NORTHWESTERN
UNIVERSITY

Electrical Engineering and Computer Science Department

Technical Report
NWU-EECS-09-09
March 25, 2009

Using Analogy to Overcome Brittleness in AI Systems

Matthew Evans Klenk

Abstract

One of the most important aspects of human reasoning is our ability to robustly adapt to new situations, tasks, and domains. Current AI systems exhibit brittleness when faced with new situations and domains. This work explores how structure mapping models of analogical processing allow for the robust reuse of domain knowledge. This work focuses on two analogical methods to reuse existing knowledge in novel situations and domains. The first method, *analogical model formulation*, applies analogy to the task of *model formulation*. Model formulation is the process of moving from a scenario or system description to a formal vocabulary of abstractions and causal models that can be used effectively for problem-solving. Analogical model formulation uses prior examples to determine which abstractions, assumptions, quantities, equations, and causal models are applicable in new situations within the same domain. By employing examples, the range of an analogical model formulation system is extendable by adding additional example-specific models. The robustness of this method for reasoning and learning is evaluated in a series of experiments in two domains, everyday physical reasoning with sketches and textbook physics problem-solving. The second method, *domain transfer via analogy*, is a task-level model of cross-domain analogical learning. DTA helps overcome brittleness by allowing abstract domain knowledge, in this case equation schemas and control knowledge, to be transferred to new domains. DTA learns a *domain mapping* between the entities and relations of the new domain and the understood domain, through comparisons between structures of explanations. Then, using this mapping, a new domain theory can be inferred and extended through an analogy between the domain theories themselves. This model is evaluated across a variety of physics domains (e.g., mechanics, electricity and heat flow). Successful cross-domain analogies result in *persistent mappings*, which support incremental learning of the target domain theory through multiple cross-domain analogies.

Keywords: Problem-Solving, Within-Domain Analogy, Cross-Domain Analogy, Analogical Learning

NORTHWESTERN UNIVERSITY

Using Analogy to Overcome Brittleness in AI Systems

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Computer Science

By

Matthew Evans Klenk

EVANSTON, ILLINOIS

June 2009

ABSTRACT

Using Analogy to Overcome Brittleness in AI Systems

Matthew Evans Klenk

One of the most important aspects of human reasoning is our ability to robustly adapt to new situations, tasks, and domains. Current AI systems exhibit brittleness when faced with new situations and domains. This work explores how structure mapping models of analogical processing allow for the robust reuse of domain knowledge. This work focuses on two analogical methods to reuse existing knowledge in novel situations and domains.

The first method, *analogical model formulation*, applies analogy to the task of *model formulation*. Model formulation is the process of moving from a scenario or system description to a formal vocabulary of abstractions and causal models that can be used effectively for problem-solving. Analogical model formulation uses prior examples to determine which abstractions, assumptions, quantities, equations, and causal models are applicable in new situations within the same domain. By employing examples, the range of an analogical model formulation system is extendable by adding additional example-specific models. The robustness of this method for reasoning and learning is evaluated in a series of experiments in two domains, everyday physical reasoning with sketches and textbook physics problem-solving.

The second method, *domain transfer via analogy*, is a task-level model of cross-domain analogical learning. DTA helps overcome brittleness by allowing abstract domain knowledge, in this case equation schemas and control knowledge, to be transferred to new domains. DTA

learns a *domain mapping* between the entities and relations of the new domain and the understood domain, through comparisons between structures of explanations. Then, using this mapping, a new domain theory can be inferred and extended through an analogy between the domain theories themselves. This model is evaluated across a variety of physics domains (e.g., mechanics, electricity and heat flow). Successful cross-domain analogies result in *persistent mappings*, which support incremental learning of the target domain theory through multiple cross-domain analogies.

ACKNOWLEDGEMENTS

I would like to thank Ken Forbus for advising this research. Throughout the countless ebbs and flows, Ken was always a steadying force, keeping me focused on the central research problems. Ken provides a wonderful example of what it takes to be a successful scientist balancing the big picture goals with day-to-day dedication. I am grateful for my thesis committee. Tom Hinrichs provided a wonderful mix of pragmatism and vision about the purpose and value of research. Meetings with Chris Reisbeck often left me exhausted but inspired as well. John Laird was instrumental in helping me present this research to greater AI community.

The past and present members of the Qualitative Reasoning Group provided invaluable intellectual and moral support over this 6 year process. Praveen Paritosh provided for energetic brainstorming sessions on research questions of cognition and artificial intelligence. This thesis would not have been written without the day-to-day support and encouragement of Kate Lockwood, Emmett Tomai, and Morteza Dehghani. Getting through graduate school requires a community, and I was very fortunate to have such colleagues with which to share the highs and the lows.

I also need to thank the people who unknowingly supported this effort. I thank my friends, who stood with me despite my unpredictable schedule and incoherent rants as I was finding my way through graduate school. They provided an escape from academia by reminding me that the human experience is constructed from interpersonal connections. I also thank the independent cafes and friendly baristas in Atlanta, Chicago, and Washington DC from which much of this research was conducted.

Most importantly I would like to thank my family. I thank my parents, Jim and Carol Klenk, for encouraging my intellectual pursuits from a young age and providing support and encouragement throughout graduate school. I thank my brother, Paul, for constantly setting a fantastic example of hard work and pursuit of excellence. Finally, I want to thank my fiancée, Katrina Connolly, for encouraging me to go to graduate school, staying with me throughout the process, and sharing with me the excitement about the world around us.

Table of Contents

List of Figures

List of Tables

1 Introduction

One of the most important aspects of human reasoning is our ability to robustly adapt to new situations, tasks, and domains. Current AI systems fall far short of exhibiting this flexibility. While modern cars have sophisticated diagnostic systems, one would not expect the system designed for a Ford Focus to work in Chevy Malibu. Consequently, AI system developers frequently design whole new systems from scratch for each problem and task. On the other hand, people learn new tasks and domains all the time. In fact, they do this with limited external feedback and without anyone reaching inside their heads and reprogramming them. When presented with faulty Chevy, an experienced Ford mechanic would be able to apply his or her knowledge to diagnose the problem. This flexibility is the subject of this dissertation.

1.1 Problems of Brittleness

Most large-scale AI systems rely on a knowledge base (KB) of rules and facts relevant for the domain. These KBs attempt to capture the knowledge and reasoning of human experts. Therefore, to build an AI system for a domain, the system designers encode domain knowledge from human experts into the KB. Given a query, the AI system uses inference rules and facts from the KB to arrive at an answer. In a perfect world, the AI system can answer the same questions about the domain as human experts.

Unfortunately, this perfect world is rarely realized outside of narrow domains. Frequently, AI systems fail to extrapolate to unseen problems from the domain. A common source of this brittleness is either incorrect or missing knowledge in the KB. This could be because the domain expert failed to provide some piece of reasoning, or the system designer failed to encode all the ways in which a piece of knowledge could be used. This results in the AI system failing to come up with solutions for questions in which humans answer with ease.

The TacAir-Soar system (Jones *et al.* 1999) is a great example of a successful AI systems using a knowledge base. TacAir-Soar was designed to provide realistic simulated aircraft for military training exercises. It generated "human like" behavior for simulated air combat missions, which represented a significant advance over the previous state of the art for automated entity behaviors in simulations. While this was a great success for AI, these agents were unable to use a variety of weapons and sensors because the system designers did not have sufficient time to build intelligent behaviors for them. Even though this is within the same domain, air combat, adapting the system to handle new sensors and weapons requires adding significant knowledge to the agent. Given a new weapons and sensor systems, human pilots likely reuse their knowledge and experiences to determine how best to employ them.

An analysis of two recent AI research projects demonstrate the magnitude of this problem in regards to broad domains and unanticipated problems. In the High Performance Knowledge Base (HPKB) project, teams constructed knowledge bases to answer questions about international responses to crises (Cohen *et al.* 1999). Tracking the development of the SRI/CYC system illustrates how brittleness affects knowledge system development. SRI required 1703 new axioms to answer the sample questions and be ready for the Phase 1 test. On 110 unseen questions about a crisis management scenario, the SRI/CYC system was able to produce at least a partially correct answer only 30% of the time. The system developers were then given 4 days to improve performance. By adding 123 axioms, the system answered at least partially correct 45% of the questions. Next, the domain was expanded. This required the system designers to add another 1485 axioms to the system. But when given a new scenario and a new set of questions, the system was only able to achieve a score of 40% partially correct. This performance was improved to 65% with the addition of 304 axioms. This failure to perform well on unseen problems and the need for large scale KB changes to be made to expand the scope of problems covered within the same domain demonstrate the brittleness of current AI systems.

In the HALO project, three teams attempted to build AI systems to answer AP Chemistry questions from a subset of a chemistry textbook (Barker *et al.* 2004). The three systems were evaluated on unseen problems. None of the teams scored over 70% on the multiple choice section. This represented only half the exam, the other sections were more difficult. On the exam as a whole, none of the systems scored above 50%. Once again, the AI systems failed to operate robustly on unseen questions.

These projects represent the traditional approach to overcoming brittleness. They seek to overcome brittleness by constructing larger KBs with increasingly general knowledge about the world (Newell & Ernst 1965). The general knowledge would allow the system to perform even when the domain specific knowledge is not available. This approach is the motivation of the CYC project, which seeks to build a KB representing human common sense reasoning (Lenat 1995). While these approaches have had many successes, the brittleness evident in current AI systems requires searching for new solutions to this problem.

1.2 Analogy for Robustness

This dissertation explores using computational models of analogy to overcome this brittleness by reusing existing knowledge. While numerous researchers have proposed analogy as a solution to this problem (e.g., Lenat & Feigenbaum 1991; Forbus & Gentner 1997), little progress has been made in terms of realizing this goal. Analogy here includes both within-domain and cross-domain analogies. For example, when reasoning about how a hot cup of tea will cool sitting by your desk, you could use your experiences with other cups of hot liquids. This is an example of a within-domain analogy. Researchers also emphasize the need for cross-domain analogies in which superficially dissimilar but systematically related pieces of knowledge allow inferences in new domains (Gentner & Gentner 1983). When trying to understand why the hot cup of tea is cooling, an analogy between heat flow and fluid flow would suggest that heat is flowing out of the tea and into the air.

Unfortunately, like the idea of more general knowledge, analogy alone is not enough to overcome this problem of brittleness. The AI system needs to know when and how to use analogy. A basic approach is to invoke analogical reasoning whenever deductive reasoning fails. This is inadequate because as our KBs grow in size, queries take hours or even days to fail. Therefore, it is necessary to integrate analogy directly into the reasoning process. In addition, most analogical reasoning methods involve comparisons between two cases to generate inferences about one of them. In these methods, it is important to understand what kind of knowledge is being transferred between the cases. Other analogical reasoning methods use the correspondences resulting from analogical comparison for generalization and learning. Because of these different uses of analogy, more investigations are required to improve our understanding of how analogy can be used to alleviate brittleness.

1.3 Claims and Contributions

This dissertation outlines two methods, *analogical model formulation* and *domain transfer via analogy* (DTA), which use analogy to robustly reuse knowledge from previous situations and domains.

The first method, analogical model formulation, applies analogy to the task of *model formulation*. Model formulation is the process of moving from a scenario or system description to a *scenario model*. The scenario model consists of assumptions, abstractions, and causal models that can be used effectively for problem-solving. Given the everyday physical reasoning problem in Figure 1 asking, "Which crane is more stable?", the problem-solver must make a number of modeling decision. While many things can potentially affect stability of objects, a useful model for this problem connects the stability of each crane to the horizontal distance between the crane and the boat it is carrying. Then, using this model, one can determine that the crane on the right is more stable.

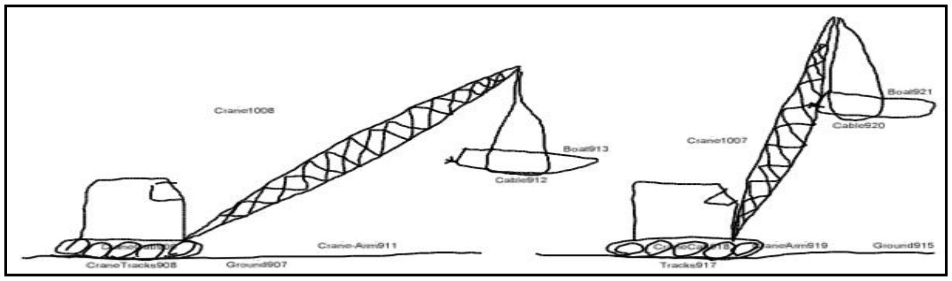


Figure 1: Which crane is more stable?

Analogical model formulation uses prior examples to determine which abstractions, assumptions, and causal models are applicable in new situations. The process begins by retrieving a similar precedent from memory. An analogy is created between the precedent and the current situation. Using analogical inferences concerning the applicable abstractions, assumptions and causal relationships, the problem-solver creates a model from which it derives the solution. The robustness of this method for reasoning and learning is evaluated in a series of experiments from two multiple choice test domains, everyday physical reasoning with sketches and AP Physics problem-solving.

