A Constraint-Based Expert Modeling Approach for Ill-Defined Tutoring Domains

Angela Woods, Brian Stensrud, Robert Wray, Joshua Haley, Randolph Jones
Soar Technology, Inc.
<angela.woods, stensrud, wray, joshua.haley, rjones>@soartech.com

Abstract

We introduce an approach for representing and diagramming machine-readable expert models for intelligent tutoring systems (ITSs) and virtual practice/training environments. Building on previous work on constraint-based expert models, we address the challenges of implementing ITS features within complex, ill-defined learning domains. Our constraint-based expert model (CBEM) can be used, in conjunction with a real-time interpreter (the *Monitor*), to make accurate, real-time observations of student behavior in ill-defined domains. These observations can be used for in-situ feedback and dynamic, individualized experience tailoring and instruction. In this paper, we detail the structure and elements of the CBEM, describe instances of successful application to several training domains, and introduce current and future research to extend and improve the paradigm.

Introduction

Intelligent Tutoring Systems (ITSs), when properly developed and deployed, can serve as cost-effective alternatives to the impractical option of providing learners with personal training assistants or tutors. ITSs typically consist of four basic components: the *expert* or *domain model*, the *student model*, the *instructor model*, and the *learning environment*. Together, those four components deliver individualized instruction or training.

Developing expert and student models for ITSs is not trivial, and it becomes even more challenging when working in an ill-defined domain. In an ill-defined domain (Omerod, 2006), the space of possible actions is typically large (Derry and Lajoie 1993), and it is not always possible to classify actions as correct or incorrect. This is, in part, because the correctness of an action is so heavily dependent on contextual factors that vary from situation to situation (Ogan, Wylie and Walker 2006). Additionally, the rules underlying the domain may not be well-understood, and are thus not formalized.

Mitrovic and Ohlsson's constraint-based modeling approach (1999) provides an alternate way to represent knowledge to address the challenges of working in an ill-defined domain. In this approach, a *constraint* is an ordered pair <R, S> where R is the relevance condition and S is the satisfaction condition. R and S are stated in the form of propositions and can represent simple or complex patterns. Constraints encode characteristics of correctness for a particular domain, thereby implicitly defining an *envelope* of acceptable activity.

The authors have developed a domain-general expert modeling approach and toolset inspired by the constraint approach. Constraints in the expert model form an implicit "envelope" that bounds behavior (Wray and Woods 2013). As long as learner behavior remains within the envelope, the system allows a wide range of learner actions. Constraint-based modeling provides an alternate way to represent knowledge that focuses on relationships between properties of a proposed solution instead of explicit assertions and declarative statements. It is a flexible representation that has proven useful in addressing expert modeling challenges presented by ill-defined domains (Wray et al. 2009).

In this paper, we describe the components of our expert modeling approach and language for ill-defined domains and describe ill-defined domains where we have successfully applied the paradigm.

Background

A requirement for ITSs is the ability to track and assess student behavior during a session or exercise. As ITS systems began to gain prominence during the late 1980s, the most popular methodology for achieving this became known as *model tracing* (Anderson et al 2014). In model tracing, student performance is compared against an ex-

plicit (typically graph-based) representation of correct performance – an *expert model*. This comparison can be used to both assess the student's deficiencies in a particular topic, and also to provide just-in-time hints to the students to unblock them.

While model tracing approaches have proven effective in classical tutoring domains (e.g. high school mathematics and physics), they begin to break down in ill-defined domains. Specifically, some frequently present students with situations for which there is no single correct response or path, but rather an *envelope* of correct strategies too numerous to explicitly model. Mitrovic and Ohlsson (1999) define this envelope implicitly as a set of state/constraint ordered pairs, called *constraints*. These constraints, while not prescriptive, can be used to identify when and how a student makes an error even in ill-defined domains. Another relevant approach for representing political and group-based problem solving domains is known as *issue-based systems* (Kunz and Rittel 1970), though its application to training and educational systems is minimal.

