Evaluating Expert-Authored Rules for Military Reasoning

Mike Pool[1], Ken Murray[2], Julie Fitzgerald[1], Mala Mehrotra[3], Robert Schrag[1], Jim Blythe[4], Jihie Kim[4], Hans Chalupsky[4], Pierluigi Miraglia[5], Thomas Russ[4], Dave Schneider[5]

1. Information Extraction and Transport, Inc. 2. SRI International 3. Pragati Synergetic Research, Inc. 4. Information Sciences Institute at University of Southern California 5. Cycorp, Inc.

ABSTRACT

Eliciting complex logical rules directly from logic-naïve subject matter experts (SMEs) is a challenging knowledge capture task. We describe a large-scale experiment to evaluate tools designed to produce SME-authored rule bases. We assess the quality of the rule bases with respect to the: 1) performance on the addressed functional task (military course of action (COA) critiquing); and 2) intrinsic knowledge representation quality. In the course of this assessment, we note both strengths and weaknesses in the state of the art, and accordingly suggest some foci for future development in this important technology area.

General Terms

Performance, Experimentation

Categories and Subject Descriptors

I.2.4 Knowledge Representation Formalisms and Methods – *representation languages, predicate logic*
I.2.1 Applications and Expert Systems

Keywords

Knowledge acquisition, evaluation, RKF, KRAKEN, SHAKEN, NuSketch Battlespace

INTRODUCTION

The authors are engaged in a joint research program—Rapid Knowledge Formation (RKF)—to develop and evaluate technology that will enable subject matter experts (SMEs) to enter and modify knowledge directly and easily without the need for training in formal logic, knowledge representation, knowledge acquisition (KA), or knowledge manipulation. We report on a large-scale evaluation conducted during the spring and summer of 2002, focusing on the challenge of eliciting complex logical rules from SMEs.

Below we describe the experimental challenge problem, the two integration teams' knowledge capture toolsets, and the methods used for assessing the quality of SMEs' authored COA critiquing rules. We also discuss the results obtained which provide us with insight into the various strengths and weaknesses of the technologies involved as well as the challenges and opportunities that lie ahead. While the challenge problem involved COA description and COA critiquing, here we attempt to identify various manifestations of knowledge representation challenges encountered during the COA critiquing rule authoring process and we consider ways to address them.

CHALLENGE PROBLEM[1]

Two teams (led by Cycorp and SRI International) responded to a challenge problem (posed by an independent evaluator—Information Extraction and Transport, Inc. (IET)) by developing integrated toolsets (Cycorp's KRAKEN system and SRI's SHAKEN system) suitable for knowledge capture and reasoning in the challenge problem domain—military course of action (COA)[2] critiquing. Two pairs of retired U.S. Army SMEs (one pair for each team's toolset) then exercised the developed toolsets over a period of one week, with time and effort divided roughly equally between capturing COAs themselves and capturing COA critiquing rules. The SMEs were given the following:

- Maps showing major terrain features such as roadways, towns, and bodies of water
- Positions of both Red and Blue forces
- COAs in text form, detailing both the overall mission and the specific COAs for various units

Knowledge-based critiquing requires capture of principles that military personnel routinely use to assess the quality of the standard COA critiquing criteria such as risk, preparedness, terrain suitability, position for follow-on, maneuver effectiveness, command and control, logistics support, resource use, synchronization, deception operation use, simplicity, relative combat power, blue reserve availability, security, time constraints, enemy vulnerability and a number

[1] For information about the RKF program generally, see http://reliant.teknowledge.com/RKF/.

[2] A military COA is a plan for capturing or defending some objective, or for engaging an opponent in a particular battle or series of battles [1].

of other considerations. From this set of criteria (developed by the evaluator and evaluator-collaborating SMEs who did not participate in the evaluation phase of the experiment), SMEs chose the criteria they deemed to be most salient to the COAs presented and wrote rules to automate COA critiquing.[3] This involved articulating rules for violation of spatial and temporal constraints on COAs, plan recognition, logistical consideration, negation reasoning, and other knowledge representation (KR) issues. For example, one SME's assessment of the "Risk" criterion for a COA was:

The greatest risk is that the Main Attack will get bogged down and the 23rd Armored Div will be unable to maintain the speed of the attack and reach Bridge 1 in time.

SMEs were allowed to interact with knowledge engineers (KEs) throughout the evaluation, but KEs were asked not to offer help unless it was requested and were prohibited from doing SMEs' work for them. Prior to the official evaluation period, SMEs were given one week of training with the tools during which these restrictions were not in force.

INTEGRATED TOOLSETS

Both teams' evaluation toolsets incorporated Northwestern University's NuSketch Battlespace tool [5]. NuSketch supports SMEs' formalization of the textual representation of the COAs which includes force placement, the timeline for battle, the motivations behind various attacks, and the actual path of travel for the various Blue units. During the evaluation both teams imported NuSketch output to develop COA representations that could then be subjected to critiquing.