The second method, domain transfer via analogy, is a task-level model of cross-domain analogy. These analogies are frequently employed when one is trying to gain an understanding of a new domain. For example, when students are learning about electricity, they frequently make use of analogies to flowing water and moving crowds (Gentner & Gentner 1983). These cross-domain analogies generate conceptual inferences concerning electricity. Frequently, teachers and textbook authors appeal directly to their students to make these analogies. For example, in an introduction to electricity, Koff (1961) writes “The idea that electricity flows as water does is a good analogy.” Providing AI systems with this capability to reuse their knowledge through cross-domain analogies would greatly improve their flexibility.

Domain Transfer via Analogy (DTA) models this process of cross-domain analogy. DTA helps overcome brittleness by allowing knowledge previously learned in other domains to be applied to new domains. Cognitive science research on cross-domain analogy has emphasized the importance of a *domain mapping* between the vocabularies of the new and understood domains (Gentner 1983). DTA learns a domain mapping through comparisons between structures of explanations in the two domains. Then, using this mapping, a new domain theory can be inferred and extended through an analogy between the domain theories themselves. This model is evaluated across a variety of physics domains (e.g., mechanics, electricity and heat flow).

This work fits into the following high level view of learning and reasoning by intelligent agents. In the initial stages, the agent has limited domain knowledge and reasons directly from experiences and examples. To account for this ability, we introduce the cognitive process of analogical model formulation. With these examples and experiences, the agent constructs generalizations of domain principles as it develops expertise within the domain (Forbus and Gentner 1986). While this dissertation does not provide an account for how this abstraction process occurs, a number of promising methods are discussed in future work sections. The application of this abstracted knowledge to new but similar domains through cross-domain analogies enables the agent to quickly learn new domain theories. The domain transfer via analogy method provides this functionality. Taking this view of domain expertise, these methods occur under very different circumstances dictated by the types of domain knowledge available to the agent.

1.4 Organization

Chapter 2 provides the relevant background on analogical processing. Included in this discussion is an overview of the structure mapping theory of analogy, the cognitive simulations of analogical processes used in this work, and the Companions Cognitive Architecture on which the systems presented here are built.

Chapter 3 describes the foundations of analogical model formulation. After arguing that traditional model formulation has critical limitations for accounting for human reasoning about the real world, I describe how analogical processing can be used to overcome these challenges.

Chapters 4 and 5 evaluate analogical model formulation in two test-taking domains: everyday physical reasoning and AP Physics problem-solving.

Chapter 6 contains related work and a general discussion of analogical model formulation.

Chapter 7 describes in detail domain transfer via analogy. This chapter includes the cognitive science foundations of cross-domain analogy, the domain transfer via analogy algorithm, a pilot study between linear and rotational kinematics, and a more rigorous evaluation between four physics domains: linear mechanics, rotational mechanics, electricity, and thermodynamics.

Chapter 8 contains related work on cross-domain analogy and a discussion of domain transfer via analogy.

Chapter 9 recaps the central claims of this thesis and discusses some important future directions.

2 Background

This dissertation builds upon ideas from cognitive psychology and artificial intelligence. This chapter presents the background concerning the analogical processes used in this work. This chapter begins with a description of the underlying psychological theory. Next, I describe computational models analogy and similarity-based retrieval based upon this theory and continued psychological research. These models have been integrated in the Companions Cognitive Architecture (Forbus *et al.* 2008). The central hypothesis of Companions is that structure-mapping operations are central to human reasoning and learning abilities.

2.1 Analogical Processing

Over the past 30 years, there has been an explosion of interest in analogy. This research has brought together computer scientists, philosophers, linguists, and neuroscientists as well as psychologists from a number of disciplines including cognitive, developmental, and animal psychology (Penn *et al.* 2008). These researchers have documented extensive use of analogy throughout the human experience. These domains include student learning (Gentner and Gentner 1983), scientific discovery (Gentner *et al.* 1997; Nersessian 1992), legal arguments (Holyoak and Simon 1999), and decision-making (Markman and Medin 2002). In cognitive science, structure mapping theory (SMT; Gentner 1983) has become an important framework for understanding analogy. This dissertation uses computational models of analogical processing based upon SMT.

In SMT, analogy is based on a structural alignment between two representations (the *base* and the *target*). These representations are collections of predicate calculus facts made up of entities, attributes, relations between entities, and higher-order relations between propositions. The alignment process constructs the maximal structurally consistent match. A structurally consistent match is one that satisfies the following three constraints: *tiered-identicality*, *parallel connectivity*, and *one-to-one mapping*. The tiered-identicality constraint provides a strong preference for only allowing identical predicates to match, but allows for rare exceptions. For example, *minimal ascension* (Falkenhainer 1988) allows non-identical predicates to match if they are part of a larger mapped structure and share a close common ancestor in the ontology. Parallel connectivity states that if two predicates are matched then their arguments must also match. The one-to-one mapping constraint requires that each element in the base corresponds to at most one element in the target and vice versa. To explain why some analogies are better than others, structure-mapping uses the principle of *systematicity*: Mappings that are highly interconnected and contain deep chains of higher order relations are preferred over mappings with an equal number of relations which are independent from each other. Such nested structures indicate explanations, which provide context to help evaluate analogical inferences.

The Structure-Mapping Engine (SME; Falkenhainer *et al.* 1989) is a cognitive simulation of the analogical matching process. Given two structured representations as input (the base and target), SME produces one or more *mappings*, each representing a construal of what items (*entities* and *expressions*) in the base go with what items in the target. Each mapping is represented by a set of *correspondences*. Mappings also include *candidate inferences* which are conjectures about the target using expressions from the base which, while unmapped in their entirety, have subcomponents that participate in the mapping's correspondences. Based upon the systematicity constraint, a *structural evaluation score* is computed to estimate the match quality. SME operates in polynomial time, using a greedy algorithm (Forbus *et al.*1994; Forbus & Oblinger 1990). At a high level, mapping in SME consists of the following steps:

- Step 1: In parallel, create local correspondences called *match hypotheses* between expressions in the base and target using tiered-identicality
    - Determine inconsistencies
        - Mark match hypotheses inconsistent which violate parallel connectivity
        - Mark pairs of match hypotheses mutually inconsistent that would violate the one-to-one constraint
    - Assign scores to each correspondence using a local score for each match hypothesis and then using trickle-down to add to the scores of the arguments.
- Step 2: Create *kernel mappings* forming maximally structurally consistent collections of match hypotheses.
- Step 3: Combine kernel mappings using a *greedy merge* algorithm to create between one and three global mappings.

To illustrate this more concretely, let us examine SME's operation over the example base and target representations from Table 1. The base description contains six facts describing a ball being released and falling. The target description describes a box getting released. In the first step, two match hypothesis are created by matching identical predicates in the expressions, ‘d’ ↔ ‘i’ and ‘c’ ↔ ‘h’. Because these match hypothesis are consistent, they form a kernel mapping. Because there is only one kernel mapping, it becomes the global mapping with the following correspondences `Release-1` ↔ `Release-2` and `Ball-1` ↔ `Box-1`. Due to its participation in both expressions in the mapping, the support score for the `Release-1` ↔ `Release-2` correspondence is greater than the score for the `Ball-1` ↔ `Box-1` correspondence. SME computes the structural evaluation score by adding together the support scores of the mapping. Partially matched expressions from the base become candidate inferences by replacing the parts of the base expression which participate in the correspondences with the target items. In this case, expression ‘b’ becomes `(Ball Box-1)`, expression ‘e’ becomes `(causes-EventEvent Release-2 (AnalogySkolemFn Fall-1))` and expression ‘f’ becomes `(primaryObjectMoving (AnalogySkolemFn Fall-1) Box-1)`. The expression `AnalogySkolemFn` is used to represent unmapped entities from the base. Candidate inferences are conjectures about the target. Some of which are plausible. For example, the `Release-2` event causes something like the `Fall-1` event from the base, and the `Box-1` is the object moving of this event. Some of which are not. Such as, `Box-1` is a Ball. The correspondences and candidate inferences are central to analogical model formulation and domain transfer via analogy.

Base “A ball is released and falls”	Target “A box is released”
Entities - `Fall-1`, `Ball-1`, `Release-1`	Entities - `Box-1`, `Release-2`
Expressions a) `(FallingEvent Fall-1)` b) `(Ball Ball-1)` c) `(ReleaseOfSupport Release-1)` d) `(objectActedOn Release-1 Ball-1)` e) `(causes-EventEvent Release-1 Fall-1)` f) `(primaryObjectMoving Fall-1 Ball-1)`	Expressions g) `(BoxTheContainer Box-1)` h) `(ReleaseOfSupport Release-2)` i) `(objectActedOn Release-2 Box-1)`

Table 1: Example base and target descriptions

The other cognitive simulation used throughout this work is the MAC/FAC (Forbus *et al.* 1995) model of similarity-based retrieval. It takes as input a *probe* and a *case library*. The probe is a structured description, representing what is currently being worked on by some system. The case library is a set of cases, each a structured description, representing the set of available examples. MAC/FAC selects a case from the case library based upon similarity with the probe. It does this in two stages. The first stage (MAC) computes a special kind of feature vector for the probe and each case in the case library, whose components have a strength proportional to the number of occurrences of individual predicates in each structured representation. The case from the case library with the highest (or up to three, if very close) dot product with the probe is returned from the MAC stage. Using the examples from Table 1, the content vectors for the base and target contain six and three entries respectively. In this example, each content vector the entries are weighted equally because each predicate occurs only once in each description. The second (FAC) stage uses SME to compare these candidates to the probe. The candidate with the highest structural evaluation score is returned by the algorithm as its reminding. (If others are very close, up to three can be returned, but in the work discussed here, only the most similar reminding is used.)

SME and MAC/FAC do not completely simulate all aspects of human analogy. Building upon structure-mapping principles, other cognitive scientists have explored a variety of psychology phenomena related to analogy. LISA explores the effects of working memory limitations and is implemented on a neural network (Hummel & Holyoak 2003). LISA currently only handles descriptions significantly smaller than those used in this dissertation. Kokinov and others have developed AMBR with an emphasis on integrating analogical reasoning modules with other reasoning modules (Kokinov & Petrov 2001). This is consistent with the *integration constraint* as specified in Forbus (2001) which states that "a cognitive simulation of an analogical process, such as matching or retrieval, should be able to serve as a component in larger-scale cognitive processes." My research looks at how analogical processes can be used into two large-scale reasoning tasks: model formulation and cross-domain learning.

2.2 Companions

Building upon the integration constraint, I have been involved in developing the Companions Cognitive Architecture (Forbus & Hinrichs 2004). The underlying hypothesis of Companions is that the flexibility and breadth of human common sense reasoning and learning arises from analogical reasoning and learning from experience. Companions is a distributed agent architecture. Each agent has its own knowledge base (KB), whose contents are periodically updated and synchronized. Communication between agents occurs through KQML messages

(Labrou & Finin 1997). This section describes the project vision and the agents which make up the architecture configurations used in this work.

2.2.1 Project Vision

The motivating vision of Companions is to create software systems which can be treated as a collaborator (Forbus & Hinrichs 2004). In order to realize this vision, Companions focuses on three important problems for human-level AI:

- *Robust reasoning and learning* – Companions should learn about their domains, users and themselves.
- *Longevity* – Companions should operate continuously over weeks and months at a time.
- *Interactivity* – Companions should have high-bandwidth interaction with their users.

The work in this dissertation focuses primarily on the first problem. Analogical model formulation and domain transfer via analogy are advances in state of the art for learning and reasoning techniques. They are evaluated here solely on domain learning, but I believe that the methods are applicable to learning about users and self-modeling. Issues concerning longevity, performance, and evaluation of progress will be discussed in future work sections. Chapter 4 includes new ideas concerning sketch-based interaction increasing the bandwidth of communication between the Companion and the user.

2.2.2 Agent Architecture

The Companions' project is still in its infancy and underlying agents have evolved over the past four years. For the systems describe in this dissertation, the following agents were used:

- Session Manager: Provides facilities for user interaction
- Sketching Agent: Provides an interface between our sketch understanding system, sKEA (Forbus & Usher 2002), and other Companions agents
- Facilitator: Manages sessions and directs communications between agents
- Executive: Monitors the Companion's responsibilities and delegates work to the appropriate agents (e.g. follows scripts describing experiments, records quiz results, checkpoints KBs, etc.)
- Session Reasoner: Performs domain reasoning, in this case physics problem-solving
- Similarity-based Retriever: Monitors the working memory of the Session Reasoner, and provides similar prior cases when there is a close match.

The Session Manager and Sketch Agent run locally on the user's machine, the rest of the agents run on cluster nodes. New problems are given either individually through the Session Manager, or by a script describing an experiment which is uploaded to the Executive. The Executive hands the problem to the Session Reasoner, which implements all but the retrieval portion of the analogical model formulation and problem-solving processes. While the MAC/FAC algorithm used in the Retriever is efficient, distributing it reduces the memory load on the Session Reasoner as the size of case libraries rises. The next chapters describe the analogical model formulation and domain transfers via analogy methods and how Companions use these methods in a variety of domains.

3 Analogical Model Formulation

Modeling is a central activity in scientific and everyday reasoning. A model enables predictions about the world around us. A given situation could have a variety of different models. Take a ball dropped off a building. Suppose we wish to know the position of the ball after its release. The simplest model merely states that objects fall downwards, therefore after its release the ball's height decreases. Kinematics provides a mathematical model of this situation, $Y = v_i t - 9.8 * t^2$. Each model is useful in a variety of tasks and requires a number of assumptions.