The authors' primary area of interest is observation and targeted student tailoring (Wray, Folsom-Kovarik and Woods 2013) in virtual and constructive training environments. Like classical ITS domains, these environments involve a student or novice who is attempting to execute a particular skill. However, particularly in virtual training environments, it is infeasible to specify and enforce a specific sequence of correct behavior even in well-defined training domains, for a variety of reasons:

- Even in restrictive domains there can a large *space of activity* that is correct or compliant at any given time
- The virtual/constructive environment is *nondeterministic*, primarily because it involves the participation of other actors
- The student can interact with the environment, and other actors in that environment, in a variety of ways, and only a subset of those interactions are relevant to proficiency observations

As such, we were compelled to exploit a constraint-based approach to expert modeling – this approach is introduced in this paper. Specifically, we sought to instantiate a constraint-based, easily composed expert model that could effectively be compared against student activity in these specific types of training and practice environments.

Concept

In this section we introduce a domain-general expert modeling paradigm, the *constraint-based expert model (CBEM)*. Using this paradigm, developers of intelligent tutoring and training systems use a graphical diagramming language in an authoring UI to create models that describe the bounds on expected student behavior. The authoring UI exports the CBEMs to a machine-readable XML representation that is used by a real-time interpreter (the CBEM *Monitor)* to detect correct and incorrect behavior according to the CBEM description.

CBEM Diagramming and Primitives

For well-defined domains, where the possible solution paths are known and can be enumerated, it is possible to build effective expert models to support diagnosis of student behavior. In ill-defined domains where this is not feasible, we can get good traction by using constraints for diagnosis. Even with constraints, it can still be expensive to author all the constraints necessary for good instruction. We address this problem with a diagramming tool for authoring diagnostic constraints in ill-defined training domains. We have developed a simple, domain-general diagramming paradigm that can be used by content developers and subject-matter experts to compose CBEM models. The diagramming language includes the following primitives:

- **Context:** A context represents a particular situation under which a set of constraints is applicable. A context describes major segmentation of the training space (e.g. phases of a mission). Given the complexity of a training scenario and the domain, the student may iterate through a variety of different contexts. Contexts are additive; more than one context may be active simultaneously.
- **Actions:** Actions are the space of all identifiable activity that can take place within the training scenario including student and non-player character actions.
- **Logic-blocks:** Logic-blocks describe conditions of the training environment that can result either as a consequence of time advancement or as side effects of actions. A logic-block can contain an arbitrary Boolean test of simulation state, including temporal or spatial conditions.
- **Clusters:** Clusters are logical groupings of actions and logic-blocks that are used to indicate control flow through the constraint space. Clusters may be nested (sub-clusters) as needed to express relationships and groupings. Clusters have a *type* that represents how actions and logic-blocks in that cluster are interpreted.
- **Dependencies:** Dependencies are causal links between clusters that specify expected temporal ordering.
- **Selectors:** Actions and logic-blocks may need parameters, similar to the way that a verb in a sentence needs a subject. The subjects and targets of actions and logic-blocks are objects in the training environment

and are specified as parameters to actions and logic-blocks using constructs called selectors.

- **Bindings:** If a specific object described by a selector is needed for use in more than one successive action or logic-block, a handle to that object is created called a binding.

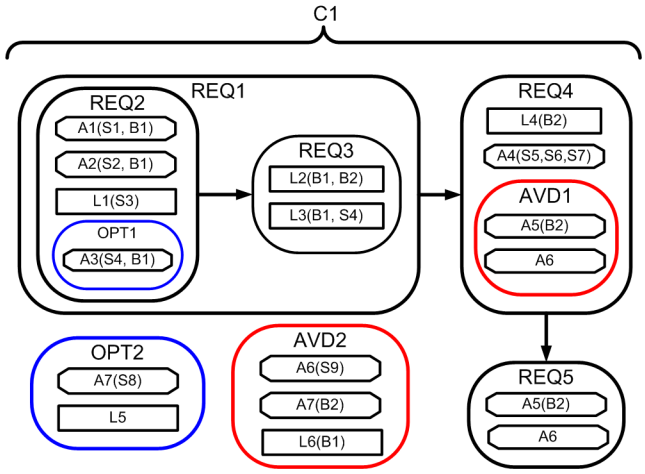

Figure 1 - The CBEM Diagramming Language

Figure 1 shows an example CBEM diagram using these primitives. This diagram represents a single *context*, *C1*. Within context *C1*, there are nine *clusters* (*REQ1-REQ5, OPT1-OPT2, AVD1-AVD2* that contain sets of possible student *actions* (*A1-A7*) and *logic-blocks* (*L1-L6*)). Actions and *logic-blocks* are further defined by specific *selectors* (S*) or *bindings* (B*) that compose the events parameters. Clusters can be one of three types:

- **Required (REQ) clusters** contain actions that *must* be executed by the student while that cluster is active
- **Optional (OPT) clusters** contain actions that *are relevant* to the domain while that cluster is active
- **Avoid (AVD) clusters** contain actions that *are forbidden* while that cluster is active

Within *C1*, an arrow connects *REQ2* and *REQ3*. The arrow is a dependency; events in *REQ2* must be executed before the actions *REQ3*. For example, the action sequence *A1→A2→L1→L2→L3* is correct, while *A1→A2→L3→L1* is incorrect. *OPT* and *AVD* sub-clusters may be nested inside of *REQ* clusters (e.g. *OPT1* is nested inside *REQ2*). By combining nested clusters with dependencies, authors can express activities whose appropriateness changes throughout the course of the context. For example, the cluster *REQ4* depends upon cluster *REQ1* that in turn requires that both sub-clusters *REQ2* and *REQ3* be completed. Action *A6* is expressly forbidden after *REQ1* is completed but before *REQ4* is completed. However, once *REQ4* has completed, *A6* becomes a required action that must be completed.

Figure 2 illustrates how easy it is to use CBEM to model any domain, for example, bread making. Here, ingredients may be added in any order, but not all of them are required.

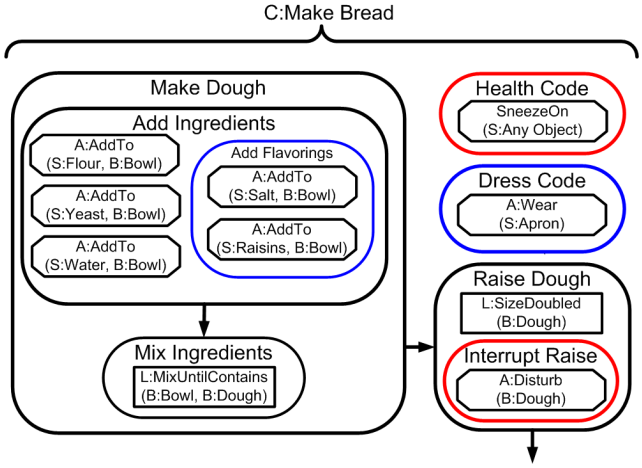

Figure 2 – Bread CBEM

Flavoring ingredients are optional. A binding (B:Bowl) ensures that all ingredients are added to the same bowl. Dependencies enforce strict ordering where appropriate (e.g. Make Dough must occur before Raise Dough, and Raise Dough would be followed by Bake Bread). Abstract domain concepts (e.g. MixUntilContains or SizeDoubled) can easily be represented as logic-blocks. Dependency nesting combined with AVD and OPT clusters enables some activities to be relevant throughout the context (e.g. Health and Dress Code) whereas others (e.g. Interrupt Raise or Add Flavorings) are only appropriate during a specific part of the context, as dictated by their containing REQ cluster.

Identifying and Classifying Student Behavior

The constraint patterns a CBEM describes are used to classify student behavior. The run time interpreter (Monitor) can be envisioned as a set of if-then statements that are continuously evaluated and activate when the 'if' conditions become true (causing the 'then' effects to occur). The types of classifications the Monitor can make include:

- **Correct behavior** occurs when all of the required actions and logic-blocks contained in a *REQ* cluster have been completed. For example *REQ2* is complete when *A1*, *A2* and *L1* are completed.
- **Commission errors** occur when an action or logic-block inside an *AVD* cluster is completed. For example, when *L6* is completed, a commission error occurs.
- **Dependency errors** occur when the student performs a context-appropriate action in the wrong order. De-

pendency errors are detected when the student executes an action in a *REQ* cluster that depends on the completion of a second *REQ* cluster that is not complete, where both clusters are in the same context. For example, if *A2* has not been completed (*REQ2)* is not complete and *L2* in *REQ3* is completed, then a dependency error occurs.

- **Omission errors** occur when the student fails to complete a required action in a context. These errors are detected when the context changes, or goes out of scope, and all of its REQ clusters have not been completed. For example, if context *C1* goes out of scope before action A5 occurs, an omission error occurs.
- **Context errors** occur when the student performs an action that is not appropriate for the context. For example if action *A10* which does not exist in context *C1* is executed, a context error occurs.