Cycorp's KRAKEN System

A large knowledge base (KB) based on a higher-order formal predicate logic supports Cycorp's KRAKEN tools. The key strategies for SME-oriented KB interaction are natural language (NL) presentation and a knowledge-driven acquisition dialog with limited NL understanding. The KB includes a vast number of predicates and contains mappings to numerous English verbs. Cycorp's approach might be described as maximalistic, domain pluralistic, and conceptually precise. The KRAKEN tools speed up an SME's learning curve by productive collaboration with Cycorp's knowledge base, Cyc, which has a sophisticated knowledge representation milieu. For a more detailed description of the state of KRAKEN before the experiment, see [10]. We note below some significant additions that were incorporated in Kraken to support the challenge experiment.

Cyc facilitates knowledge construction in various ways. First, a key element of the KRAKEN tool is the Salient Descriptor. This tool uses KB content to help guide SMEs through natural language dialogues, for generating relatively complete axiomatizations. For example, Cyc contains a number of rules that are of the form, "If X is a subclass of Y, then it is useful to know, to what thing or kinds of things it bears the relation P, or whether it is also a kind of Z, etc.". A rule might state that if X is a Military Vehicle, it is useful to know which military forces are in possession of such objects or what kinds of weaponry are found on such vehicles, and so on. By posing queries to the user based on such rules, the Salient Descriptor is able to elicit more knowledge from SMEs based on initial SME inputs. Note that the tool can induce dialog queries from any ordinary KB rule as well: in essence, it attempts to determine the relevance of further knowledge from the current state of the KB. It also guides users further down the genls (subtype) hierarchy so that users can quickly identify more accurate subclasses under which to categorize the new concept if a more specific subclass applies. For example, if an SME attempts to create a type of #$MilitaryTank, the system would use facts in its knowledge base to elicit further relevant facts about the tank type, such as, size, fording depth, and kinds of military unit with which it is typically associated.

Compared to a previous version of the KRAKEN system, described in [10], the experimental system included improved rule-writing support and NL processing—significantly advancing KRAKEN's support for users' authoring rules. Also, KRAKEN includes a means to translate NuSketch outputs into KB assertions. For the evaluation, NuSketch was used only as an initial segment in the COA development process for KRAKEN: the users substantially refined their COAs using the other facilities in the toolset. As compared to earlier versions of the tool, the interface on the KRAKEN query tool was simplified and a set of very general queries to elicit COA critiquing values was created prior to the evaluation process. An example of such a query is, "Does COA X score positive with respect to Enemy Engagement?". This allowed SMEs to invoke the general queries throughout the process. The presentation of answers and justifications allowed SMEs to identify lacunae and errors in either the SME-created knowledge or the Cyc KB.

SRI's SHAKEN System

The pre-experimental SHAKEN KR and KA system was designed for using graphs, or concept maps (CMaps), to represent concepts, processes, and situations [13]. A root node represented a universal quantification, while other nodes in the graph represented existentials bound to the universal via relations specified in graph arcs (see Figure 1). However, the COA critiquing task is quite different, requiring articulation of which kinds of situations are relevant to the critique, and to what extent instances of such situations are desirable.

[3] More comprehensive information about the experiment—including a full challenge problem specification—is available at http://www.iet.com/Projects/RKF/.

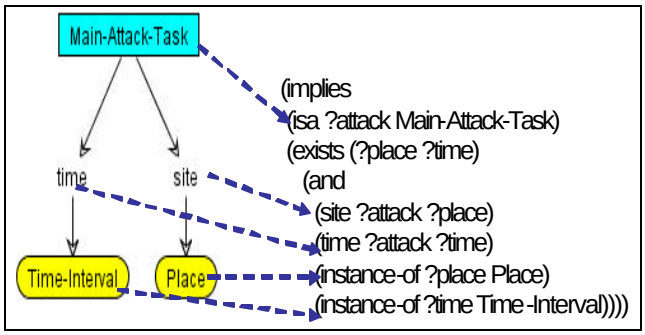

Figure 1. Simple CMap and Logical Mapping

Hence, representing COA critiquing principles required the ability to represent concept maps corresponding to a very different general logical schema than that which was implemented in earlier versions of the tool (see Figure 2). In COA critiquing patterns, the root node relates to a universal characterization of some property that instances of that universal generalization will satisfy. For a more detailed discussion of improvements to SHAKEN in support of the experiment, see [1].

SHAKEN uses *patterns* to capture an important form of COA critiquing knowledge. Each pattern captures an implication and is authored as a kind of CMap having two special edges: the critique-score edge identifies the consequence of the implication; the has-pattern edge identifies the antecedent of the implication. Effectively, SHAKEN interprets the 'hasPattern' predicate as having a radically different meaning than other predicates in the system. Using 'hasPattern' is like moving the non-root nodes in a pattern from being existentially quantified conjuncts in the consequent to universally quantified conjuncts in the antecedent; compare the logical rule represented in Figure 1 to the one represented in Figure 2. Patterns differ from conventional logic in that they do not necessarily require the entire antecedent to be satisfied (see [14] for a more complete discussion of the inference supported by SHAKEN patterns).