This chapter explains the method of analogical model formulation. I begin with a description of the task of model formulation and an example. Next, I describe the current approaches to this task along with their limitations. These limitations are particularly constraining in comparison with the flexibility of human reasoning. Analogical model formulation overcomes these limitations by using analogy during model formulation.

3.1 Model Formulation

Creating systems capable of the breadth and flexibility of human reasoning is one of the central problems of artificial intelligence. The qualitative reasoning community has offered progress toward this goal by providing formalisms for reasoning with incomplete knowledge. An important contribution of this community is formalizing the task of *model formulation* (Falkenhainer & Forbus 1991). Given a scenario description, a domain theory, and a query, model formulation produces a *scenario model*, which consists of the relevant abstractions, processes, and causal relationships useful for answering the query.

Model formulation is not just applicable in qualitative reasoning, but it is central for quantitative reasoning as well. Consider the following question from *Mathematics in Nature* (Adams 2003): "Half of a snowball melts in an hour. How long will it take for the remainder to melt?" To arrive at a coherent answer for this problem, the following simplifying assumptions are extremely helpful:

- Assume the snowball is a uniform sphere at all times. Therefore, the surface area of the snowball is a function of the radius.
- Assume the density of the snowball is uniform throughout. Therefore, the mass of the snowball is directly proportional to its volume.
- Assume the mass of the snowball decreases at a rate directly proportional to its surface area, and that no other process affects the melting of the snowball.

While each of the assumptions makes sense from a problem-solving perspective, none of these are deductively valid. As any child will tell you, snowballs are hand-packed making a uniform density spherical snowball is virtually impossible. Also, once the snowball begins melting, certain parts of the snowball will melt faster than others. However, using the above assumptions, it is possible to formulate a model which allows one to answer that it will take 4.8 hours for the snowball to melt. There are other potential models. An even simpler model would be that the melting rate is constant throughout the entire process. But this simpler model would likely be more erroneous than the model constructed here. Model formulation is the task of constructing a scenario model to provide a useful answer to a query about a given situation while minimizing extraneous detail.

3.2 Traditional Approaches to Model Formulation

The initial approach to model formulation has been to create scenario models by hand. When engineers needed to monitor or make predictions about a particular mechanical system, they would manually construct special purpose simulators. Changing the mechanical system or the purpose of the simulator required rewriting all or part of the simulator.

To improve the reusability, one could construct a *domain theory* to describe a class of systems. The core of the domain theory consists of *model fragments* (Falkenhainer and Forbus 1991), each describing a single phenomenon. Answering a particular query about a scenario can be automated by (1) finding applicable model fragments, (2) instantiating them within the scenario context, and (3) composing a scenario model. This *compositional modeling* formalism is a vast improvement over manually encoding a custom model for every of scenario and query. By explicitly reasoning about assumptions, compositional modeling considers multiple levels of detail and mutually exclusive models for the same phenomena (e.g. relativistic versus Newtonian motion). Falkenhainer and Forbus (1991) demonstrated that compositional modeling can reason with large scale multi-grain, multi-perspective models of engineering thermodynamics. Domain experts provide the domain theory, including knowledge about the model fragments, assumptions and approximations. Further compositional modeling research provides methods for identifying the level of detail and perspectives to take in a scenario model. In the domain of electrical circuits, Nayak (1994) introduced *causal approximations* which provide a monotonic ordering between different models of the same object. This ordering enables tractable model formulation. To lessen the burden on the knowledge engineer, Rickel and Porter's TRIPEL (1994) builds scenario models using a domain theory consisting solely of the domain variables and influences between them. Then using domain-independent constants, TRIPEL constructs the simplest adequate model to answer a given prediction question about plant physiology.

Can these techniques account for the flexibility people show in their abilities to formulate models about the world? I believe not, for three reasons. First, traditional model formulation relies on having a complete and correct domain theory. Even in engineering and science applications, complete and correct domain theories can be difficult to construct. This problem becomes much more difficult when accounting for people's ability to reason about the world around them where the number of model fragments and entity types is orders of magnitude larger than engineering domains. The process is even more difficult as new conceptual types are created all the time. For example, before the year 2005, no knowledge base would have a concept for a WiiMote. Second, work in model formulation tends to start with fairly abstract scenario descriptions, e.g. circuit schematics (Flores & Cerda 2000), instead of everyday entities. While this is fine for engineering applications, the ability to create qualitative and quantitative models of everyday situations is a hallmark of the flexibility of human problem-solving. Third, also due to an emphasis on engineering domains, model formulation research has largely ignored learning. Humans are constantly learning about new domains and applying their knowledge to reason about the world. These limitations make it unlikely that traditional model formulation techniques can account for human-level domain reasoning. The next section describes a method to mitigate these problems of inflexibility by integrating analogical reasoning within the model formulation process.

3.3 Analogical Model Formulation

The solution proposed here is to use analogy to drive the model formulation process. By analogy, I do not mean cross-domain analogies, such as understanding an electrical circuit in terms of mechanical systems. Instead, within-domain analogies allow a new situation to be understood in terms of a prior example. For example, given instructions for estimating the melting time of an ice cube, one could apply similar modeling assumptions, approximations, and abstractions to model the above scenario of the melting snowball. Specifically, if assuming uniform density and a perfect cube shape proved useful to solving the melting time of the ice cube, the problem-solver would be wise to assume the snowball is also of uniform density and a perfect sphere. There are reasons to believe that this kind of within-domain analogical reasoning is ubiquitous during human common sense reasoning (Forbus & Gentner 1997). When faced with a new problem, one is reminded of similar experiences. The explanations for these prior experiences can be used to formulate a model for the new situation. The process of using an analogy with a prior example to construct a model is called *analogical model formulation.*

Figure 2: Overview of Analogical Model Formulation (blue items are inputs, purple items are processes, and orange items are results)

Figure 2 describes this process of analogical model formulation. Given a problem description, the process begins by retrieving an analogous example or examples using the computational models of analogy and similarity-based retrieval discussed in Chapter 2. This process is guided pragmatically to retrieve the sufficient analogue(s). E.g., if the retrieval does not provide the means to understand the entire scenario, retrieve additional analogs focusing on the unmatched aspects of the problem description. Next, the model formulation step uses inferences suggested by the analogy between the example(s) and the problem scenario to make modeling decisions concerning abstractions, assumptions, and causal models. Here, analogy bridges the gap between the problem scenario and any available domain knowledge. For example, if an agent has an abstract qualitative mechanics domain theory, but does not know how to map everyday objects into the technical vocabulary of the domain theory, then the agent uses the analogy to decide which abstractions are applicable to which objects in the problem scenario. On the other hand, if the agent does not have any domain knowledge, then the entire scenario model is constructed by analogy with the example. Finally, the agent uses rule based problem-solving over the model to determine the solution for the query.

Analogical model formulation addresses the shortcomings of traditional model formulation systems as follows. First, by relying on examples, analogical model formulation alleviates the need for a complete and correct domain theory. It is well-known in the knowledge acquisition community that getting domain experts to tell stories (i.e., concrete examples) is considerably easier than getting experts to articulate complete and correct domain theories. Second, analogical model formulation can build scenario models of everyday situations. The breadth of entity types is limited only by the underlying ontology. Third, analogical model formulation provides a way to create systems that can incrementally learn by making effective use of what knowledge they have, even when it is incomplete. Extending the range of scenarios or domains over which the system can reason requires only adding new examples.

3.4 Benefits of Integrating Analogy

While qualitative reasoning has made many advances in generating and reasoning about models, automating the model construction process is still a difficult task. This is especially true for broad domains. The fact that people are able to form useful models about the world all the time highlights the importance of this problem. The use of experience in model formulation was first proposed by Falkenhainer (1992). Falkenhainer's *credibility extrapolation* used error measured in prior scenarios to bound the error of a current model. Through analogies with previous experience, the model formulation process was informed about the consequences of the approximations and assumptions under consideration from its domain theory. Like other compositional modeling efforts, Falkenhainer’s work still focused on engineering problems, and the domain theory included the complete set of model fragments, assumptions and approximations for the domain.

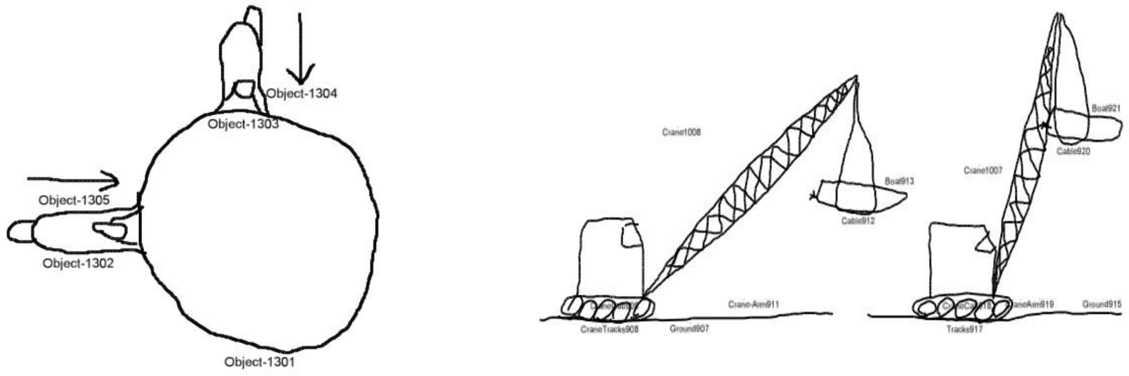
Analogical model formulation places analogy at the center of the model formulation process. The examples provide the scenario model with the assumptions, approximations, causal relationships and equations. Analogical model formulation offers a promising approach to account for the human ability to form models about the world around them. This method integrates existing analogical processing and model-based reasoning techniques. The systems presented in the proceeding chapters utilize qualitative reasoning to derive solutions from models, and leverage analogical reasoning as described above to make modeling decisions that are otherwise difficult in broad domains.

In the next two chapters, I will present support for the hypothesis that analogical model formulation provides a robust method for reasoning and learning in two domains with relatively unconstrained scenarios: everyday physical reasoning and AP Physics. The evaluations are designed evaluate to what extent analogical model formulation allows a Companion with limited domain knowledge to use examples to solve new problems.

4 Everyday Physical Reasoning with Analogical Model Formulation

Evaluating the claims of analogical model formulation requires broad domains. Finding domains which test the breadth of AI systems, while still allowing for progress, is difficult. The Bennett Mechanical Comprehension Test (BMCT; Bennett 1969) is one such domain. The BMCT is used to evaluate applicants for technical positions. BMCT problems consist of diagrams depicting physical situations with multiple choice questions about their qualitative properties. For concreteness, two examples of BMCT questions are illustrated in Figure 3[1]. The BMCT is broad in two ways. First, it involves a variety of domains, including statics, dynamics, acoustics, heat, and electricity. This is the *domain breadth* problem. Second, it involves a variety of everyday objects and systems: Bicycles, railroad cars, cranes, hoists, boats, and many others. This is *everyday breadth* problem. This makes it a valuable domain for assessing everyday physical reasoning, which is an important part of human common sense.

[1] To protect the security of the test, we cannot provide a full list of the problems used in this evaluation.



Outcome Problem:
Which way will the ball go?
A) down, B) right, or C) down and right

Comparative Analysis Problem:
Which crane is more stable?
A) the left one, B) the right one, or C) equally stable

Figure 3: Bennett Mechanical Comprehension Test problems

Figure 3 illustrates the two general types of problems that appear on the BMCT. The ball problem is an example of an outcome problem, which asks for a prediction about a property of a system. The crane problem is an example of a comparative analysis problem, where the question concerns a comparison between two (or three) situations, or two aspects of the same situation. To solve either of these problems requires moving from these everyday descriptions to a formal model. Once the model has been formulated, these problems are straightforward. The ball problem requires the addition of forces applied by the people, and the crane problem requires comparing the horizontal distances between the bases of the cranes and their loads. The hard part is formulating the right model. For both of these examples, many other parameters were potentially relevant (e.g. the materials of the objects involved, the types of connections between them, the kinds of surfaces they are resting on, etc). In the outcome problem, the types of objects play a central role in determining the relevant abstractions. For example, if the ball was instead a lake, the answer would be drastically different. In the comparative analysis problem, one must focus on the differences that are visible between the two scenarios, which requires understanding how conceptual properties (like stability) depend on visual properties (like distances). To be successful on the BMCT, one must be able to form a broad range of models, integrating spatial and conceptual reasoning, over a wide range of input descriptions. This makes the BMCT a great domain for evaluating analogical model formulation.

This chapter describes how analogical model formulation allows a Companion to construct models about a diverse group of everyday scenarios. This enables a Companion to perform model formulation without a complete domain theory, as required for traditional first-principles only qualitative reasoning. In addition to analogical model formulation, this chapter presents two additional contributions: *Sketch annotations* communicate linkages between visual and conceptual properties in sketches, and *analogical reference frames* enable comparative analysis to operate over a broader range of problems than prior techniques.

This chapter describes an experiment to see how well a Companion using analogical model formulation could perform on a subset of the BMCT. The subset was chosen to minimize encoding efforts while still allowing us to evaluate the performance of analogical model formulation. The subset design and the analogical model formulation approach, which will be covered in detail later in the chapter, pose both the everyday and domain breath challenges to the Companion. In addition to simply evaluating the Companion's ability to solve BMCT problems, it is necessary to evaluate the claim that analogical model formulation is a robust solution. To evaluate robustness, three people were used to provide the Companion with three different sets of examples for analogical model formulation. Evaluating each set independently illustrates that this variation is important (i.e. the Companion is able to solve more problems with some sets of examples than others). By combining the sets of examples, we can evaluate the robustness in the face of poor examples.

I begin by describing the software systems and research this work draws upon: the sKEA sketch understanding system (Forbus & Usher 2002) and qualitative mechanics from Nielsen (1988) and Kim (1990). The next section describes sketch annotations and how the examples are created. This is followed by a detailed explanation of how analogical model formulation is applied to BMCT problems. Before describing the problem-solving algorithm, I introduce analogical reference frames to extend the range of applicable problems for comparative analysis (Weld 1988). This chapter closes with a description of the experiment and a discussion of the results.

4.1 System Components

4.1.1 Sketch Understanding with sKEA

Sketching is a powerful way to work out and communicate ideas. The nuSketch model (Forbus *et al.* 2004) takes sketching to be a combination of interactive drawing and conceptual labeling. While most sketch understanding systems focus on the problem of recognition, nuSketch systems are based on the insight that recognition is not necessary in human-to-human sketching. The sketching Knowledge Entry Associate (sKEA) was the first open-domain sketch understanding system. Anything that can be described in terms of sKEA's knowledge base can be used in a sketch. For this evaluation, sKEA's knowledge base consists of a 1.2 million fact subset of Cycorp's Cyc Knowledge Base, which includes over 38,000 concepts, over 8,000 relations, and over 5,000 logical functions. The KB for this evaluate also includes representations for qualitative physics, visual properties and relationships, spatial knowledge, and analogical reasoning. The breadth of this KB makes it an excellent platform for exploring reasoning in a broad domains such as the Bennett Mechanical Comprehension Test, because the entity types and relations necessary to define problems, such as "crane" and "wheelbarrow", are already defined, as opposed to having to generate them specifically for this project.

Glyphs are the basic constituent of sketches. A *glyph* consists of its *ink*, representing its visual properties, and its *content*, representing the conceptual entity depicted by the glyph. The content can be instances of any of the concepts in the KB. Each sketch is divided into layers, which decompose a sketch into pieces. For example, two systems being compared side by side would be drawn in the same sketch, but drawn on different layers. Sometimes systems must be viewed at different levels of abstraction. In understanding how a wheelbarrow works, for example, it makes sense to draw the individual parts, since each contributes differently to how it functions. But if we are considering how hard it will be to lift a wheelbarrow, we need to consider the wheelbarrow as a single rigid object. sKEA includes group glyphs, which introduce a new entity to represent a selected set of entities, to handle such situations.

Using the electronic ink, sKEA computes the following visual relationships between glyphs (Forbus *et al.* 2003):

- Qualitative topological relationships: sKEA uses the RCC8 algebra (Cohn 1996) to describe the topological relationships between glyphs in a sketch. In RCC8, two glyphs might be disjoint (DC), touching exactly on their boundaries (EC), partially overlapping (PO), equal (EQ), one completely inside the other (NTPP), or one inside the other but their boundaries touching (TPP). With inverses for NTPP and TPP, these eight relationships completely characterize the possible connectivity relationships between two 2D regions.

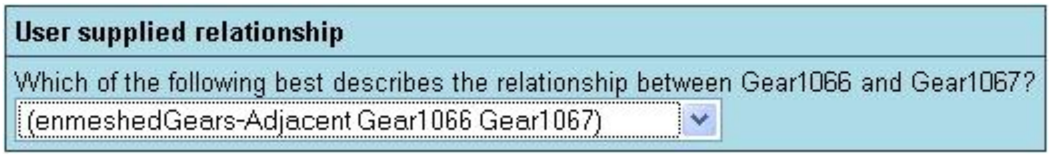
- Positional relationships: Relationships such as above/below and left/right are computed for pairs of glyphs that are adjacent. Adjacency is determined via a Voronoi diagram.

- Visual grouping relations: The RCC8 relationships naturally impose two visual grouping relationships. *Connected glyph groups* consist of sets of glyphs that are pairwise PO or EC with each other. *Contained glyph groups* consist of the set of glyphs which are TPP or NTPP with some larger glyph (called the *container* for the group).

- Orientations: An axis is computed for each glyph, which is characterized as primarily vertical or horizontal, as appropriate.

- Sizes: Each glyph in the sketch is classified as one of five qualitative categories, from tiny to very large, depending on the distribution of glyph sizes in the sketch. This is done by computing the minimum bounding rectangle of each glyph normalized against the minimum bounding rectangle of entire sketch.

Sometimes the visual relationship between a pair of glyphs and the nature of their contents implies a conceptual relationship between their contents. For example, if a glyph representing a wheel is Edge Connected (EC) to a glyph representing the ground, then it is reasonable to assume that the wheel is touching the ground. These visual-conceptual relationships represent commonly used conventions for depicting situations in sketches (Forbus *et al.* 2005). sKEA automatically infers a large candidate set of relationships, and provides an interface for users to select which of them, if any, is appropriate. Figure 4 contains a screen shot of a user selecting the `enmeshedGears-Adjacent` relationship for a pair of gears. The selected conceptual relationship is added to the predicate calculus representation for the sketch.


User supplied relationship

Which of the following best describes the relationship between Gear1066 and Gear1067?

(enmeshedGears-Adjacent Gear1066 Gear1067)

Figure 4: sKEA allows the user to specify the conceptual relationship between sketched entities

The result of drawing in sKEA is a sketch which contains the ink of the glyphs and a corresponding case of predicate calculus facts. These facts include the spatial relationships concerning the glyphs as well as the conceptual relationships concerning the conceptual entities depicted by the sketch. In addition, when reasoning about the sketch, sKEA accepts queries for additional facts to be computed on demand. For example, the Companion asks sKEA when it needs to know the outside normal direction of a surface, the overlapping parts of two glyphs, or the distance between two points. sKEA responds to these queries by creating a conceptual representation that is tied directly to the ink of the glyphs. Given the spatial and conceptual components of the BMCT, sKEA is an excellent platform for this task.

4.1.2 Qualitative Mechanics

Mechanics is traditionally concerned with forces, motion, and how bodies interact. Qualitative mechanics provides a set of abstractions (e.g., rigid body, surface normal, qualitative descriptions of angle, etc.) that support qualitative reasoning about traditional mechanics phenomena. The technical vocabulary and model fragments of qualitative mechanics are part of the starting endowment of the system, rather than as something to be learned. The qualitative mechanics domain theory is drawn from (Nielsen 1988; Kim 1993). Specifically, their qualitative representations of objects, surfaces, force transfers, and centers of rotation are included. The key aspects of the domain theory are summarized here.

The Companion's domain theory consists of five model fragments, `XForceTransfer`, `YForceTransfer`, `TorqueTransfer`, `TorqueEquilibrium`, and `ForceDistribution`. Figure 5 contains the CML descriptions (Bobrow *et al.* 1996) for `XForceTransfer` and `TorqueEquilibrium`. The `XForceTransfer` model fragment has five participants, the forcer, the object, each of their surfaces and the direction of the force. This model fragment is active when the forcer is applying a force on the object's surface and results in the force being added to the net force on the object. The `TorqueEquilibrium` model fragment has seven participants and is active when there are two applied torques in opposite directions. While this definition is a simplification[2], it is sufficient for these problems. When a `TorqueEquilibrium` model fragment is active, its consequences are believed. These consequences are qualitative proportionalities describing causal functional dependencies between quantities. Since the Cyc KB uses a relational form for quantities, the function `QpQuantityFn` converts them to an equivalent fluent. Here two functions are used to represent the four quantities affected by the model fragment. The function `(QpQuantityFn ForceAppliedToSurfaceBy)` takes two arguments and denotes the amount of force applied to a surface (its first argument) by an object (its second argument). The function `(QpQuantityFn DistanceToOrigin)` denotes the distance from a surface to the point it rotates around. When instantiated, these model fragments provide causal relationships (i.e., qualitative proportionalities) needed to solve BMCT problems.

[2] Torque equilibrium also requires that the opposing torques be equal. The system currently assumes this by default.

```
(defModelFragment XForceTransfer
  :participants ((TheObject :type RigidObject)
                 (TheForcer :type RigidObject)
                 (TheSurface1 :type Surface :constraints ((hasSurface TheObject TheSurface1)))
                 (TheSurface2 :type Surface
                            :constraints ((hasSurface TheForcer2 TheSurface2)
                                          (surfaceContact TheSurface1 TheSurface2)))
                 (TheXDir1 :type Sense))
  :conditions ((xForceApplied TheSurface1 TheXDir1 TheForcer))
  :consequences ((c+ ((QpQuantityFn XNetForce) TheObject)
                    ((QpQuantityFn XForceAt) TheSurface1))))

(defModelFragment TorqueEquilibrium
  :participants ((TheObject :type RigidObject)
                 (TheForcer1 :type RigidObject)
                 (TheForcer2 :type RigidObject)
                 (TheSurface1 :type Surface :constraints ((hasSurface TheObject TheSurface1)))
                 (TheSurface2 :type Surface :constraints ((hasSurface TheObject TheSurface2)))
                 (TheRotDir1 :type RotDirection)
                 (TheRotDir2 :type RotDirection
                           :constraints ((inverseRotDirection TheRotDir1 TheRotDir2))))
  :conditions ((torqueApplied TheSurface1 TheRotDir1 TheForcer1)
               (torqueApplied TheSurface2 TheRotDir2 TheForcer2))
  :consequences ((qprop ((QpQuantityFn ForceAppliedToSurfaceBy)  TheSurface1 TheForcer1)
                        ((QpQuantityFn ForceAppliedToSurfaceBy)  TheSurface2 TheForcer2))
                 (qprop ((QpQuantityFn ForceAppliedToSurfaceBy)  TheSurface1 TheForcer1)
                        ((QpQuantityFn DistanceToOrigin)  TheSurface2))
                 (qprop- ((QpQuantityFn ForceAppliedToSurfaceBy)  TheSurface1 TheForcer1)
                         ((QpQuantityFn DistanceToOrigin)  TheSurface1))))
```

Figure 5: Example model fragments from the qualitative mechanics domain theory

Notice that the qualitative mechanics domain theory is defined in terms of a technical vocabulary of abstract concepts, including `RigidObject`, `Surface`, and `Fulcrum` as well as the relationships `xforceApplied` and `torqueApplied`. The domain theory includes inference rules which, given conceptual relationships between sketched objects (e.g., `canPivotAround`, `touchesDirectly`, and `on-Physical`), can conclude the `xforceApplied` and `torqueApplied` relationships. However, since problems on the BMCT are given in terms of everyday situations, the appropriate abstractions must be inferred in order to determine which model fragments are applicable. Section 4.3 describes how this is done via analogical model formulation.

4.2 Sketch Annotations

In sketching, people annotate sketches of physical entities with conceptual information that would not appear in the actual situation. In architectural drawings, annotations indicate distances between walls and the widths of windows. In sketches explaining principles, annotations indicate important spatial properties, such as the radius of a gear and where forces are applied. Annotation glyphs provide this capability in sKEA. Figure 6 contains sketches that illustrate each type of annotation. Like other glyphs, an annotation glyph consists of its ink and its content, i.e., the entity it is representing. However, annotation glyphs also refer to one or more other glyphs in the sketch, indicating the entity (or entities) about which they are providing information. These glyphs are the *references* for the annotation glyph.

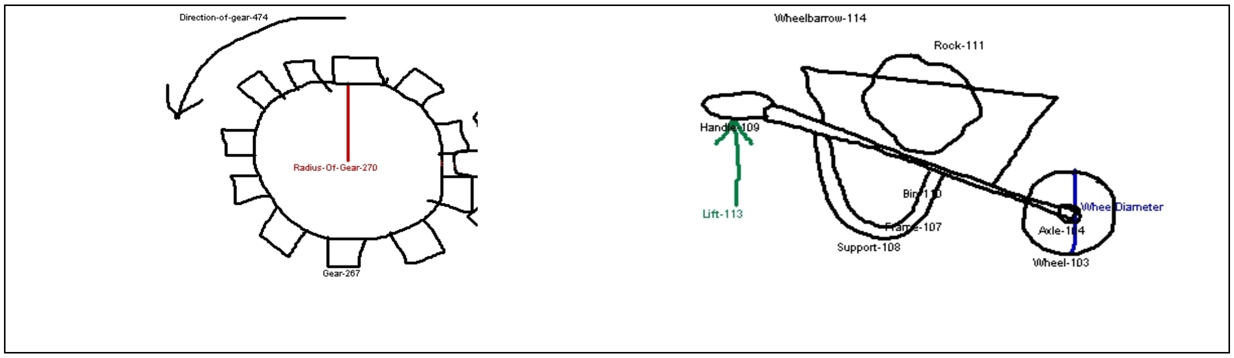


Figure 6: The gear has a rotational annotation indicating counter-clockwise motionan and a linear annotation indicating its radius. The wheelbarrow has a force annotation indicating an applied force upward on the handle and a length annotation indicating the diameter of its wheel.

This work introduces three types of annotation glyphs. *Force annotations* indicate the location and direction of an applied force on a reference. sKEA computes the direction and application surface of a force annotation from its ink and referenced glyphs. If there are two references, sKEA uses the direction of the arrow and the relative positions of the references to determine which object is applying the force. If there is only one reference (e.g., the handle from the wheelbarrow example), sKEA assumes a new object which is applying a force at the point of the arrow onto the reference. *Rotational annotations* indicate a reference's direction of rotation. sKEA assumes the qualitative rotational motion of the reference as either clockwise or counter-clockwise. *Linear annotations* indicate linear distances, either along a single reference or between two references. Two special subclasses of linear annotations are *X-coordinate* and *Y-coordinate annotations*, which refer to the projection of the measurement onto the appropriate axis. sKEA computes distance measurements using the closest *anchor points* on the reference(s) to the endpoints of the linear annotation. Anchor points are used to specify which parts of the reference(s) that the annotation is tied to. Each glyph has nine anchor points: the centroid, the rightmost top, leftmost top, top rightmost, and so on clockwise around the glyph. Anchor points provide symbolic descriptions that can be projected as candidate inferences from an example to a new situation (e.g., the distance from the left bottommost point to the right bottommost point of a reference).

4.2.1 Creating Examples using Annotation Glyphs

Examples in the Companion's case library represent its experience. Users create examples of physical scenarios using sKEA and a concept map system. The concept map system allows users to enter predicate calculus expressions about the sketched entities. The process begins with the user drawing the scenario, labeling their glyphs with the everyday concepts used to describe the entities depicted. Next, the user sketches force and rotation annotations where applicable in the sketch. They use sKEA's conceptual labeling interface to identify all of the appropriate qualitative mechanics abstractions (e.g., `RigidObject`) and sKEA's visual/conceptual relationships interface to identify the appropriate conceptual relationships between entities (e.g., `touchesDirectly`). At any time the user can invoke a traditional model formulation algorithm (Forbus & Falkenhainer 1991) to see if the appropriate qualitative mechanics model fragments are instantiated. Once they are satisfied with the resulting qualitative mechanics model, the final step is to create causal models that describe conceptual quantities in terms of visual quantities. For each relevant conceptual quantity (e.g., the revolution rate of a gear), the user draws a linear annotation for the causally related visual quantity (e.g., the distance from the center of the gear to its top right most point). Next, using the concept map system, the user adds a qualitative proportionality linking the conceptual quantity to the visual quantity (e.g., that the revolution rate is qualitatively proportional to the radius of the gear). This completes the process of constructing the scenario model for the example. Note that the model fragments, if any, are instantiated from an incomplete domain theory, while conceptual/visual quantity causal relationships are defined in an example-specific manner. All of the visual and conceptual representations for the example are stored as a case in a library, to be used for subsequent analogical model formulation.

To illustrate this process more concretely, the following steps are required to construct the example wheelbarrow in Figure 6. First, a user draws the wheelbarrow in sKEA. To do this, the user draws seven glyphs representing the wheel, axle, bin, frame, support, handle and rock. Next, a group glyph is created representing the wheelbarrow including all the glyphs except the rock. The wheelbarrow's handle is annotated with a force arrow indicating that there is an assumed force in the upward direction. Then, the user adds qualitative mechanics abstractions by labeling the wheelbarrow group glyph and the rock as instances of the type `RigidObject` and the axle as a Fulcrum. sKEA's visual/conceptual relations interface is then used to add some of the conceptual relationships needed by the qualitative mechanics domain theory. Here, the user stated that the rock is on-Physical the bin. The domain theory includes rules to determine that the on-Physical relationship results in a downward force from the rock onto the bin. A standard model formulation algorithm instantiates whatever model fragments are appropriate, based on these abstractions and relationships. This results in the instantiation of a ForceDistribution and two TorqueEquilibrium model fragments. The ForceDistribution consists of the rock pushing down on the wheel and the assumed object in contact with the handle. The two TorqueEquilibrium model fragments are symmetric. In one, theForcer2 is the assumed object in contact with the handle, and theForcer1 is the rock. In the other TorqueEquilibrium model fragment, the participants are reversed. The consequences of the first model fragment appear in Figure 7. The first qualitative proportionality states that the force applied by the rock on the surface between the rock and the bin is proportional to the force applied on the handle by the assumed force. Surfaces are defined using the function ContactSurfaceFn which takes two arguments. The term denotes the surface of the first argument that is in contact with the second argument. For example, (ContactSurfaceFn Bin-110 Rock-111) denotes the surface of Bin-110 which is in contact with Rock-111. (ContactObjectFn Handle-109) denotes the assumed object in contact with the handle. The next two statements say that the force applied on the handle is proportional to the distance from the surface of the rock to the origin of rotation and inversely proportional to its distance from the origin of rotation.

```
 (qprop
  ((QpQuantityFn ForceAppliedToSurfaceBy) (ContactSurfaceFn Handle-109
                                                            (ContactObjectFn Handle-109))
                                           (ContactObjectFn Handle-109))
  ((QpQuantityFn ForceAppliedToSurfaceBy) (ContactSurfaceFn Bin-110 Rock-111) Rock-111))
(qprop
  ((QpQuantityFn ForceAppliedToSurfaceBy) (ContactSurfaceFn Handle-109
                                                            (ContactObjectFn Handle-109))
                                          (ContactObjectFn Handle-109))
  ((QpQuantityFn DistanceToOrigin) (ContactSurfaceFn Bin-110 Rock-111)))
(qprop-
  ((QpQuantityFn ForceAppliedToSurfaceBy) (ContactSurfaceFn Handle-109
                                                           (ContactObjectFn Handle-109))
                                          (ContactObjectFn Object-109))
  ((QpQuantityFn DistanceToOrigin) (ContactSurfaceFn Handle-109
                                                     (ContactObjectFn Handle-109))))
```

Figure 7: Resulting qualitative proportionalities from the torque equilibrium model fragment

Before the example is complete, any causal dependencies of conceptual quantities on visual quantities must be entered. All of the entities and quantities in the scenario model are automatically added to the concept map interface. Similarly, every time an annotation glyph is used to create a visual quantity, that quantity is automatically added to the concept map interface. In this example, the user might want to convey that the wheelbarrow's smoothness of the ride is determined in part by the diameter of its wheel. To do this, they first add a linear annotation to the wheel. Next, using the concept map interface, they enter the qualitative proportionality between the quantity representing the wheel's diameter and the wheelbarrow's smoothness of ride. Figure 8 shows a portion of the symbolic representation for the annotation glyph and the qualitative proportionality entered via the concept map. The first fact links the visual quantity, named "Wheel Diameter" by the user, to the annotation glyph. The next two facts indicate the start and end points of the annotation glyph, in terms of anchor points on the reference glyph. In this case, the start and end points are the bottom right point and the top left point of `Wheel-103`, respectively. The final fact shows the qualitative proportionality between the quantity representing the wheel's diameter and the wheelbarrow's smoothness of ride.

Automatically added by sKEA

```
(visualRepresentationOfQuantity
  ((ConceptKnownAsFn "WheelDiameter")
       (GlyphFn Wheelbarrow-114 User-Drawn-Sketch-Layer-114))
  (AnnotationGlyphFn WheelDiameter-203 User-Drawn-Sketch-Layer-114))
(startPointOf
  (AnnotationGlyphFn WheelDiameter-203 User-Drawn-Sketch-Layer-114)
  (RightmostBottomPointFn
       (GlyphFn Wheel-103 User-Drawn-Sketch-Layer-114)))
(endPointOf
  (AnnotationGlyphFn WheelDiameter-203 User-Drawn-Sketch-Layer-114)
  (LeftmostTopPointFn
       (GlyphFn Wheel-103 User-Drawn-Sketch-Layer-114)))
```

Entered manually through the Concept Map

```
(qprop ((QPQuantityFn RideSmoothness) Wheelbarrow-114)
       ((ConceptKnownAsFn "WheelDiameter")
        (GlyphFn Wheelbarrow-114 User-Drawn-Sketch-Layer-114)))
```

Figure 8: A subset of the facts (simplified for readability) to represent the wheel diameter annotation and the causal relationship between the wheel diameter and the wheelbarrow's smoothness of ride

To summarize, examples include three types of information:

1. The everyday entities represented in the sketch.
2. Instances of model fragments, constructed automatically from the sketched entities and relationships, using the conceptual labeling and the visual/conceptual relationship interfaces to provide the necessary abstractions for model formulation.
3. Example-specific causal relationships between visual quantities (measurable in the sketch) and conceptual quantities.

The next section shows how these examples can be used via analogy to construct scenario models in new situations.

4.3 Analogical Model Formulation

Analogical model formulation creates a scenario model for a new situation by analogy with a previously understood example. The process begins with MAC/FAC being used to retrieve a relevant example. SME then creates a mapping between the example and new situation. This mapping includes a set of candidate inferences which suggest modeling decisions for the new situation. For BMCT problems, candidate inferences provide qualitative mechanics abstractions and relationships, definitions for visual quantities, and causal relationships. Together, this information provides the basis for a scenario model which can be used to solve BMCT problems.

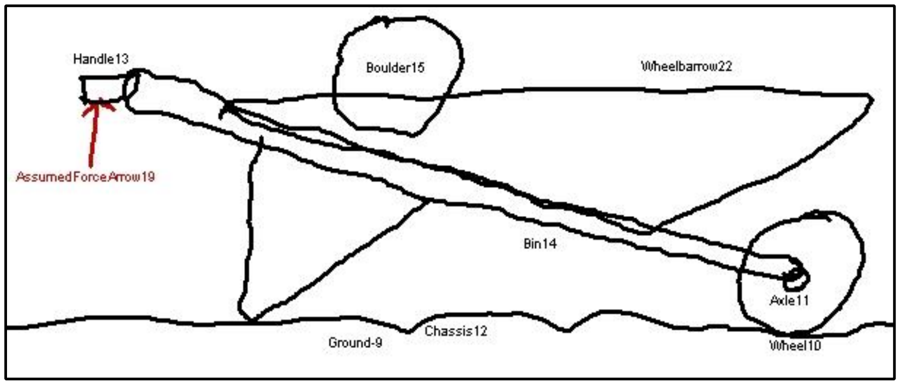


Figure 9: A wheelbarrow from a problem sketch

This section presents how analogical model formulation constructs a scenario model for the wheelbarrow shown in Figure 9 using the example wheelbarrow from Section 4.2.1. SME creates a mapping, a portion of which is shown in Table 2, between the predicate calculus representations of the problem and the example. Because SME is guided by shared relational structure, the resulting mapping does not necessarily include all of the entities in the problem and example. Expressions from the example that do not participate fully in the mapping become candidate inferences, in which the mapped portions are replaced by the corresponding expressions in the problem. Analogical model formulation depends upon these candidate inferences to make modeling decisions in the problem scenario.

Example entities (Base)	Problem situation entities (Target)
Wheelbarrow-114	Wheelbarrow-22
Rock-111	Boulder-15
Handle-109	Handle-13
Frame-107	Chassis-12
Bin-110	Bin-14
Axle-104	Axle-11
Wheel-103	Wheel-10
Lift-113	AssumedForceArrow-19
Support-108	--
Wheeldiameter-203	--
--	Ground-9

Table 2: Mapping between the problem and example entities

Analogical model formulation uses the example to infer three types of information about the problem scenario: causal models, qualitative mechanics abstractions and relationships, and information regarding the measurement of visual quantities. Causal models are inferred from the example as follows. The qualitative proportionalities in Figure 7 and Figure 8 become candidate inferences with the entities for `Wheelbarrow-114`, `Handle-109`, `Wheel-103`, `Bin-110` and `Rock-111` replaced with `Wheelbarrow-22`, `Handle-13`, `Wheel-10`, `Bin-14` and `Boulder-15` respectively. The system searches the candidate inferences for qualitative proportionalities and assumes them into the problem representation. Qualitative mechanics abstractions and relations are inferred in the same way. Candidate inferences concerning abstractions and relations are assumed into the problem (e.g., `(isa Axle-11 Fulcrum)` and `(on-Physical Bin-14 Boulder-15)`). Visual quantity measurement information is imported in two ways. First, measuring `DistanceToOrigin` quantities requires qualitative mechanics knowledge about the center of rotation. For example, the distance to the origin from the surface between the boulder and the bin requires knowing that the axle is the fulcrum. The second type of visual quantity concerns user defined annotations, such as `WheelDiameter-`

203. In this case, the expressions in the example concerning the annotation become candidate inferences. The Companion uses these candidate inferences to automatically create a corresponding annotation in the problem. Figure 10 contains the candidate inferences which define the wheel diameter quantity and provide instructions as to how to draw the annotation based upon anchor points. Since the entity for the annotation WheelDiameter-203 does not participate in the mapping, these candidate inferences contain AnalogySkolemFn expressions. These expressions represent entities which appear in the base but do not have a corresponding entity in the target. Using this automatically constructed annotation, the scenario model includes information concerning the measurement of a visual quantity that was defined only in terms of the example.

```
(visualRepresentationOfQuantity
  ((ConceptKnownAsFn "WheelDiameter")
    (GlyphFn Wheelbarrow-22 User-Drawn-Sketch-Layer-114))
  (AnalogySkolemFn
    (AnnotationGlyphFn WheelDiameter-203 User-Drawn-Sketch-Layer-114)))
(startPointOf
  (AnalogySkolemFn
    (AnnotationGlyphFn WheelDiameter-203 User-Drawn-Sketch-Layer-114))
  (RightmostBottomPointFn
    (GlyphFn Wheel-103 User-Drawn-Sketch-Layer-114)))
(endPointOf
  (AnalogySkolemFn
    (AnnotationGlyphFn WheelDiameter-203 User-Drawn-Sketch-Layer-114))
  (LeftmostTopPointFn
    (GlyphFn Wheel-103 User-Drawn-Sketch-Layer-114)))
```

Figure 10: Candidate inferences concerning the wheel diameter visual quantity

The next section describes how analogy is used to frame comparative analyses allowing for its application between scenario models necessary for solving BMCT problems.

4.4 Analogical Reference Frames for Comparative Analysis

Comparative analysis, and in particular *comparative analysis*, seeks to understand how and why the behavior of a system will change given some changes to its parameters (Weld 1988). For example, comparative analysis can explain why the period of an oscillating block system would increase if the mass of the block was increased. A *CA value* is a qualitative description of how one particular parameter in a system will change given other parameter changes. There are four possible CA values: unchanged, increased, decreased and ambiguous. The comparative analysis problems on the BMCT do not fit directly into the traditional perturbed system framework. First, they involved comparisons between scenarios, rather than describing a perturbation in a single scenario. Second, some of the problems require comparisons between different parts of the same system. For example, the BMCT problem in Figure 11 asks "which wheel of a railcar presses harder on the rail?"

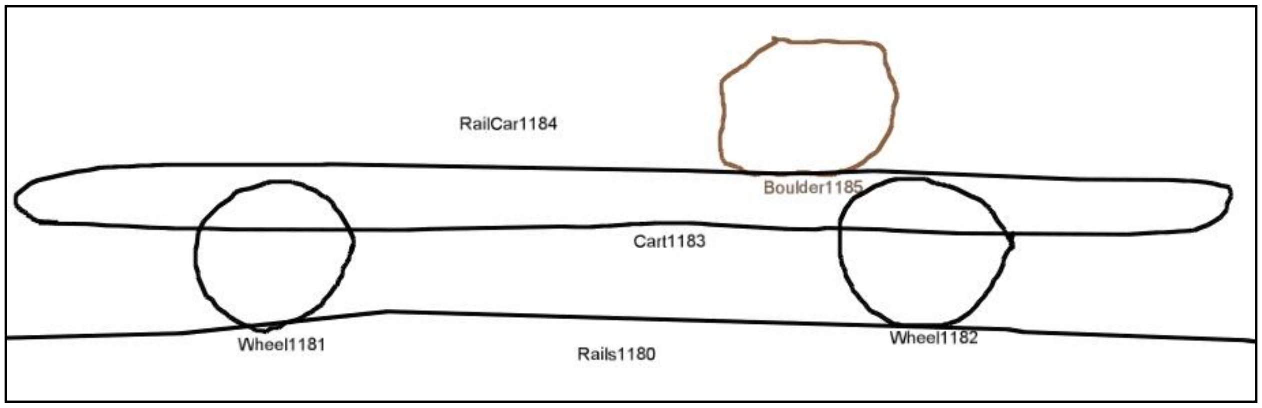


Figure 11: A comparative analysis problem concerning aspects of the same system, "which wheel of the railcar presses harder on the rail?"

Analogy provides a general mechanism for framing comparative analyses. Using SME, an *analogical reference frame* is created to determine correspondences between either two systems, or different aspects of the same system. These correspondences frame a comparative analysis problem by defining what each parameter is compared against. In problems with multiple systems, SME creates an analogy between the systems with one of them as the base and the other as the target. In the case of single system problems, the system is compared with itself while constraining the entities being compared to match each other. For instance, in the rail car problem from Figure 11, SME matches the scenario to itself but requires that `Wheel-1181` correspond with `Wheel-1182`.

The analogical reference frame lines up quantities whose differences can be reasoned about via standard comparative analysis. Returning to Figure 11, we want to find the CA value of the force applied to the rail at `Wheel-1181` through a comparison with its corresponding quantity, the force applied to the rail at `Wheel-1182`. The causal models indicate that the force on each wheel is inversely qualitatively proportional to the distance between the surface of the boulder in contact with the cart and that wheel. Since the two distances are aligned by the analogical reference frame, we determine that the CA value for the distance concerning `Wheel-1181` is decreased. Since the relationship to force is inversely qualitatively proportional this distance, the CA value for the force applied at `Wheel-1181` is increased, i.e., `Wheel-1182` is pressing harder on the rail. Analogical reference frames are important because they allow a wider class of systems to be analyzed, since the correspondences between aspects of a problem are computed dynamically.

4.5 Solving BMCT Problems via Analogy

This section describes how sketch annotations, analogical model formulation, and analogical reference frames come together to allow a Companion to solve BMCT problems from examples. Problems are presented as sketches of the situation and a query. The session reasoner solves these problems using the AND/OR suggestion architecture from (Forbus & De Kleer 1994). The problem-solving knowledge consists of 19 methods and 136 backchaining rules. The entire process is implemented on the Companions cognitive architecture with the following agent configuration:

- Facilitator: Manages sessions, starts up other agents, and helps set up communication channels between agents.
- Session Manager: Provides generic facilities for user interaction, including startup, shutdown, and making queries
- Sketching Agent: Provides an interface between sKEA and other Companion's agents
- Session Reasoner: Responsible for the domain reasoning, in this case solving BMCT problems.
- Similarity-based Retriever: Provides analogical remindings to the session reasoner based upon the current contents of working memory.

The agent architecture allows specialized reasoning to be distributed to different agents across the entire Companion. For example, when the session is started, subscriptions brokered by the Facilitator are set up between the Session Reasoner and the Retriever, so that the Retriever receives updates in the Session Reasoner's working memory. Using these updates, the Retriever runs MAC/FAC to retrieve a case from its case library as an analogical match to the working memory contents of the Session Reasoner. When the user makes changes to the sketch altering the Sketching Agent's working memory, subscriptions update the working memory contents in the Session Reasoner. These brokered subscriptions basically become remote procedure calls once set up: If the Session Reasoner needs to know the distance between two points, the query is automatically forwarded to the Sketching Agent, which measures the distance on the sketch and sends the result back to the Session Reasoner.

Solving BMCT problems using analogical model formulation involves three steps. First, the Companion retrieves a relevant example. Second, the Companion creates a scenario model based upon the example using analogical model formulation. Third, the Companion uses the model to compute the answer. Figure 12 shows how analogical model formulation is implemented on the Companion cognitive architecture.

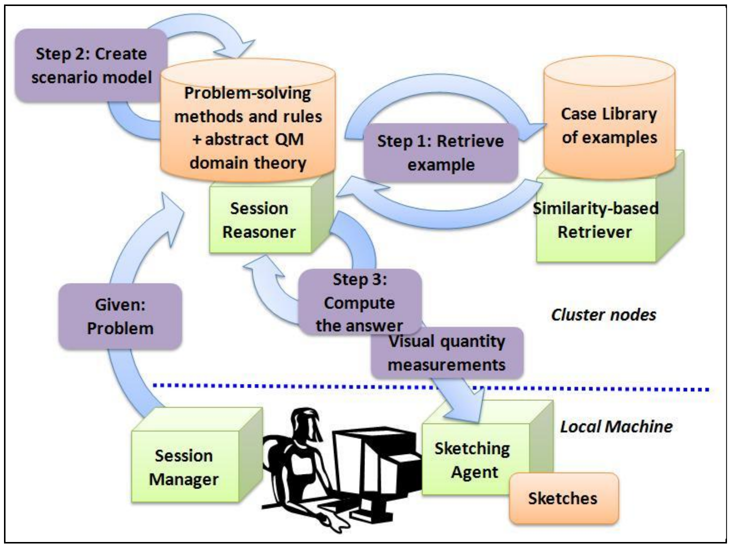
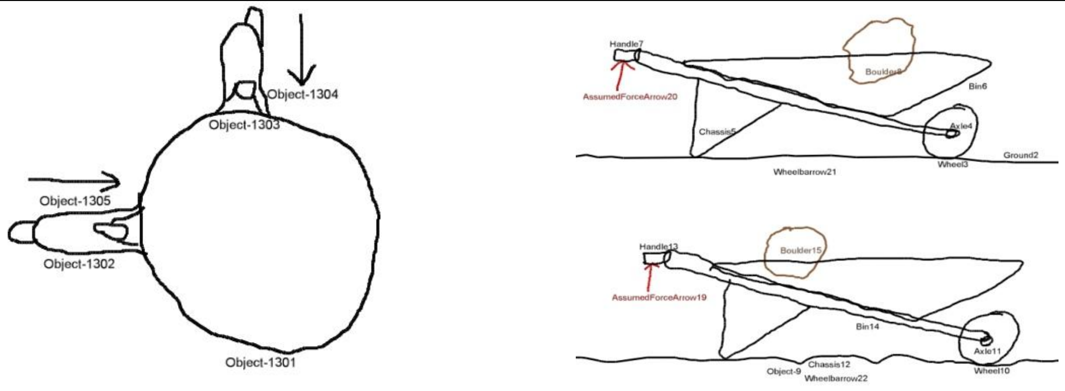

Step 2: Create scenario model
Problem-solving methods and rules + abstract QM domain theory
Step 1: Retrieve example
Case Library of examples
Session Reasoner
Similarity-based Retriever
Step 3: Compute the answer
Given: Problem
Cluster nodes
Visual quantity measurements
Local Machine
Session Manager
Sketching Agent
Sketches

Figure 12: Solving BMCT problems on a Companion

The Session Reasoner performs the majority of the reasoning. It relies on the Retriever to find relevant analogs, and the Sketching Agent to perform visual quantity measurements. In addition to running sKEA, the Sketching Agent also maintains the concept map used in example entry.

As previously discussed, there are two types of questions on the BMCT: outcome questions and comparative analysis (CA) questions. Figure 13 contains an example of each type and the associated query stored with each problem. The predicate of the query indicates the type of problem. Because the ball problem is a single situation, it is sketched on a single layer. The predicate `solveQMOneSketchProblem` designates that this is an outcome question. The first argument is the query for the outcome question, "what is the net force on the ball?" In CA problems with comparisons between scenarios, the sketch consists of two layers, one for each scenario (e.g., one layer for each wheelbarrow). The predicate of the query in the wheelbarrow problem, `solveCAProblem`, indicates that this is a comparative analysis problem. The first argument is the context containing the facts representing the sketch. The next two arguments are the objects being compared, in the case the wheelbarrows. The next argument is the quantity being compared between these situations, the force applied to the handle of the top wheelbarrow. The last argument is the answer, which is the CA value indicating the change in the quantity from one situation to the other. The correct answer to this problem is increasing (i.e., `IncreasedCA)` indicating that the force applied to the handle increases from the top wheelbarrow to the bottom wheelbarrow.



```
(solveQMOneSketchProblem
 (and
  (valueOf
   ((QpQuantityFn XNetForce)
       Ball-1301)
   ?x-dir)
  (valueOf
   ((QpQuantityFn YNetForce)
       Ball-1301)
   ?y-dir)))
```

```
(solveCAProblem
  BMCT-S-1-MEK
  Wheelbarrow-21
  Wheelbarrow-22
  ((QpQuantityFn
           ForceAppliedToSurfaceBy)
     (ContactSurfaceFn Handle-7
          (ContactObjectFn Handle-7))
     (ContactObjectFn Handle-7))
  ?value)
```

Figure 13: BMCT problems and predicate calculus queries - "Which direction will the ball travel?" (Outcome Problem) and "Which wheelbarrow is easier to carry?" (CA Problem)

4.5.1 Retrieve Analogous Example

The first step of problem-solving is retrieving an analogous example. The Companion does this for each layer in the problem sketch. To retrieve a relevant example, the Retriever uses MAC/FAC to generate a reminding for the situation depicted by the layer. MAC/FAC determines the most similar example from its case library using the situation with the low level visual properties removed (i.e. visual groupings, glyph orientations, and relative sizes) as the probe. For outcome problems, the first retrieval is used. For CA questions, the retrieval must also contain candidate inferences that causally constrain the goal quantity. For the wheelbarrow problem, the candidate inferences must include qualitative proportionalities constraining the force applied to `Handle-7`. This is a useful heuristic, because without these candidate inferences, the Companion will not be able to construct a useful scenario model for solving the problem. Should the first retrieval prove unsatisfactory, a second retrieval is performed with the unsatisfactory case removed from the case library. If that, too, fails, the low level visual properties are added back into the probe and up to two more retrieval attempts are made. The multiple retrievals enable different aspects of the problem scenario (i.e. the conceptual and visual aspects) to guide the retrieval and mapping process.

4.5.2 Perform Analogical Model Formulation

As described in Section 4.3, analogical model formulation creates a scenario model for the problem, consisting of causal models, qualitative mechanics abstractions and relationships, and information regarding the measurement of visual quantities. Consider for example the ball problem. Analogical model formulation infers that the ball and the two people are instances of the collection `RigidObject`, and the ball `touches-directly` each of the people. These facts allow the Companion to formulate a qualitative mechanics model of the problem using its domain theory and a standard model formulation algorithm. This scenario model consists of two model fragments: `XForceTransfer` and `YForceTransfer`. Turning to the wheelbarrow problem, the results of analogical model formulation for each situation are described in detail in Section 4.3. For this particular problem, the qualitative proportionality between the force applied at the handle and the distance between the rock and the center of rotation is the crucial aspect of each resulting scenario model.

4.5.3 Solving Outcome Problems

Solving an outcome problem involves standard qualitative reasoning. For example, in the ball problem, the Companion calculates the net force, down and to the right, from the consequences of the force transfer model fragments.

4.5.4 Solving Comparative analysis Problems

Solving a comparative analysis problem requires the additional step of constructing an analogical reference frame. The Companion uses the causal model and analogical reference frame to ascertain the relevant visual properties to compare. These visual properties are measured and their numerical values compared to produce CA values for the causally independent parameters. These CA values are then propagated through the causal model to derive a CA value for the query parameter. These problems can be quite difficult: One problem on the BMCT involves comparisons between aspects of three situations. In this case, the Companion sets up three reference frames, one for each pair of situations, and carries out the same analysis for each reference frame in order to derive the correct answer

4.5.4.1 Creating the Analogical Reference Frame

As described in Section 4.4, the Companion creates an analogical reference frame for the problem. It uses SME to create a mapping between the two scenarios, or the scenario with itself. In the latter case, the mapping is constrained by requiring a correspondence between the aspects of the scenarios being compared by the query. The resulting correspondences indicate how quantities should be compared. Specifically, the CA value for a parameter refers to a comparison between its value and that of the parameter corresponding to it in the mapping. In the wheelbarrow problem, the Companion sets up the analogical reference frame by mapping the top wheelbarrow onto the bottom wheelbarrow.

4.5.4.2 Backward Chaining through the Causal Model

Once the reference frame is set up, comparative analysis proceeds by chaining backward from the sought quantity through the causal model. Non-visual quantities that are either not causally constrained by other parameters or are not known to be different are assumed to be the same across the scenarios, i.e., a CA value of unchanged. For example, in the wheelbarrow problem in Figure 13, the Companion assumes that the rocks apply the same amount of force on the wheelbarrow's bins because force applied is not a visual quantity. CA values for visual quantities are computed by comparing measurements between the corresponding quantities in the sketch. In the wheelbarrow problem, the Companion determines that the force applied on the handle is proportional to the distance from the rock to the wheelbarrow's origin of rotation. Therefore, since the CA value for distance from the surface of the bin touching the rock to the wheelbarrow's origin of rotation increases, the force applied on the handle also increases from the top wheelbarrow to the bottom wheelbarrow.

4.5.4.3 Measuring Visual Quantities

The Companion uses the Sketching Agent to measure visual quantities. In this work, there are three types of visual quantities: `DistanceBetweenSurfaces`, `DistanceToOrigin`, and example-specific visual quantities defined by a linear annotation. The `DistanceBetweenSurfaces` quantity represents the distance between two surfaces. sKEA reduces each surface to a point by averaging the X and Y coordinates of the surface's endpoints and computes the distance between these points. The `DistanceToOrigin` quantity represents the distance from a surface to the center of rotation of the object the surface is on. The center of rotation is determined by two methods, each of which depends upon the results of analogical model formulation. First, the object may participate in the conceptual relationship, `canPivotAround`, with another object. In this case, the surface between these objects would be the center of rotation. Second, the object may be part of a conceptual glyph group. In this case, if there is another glyph also in the group who is an instance of the collection `Fulcrum`, the centroid of this glyph is the center of rotation. Example-specific visual quantities created by annotations are measured from the anchor points transferred via candidate inferences. Linear annotations measure the distance between anchor points, while X-coordinate and Y-coordinate annotations measure distance between the anchor points along the appropriate axis.

In the wheelbarrow problem, the Companion uses the causal model to determine that the distance to the origin of rotation from the surface on the wheelbarrow's bin, defined by the contact with the rock, is a relevant visual quantity. sKEA uses the ink of the bin and rock glyphs to determine the line segment of the bin which represents the surface. sKEA averages the endpoints of this line segment to calculate one end of the distance measurement. The center of rotation of this wheelbarrow depends upon the qualitative mechanics abstraction `Fulcrum`. Because the wheelbarrow is a group glyph and the axle is an instance of the `Fulcrum` collection, the Companion selects the centroid of the axle as the center of rotation for the wheelbarrow. Next, sKEA computes the distance between these points and provides the Session Reasoner with the results. The same process occurs for the other wheelbarrow. Recall that not only was the relevance of this quantity established via analogy, its measurement also depended upon the qualitative mechanics abstraction inferred during analogical model formulation.

4.6 Experiment

The purpose of this experiment was to determine how well a Companion performs on problems from the Bennett Mechanical Comprehension Test using analogical model formulation. A subset of the test was chosen to minimize domain encoding efforts and to focus on what could be achieved via analogical model formulation. I selected 13 of the 68 problems on the BMCT, focusing on problems involving net force, revolution rate, stability, and smoothness of ride. Given that all of the problems include real world objects, this subset is an example of the everyday breath problem. By including problems not handled by pre-existing qualitative mechanics theories, this subset is useful for evaluating the domain breadth problem as well.

Eleven problems involved differential qualitative analyses, six of which involved phenomena not covered by the Companion's qualitative mechanics domain theory. The other two problems were outcome problems. The experiment and analyses below provide evidence concerning three questions:

1. Can a Companion using analogical model formulation solve Bennett Mechanical Comprehension Test questions?
2. How does a Companion perform when the number of explanations increases? This is important for assessing how well learning by adding examples scales.
3. How well do the retrieval and mapping mechanisms perform? That is, when there are errors, how often are these mechanisms the cause, as opposed to some other part of the system?

4.6.1 Method

A central assumption concerning analogical model formulation is that intelligent agents have access to explanations of situations similar to the current problem. These explanations may have been given to agent or just inferred from experience. To model these explanations, a list of 18 example situations was created. 15 examples were intended to be good analogs for specific test questions, with each problem from the BMCT subset having at least one relevant example. The other three examples provided additional distracters. Three graduate students, with varying degrees of familiarity with sKEA, served as knowledge enterers (KEs) to create the examples. Each example was described to the KEs by a short English phrase, e.g. "a tricycle". They were instructed to draw each example in two dimensions, avoiding perspective projections. They were also instructed to break objects into separate glyphs whenever they were going to be referring to a named part in describing how that example worked. For example, because a wheelbarrow is lifted at its handle and rotates around its axle, the handle and axle would be separate glyphs. While drawing, they used sKEA's conceptual labeling interface to apply appropriate domain abstractions from a list provided to them. They also used sKEA's visual/conceptual relationship interface to select relevant relationships. KEs were also given a list of physical quantities that were relevant in this subset of the BMCT (i.e., smoothness of ride, revolution rate, and stability). When one of these quantities was relevant, they were instructed to include it in their example, and explain what other properties of the system that it depends on, using annotation glyphs as necessary to define physical quantities in terms of visual measurements. To evaluate their progress, they ran a standard qualitative mechanics model formulation system to derive model fragment instances. If there were missing or inappropriate instances of model fragments, they were encouraged to modify their sketch until they were satisfied with the model fragments generated. For example, in a sketch depicting two meshed gears where one of the gears is rotating (as indicated by an annotation), the KE would know that something was wrong if there was no mention of torque transfer in the active model fragment list. Once finished, each example sketch was stored in a case library for that particular KE.

The 13 problems were drawn by a fourth graduate student, an experienced sKEA user. The problem sketches did not include any qualitative mechanics structural abstractions or conceptual relationships. Thus, all the problems required analogical model formulation to arrive at the correct answer. Every answer the Companion provided had to be justified by its own model of the scenario. No guessing was allowed.

The problems were presented to the Companion in a series of seven trials. In the first three trials, the Companion had access to each KE's case library individually. In the next three trials, the Companion was given each pair wise combination of case libraries, and in the final trial, the Companion had all the examples from the three cases libraries.

While the 13 problems represent a subset of the test, they still cover a broad range of situations. The predicate calculus generated for the sketches of these problems contains 164 entities of 84 different types. These entities are related to each other by 37 unique relations. The problem representations contain on average 182 facts, with the largest and smallest problems having 397 and 40 facts respectively. Because annotations and conceptual relationships were added to the examples, the example representations are slightly larger, ranging from 52 to 467 facts with an average of 201. They include 100 conceptual types and 212 unique relations.

4.6.2 Results

Library	# of Correct Retrievals (out of 13)	# of Correct Answers (out of 13)
KE1	7	6 (p < .24)
KE2	10	10 (p < .001)
KE3	5	2 (p < .96)
KE1+2	11	9 (p < .008)
KE1+3	9	6 (p < .24)
KE2+3	10	10 (p < .001)
KE1+2+3	12	10 (p < .001)

Table 3: Problem-solving results by case library

A summary of the results appears in Table 3. The correct retrieval column lists the number of times the system retrieved one of the appropriate analogues for the problem sketch. Given the large number of distracters, 16 or 17 depending on the problem, all of the retrieval results are statistically significant ($p < 0.001$). The correct answer column lists the number of times the system provided the correct answer. In four of the seven trials, the Companion produced the correct answer on a statistically significant number of problems ($P < 0.05$). Every problem was solved in at least one of the trials. KE2 had the most experience with sKEA, leading to similar representations to the problems, and KE3 had the least experience, providing some serious variability. Table 3 demonstrates that Companions can indeed solve BMCT problems via analogical model formulation: 77% correct, under the best conditions. Furthermore, as the number of available examples grows, the Companion's performance improves. Notice that every combination of KEs except for KE2+KE3 provides an improvement in correct retrievals. This is important because it means greater breadth can be achieved to some degree by increasing the system's experience. A close inspection of the results reveals that in each of the combination trials, example sketches from at least two of the case libraries were used to formulate correct answers. This indicates that the methods have some degree of robustness across examples entered by multiple people.

Question Type (number)	# Answers Produced	# Correct Retrievals	# Answers Correct
BMCT Questions (91)	71 (78%)	65 (71%)	53 (58%)
Outcome Questions (14)	14 (100%)	10 (71%)	10 (71%)
CA Questions (77)	57 (74%)	55 (71%)	43 (56%)
Net Force Questions (35)	24 (68%)	23 (65%)	15 (42%)
Other Questions (42)	33 (78%)	32 (76%)	28 (66%)

Table 4: Companion's performance by question type

Table 4 looks at the same data, but broken down by question type, to get a better understanding of the Companion's performance. Overall, the Companion answered correctly 58% of the time across all problem/memory conditions. However, since errors in the fixed components of the system have been ruled out via by-hand analysis, the difference between the number of answers produced (78%) and correct retrievals (71%) suggests that there are occasional problems in mapping or in using candidate inferences. This was not the case for the outcome questions – if the retrieval was correct, the Companion derived the correct result. Recall in outcome questions, the Companion always uses first analog retrieved. Finding a criterion for testing the analogically derived model would make a difference here. This problem was worse in the CA problems, despite the use of a relevance heuristic to filter retrievals. In 20 of the CA questions (26%), the Companion was unable to find an example that causally constrained the sought quantity, and thus was unable to produce an answer. Also, when the Companion retrieved a relevant example, it still missed 12 out of 55 problems (22%). These failures are examined more closely below.

Analogical model formulation's performance without a complete domain theory is evaluated by distinguishing between different types of CA questions. In net force questions, the Companion transferred causal models from examples that were generated from the qualitative mechanics domain theory. The Companion performed slightly worse on these problems averaged across all memory conditions. It retrieved a relevant example 65% of the time and answered the problem correctly on 42% of the problems across all memory conditions. The other questions concern the phenomena not covered in the Companion's qualitative mechanics domain theory: stability, revolution rate, and ride smoothness. The causal models required to solve these problems were defined in an example-specific manner via linear annotations. The Companion answered 66% of these problems correctly across all memory conditions, supporting the hypothesis that analogical model formulation is a promising approach to addressing the domain breadth problem posed by the BMCT.

To better evaluate the retrieval heuristic used in CA problems, Table 5 organizes the CA results based upon which retrieval strategy produced the answer. While the majority of the answers were based upon the first retrieval, 24 answers across all memory conditions required additional retrievals. Of these, the Companion retrieved an appropriate analogue 22 times and answered the question correctly 15 times. These results support the hypothesis that the retrieval method is useful in for solving BMCT problems from examples.

CA Retrieval Strategy Type	# Uses	# Correct Retrieval	# Correct Answer
1st retrieval: No spatial relations, full case library	33	32	28
2nd retrieval: No spatial relations, case library – 1st retrieval	5	5	3
3rd retrieval: including spatial relations, full case library	11	11	8
4th retrieval: including spatial relations, case library – 3rd retrieval	8	6	4
Total:	57	54	43

Table 5: CA results by retrieval number and strategy

4.6.3 Analysis of failures

It is useful to understand why systems fail. First, failures in outcome problems are analyzed followed by a discussion of the failures in CA problems.

The four failures on outcome problems occurred because the Companion failed to retrieve the correct example from memory. The Companion confused a gear rotating inside another gear with two gears rotating side by side, due to annotation glyph placement. This results in incorrect qualitative mechanics abstractions and relationships being assumed in the problem, which in turn leads to the Companion constructing an incorrect scenario model. Currently, the problem solving method does not have a method for evaluating the retrieval on outcome problems. People seem to handle this problem by recognizing contradictions in their reasoning. Recognizing contradictions is difficult in analogical model formulation because the mapping with the example is assumed to provide the correct structural abstraction and conceptual relationships. A promising direction for overcoming this problem is to explore ways of combining reasoning from multiple examples.

The failures on CA problems occurred during both the retrieval and mapping stages of the algorithm. As noted above, the retrieval must yield a mapping which causally constrains the quantity in question. For 20 of the 77 problem/memory condition pairs, such an example could not be found. Frequently, this was because the KEs sketched the example at a different level of abstraction than the problem. Even when an example is found that constrains the goal quantity, there are still two failure modes for mapping. First, the causal model may include surfaces or objects which do not exist in the problem sketch. Figure 14 illustrates this problem. The problem sketch contains glyphs for the ground and the axle, but the example does not. Furthermore, the chassis in the problem sketch is conceptually labeled as a leg in the example. While SME handles incomplete matches, significant differences in depiction, such as divergence in number of glyphs, can cause mapping failures. In this case, the bin in the example maps to the chassis in the problem. The candidate inferences postulate surfaces and/or glyphs that do not exist (e.g, the surface between the boulder and the chassis), which sKEA is unable to reason about. The Companion fails whenever a candidate inference for a necessary causal relationship includes references to either surfaces or glyphs that do not exist.

The second kind of mapping failures in CA problems are errors in measurements of user defined visual quantities. When measuring the wheel diameter in the problem scenario of Figure 14, if the wheel in the example was mapped to the axle in the problem, then the visual measurement for the wheel diameter would be the width of the axle in the problem. These mapping failures lead the Companion to produce incorrect answers.

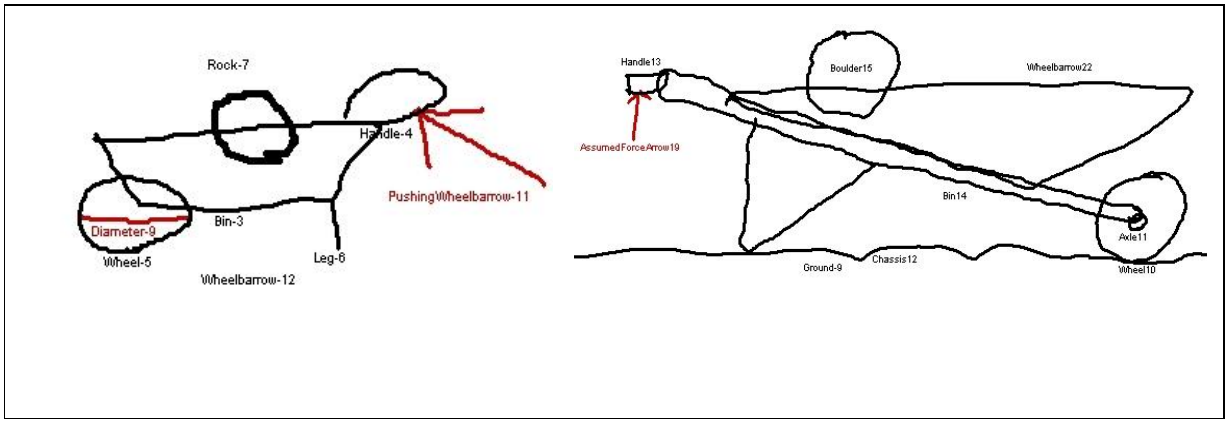


Figure 14: Example wheelbarrow (left) and problem wheelbarrow (right)

People seem to have several methods for dealing with such problems. First, they try other examples, going back to memory to find an example that is more productive. A simple version of this is already implemented in the retrieval method. Also, people use *rerepresentation* (Yan *et al.* 2003) to bring the base and target into closer alignment. Knowledge about depiction seems crucial: If two sketches are misaligned, simplifying the more complex one, or postulating new glyphs in the simplified one, seems to be a promising strategy.

As illustrated in Table 5, the Companion performed slightly worse on net force CA problems which relied on causal models generated by its qualitative mechanics domain theory than on problems which relied upon example-specific causal models. These causal models from the qualitative mechanics domain theory were larger which increases the likelihood of mapping failures. The net force causal models contained more entities than the example-specific causal models. This is evident in the example wheelbarrow in Figure 14. The causal model concerning the force applied at the handle references the following entities: `Handle-4`, `(ContactObjectFn Object-4)`, `Rock-7`, `Bin-3`, and `Wheel-5`. All five of these objects must map appropriately to the problem to create the necessary scenario model. On the other hand, the example-specific explanation for the wheelbarrow's ride smoothness references only two entities: `Wheel-5` and `Wheelbarrow-12`. While the causal models concerning net force CA problems are more complex, the Companion does have access to a qualitative mechanics domain theory. One avenue for future work is to develop a theory which uses the first-principles explanation to verify and repair these analogically inferred causal models.

4.7 Summary and Future Directions

In this chapter, I have argued that qualitative reasoning combined with analogical processing is a promising avenue for addressing the problems of domain breadth and everyday breadth that must be solved for human-like reasoning. The Companion's performance on a subset of the Bennett Mechanical Comprehension Test provides important evidence for this claim. Analogical model formulation enables a Companion to build scenario models over a broad range of input descriptions, even with an incomplete domain theory. Sketch annotations provide a means for defining visual quantities in examples, which can be used in example-specific causal explanations and applied via analogy to new situations. This extends the Companion's reach by enabling it to reason about phenomena for which it does not have a corresponding domain theory. Analogical reference frames extend the scope of comparative analysis to include the types of problems found on the BMCT.

This work opens a number of promising directions for future research. Within the analogical model formulation framework, there are two important places for improvement: learning evaluation rules for analogical inferences and representation to improve mappings. As indicated in the results section, more methods for evaluating analogical inferences would have benefitted analogical model formulation on outcome problems. Specifically, as an agent accumulates multiple examples of a scenario, there is the potential to abstract commonalities between them. For example, regarding analogical inferences concerning the qualitative mechanics abstractions and relationships, one could observe that qualitative mechanics abstraction `touches-directly` only occurs when the glyphs representing the two objects spatially intersect. Therefore, when the system encounters an analogical inference suggesting this relationship for two objects, it could reject the inference. This knowledge could also be a trigger for rerepresentation (Yan *et al.* 2003). Similar relationships between objects result in the model fragments for net force CA problems, such as the problem wheelbarrow in Figure 14. Recall that this problem failed because the mapping resulted in a causal model which included a non-existent surface between the boulder and the chassis of the wheelbarrow. One strategy would be to merge glyphs in the problem until the proper spatial relationship held for the analogically inferred qualitative mechanics relationship. For example, merging the chassis with the bin would create a spatial intersection between with the boulder. This intersection would then allow the Companion to reason inferred causal model. This repair would allow the Companion to successfully reason about a problem scenario where analogical model formulation initially produced a flawed model.

Another direction for future research on analogical model formulation is to explore how new abstract domain theories are learned from examples. In this work, the majority of the

Companion’s domain knowledge is in either examples or the qualitative mechanics domain theory. A common hypothesis in cognitive science is that people are able to generalize from examples into a more abstract domain theory. Taking the examples from this work, the Companion could create generalizations across causal models concerning the same quantities. This could lead to rules such as “things with wider bases are more stable.” There are still many open questions regarding how this knowledge gets formed and how it is applied in future problem-solving episodes.

An important aspect of this evaluation is using the multi-modal reasoning requirement of the BMCT. While sketch annotations provided an excellent way to tie visual and conceptual information together in sketches, the linear distance annotations are tied to the reference glyphs by anchor points. While the nine anchor points were sufficient for this work, experience with everyday sketch points out a number of other important areas. Ink intersections, either within an individual or between a pair of glyphs, represent an additional type of reference point. Also, additional research is required to understand the types of annotations which are used in everyday sketching. For example, sketches of simple machines frequently include hashing to indicate solid regions and sketches of military plans include circles indicating regions of importance. An important lesson from this work is that qualitative representations provide a useful bridge between symbolic representations necessary for analogy and complex reasoning and the spatial representation given as input in the problem.

By employing the BMCT to evaluate the system, this work falls within the umbrella of

Psychometric AI (Bringsjord & Schimanski 2003). Psychometric AI holds human intelligence tests as an important progress meter for AI. Building a system that could ace the BMCT would be a substantial achievement. While tests are an important way for measuring progress, how the system accomplishes its performance is of particular interest to the AI and cognitive science community. In this work, analogical processing, comparative analysis, qualitative reasoning and sketch understanding all played integral roles in achieving this result. Continuing in this direction, the next chapter describes how a Companion uses analogical model formulation to solve AP Physics problems.

To overcome the brittleness of AI systems, the robustness and flexibility of the system is very important. In this evaluation, robustness was evaluated by having multiple knowledge enterers provide the Companion with a broad range of inputs. A failure analysis indicated some of the representational differences that are difficult for the Companion to overcome. Specifically, analogical model formulation was insufficient when the situations were depicted at different levels of abstraction. This motivates future work to incorporate rerepesentation strategies (Yan *et al.* 2003) to better align the sketches. Also, the Companion's performance as measured with different combinations of examples provides information concerning the robustness with both good and bad examples. In the next chapter, two other measures of robustness are employed: external evaluation and systematic differences between problems and examples.

5 AP Physics problem-solving with analogical model formulation

This chapter describes how the same method, analogical model formulation, allows a Companion to solve Advanced Placement (AP) Physics problems designed by the Educational Testing Service (ETS). Physics problem-solving is an important domain for moving beyond traditional approaches for model formulation, because, as on the BMCT, the problems are presented as everyday scenarios. While the BMCT included spatial reasoning, physics problems include quantitative as well as qualitative reasoning. Figure 15 provides four examples, illustrating types of problems that the system learns to solve. Natural language understanding is factored out by using predicate calculus problem descriptions. Unlike previous systems such as MECHO (Bundy 1979) or ISSAC (Novak 1977), this translation process leaves everyday concepts in place. That is, balls, buildings, astronauts, boxes, baseball bats, flying, falling, and pulling all appear in the formal problem descriptions[3]. Therefore, one must still overcome the everyday breadth problem to determine the relevant abstractions and assumptions for each physics problem. The generalizations in any commonsense ontology are unlikely to provide much help: cats, coins, and pianos can all be considered as point masses in particular situations, but they are not likely to be closely related in any non-trivial ontology. Furthermore, modeling decisions are contextual. For example, a coin falling off a building can be considered to be a point mass. But the exact same coin spinning on a table cannot be considered a point mass since its shape and size must be considered.

[3] The subset of the ResearchCyc ontology used in this work contains over 30,000 concepts. See research.cyc.com for details.

1. A ball is released from rest from the top of a 200m tall building on Earth and falls to the ground. If air resistance is negligible, which of the following is most nearly equal to the distance the ball falls during the first 4s after it is released? (a) 20m; (b) 40m; (c) 80m; (d) 160m.
2. An astronaut on a planet with no atmosphere throws a baseball bat upward from near ground level with an initial speed of 4.0m/s. If the baseball bat rises to a maximum height of 5.0m, what is the acceleration due to gravity on this planet? (a) 0.8m/s^2; (b) 1.2m/s^2; (c) 1.6m/s^2; (d) 20m/s^2;
3. A box of mass 8kg is at rest on the floor when it is pulled vertically upward by a cord attached to the object. If the tension in the cord is 104N, which of the following describes the motion, if any, of the box? (a) It does not move; (b) It moves upward with constant velocity; (c) It moves upward with increasing velocity but constant acceleration; (d) It moves upward with increasing velocity and increasing acceleration.
4. A block of mass M is released from rest at the top of an inclined plane, which has length L and makes an angle q with the horizontal. Although there is friction between the block and the plane, the block slides with increasing speed. If the block has speed v when it reaches the bottom of the plane, what is the magnitude of the frictional force on the block as it slides? (a) $f = Mg\sin(q)$; (b) $f = Mg\cos(q)$; (c) $f = MgL\sin(q)$- ½Mv^2 ;(d) $f = [MgL\sin(q)$- ½$Mv^2]$/L.

Figure 15: Example AP Physics problems of the four types used in this work

While complex, there is ample evidence that people are able to solve physics problems stated in everyday terms. The problems used throughout this work were generated by the Educational Testing Service, which administers the AP Physics examination in the United States. High school students take the AP Physics exam for college credit. Students' performance on this exam indicates that they do learn to categorize everyday objects in terms of domain abstractions, determine what equations are relevant, infer parameter values from scenarios, and assume default circumstances when necessary. The problems used in this work were generated automatically, from templates. The four problems, one from each problem type, shown in Figure 15 represent roughly 20% of the typical Mechanics portion of the AP Physics examination.

Characterizing how well learned knowledge is transferred to new problems is complex. One way involves identifying different *transfer levels*, each representing a particular type of relationship between a known *source* problem and a novel *target* problem. This research uses an externally-developed set of six transfer levels[4]. Using Problem 1 from Figure 15 as an example of a source problem, the following are representative examples of the six transfer levels:

A ball is released from rest from the top of a 200m tall building on Earth and falls to the ground. If air resistance is negligible, which of the following is most nearly equal to the distance the ball falls during the first 4s after it is released?

1. