Analogy, Intelligent IR, and Knowledge Integration for Intelligence Analysis: Situation Tracking and the Whodunit problem

Kenneth D. Forbus, Larry Birnbaum, Earl Wagner, James Baker, Michael Witbrock[1]
Computer Science Department, Northwestern University
1890 Maple Avenue, Evanston, IL, 60201, USA
forbus@northwestern.edu

Keywords: Knowledge Discovery and Dissemination, Predictive Analysis and Hypothesis Management, Terrorism

Abstract

This paper outlines how we are exploring ways to combine analogical reasoning, deductive knowledge integration, and intelligent information retrieval to create new power tools for intelligence analysis. We summarize efforts on two problems: (1) *situation tracking*—where the goal is to maintain a conceptual understanding of an ongoing situation over time, extending it with new information (often gleaned proactively)—and (2) the *whodunit* problem—identifying a small set of likely perpetrators for an event.

1. Introduction

Intelligence analysts must sift through massive amounts of data, using perspective gained from history and experience to pull together from disparate sources the best coherent picture of what is happening. Information Technology research has the potential to create new software tools that could aid analysts in their use of precedents and analogies, in scenario generation, and in searching and assessing immense corpora of data.

Current technology is capable of providing some of this functionality, but in a limited and piecemeal manner. Knowledge-based systems offer fine-grained and logically coherent inferences and hypotheses—deduction and induction—but only when a sufficiently large fraction of all relevant information is represented precisely (e.g., in formal logic). Analogical reasoning systems offer the prospect of "thinking outside the box"—but again depend upon structured representations. IR (Information Retrieval) systems can handle the quantity and diversity of unstructured information that exists in the world, but cannot generate new inferences or hypotheses, due to their lack of structured representations.

Our project is aimed at integrating and extending these three technologies to create power tools for intelligence analysts. Our goal is to discover interesting and powerful functional integrations that permit these technologies to exploit each others' strengths in order to mitigate their weaknesses. From the perspective of knowledge-based AI technology, the goal of the project is to extend the reach of such systems into the world of unstructured data and text. From the perspective of IR technology, it is to leverage the application of inferential and analogical techniques to structured representations in order to achieve significant new functionality.

[1] Cycorp, Inc., 3721 Executive Center Drive Austin, Texas 78731, USA

2. Review of component technologies

Analogy and Similarity

Analysts use analogy constantly in their work, comparing and contrasting precedents to generate explanations and make predictions (Heuer, 1999; Neustad & May, 1988). We believe that human-like analogical processing systems can provide important new capabilities for analysts, helping them overcome working memory limitations and confirmation biases. The theory of analogical processing underlying our approach is Gentner's (1983) *structure-mapping theory*, which describes how the comparison process underlying human analogy and similarity works. There is a large body of psychological evidence supporting the theory, which is important for two reasons. First, people are the most robust reasoners we know of, so emulating them is a wise strategy. Second, we want our software to share our sense of similarity, if we are going to trust its results. In this project we are using three cognitive simulations of structure-mapping processes. The Structure-Mapping Engine (SME) provides analogical matching (Falkenhainer *et al.,* 1989; Forbus *et al.,* 1994). Given two structured representations, SME produces *mappings* consisting of a set of correspondences (i.e., what goes with what), a set of *candidate inferences* (i.e., what might be inferred on the basis of the mapping), and a *structural evaluation score* indicating overall match quality. MAC/FAC (Forbus *et al.,* 1994b) provides retrieval, by modeling similarity-based reminding. The first stage of MAC/FAC uses a special kind of feature vector, automatically constructed from structured representations, as a cheap means for extracting two or three best candidates from a large pool of memories. The second stage uses SME to compare the structured representations retrieved against the current situation. The third simulation, SEQL (Kuehne *et* *al.,* 2000), models generalization. Given a stream of examples, it uses SME to construct and extend generalizations by maintaining the overlap when two examples (or an example and a generalization) are very close matches. Recently Daniel Halstead at Northwestern has been extending SEQL to include probability information. All three of these simulations have been used to model a variety of psychological results, and have been used to generate predictions that have been subsequently been confirmed by psychological experiments, and SME and MAC/FAC have been used in performance systems previously (Forbus, 2001).