The consequence of a COA-critiquing pattern is a proposition comprising a qualitative desirability attribute (*e.g.*, very good, good, poor, very poor) that represents a quality gradation according to (a) a SME subjective assessment and (b) a COA-critiquing dimension. Similar vocabulary is used in the KRAKEN system.

```
(implies
    (and (isa ?engage Engagement-Military-Task)
         (isa ?blue-force Armored-Battalion)
         (isa ?red-force Mechanized-Infantry-Unit)
         (agent ?engage ?blue-force)
         (object ?engage ?red-force)
         (allegiance ?blue-force *Blue)
         (allegiance ?red-force *Red)
         (not (exists ?city
                 (and (isa ?city City)
                      (is-inside ?blue-force ?city)))))
    (coa-score Good Enemy-Maneuver-Engagement))
```

Figure 2. Logic for the Pattern in Figure 3

Another extension important to eliciting accurate rules from SMEs, which was introduced into SHAKEN, was the ability to assert negations. An example is given in Figure 3 (in the diagrams, link names with a (green) box around them represent negations in the graphical language) and the corresponding logic is illustrated in Figure 2.

EVALUATION

During RKF Year 1, SMEs' KR requirements were largely met by building tools for representing and extending subsumption hierarchies, and for describing complex processes (see [12]). COA critiquing presented a greatly extended set of challenges; in particular, the requirement to represent rules representing principles of COA critiquing.

SMEs' authored critiquing rules were evaluated using three approaches with different bases, as follows: 1) functional performance; 2) economics; and 3) intrinsic quality. Further information and documentation is available at [6].

Functional performance evaluation had two phases. The first phase assessed the quality of the SME-authored COAs by posing test questions to the tools for reasoning about the authored KBs and scoring returned answers against a predefined set of criteria. The second phase evaluated the system-generated critiques by comparing them to (textual) manually authored critiques for the same COAs (prepared by the evaluation-participating SMEs in advance of critiquing rules capture). In economic evaluation (following [3]), we measured *in situ* knowledge reuse as an indication of knowledge generality.

It is important to note that this evaluation was not designed as a "bakeoff" between the two systems, *i.e.*, program participants and evaluators were not interested, or at least not primarily interested, in the correctness of a hypothesis of the following form, "System A better facilitates the elicitation of SME COA critiquing knowledge than does System B". The amount of data that can be collected in the kind of evaluation described, *i.e.*, one involving complex knowledge representation for a large comprehensive reasoning task, makes it very difficult to generate enough evidence to be able to confidently reject the related null hypothesis. In addition, the tools were bringing potentially complementary KA methods to the evaluation. For example, the KRAKEN participants devoted far more effort to detailed COA description than did the SHAKEN participants. The evaluation served to subject the systems to a comprehensive KA test. It showed the system designers how effective their tools were for a nontrivial knowledge representation and reasoning (KR&R) task of interest to tool-using SMEs.

The Evaluation-participating SME evaluated their system critiques for correctness and quality, as determined by a manual comparison to a critique generated by the participating SME. Beyond just considering the raw judgments, evaluators also considered explanations—whether each system gave appropriate rationale and considered relevant

circumstances. Quality considerations involved correctness and completeness of explanation. Table 1 presents system success with respect to criteria that the pair of SMEs associated with each system was able to address. The first three columns indicate the system, SME and COA being critiqued. The third and fourth columns indicate the average score for correctness and explanation quality over all criteria considered for a given COA (out of 100). The last column indicates the number of criteria considered for the specified COA and for the specified critique.

Table 1. COA Critique Results

System	SME	COA	Correctness	Quality	# of Criteria
SHAKEN	1	1	55	61	12
SHAKEN	2	1	54	54	6
KRAKEN	1&2	1	70	50	6
KRAKEN	1&2	2	62.5	45	5

The elicited KBs contained sufficient knowledge, in terms of volume and complexity, for each system's reasoning tools to offer high-level critiques of knowledge-rich COAs. In the judgment of the military SMEs, these critiques were plausible from a number of diverse perspectives

As noted, the scores reported for COA critiquing performance do not lend themselves well to cross-system comparison because the averages are over the number of criteria that the SME addressed in creating critiquing rules. No attempt was made to reward or penalize with respect to the number of criteria addressed (see [6] for more detail). For example, SME 1 of the SHAKEN team attempted to represent critiquing principles for 12 different criteria, far more than any of the other SMEs. Attempting to address more critiquing criteria may have brought down this SME's average score. Similarly, SMEs were given complete latitude in choosing the criteria for which they represented rules.

The number of rules created by SHAKEN's SME users, an average of 30 per complete elicitation session, reflects well on its graphical elicitation process. The rule elicitation interface on the KRAKEN system is robust and functional and includes a number of useful tools that ensured effective quality control and system integration, but SMEs using it were not able to use it to create rules at the same pace, Kraken SMEs created 8 rules per complete elicitation session. As the examples below indicate, the elicited rules were comparable in terms of complexity, as measured by the number of distinct terms appearing in the rules, and granularity in the level of representation. The rules had similar critiquing foci but a noteworthy difference is the fact that KRAKEN SMEs did create different pairs of rules that produced a reasoning chain, *i.e.*, a rule in which the consequence of one rule could be used to help determine whether the antecedent of another SME rule held. Also, the SMEs working on the KRAKEN system spent a larger proportion of their time on another component of the evaluation, *i.e.*, formulating COA descriptions that went beyond the rudimentary descriptions elicited via NuSketch. Hence, while the quality of many of the rules produced in KRAKEN was high and KRAKEN COAs were very well axiomatized, the number of critiquing rules created was low compared to the number of SME-created critiquing rules in SHAKEN. The higher number of SHAKEN rules also explains the *prima facie* disproportionate focus on SHAKEN rules below.

