic method distinguishes between two cases. In the first case the example was just started (i.e. it does not yet contain a question) and the if-task was already used in other examples. Because the if-task was already used in other examples there is at least one question associated with it. Therefore, based on the above modeling guideline, the method will propose questions which are similar with the existing ones. In the second case either the task is used for the first time (i.e. no other rules or examples use it, which means that there is no question associated with the if-task), or the user started a new question. If the user started a new question, this suggests that none of the proposed ones was useful. In this second case the method will propose questions based on similar rules and examples. We will discuss each of these cases separately.

First we will discuss how to propose questions similar with the ones already associated with this if-task. Such a question may appear in a rule having the same if-task or in an example having the same if-task (example that was not yet generalized into a rule). Because usually a question contains values that refer to a particular problem solving step the method must construct a similar question, with the same pattern but with the values determined by analogy with current task instantiation. Table 1 summarizes this method.

In order to compare the plausibility of the generated questions we consider the confidence in our analogical reasoning. We assume that the confidence is a number between 0 and 1 where 1 represents the highest confidence and is computed based on the following heuristics:

Case 1: For the questions similar with the questions from the rules with the same if-task (in method SQT-R), we will use a measure of their instantiation: an analogous example that is better instantiated with the task values has more chances to be similar with the currently edited example. The confidence function used is: *f_1(E, R, T(i_1,i_2,…))= (percent of instantiated variables in question(E))/2 + (percent of instantiated variables in example(E))/2.*

Case 2: For the questions similar with the questions from the unformalized examples with the same if task (in method SQT-E), we will use a measure of their instantiation: *f_2(IE, ET, T(i_1,i_2,…))=(percent of instantiated variables in question(IE))/2 + (percent of instantiated variables in example(IE))/2.*

Case 3: The main method SQT, combines these measures of confidence. We are less confident in the questions from the examples which are not yet formalized in a rule, because such a question may be further changed before or during the rule learning process.

The second approach is to propose questions based on rules and examples having if-tasks which are similar with the current one. For each similar task this method will generate the questions by using the method from Table 2.

5.2 Explanation-based heuristics

This class of heuristics uses plausible explanations of the current partial problem solving step in order to improve the

Table 1: Method to suggest questions used with a task (SQT)

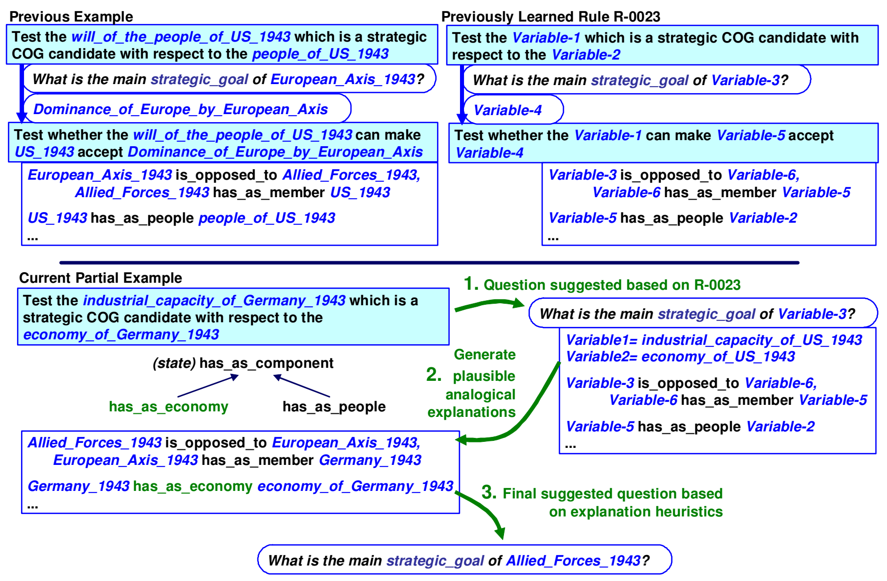
```
Suggest-questions-from rules: SQT-R(T($i_1$,..., $i_k$), n)
Find the n most plausible questions $(Q_j)_{j\le n}$ for the task T($i_1$,..., $i_k$)
similar with the questions from the rules with the same if-task T
  SQ←∅
  for each rule R to solve task T do
    for each instantiated example E generated by R($i_1$,..., $i_k$) do
      Q←the partially instantiated question from example E
      confidence(Q)←$f_1$(E,R,T($i_1$, $i_2$,..., $i_k$))
      if Q∉SQ then add Q in SQ
              else increase confidence for Q in SQ
  return first n elem from SQ in the decreasing confidence
  order

Suggest-questions-from examples: SQT-E(T($i_1$, $i_2$,..., $i_k$), n)
Find the n most plausible questions $(Q_j)_{j\le n}$ for the task T($i_1$..., $i_k$)
similar with the questions from the examples for if-task T
  SQ←∅
  for each example E (not generated by a rule) to solve task T
  do
    ET← template of E (values replaced with variables)
    IE←the instantiation of the template ET with $i_1$, $i_2$,..., $i_k$
    Q←the partially instantiated question from example IE
    confidence(Q)←$f_2$(IE, ET, T($i_1$, $i_2$,..., $i_k$))
    if Q∉SQ then add Q in result SQ
            else increase confidence for Q in SQ
  return first n elem from SQ in the decreasing confidence
  order

Suggest-questions-used-for-a-task: SQT(T($i_1$, $i_2$,..., $i_k$), n)
Find the n most plausible questions $(Q_j)_{j\le n}$ for the task T($i_1$, $i_2$,...,
$i_k$) using the questions already used for it
  SQ1←SQT-R(T($i_1$, $i_2$,..., $i_k$),n)
  for Q∈SQ1 do confidence(Q) ← 1/2 + confidence(Q)/2
  SQ2←SQT-E(T($i_1$, $i_2$,..., $i_k$),n-|SQ|)
  for Q∈SQ2 do confidence(Q) ← confidence(Q)/2.01
  return SQ1+SQ2
```