Large knowledge bases and deductive reasoning

The Cyc Knowledge Server is a very large, multi-context knowledge base and inference engine. At its heart is an ontology of over 200,000 general terms and a knowledge base (KB) of over two million facts and rules about those terms expressed in arbitrary-order predicate calculus plus modal operators. This KB represents both broad human common sense and application-specific domain knowledge—including in particular the domains of intelligence analysis and terrorism—in a semantically rich and precise manner. Cyc has a general-purpose inference engine plus a library of over 600 special-purpose inference modules each optimized to make some common class of inferences (e.g., transitive relation reasoning) very efficient. Cyc is OWL compliant, has a formal API, and can interface to structured external sources such as databases and (via clarification English dialogue) to human experts. Cyc's natural language system includes an English lexicon containing syntactic and semantic information for over 20,000 words and phrases, as well as parsing and generation capabilities.

Intelligent Information Systems

The third component of this project draws from our previous efforts to develop *frictionless*, *proactive* technologies that bridge the gap between users and the information systems that serve them, helping them to carry out information search and access tasks—or, in the best case, automating these tasks entirely. To accomplish this, a characterization of the user's current task context is used to determine their information needs and to gather the appropriate information. Our most notable success to date is Watson (Budzik *et al.,* 2001), a system that analyzes the document that you are currently reading or writing in a variety of applications (e.g., word processors, email clients, web browsers, etc.) and automatically retrieves information relevant to that document from a variety of sources (the public web, corporate intranet, or proprietary data services). Watson has proven effective in a number of studies (Budzik, 2003). We have also used Watson as a test-bed for moving beyond similarity as a measure of relevance, towards more "transformational" models in which specific kinds of relevance can be specified and utilized (e.g., Budzik *et al.,* 2000), a key goal of the current project.

3. Situation-tracking

Most analytic jobs require keeping up with what is happening. This means tracking what is happening in some area or topic of concern, and updating one's conceptual model as things change. Modern information retrieval and QA systems are aimed at finding relevant information, but little to no attention has been paid to automatically (or even semi-automatically) constructing and maintaining a human-like conceptual model of a situation in software—i.e., not just unstructured sets of documents on the same topic, but structured scenarios unfolding over time. Yet having such software would be a boon to analysts. Freed of human short-term memory limitations and the distractions of "having a life," software systems could help analysts work through complex arguments more quickly and accurately, and help find precedents that otherwise might be missed. The software could proactively seek out relevant information, based on a shared understanding with the analyst about what is important.
The situation tracking task in this sense also encompasses a meaningful cross-section of the technical challenges we aim to address. What are the right levels of representation to use in the conceptual models? How should incoming data feeds be processed to provide useful information both for the analyst and for updating these formal models? Can reasoning based on these models, including properties and motivations of sources, help estimate plausibility and determine how new information should be assimilated? Can relational patterns extracted from situation models via analogical processing be used to generate targeted information retrieval queries to look for "the other shoe dropping"?

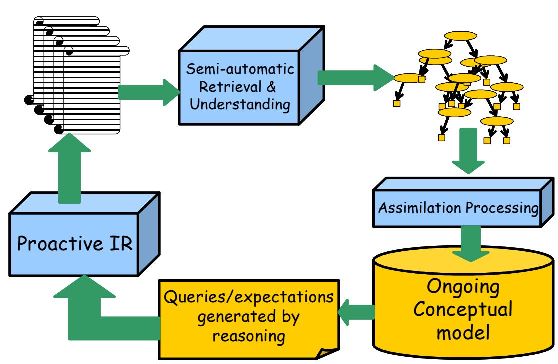

Figure 1: Situation Tracking Testbed architecture

Our first cut at the overall architecture of the situation tracker is shown in Figure 1. The data feeds are text stripped from web sources, found either by users or as a result of proactive searches by the system itself. The system has some capacity to analyze and represent the content of these texts on its own. We are also developing an interface for interactively translating texts into formal representations, built on Cycorp's natural-language based knowledge capture tools, which rely heavily on clarification dialogues and follow-up questions to facilitate building representations. The results of these interactions will be represented as transformations from the source text to formal representations, and accumulated as part of the system's knowledge.

We see these transformations being used in two ways. First, in processing subsequent inputs, these transformations will be applied via analogical processing to suggest how that new document should be understood, in a manner analogous to example-based machine translation (Brown 1996; Somers 2001). Our hypothesis is that this will enable the rate of knowledge entry to speed up over time, with experience in a situation. Second, these transformations will provide an input to our intelligent information retrieval system, which will essentially "invert" them to help generate better queries for finding texts that are closely related to something that is formally represented, in particular predictions based on the situation representation. Our hypothesis is that this will enable us to automatically retrieve documents that provide relevant evidence regarding predictions made by the system.