REPRESENTATION ISSUES

The remainder of this paper focuses on our (the authors') evaluation—with respect to intrinsic quality—of the rules captured by SMEs. We organize the presentation of critiquing rule quality assessment around particular knowledge representation and reasoning issues. Tools used to facilitate the analysis below included Pragati's Multi-ViewPoint Clustering Tool [8], the WhyNot Debugging tool [2] and the KANAL tool [7].

Necessary-Sufficient Conditions

SMEs occasionally considered representation problems from an awkward perspective. At times SMEs confused necessary conditions with sufficient conditions or wrote rules for the purpose of detecting a large number of necessary conditions for a positive assessment with respect to a criterion while a more efficient representation may have been to simply query for the presence of sufficient conditions for a negative assessment.

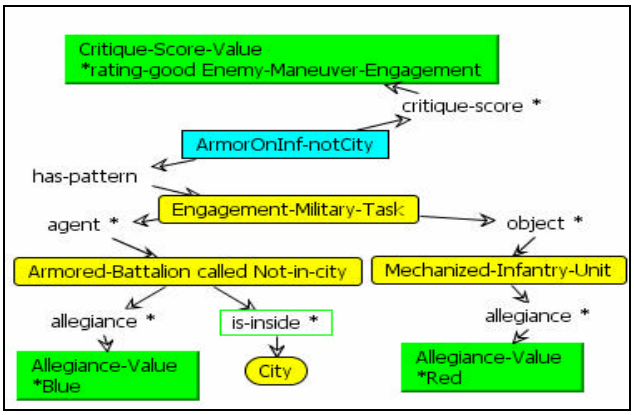

Figure 3. Example of Overly Specific Rule

For example, Figure 3 supports an inference that a given COA has specified an adequate plan for managing force maneuverability if it deploys an armored battalion to engage enemy forces in a non-urban location. However, perhaps deployment within a marsh, for example, would also pose problems for an armored force. A non-urban setting may be necessary for desirable maneuverability of armored forces but it is not sufficient. Rather than stating this as a sufficient criterion for establishing a positive score for maneuverability it might have been preferable to state that the negation of this condition suffices to establish a negative

score; that is, deploying armored forces within a city is sufficient to infer that a COA scores poorly for maneuverability.

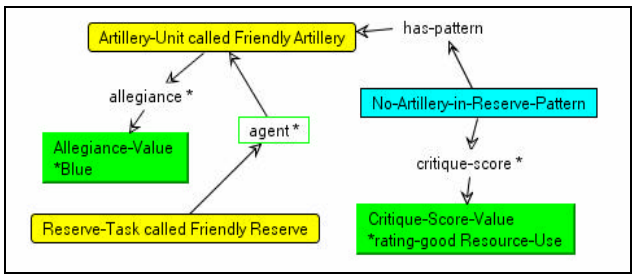

Figure 4. A Pattern Implementing Negation

Figure 4 further illustrates possible difficulties in failing to recognize the most efficient method of implementing a critiquing principle. The SME created a pattern to find all instances in which a reserve task fails to implement artillery and scores this as "good". While this is a necessary condition for good resource use when instantiated for every reserve, the SME could have also used a rule reflecting that a single failure is sufficient for poor resource use.

Compare the rule in Figure 4 with the SHAKEN rule represented in Figure 10 or the KRAKEN rule in Figure 5. Both indicate better approaches to the principles in question. Figure 5, for example, queries the COA for situations in which no unit has been assigned to another unit, or more precisely situations in which the system knows of no assigned unit. Here the sufficient conditions for ineffectiveness rather than the necessary conditions for effectiveness have been represented. This representation may have resulted from an effective consultation with a KE after the SME attempted a less direct representation, thus offering a possible example of how brief KE/SME interaction may facilitate effective KR. It is noteworthy that one of the tools developed but not fully integrated prior to this evaluation, Teknowledge's SCOOP tool, was designed for such purposes, that is, creating a collaboration environment in which appropriate kinds of rules can be flagged and automatically directed to SMEs for further consideration before being integrated into a KB [11].

```
(implies
    (and
      (isa ?COA CourseOfAction)
      (coaRedUnit ?COA ?RED)
      (unknownSentence
        (thereExists ?BLUE
          (and
            (isa ?BLUE BlueUnit)
            (unitAssignedToUnit ?BLUE ?RED)))))
    (evaluatesDimensionOfCOA ?COA engagementOfEnemyCombatPowerInCOA
IneffectiveForPurpose))
```

Figure 5. KRAKEN Rule Concerning Unit Assignments

One can also envisage an elicitation tool designed to address such problems. Tools might query an SME as to whether a given state of affairs is necessary, sufficient, or merely indicative of a type of assessment being applicable to a COA. If the SME needs to indicate that it is merely necessary, one can have the system offer an obversion-like transformation of the proffered rule as a possible sufficient condition for the opposing assessment (*e.g.*, 'good' or 'effective' for the criterion rather than 'poor' or 'ineffective') in its stead. Nevertheless, given representational subtleties, especially those associated with negation and quantification, fully automated elicitation of any arbitrarily complex rule remains somewhat elusive.