Note that this constraint classification makes the tenuous assumption that student/trainee activity can always be assessed as correct or incorrect. In the summary section, we provide thoughts on possible extensions and modifications to the classification breakdown above to support 'gray areas' of student behavior and activity with respect to our constraint methodology.

RUNTIME INTERPRETATION

The CBEM diagramming constructs can be represented in a machine-readable format. The authors have developed a run-time interpreter (the Monitor) for the machine-readable format. The Monitor is data driven and loosely coupled to the training environment. To configure the Monitor for use with a specific training environment, it is instantiated using a set of data known as *domain bindings*. Domain bindings enumerate the type of objects, actions and logic-blocks supported by that particular training environment.

Scenario authors create CBEMs to specify machine-readable constraints for student activity that are relevant to specific training scenarios. The Monitor imports the CBEMs and actively observes the sequence of actions generated in the training environment, comparing the student actions to the constraints found in the CBEMs to detect correct and incorrect student behavior. To do this, the Monitor uses an efficient pattern-matching engine (Laird, 2012). The representation of the CBEM converts easily into a working-memory representation, while activity in the training environment is converted into the agent's working memory. The Monitor uses its pattern-matching machinery to identify situations in the runtime activity feed coming from the training environment that match patterns described by the CBEM model author.

The Monitor is part of a larger, general-purpose Dynamic Tailoring System (Wray, Folsom-Kovarik and Woods 2013). The Monitor's output is consumed by a component called the *Pedagogical Manager,* which maintains an estimate of student proficiency and selects mediation strategies.

APPLICATIONS

The authors have successfully applied the CBEM to a variety of different domains containing ill-defined content. We describe these applications below.

Course of Action Analysis

We have developed an instructional game (illustrated in Figure 3) focusing on Course of Action Analysis (COAA) which incorporates CBEMs and the runtime Monitor (Wray, Woods and Priest 2012). COAA involves step-by-step evaluation of candidate courses of action (COAs) to find gaps, to recognize the need for synchronization (combat power), to coordinate unit actions with those of other units, and to improve understanding of the plan. The COAA instructional game allows a student to practice creating a COA and executing the wargaming process. Effective practice requires guidance and feedback. We integrated the Monitor and supporting CBEMs to deliver guidance and feedback based on the specific actions of individual students.

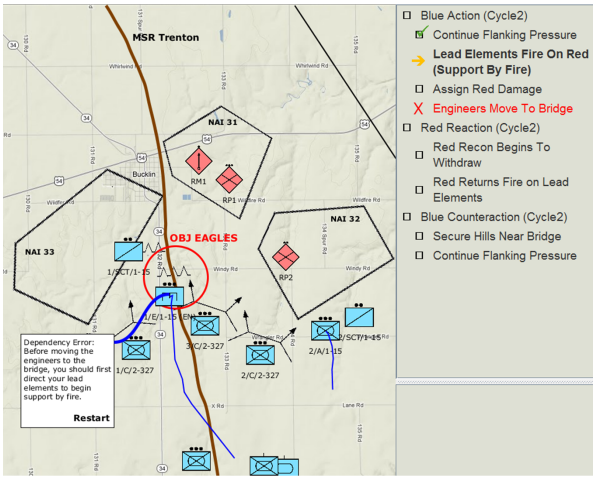

Figure 3 – COAA Using CBEM and Monitor

To explore the effectiveness of this approach, Soldiers were run through a study to compare the COAA instructional game to a control that approximates how games are typically used in simulation centers (Priest Walker and Wray 2014). In the control, the number of omission and commission errors indicated an inability to self-diagnose and correct for those types of errors based on static, after-

practice feedback. Furthermore, control participants were making the same errors over and over, indicating that these errors were a sign of trainees not learning from their mistakes. In contrast, participants using the COAA instructional game were able to virtually eliminate errors of omission and commission from later, more complex scenarios, indicating a learning effect.

Interpersonal Communication

A second application is being developed for the Immersive Naval Officer Training System (INOTS) that seeks to provide a controlled role-play environment for Navy officer training (Campbell et al. 2011). INOTS focuses on interpersonal leadership skills by providing practice in helping subordinates handle conflict, whether it is with shipmates or problems in their personal lives.