As new information is discovered, analyzed, and represented, it must be assimilated into the system's ongoing conceptual understanding of the situation. This involves a combination of first-principles reasoning (for constraint and consistency checking) and analogical processing (to see if it is very unusual in some way compared to previous inputs or to expectations previously generated by the model). Part of the model will be a running set of predictions. These predictions will be checked against incoming information, and indeed will be used to proactively seek relevant information.

As part of the situation-tracking test-bed we have constructed a prototype semi-automatic retrieval and understanding system based on the use of scripts (Schank & Abelson, 1977) as situation models. Scripts are explicit, structured representations of stereotypical situations as they unfold over time, e.g., kidnappings. They provide the full inferential power of logical representations, including variables and variable bindings, reasoning over time, and distinguishing among alternative pathways and outcomes. They also serve as a backbone or "glue" integrating and mediating between the IR-based natural language technology on the one hand, and knowledge-based mechanisms, including Cyc and analogical reasoning, on the other. Thus the full power of these mechanisms can be brought to bear on information retrieval, while, at the same time, the IR mechanisms can be used to find information relevant to ongoing instances of represented situations.

In the current prototype, a user selects and partially specifies a situation to track, e.g., the kidnapping of Nicholas Berg, by selecting the script (kidnapping) and then specifying one or more of its roles (in this case the victim). (Automatic retrieval of appropriate scripts to seed the process is a problem we will tackle later.) The assignment of Berg as the victim is propagated to constituent scenes of the kidnapping script, which consist of logical sentences describing the actions that make up a kidnapping, e.g., that the victim is seized by the kidnappers, then taken to a location and held prisoner there, etc. Every script and its scenes have specialized retrieval and recognition rules, parameterized by the appropriate role variables. These identify and pull out stories related to the situation being tracked, and organize them under the appropriate scene in the script, i.e., determine whether they relate primarily to the initial abduction, to holding the victim prisoner, to announcing the act or demanding ransom, to the release, ransom, or death of the victim, and finally to the escape or capture of the kidnappers.

Each script and all of the scenes in it have associated extraction rules that try to fill other roles in the script, e.g., location, perpetrators, times, reasons, method of execution, etc. There is some overlap between the scene identification rules and the extraction rules. For example, the rule for identifying a hostage release scene consists of a set of patterns for recognizing descriptions of that event. If the hostage is already known, then those patterns will be parameterized appropriately by the script, leading to highly specific patterns including, e.g., the name or names of the victims. On the other hand, if the victim isn't known, then one of these patterns, upon matching, might provide this information to the script by binding that role variable. As each role is filled, this further parameterizes other recognition/retrieval and extraction rules, increasing their specificity and effectiveness. The output is a fully fleshed-out script, with roles extracted, and stories categorized by scene, presented to the user as shown in Figure 2.

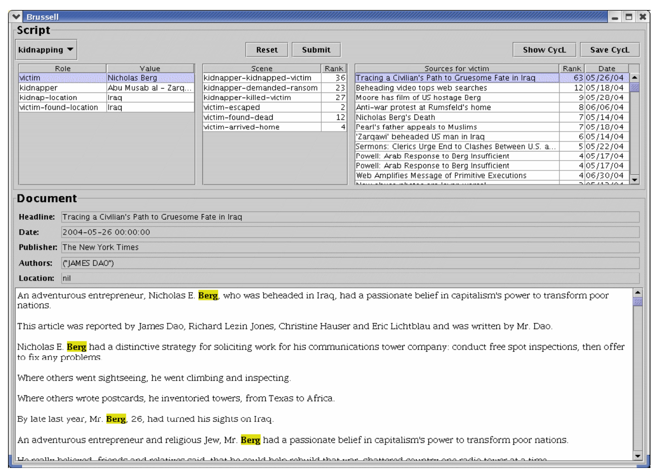

Figure 2: Script-based IR GUI

From the perspective of IR, the use of a script in situation tracking, and the propagation of constraints and information among the script's scenes, enables chaining, i.e., the identification and retrieval of stories that are only inferentially related to what was originally specified by the user. For example, if the user specifies a kidnapping situation to track by specifying the kidnapper (e.g., Zarqawi), once a story that mentions a particular victim (e.g., Berg) is found and analyzed, that role will also be bound in an instance of the script. This in turn means that new queries can be launched looking for, e.g., stories describing the scene in which that victim (Berg) was first seized, and these can be retrieved and properly placed in the evolving situation model even if the kidnapper was not mentioned in those stories, for example because his identity was not yet public knowledge. Moreover in this way additional information, e.g., the location of the victim's seizure, might be extracted. In other words, the inferential capacity of the script (or any other explicit representation of the situation) can be used to automatically parameterize queries that may find related stories even when those stories could not be retrieved given the information originally provided.