Table 2: Heuristic method to suggest questions from examples with similar if tasks (SQA)

```
Suggest-questions-from-similar-examples: SQA(E($i_1$,...$i_k$), n)
Find the n most plausible questions $(Q_j)_{j\le n}$ for the example
E($i_1$,..., $i_k$) similar with questions from the rules and examples
having an similar if-task
  SQ←∅; Temp-SQ←∅
  T($\pi_T$($i_1$, $i_2$,..., $i_k$))←the if task from the example E($i_1$, $i_2$,..., $i_k$)
  for each task AT($\sigma$($i_1$, $i_2$,..., $i_k$)) which is $\sigma$-similar with
  T($\pi_T$($i_1$, $i_2$,..., $i_k$)), generated in decreasing order of their
  similarity-conf(AT,T) do
    move to the end of SQ the questions from Temp-SQ with
      the confidence ≥$f_3$(max-question-conf, similarity-
      conf(AT,T))
    if |SQ|≥n then return first n elements from SQ
    New-SQ←SQT(AT($\sigma$($i_1$, $i_2$,..., $i_k$)),n-|SQ|)
    for each Q∈New-SQ do
      conf(Q)←conf(Q)/2+similarity-conf(AT,T)/2
      merge New-SQ into Temp-SQ
  add Temp-SQ to the end of SQ
  return first n elements of SQ
```

suggestions already made through the previously presented methods or to make new ones. There are three cases when such heuristics are used. The first case is when the questions, answers or tasks are generated by analogy with a rule and have missing values. The second case is when the analogy is made with an unformalized example.

Let us consider the situation from Figure 4. It illustrates the method for the case when it completes the definition of a question. In the upper-left part side of the figure there is a previously entered example. From this example the agent has learned a general task reduction rule, a fragment of which is presented in the upper-right part of the figure. The bottom part of the figure shows the steps of the method. The current partial example contains only the if-task: *"Test the industrial_capacity_of_Germany_ 1943 which is a strategic COG candidate with respect to the economy_of_Germany_1943"*. This task has the same pattern with the if-task of the rule from the top of the figure. Therefore, the previously presented SQT method will generate an incomplete analogous question *"What is the main strategic_goal of variable-3?"* However, there is no direct correspondent of *variable-3* that appears in rule. To find the value of *variable-3* the proposed method use the rule's explanations. These explanations represent an ontology pattern that links the matched variables (*variable-1* and *variable-2*) and the unmatched variable (*variable-3*), through features and other variables, as presented in the bottom-right part of Figure 4. Now, the method searches the object ontology for analogical structures constructed with the known values from example (*industrial_capacity_of_Germany_1943* and *economy of Germany_1943*). The most complete such analogous explanation structure is presented in the bottom-left part of the figure. It replaces the feature *has_as_people* with the similar feature *has_as_economy*, constructing a plausible explanation for the proposed question. This explanation determines the value for *variable-3* as *Allied_Forces_1943*. Usually, there is more than one plausible explanation. In such cases the method must analyze them and use only the most plausible ones. To estimate their plausibility the method uses the confidence in the similarity between the proposed explanation and the explanation of the rule from the top-left part of the figure. Because the plausible explanation shown in Figure 4 has the best confidence it will be first selected. Finally the completed question will be proposed to the user.