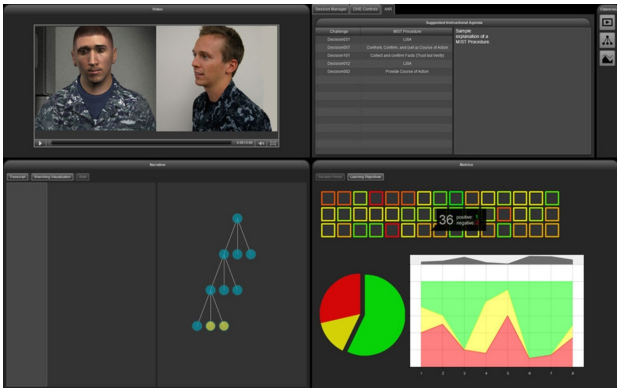

Figure 4 – INOTS

The INOTS system evaluates the student's understanding of scenario learning objectives by monitoring the dialog choices they make within the scenario. The monitoring is currently supported by a detailed hand-authoring process. A knowledge engineer annotates each action in the dialog tree with assessments about learning objective performance that serves as both an expert model and the assessment portion of the pedagogical model for the INOTS ITS. Our work is focused on authoring tools that replace the tedious hand authoring process with user interfaces that are more streamlined and focused. Rather than an implicit expert model, we are using a CBEM model as the explicit representation, and are developing the Emma authoring UI for creating CBEMs using the visual diagramming language.

Observational Skills

A third application of CBEMs and the Monitor is within a training application in which US Marines observe a village from a Virtual Observation Platform (VOP) (Wray and Woods 2013). Marines learn to construct a general baseline of understanding from sustained attention to activities in a village. Needed skills range from low-level signals (recognizing the proxemics and kinesics of individual villagers), to developing an abstract mental representation of the "patterns of life" (Schatz et al. 2012). Figure 5 shows the Monitor evaluating logic-blocks and detecting a commission error when the student does not properly report on the observed village activity.

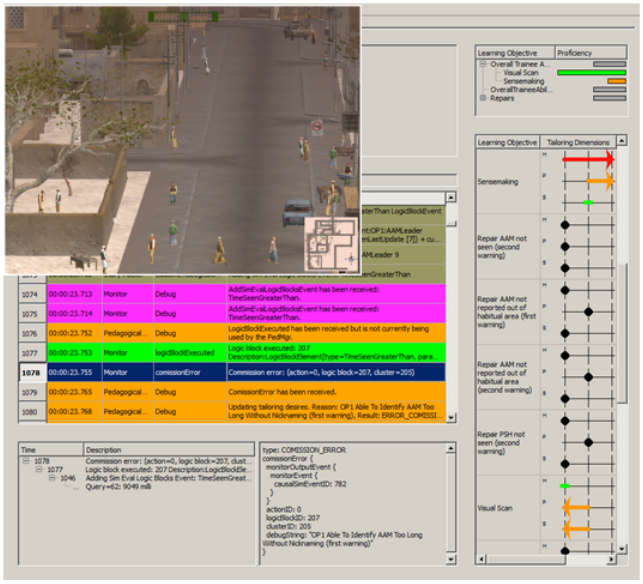

Figure 5 – Monitor Activity in VOP

SUMMARY

Inspired by Mitrovic and Ohlsson's unordered, constraint-based modeling approach (1999), we have constructed a hybrid system capable of representing both fully unordered constraint specifications as well as limited representation of ordering constraints. By using clusters to group unordered constraints, and dependencies to give temporal ordering to some parts of the constraint model, we have developed a wide range of student behaviors that can be represented.

Our approach to expert modeling consists of three primary components:

- A **machine-readable** constraint-based expert model (or CBEM) specification
- An authorable **diagramming language** used to build CBEMs
- A **real-time interpreter**, or *Monitor*, capable of detecting student errors based on the CBEM

The constraint-based approach has been successfully applied to address the challenges of modeling student behavior in several ill-defined domains, including mediation skills, tactical course-of-action planning, and perceptual skills. In these and other training domains, CBEMs are capable of characterizing the bounds on correct behavior,

such that a domain-general monitoring system can examine and classify student actions. By combining this flexible representation with a diagramming paradigm that can be used by content developers and subject matter experts, we have created a methodology and set of software tools with which intelligent tutoring strategies can be applied to ill-defined learning and training domains.