Our next step will be to integrate situation representations constructed by the tracker to analogical mechanisms for assimilation and reasoning support. We have already implemented a parameterized question interface for asking compare/contrast questions and for retrieving similar cases, similar to that reported in (Forbus *et al.,* 2002).

4. The Whodunit Problem

An important task for analysts is coming up with plausible hypotheses about who performed an event. Recall the pre-election bombing in Madrid. While the Spanish government originally claimed that the Basque Separatist group ETA was the most likely suspect, evidence quickly mounted that Al Qaeda was very likely responsible. Multiple, highly coordinated attacks, for example, are more similar to Al Qaeda's *modus operandi* than previous ETA actions. This is an example of what we call *the whodunit problem.*

Stated more formally, given some event E whose perpetrator is unknown, the whodunit problem is to construct a small set of hypotheses {Hp} about the identity of the perpetrator of E. These hypotheses should include explanations as to why these are the likely ones, and be able to explain on demand why others are less likely.

This is of course an extremely difficult problem, but one which concisely expresses a key task that intelligence analysts perform. We define a more restricted class of whodunit problems to begin with:

- *Formal inputs*. We assume that the input information is encoded in the form of structured descriptions, including relational information, expressed in a formal knowledge representation system.
- *Accurate inputs*. We assume that the input information is completely accurate.
- *One-shot operation.* Once the outputs are produced for a given E, the system can be queried for explanations, but it does not automatically update its hypotheses incrementally given new information about E.
- *Passive operation*. The hypotheses are not processed to generate differential diagnosis information, i.e., "tells" that could be sought in order to discriminate between the small set of likely hypotheses.

The assumption of formal inputs is reasonable, given that one of the problems we are tackling in the Situation Tracking Testbed is producing such representations from news sources. The assumptions of accurate inputs and of one-shot, passive operation simplify the problem, but in a way that we believe preserves many essential elements.

The whodunit problem is an excellent candidate for analogical reasoning. We have defined two basic whodunit algorithms. Both assume a large case library of relational descriptions of events, tagged with relationships indicating the perpetrator of each. The first algorithm is purely exemplar-based:

Method 1: Closest Exemplar

1. Use MAC/FAC to retrieve events similar to E.
2. For each similar event, remove it if it doesn't include a candidate inference about the perpetrator.
3. Iterate until enough hypotheses are generated.
4. (Optional) Generate explanations and expectations by analyzing the similarities and differences between each Hp and E.

Intuitively, this method corresponds to taking what one is reminded of when hearing about E as the most likely suspects. People can be surprisingly biased about such decisions, e.g., the Spanish government stuck with its ETA hypothesis long enough to lose credibility. People also have their own lives, with many other kinds of things in their memories. A cognitive simulation need not have either of those limitations.

While examples are important, one powerful aspect of human cognition is our ability to make generalizations. Generalizations are important because they strip away what is accidental, and thus highlight what is essential about a class of similar examples. The other whodunit algorithm uses our SEQL model to automatically produce generalizations.

Method 2: Closest Generalization

Preprocessing:

1. Partition case library according to perpetrator.
2. Use SEQL to construct generalizations for each perpetrator.

Generating hypotheses:

1. Given an incident E, pick the *n* closest generalizations, as determined by SME's structural evaluation score.

We have recently been experimenting with an extended version of SEQL, which generates probabilities when merging a new example into a generalization. That is, instead of completely eliminating facts which do not appear in the overlap, they are simply attenuated, based on frequency information computed from multiple matches. This version of SEQL can thus provide probabilities for different aspects of the match, potentially indicating what properties were most important (e.g., coordinated attacks, in the Madrid bombing example).

To test these algorithms, we compared them on a set of incidents. The case library we used was Cycorp's Terrorist Knowledge Base, a collection of 3,379 terrorist incidents hand-entered by domain experts. These incidents are expressed using the vocabulary of the Cyc KB, and range from 6 to 158 propositions, average = 20.