Previous Example
Test the will_of_the_people_of_US_1943 which is a strategic COG candidate with respect to the people_of_US_1943
What is the main strategic_goal of European_Axis_1943?
Dominance_of_Europe_by_European_Axis
Test whether the will_of_the_people_of_US_1943 can make US_1943 accept Dominance_of_Europe_by_European_Axis
European_Axis_1943 is_opposed_to Allied_Forces_1943, Allied_Forces_1943 has_as_member US_1943
US_1943 has_as_people people_of_US_1943
...
Previously Learned Rule R-0023
Test the Variable-1 which is a strategic COG candidate with respect to the Variable-2
What is the main strategic_goal of Variable-3?
Variable-4
Test whether the Variable-1 can make Variable-5 accept Variable-4
Variable-3 is_opposed_to Variable-6, Variable-6 has_as_member Variable-5
Variable-5 has_as_people Variable-2
...
Current Partial Example
Test the industrial_capacity_of_Germany_1943 which is a strategic COG candidate with respect to the economy_of_Germany_1943
1. Question suggested based on R-0023
What is the main strategic_goal of Variable-3?
Variable1= industrial_capacity_of_US_1943
Variable2= economy_of_US_1943
Variable-3 is_opposed_to Variable-6, Variable-6 has_as_member Variable-5
Variable-5 has_as_people Variable-2
...
(state) has_as_component
has_as_economy
has_as_people
2. Generate plausible analogical explanations
Allied_Forces_1943 is_opposed_to European_Axis_1943, European_Axis_1943 has_as_member Germany_1943
Germany_1943 has_as_economy economy_of_Germany_1943
...
3. Final suggested question based on explanation heuristics
What is the main strategic_goal of Allied_Forces_1943?

Figure 4: Completion of an element generated by analogy with a rule

This method is presented in Table 3. This is a recursive method that returns only the best question, but it may easily be adapted to return the best *n* questions.

The second case is when the analogy to generate the initial question was made with an unformalized example *E1* and not with a rule. The difference is that we do not have the explanations based on which to construct the analogy. Therefore, in the first phase we will construct the plausible justification of *E1*. Then the method will continue in a similar way. However, we will determine plausible explanations for *E* analogous only with some of the constructed explanations for *E1*.

Table 3: Heuristic method for explanation-based completion

```
Explanation based completion
EBC(E, Q, R, BoundedVars(($v_1$,$i_1$),($v_2$,$i_2$)...($v_k$,$i_k$)),
UnboundedVars($v_{k+1}$, $v_{k+2}$,...$v_p$))
Complete the question Q generated from rule R with the most
plausible instantiations of variables based on the current
example E.
    V(Q)← variables from question Q which are not
    instantiated
    if V(Q)=∅ then return Q($i_1$, $i_2$,...$i_k$)
    for each variable v∈ V(Q) do
        Path(v)←acyclic paths in rule explanations from v
        to bounded variables
    V(Q)←{v∈ V(Q)|v has the minimum distance to one of
            bounded variables}
    V(Q)←{v∈ V(Q)|v has links with the maximum number
    of bounded vars.}
    v←first element in V(Q)
    AE(v)←the best analogous fully instantiated
    explanations with Path(v)
    SQ←∅; confidence(SQ)←0;
    for each expl∈ AE(v) do
        BV←newly bounded variables in expl
        Q←EBC(E, Q, R, boundedvars∪BV,
        unboundedVars\BV)
        confidence(Q)←(confidence(Q)+confidence(expl))
        /2
        if confidence(Q) > confidence(SQ) then SQ←Q
    return SQ
```

6 Experimental results

We have performed two knowledge acquisition experiments at the US Army War College. In the first experiment we have used a modeling editor that allowed the expert to define and modify a reasoning tree, as illustrated in Figure 1, but offered no help. In the second experiment the experts used the presented modeling assistant that provided help during the modeling process.

One important observation is that in the second experiment, using the modeling assistant, the experts succeeded to model more reasoning steps (31 on average, as compared to 19 in the other experiment) containing creative solutions, with relatively reduced knowledge engineering help.