Generality Issues

In this evaluation SMEs were working on fairly focused concrete scenarios. While this focus on a particular task seemed to usefully motivate the rule representation effort, an unfortunate consequence of the scenario focus is that SMEs occasionally tended to overly restrict a rule by including unnecessary details from a particular scenario.

Consider again the pattern in Figure 3. Is the maneuverability of an armored force compromised only at the battalion echelon, or would the maneuverability of an armored division also be compromised within a city? Need the opponent be a mechanized infantry? Is this true only for forces having Blue allegiance or might a Red armored battalion also have compromised maneuvering within a city? Such details may have served to unnecessarily restrict the pattern by making it too specific.

SMEs also exhibited a tendency to include details that, while true of the scenario in question, were unnecessary for the critique's success. In the example in Figure, the fact that the bridge supports traffic by armored vehicles may be a detail specific to the given COA being considered and may not be generally necessary or useful.

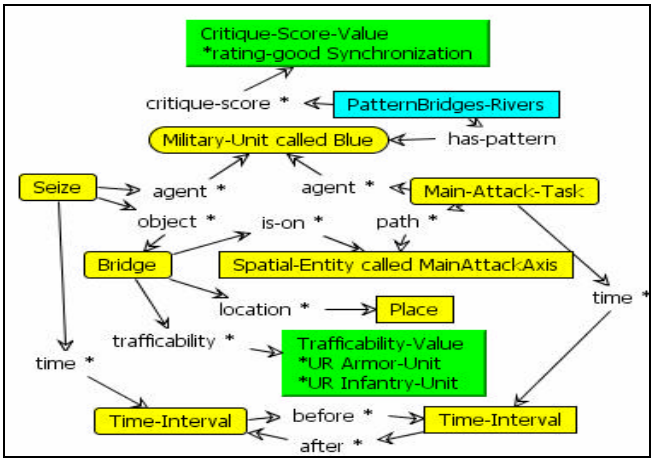

Figure 6. Example of Extraneous Detail

Conversely, while the aforementioned rules were too specific, the KRAKEN rule depicted in Figure 7 seemed to be too general for effective reuse. It requires a minimum 3:1 ratio for task effectiveness for any task in a COA or at least for any task with which a force ratio is associated. However,

many tasks, such as screening or defend tasks, could presumably be carried out effectively without such a ratio.

```
(implies
 (and
  (genls ?THECOA Event)
  (isa? THECOA CourseOfAction)
  (componentTaskOfCOA ?THECOA ?THETASK)
  (genls ?THETASK Event)
  (isa ?THETASK AssignedTaskType)
  (forceRatioForTask ?THETASK ?THEQUANTITY)
  (isa ?THEQUANTITY ScalarInterval)
  (greaterThan 3 ?THEQUANTITY))
 (evaluatesDimensionOfCOA ?THECOA
   enemyManeuverEngagementInCOA IneffectiveForPurpose))
```

Figure 7. General KRAKEN Rule

Allowing SMEs to develop rules based on a concrete reasoning situation was usually helpful but it also seems that such an elicitation may need to be followed by an explicit generalization step in the knowledge-authoring process after the initial conception of new knowledge. KRAKEN did query the user about generalizing specific type-level ground atomic formulas. Finding ways to extend this to more complex rules may be worthy of consideration by both development teams. After allowing the SMEs to author new knowledge in very concrete terms, it subsequently becomes desirable to reconsider the knowledge, to generalize it or refactor it, and to explain shallow rules in terms of first-principles, generalizing the concrete knowledge to apply to a broader and more appropriate set of data. One approach to doing this is to explain shallow rules with first-principle rules as they become available [9].

Negation Reasoning

In SHAKEN, a very high proportion of patterns were designed to reason about the positive dimensions of a COA rather than about negative assessments. Of 60 consequents or conjuncts in consequents of critiquing rules, 51 assessments generated a positive conclusion about the COA, while only 9 generated a negative conclusion. We noted above that this tendency often resulted in less than efficacious rules.

Other evidence about SME confusion regarding the implementation of negation is found in the pattern depicted in Figure 8.

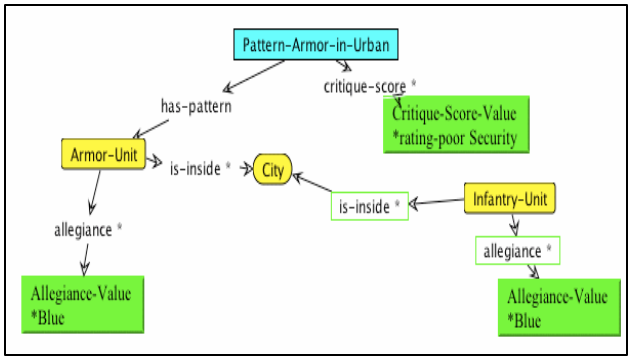

Figure 8. Negation Implementation

This rule suggests that when a Blue armored unit is found inside a city, the existence of any non-Blue infantry unit outside (not is-inside) that city decreases the security score for that COA. This probably fails to represent what the SME intended. This may simply be a case of there being too many negations in the graph. A more plausible rule would be one in which the allegiance *is* negated, but the is-inside *is not*. That would yield a rule saying security is poor if you have one of your armor units in a city which has unfriendly infantry in it.

Negation seems to require special care in such elicitation. SMEs should be encouraged to use it effectively so as to avoid representational infelicities. This may require occasional KE intervention and tools to help SMEs understand and implement effectively (*e.g.*, "You just used the negation of a predicate, do you really mean to say …")

Inconsistency/Redundancy Issues

Consideration of the SME KBs also raises some issues regarding inconsistencies, redundancies, and extraneous knowledge.

For example, the pattern of Figure 9 captures domain knowledge that the available forces of a main attack should exceed that of a supporting attack. However, it unnecessarily commits to particular available-force-ratio values. Furthermore, the given values are equal and so violate the specified inequality, that is, logically the rule supporting this pattern would never be bound.

The patterns in Figure 4 and Figure 10 represent a related phenomenon. These are patterns created by different SMEs so neither SHAKEN nor the SMEs should be faulted for their existence. However, their respective meanings are similar to the extent that 'poor' is the negation of 'good'. One of the rules might be seen simply as something quite like the obverse form of the other. One can imagine that a mature system might detect such reusability across SMEs' KBs and eliminate one or query for more information where the form of the rules differs to some extent.

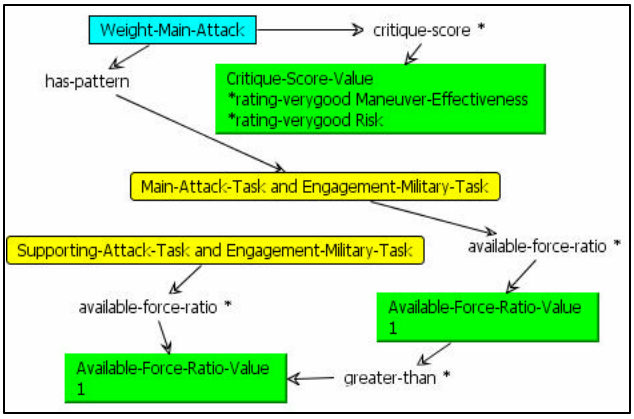

Figure 9. Pattern Re Relative Force Ratios

In KRAKEN, there were also observed redundancies. Consider the rule depicted in Figure 11. A second version of it was created in which the SME replaced 'parts' with 'suborganizationOf', but 'suborganizationOf' is a specialization of 'parts'—that is, if (suborganizationOf X Y), then (parts X Y), so it is not clear that both forms of the rule are necessary.

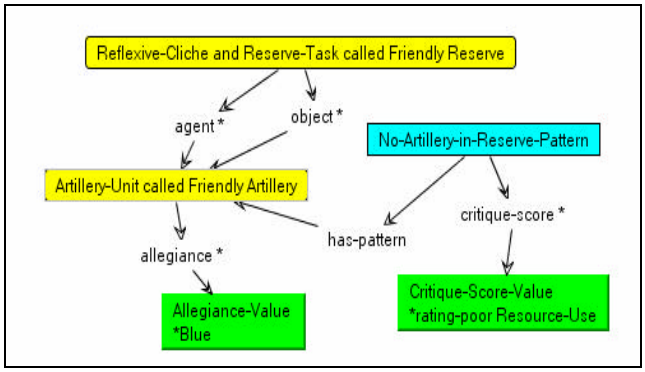

Figure 10. No-Artillery-In-Reserve-Pattern-2

```
(implies
 (and  (isa ?THE-1ST-TASK AssignedTaskType)
       (targetInAttackTask ?THE-1ST-TASK
       ?THE-1ST-BATTALION)
       (isa ?THE-1ST-BATTALION
       Battalion-MilitaryEchelon)
       (parts ?THE-BRIGADE ?THE-1ST-BATTALION)
       (isa ?THE-2ND-TASK AssignedTaskType)
       (different ?THE-1ST-TASK ?THE-2ND-TASK)
       (targetInAttackTask ?THE-2ND-TASK
       ?THE-2ND-BATTALION)
       (isa ?THE-2ND-BATTALION Battalion-MilitaryEchelon)
       (parts ?THE-BRIGADE ?THE-2ND-BATTALION)
       (different ?THE-1ST-BATTALION
       ?THE-2ND-BATTALION))
(targetInAttackTask ?THE-2ND-TASK ?THE-BRIGADE))
```

Figure 11. Redundant Rule

Consistency and redundancy checking might be implemented with a KR system's judicious implementation of its own inference tools. Inconsistencies should serve as prompts to the system that the SME has failed to represent what he or she intended or the need to contextualize knowledge appropriately. The KRAKEN system already implements some such safeguards insofar as the system detects ill-formed assertions, *e.g.*, violating argument constraints, and advises users how they might revise appropriately. In addition to using the system's own inference tools, it would be useful to implement existing tools for finding areas of overlap in KBs.

Incompleteness

There were several cases in which SMEs created new classes and relationships. A common error when doing so was incorrect placement of the new concept into the preexisting KB. Also, SMEs did not always state sufficient information about a new concept to differentiate it from other preexisting concepts in the KB. For example, one SME authored a class that represented air-defense forces. It was defined as a new specialization of the predefined Military-Unit. However, it differed only in its name from the superclass. It included no local slot values (other than the values of superclass). So, either it is a poorly defined term or it is incomplete work that the SME should have deleted or completed. A more helpful interface would better aid the SMEs in managing the status of their work to help make it clear when there are incomplete formalizations that need to be removed or completed.

SMEs occasionally failed to give enough information concerning spatial and temporal constraints. For example, **Figure 12** shows a rule in which the SME has failed to specify that the place at which the Fire-Support-Task occurs must be spatially proximate to the location of the artillery unit.

Also noteworthy are other kinds of content that SMEs failed to consider. Several patterns authored by SHAKEN SMEs checked dependencies between actions. For example, one assessed a unit based on the ordering of its assigned tasks, and another checked for the presence of a follow-on mission. Another pattern linked the actions of friendly and opposing forces by checking whether an aviation attack is used to inhibit an opposing counterattack. However, no SMEs authored any knowledge that checked causal or parent/child relationships in the plans.

SMEs also added knowledge that checked the global structure of the plan. In SHAKEN, expected effects in Kanal [7] were used to test that the plan achieved its goals. For example, both KRAKEN rules and SHAKEN patterns were used to test that each of the Red reserve units was engaged by some task. However, no knowledge was authored by the SMEs to test the clarity or simplicity of the COA.

Ultimately, systems might benefit from a more general application of tools like the KRAKEN Salient Descriptor that reason about expected knowledge based on knowledge found in the KB. So, for instance, when an SME creates a pattern for critiquing a particular kind of movement, the system might prompt for spatial or temporal information, should the rule fail to implement any spatial or temporal predicates. Similarly, attack information might be expected to make men-

tion of particular weapon types, spatial proximity, and so on. Failure to include such information would be indicative of possibly abandoned or incomplete pattern/rule representations and would prompt a query to the user.

Discussion

With respect to improving KA tools, several of the errors made by SMEs stem from the inability of KB systems to adequately explain the meaning of knowledge contained in the system. SMEs regularly reported that the systems did not tell them clearly enough all they needed to know. In some cases, the knowledge was in the KB, but the interface left it difficult for SMEs to find. In others cases, such as system-generated explanation of predicates, rules, and inference results, their interpretation required KE-level skills.

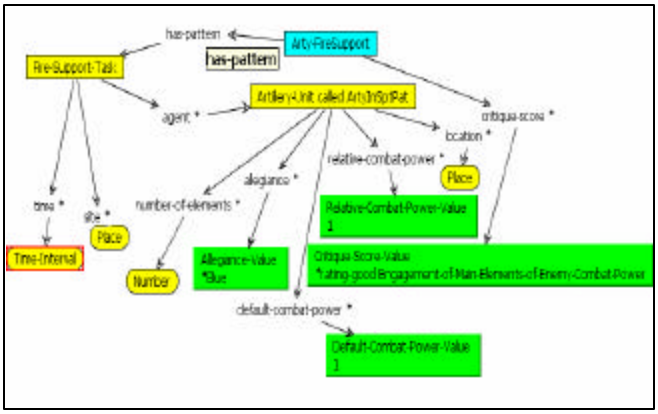

Figure 12. Missing Information about Location Relations

An important lesson to be learned from the KA effort in this evaluation process was the utility of having SMEs progress from representing a particular scenario to the representation of general principles for critiquing. This facilitated the knowledge elicitation in a number of ways. It gave SMEs a concrete introduction to the KR language and KA environment. For purposes of rule elicitation, the focus on a particular scenario rendered the initial rule articulation much more manageable for the SME. It would have been more difficult to articulate rules from more universally acceptable general principles initially. Upon completion of the initial phase of scenario-based rule articulation, these rules could then be examined for generalization, either by a system tool or in consultation with a KE. Finally, the concrete scenario provided an excellent test case for the rules that the SMEs entered into the system. SME familiarity with the scenario facilitated analysis and revision of knowledge based on results of queries posed against that scenario.

A clear example of the utility of this testing process can be given from the implementation of the KRAKEN tools. Its users used the querying facility (matrices for diagnosis and evaluation) to generate (relatively) easy-to-read baseline analysis and thereby motivate new COA KR. The SMEs regularly queried the system to determine whether the system was drawing the kinds of inference intended. The explanation interface allowed for KR and query diagnosis. For example, upon posing the query indicated in Figure 5 an SME made the following observation:

[This critique] failed to connect knowledge about specific tasking of individual Blue units to attack and destroy specific individual Red units. Thus, concluded that the COA was disadvantageous with respect to engagement of main elements of enemy combat power because no Blue units were responsible for any Red Units. A new rule must be developed to link each Blue unit with a targeted Red unit.

As a result, the SMEs added an additional rule to the KB, as indicated in Figure 13. Querying a scenario with which an SME is well acquainted helps the SME adjudicate correctness and revise principles or add reasoning support as necessary.

```
(implies
 (and
  (isa ?UNITA ModernMilitaryUnit-Deployable)
  (different ?UNITA ?UNITB)
  (isa ?UNITA ModernMilitaryUnit-Deployable)
  (targetInAttackTask ?THE-ATTACKING-TASK ?UNITA)
  (unitAssignedToAction ?THE-ATTACKING-TASK ?UNITB)
  (isa ?THE-ATTACKING-TASK AssignedTaskType))
(unitAssignedToUnit ?UNITB ?UNITA))
```

Figure 13. Additional Rule to Prevent "enemy engagement" Rule from Misfiring

CONCLUSIONS

The evaluation presented a significant challenge to the integration teams in terms of designing systems that SMEs could use to articulate reasoning principles requiring fairly complex logical representation. Evaluation results showed that SMEs were able to write KB rules that enabled correct analyses of COAs for some criteria, and in a manner that presented coherent explanations for the critique. Above we have analyzed some of the shortcomings in those representations and possible means of addressing them in terms of the focus of future work on development of KA tools.

Some of the elicited formal rules reflect KA challenges that might motivate interface extensions, including helping to manage work in progress and removing unwanted work, making the knowledge actually captured more transparent, connecting authored knowledge to prior knowledge, facilitating consistency checking within an SME's work and across SME efforts, extending dialogue tools to help ensure completeness and implementing more complete integration of tools for regular cycles of querying and resolution. One of the interesting questions concerns ways in which these systems present complementary KA tools that might be reintegrated into an even more effective system. We have noted a number of useful tools that exist within KRAKEN, for example, the Salient Descriptor, query suite, and generalization tool, that could be applied to some of the knowledge elicitation problems arising in SHAKEN. However,

also notable is the fact that the SHAKEN rule elicitation tool – that is, the graphical interface within which a rule is mapped to a graph -- facilitated the elicitation of large numbers of patterns, many more than could be rapidly elicited in KRAKEN.

ACKNOWLEDGMENTS

This research has been supported under DARPA's RKF program [4] contract and has been approved for public release, distribution unlimited. Thanks also to Vinay Chaudhri for helpful feedback on an earlier draft.

REFERENCES

[1] Barker, K., Blythe, J., Borchardt, G. Chaudhri, V., Clark, P., Cohen, P., Forbus, K., Gil, Y., Katz, B., Kim, J., King, G., Mishra, S., Murray, K., Otstott, C., Porter, B., Schrag, R., Uribe, T., Usher, J., and Yeh, P. "A Knowledge-Based Tool for Course of Action Analysis ", to appear in the Fifteenth Conference on Innovative Applications of Artificial Intelligence (IAAI-03), 2003, Acapulco, Mexico.

[2] Chalupsky H., Russ T., "WhyNot: Debugging Failed Queries in Large Knowledge Bases" In *Proceedings of the Fourteenth Conference on Innovative Applications of Artificial Intelligence*, pp. 870-877, 2002.

[3] Cohen, P., Schrag, R., Jones, E., Pease, A., Lin, A., Starr, B., Gunning, D., and Burke, M. "The DARPA High-Performance Knowledge Bases Project." *AI Magazine* 19(4):25, 1998. Panther, J. G., *Digital Communications*, 3rd ed., Addison-Wesley, San Francisco, CA (1999).

[4] DARPA, *The Rapid Knowledge Formation Project*, http://reliant.teknowledge.com/RKF, 2000.

[5] Forbus, K., Usher, J., and Chapman, V. 2003. "Sketching for Military Courses of Action Diagrams", Proceedings of the Intelligent User Interfaces Conference (IUI-03), January 2003. Miami, Florida

[6] IET's RKF webpage. http://www.iet.com/Projects/RKF/.

[7] Kim, J., and Blythe, J. "Supporting Plan Authoring and Analysis, Proceedings of the Intelligent User Interfaces Conference" (IUI-2003) , pp. 109-116, Miami Beach, Fl, January 2003.

[8] Mehrotra, M. "Ontology Analysis for the Semantic Web ", AAAI-02 Workshop on Ontologies and the Semantic Web. Edmonton, Canada. July 28-29, 2002.

[9] Murray, K., 1990. Improving Explanatory Competence, *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, 309-316.

[10] Panton, K., Miraglia, P., Salay, N., Kahlert, R., Baxter, D., and Reagan, R. "Knowledge Formation and Dialogue Using the Kraken Toolset", in *Proceedings of the IAAI-02*, pp. 900-905. Menlo Park, CA. 2002.

[11] Pease, A., and Li, J. "Agent-Mediated Knowledge Engineering Collaboration", in *Proceedings of the AAAI 2003 Spring Symposium on Agent-Mediated Knowledge Management*. AAAI Technical report SS-03-01, 2003.

[12] Schrag, R., Pool, M., Chaudhri, V., Kahlert, R., Powers, J., Cohen, P., Fitzgerald, J., and Mishra, S. "Experimental Evaluation of Subject Matter Expert-oriented Knowledge Base Authoring Tools", in *NIST Special Publication 990: Performance Metrics for Intelligent Systems*, NIST: Proceedings of the 2002 PerMIS Workshop, Gaithersburg, Maryland, pp. 272-279, August, 2002.

[13] Thoméré, J., Barker, K., Chaudhri, V., Clark, P., Eriksen, M., Mishra, S., Porter, B., and Rodriguez, A. "A Web-Based Ontology Browsing and Editing System", in *Proceedings of the IAAI-02*, pp. 927-934. Menlo Park, CA.

[14] Yeh, P., Porter, B., and Barker, K. "Transformation rules for knowledge-based pattern matching", The University of Texas at Austin, Computer Sciences Technical Report UT-AI-TR-03-299. 2003