We selected 98 perpetrators (out of 450, based on having at least three attacks in the KB) to use in the experiment. One case was pulled from each of these sets at random and the perpetrator information was removed, to provide the test inputs. We used inference involving the KB to automatically flesh out the examples slightly. For instance, suppose there is a group which operates in both Pavia and Florence. The location information about an incident describes some specific location (e.g., what city), but does not include any background information about that location or other entities that play roles in the incident. Given that these cities are not identical, SEQL would replace them with an arbitrary new entity. But it would not know that this new city is in Italy, even though that is the most reasonable assumption, given that both of the cities in the examples were. Our solution to this problem is to add extra information to the cases, as part of the preprocessing phase, that is likely to provide those relevant constraints. Currently this consists of including all of the attribute information for every entity in a case. For example, if one case occurred in `CityOfRomeItaly` and another occurred in `(CityNamedFn "Pisa" Italy)`, they would still match because both would be known to be instances of the concept `(CityInCountryFn Italy)`.

We used three criteria for bounding the size of the set of hypotheses Hp. The most restrictive is producing only a single perpetrator, i.e., guessing directly who did it. The least restrictive is a "top 10" list, rank ordered by estimated likelihood. (In the case of probabilistic SEQL, we used probabilities; for the other algorithms we used the SME structural evaluation score.) In some ways a top-10 list is too large: it would certainly not be optimal as a final output. But it may be very useful to use this broader cutoff in combination with tackling broader versions of the problem, i.e., proactive information gathering instead of passive operation. The middle ground is the "top 3" list, which has the virtue of providing both the best and some (hopefully mind-jogging) alternatives.

Figure 4 illustrates the results. Closest Exemplar does surprisingly well, identifying the correct perpetrator 29% of the time, and including it in its top three 31% of the time. However, continuing to construct hypotheses from MAC/FAC beyond that point proved useless: no additional correct identifications were included. On the other hand, using Closest Generalization does not do as well as MAC/FAC in zeroing in on a single best hypothesis, getting it only 18% of the time for standard SEQL, and 23% of the time for the probabilistic version of SEQL. Both versions do slightly better than MAC/FAC on the top 3 list (34% and 37%, probably not statistically significant). Where generalization seems to really be adding value is in the top 10 list, where both versions of SEQL include the correct perpetrator 53% of the time.

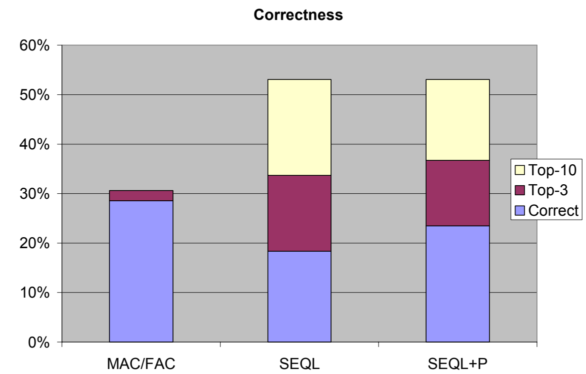

Figure 3: Results for the whodunit experiment

We find these results encouraging for several reasons. First, the amount of automatic elaboration we have done is quite small, and it is very likely that this process can be optimized by treating it as a machine learning problem, tuning the encoding process based on what improves discrimination. Second, an analysis of the detailed results suggests that both algorithms perform better as the number of examples available rises. All of them do better when there are 15 or more examples available per perpetrator. This suggests that as more knowledge is available these algorithms will perform better rather than worse. Finally, while we are surprised that the probabilistic version of SEQL was indistinguishable in terms of its results from traditional SEQL, the explanations produced by probabilistic SEQL may be still be more useful, since they include what in effect is frequency information for the various aspects of the situation. Further experiments are underway as the TKB contents expand.

4. Related Work

Other simulations of analogical mapping have been developed, but all have limitations that make them less suitable for this purpose. For instance, some are domain-specific (cf. Mitchell 1993), which would sacrifice breadth. Others are based on connectionist architectures (cf. Eliasmith & Thagard, 2001; Hummel & Holyoak, 1997), and are known to not scale up to even medium-scale examples, e.g., the descriptions used in the whodunit experiment.

In many ways the script-based component of the system resembles script-based text “skimmers” of 25 years ago, in particular Frump (DeJong, 1977) and IPP (Lebowitz, 1980). The biggest change has been the tremendous development of information retrieval, and to a lesser extent, of textual information extraction (e.g., Baeza-Yates and Ribeiro-Neto, 1999). These improvements provide a robust technology substrate that wasn’t available two and a half decades ago. On the other hand, the successes of these systems came at a price of decreased semantic sophistication, and, taken together, their strengths and weaknesses provide us a far clearer picture of the role and value of higher-level interpretive mechanisms.