Future work: *Expanded temporal constraint management* such as relative timeframes is desired. For example, instead of specifying that a student perform an action within a fixed time period such as "within 30 seconds", it would be preferable to more loosely specify a relative timeframe such as "soon enough", where soon enough can be relative to both the current CBEM context and the student's skill profile.

Another direction for future work is *fuzzy distinctions*. *REQ* and *AVD* cluster types are appropriate for training domains that have clear right and wrong distinctions. But in some domains, the student could perform a range of actions some that are better or worse than others, but none of which are strictly right or wrong. For example, in a social situation the student may be required to give a greeting, but whether a "terse", "friendly" or "wary" greeting is "best" is highly dependent upon context. A CBEM construct that can indicate a fuzzy relative ranking of similar actions such as greetings is desired for such domains.

Acknowledgments

The work described in this paper was supported in part using funding from the Office of Naval Research (ONR) under contract #N00014-13-C-0156.

References

Campbell, J. C., Hays, M. J., Core, M., Birch, M., Bosack, M., & Clark, R. E. (2011, January). Interpersonal and leadership skills: using virtual humans to teach new officers. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)* (Vol. 2011, No. 1). National Training Systems Association.

Derry, S. J. & Lajoie, S. P. (1993). A middle camp for (un)intelligent instructional computing: An introduction. In *S. P. Lajoie & S. J. Derry (Eds.), Computers as cognitive tools (pp. 1-11)*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Laird, J.E. (2012). *The Soar Cognitive Architecture*, 2012, MIT Press.

Mitrovic, A., & Ohlsson, S. (1999). Evaluation of a constraint-based tutor for a database language. *International Journal of Artificial Intelligence in Education, 10(3-4)*, 238-250.

Ogan, A, Wylie, R., & Walker, E. (2006). The challenges in adapting traditional techniquest for modeling student behavior in ill-defined domains. *Workshop Proceedings on Ill-Defined Domains at the 8th International Conference on Intelligent Tutoring Systems*, Jhongli, Taiwan, June 26-30, 2006.

Ormerod, T.C. (2006). Planning and ill-defined problems. Chapter in R. Morris and G. Ward (Eds.): *The Cognitive Psychology of Planning*. London: Psychology Press.

Walker, Heather A.P., & Wray R. E. (2014). Effectiveness of Embedded Game-Based Instruction: A Guided Experiential Approach to Technology Based Training. *Proceedings of the 2014 Interservice/Industry Training, Simulation and Education Conference (I/ITSEC)*. Orlando, FL, December 1-4, 2014.

Schatz, S., Wray, R., Folsom-Kovarik, J. T., & Nicholson, D. (2012). Adaptive Perceptual Training in a Virtual Environment. Paper presented at the *Human Factors and Ergonomic Systems Conference (HFES-2012)*.

Werner, K. and Rittel, H. (1970). Issues as Elements of Information Systems, *Working paper No. 131, Studiengruppe für Systemforschung*. Heidelberg, Germany, July 1970

Wray, R. E., Folsom-Kovarik, J. T., & Woods, A. (2013). Instrumenting a Perceptual Training Environment to Support Dynamic Tailoring. In D. Schmorrow & C. Fidopiastis (Eds.), *Proceedings of 2013 Augmented Cognition Conference*. Las Vegas: Springer-Verlag (Lecture Notes in Computer Science).

Wray, R., Lane, H.C., Stensrud, B., Core, M., Hamel, L. and Forbell, E. (2009). Pedagogical Experience Manipulation for Cultural Learning,. *Proc. of the 2nd Workshop on Culturally Aware ITS*, Brighton, UK.

Wray, R. E., Woods, A., & Priest, H. (2012). Applying Gaming Principles to Support Evidence-based Instructional Design. *Proceedings of the 2012 Interservice/Industry Training, Simulation and Education Conference (I/ITSEC)*. Orlando, FL, December 2012.

Wray, R. E., & Woods, A. (2013). A Cognitive Systems Approach to Tailoring Learner Practice. In *Proceedings of the Second Annual Conference on Advances in Cognitive Systems ACS* (Vol. 21, p. 38).

,