At the end of each experiment the experts completed a detailed questionnaire. The most relevant results obtained for the modeling assistant are presented in Figure 5 and Figure 6.

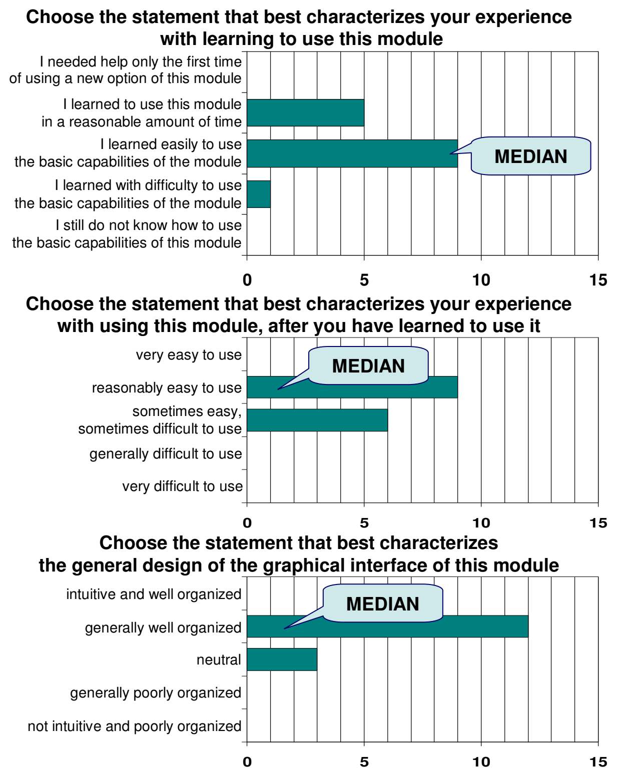


Figure 5: General evaluation of the modeling assistant

These results represent clear improvements on all the aspects taken into account, as compared to the Modeling Editor. For example, all but one expert considered that it was easy to learn to use the modeling assistant. Also this module was considered reasonably easy to use and its graphical user interface was generally considered well organized.

Figure 6 shows the evaluation of the suggestions given to the experts by the modeling assistant. Notice that the experts considered these suggestions as being generally useful and understandable. We consider these results as being very good. Although the suggestions are based on plausible reasoning, and are expected to not always be good, they clearly helped the creative process of modeling.

In conclusion, the use of the modeling assistant allowed the experts to better express their expertise, facilitating their interaction with the Disciple agent and reducing the need for help from a knowledge engineer.

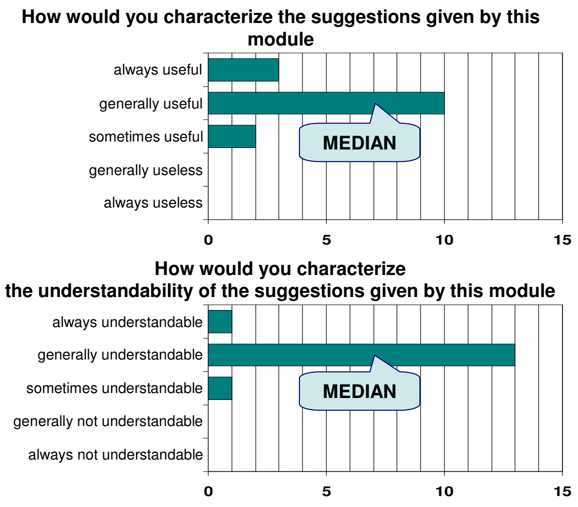


Figure 6: Evaluation of modeling assistant's suggestions

Acknowledgements. This research was sponsored by several US Government agencies, including DARPA, AFRL, AFOSR, and US Army War College.

References

Boicu, M. 2002. Modeling and Learning with Incomplete Knowledge, Ph.D. Dissertation, Department of Computer Science, George Mason University.

Bowman, M. 2002. A Methodology for Modeling Expert Knowledge that Supports Teaching Based Development of Agents. Ph.D. Dissertation, Department of Computer Science, George Mason University.

Buchanan, B.G. and Wilkins, D.C. eds. 1993. *Readings in Knowledge Acquisition and Learning: Automating the Construction and Improvement of Expert Systems.* San Francisco, CA: Morgan Kaufmann.

Michalski, R. S. and Tecuci, G., eds. 1994. *Machine Learning: A Multistrategy Approach* Volume 4. San Mateo, CA.: Morgan Kaufmann.

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: Academic Press.