5. Discussion

We believe that by combining analogical processing, large knowledge bases, and intelligent information retrieval, we can develop new power tools for intelligence analysts. Situation-tracking, where a shared conceptual model is created semi-automatically, is we think a promising example, as the IR-driven script understander, using scripts from Cycorp's KB, illustrates. Part of the value of having a shared, formally represented conceptual model is that software can then take on more of the analytic burden. Our formulation of the whodunit problem provides an example of one such service. We are encouraged by how well fairly simple analogical processing algorithms do on this (admittedly restricted) version of the problem.

Of course, more work lies ahead than behind. Our current major goal is to bring the full situation-tracking testbed to the point where it is robust enough for daily use, and put it to work on a daily basis by members of the project. This will help generate the experience and experimental data about the framework and each of the services that will be necessary to evolve them, as well as discovering what new problems lie ahead.

Acknowledgments

This work was supported by the National Science Foundation under the KDD program. Daniel Halstead conducted the whodunit experiments.

References

Baeza-Yates, R. and Ribeiro-Neto, B. 1999 *Modern Information Retrieval*. Addison Wesley.

Brown, R. 1996. "Example-Based Machine Translation in the Pangloss System". In Proceedings of the 16th International Conference on Computational Linguistics (COLING-96), p. 169-174. Copenhagen, Denmark.

Budzik, J. Information Access in Context: Experiences with the Watson System. PhD thesis, Northwestern University, June 2003.

Budzik, J., Hammond K., and Birnbaum, L., Information access in context. Knowledge based systems 14 (1-2), (2001), Elsevier Science 37-53.

Budzik, J., Hammond, K. J., Birnbaum, L., and Krema, M. "Beyond Similarity," in *Working notes of the AAAI 2000 Workshop on AI for Web Search* (Austin, TX USA), July 2000, AAAI.

DeJong, G. 1977. Skimming newspaper stories by computer. *Proceedings IJCAI-77.*

Eliasmith, C. and Thagard, P. 2001. Integrating structure and meaning: A distributed model of connectionist mapping. Cognitive Science.

Falkenhainer, B., Forbus, K. and Gentner, D. The Structure-Mapping Engine. *Proceedings of the Fifth National Conference on Artificial Intelligence.* 1986.

Forbus, K. Exploring analogy in the large. In Gentner, D., Holyoak, K., and Kokinov, B. (Eds.) *Analogy: Perspectives from Cognitive Science.* MIT Press. 2001.

Forbus, K., Ferguson, R. and Gentner, D. Incremental structure-mapping. *Proceedings of the Cognitive Science Society*, August. 1994.

Forbus, K., Gentner, D., and Law, K. MAC/FAC: A model of similarity-based retrieval. *Cognitive Science*, 19, 141-205. 1994.

Forbus, K., Mostek, T. and Ferguson, R. 2002. An analogy ontology for integrating analogical processing and first-principles reasoning. *Proceedings of IAAI02.*

Gentner, D., Structure-mapping: A theoretical framework for analogy, *Cognitive Science* 7(2), 1983

Heuer, R. J. 1999. Psychology of Intelligence Analysis. Center for the Study of Intelligence. Government Printing Office, US Government.

Hummel, J.E., & Holyoak, K.J. (1997). Distributed representations of structure: A theory of analogical access and mapping. Psychological Review, 104, 427-466.

Kuehne, S., Forbus, K., Gentner, D. and Quinn, B. SEQL: Category learning as progressive abstraction using structure mapping. *Proceedings of CogSci.* August, 2000.

Lebowitz, M. 1980. "Language and Memory: Generalization as Part of Understanding." In Proceedings of AAAI-80. pp. 324-326.

Mitchell, M. (1993) Analogy-making as perception: A computer model. MIT Press, Cambridge, MA.

Neustad, R. and May, E. 1988. *Thinking in time: The uses of History for Decision Makers.* Free Press.

Schank, R. and Abelson, R. 1977. Scripts, plans, goals, and understanding: An inquiry into human knowledge structures. Hillsdale, NJ: Lawrence Erlbaum.

Somers, H. 2001. EBMT seen as case-based reasoning. In Proceedings of the Workshop on Example-Based Machine Translation, Santiago de Compostella, Spain, September 18-22, 2001.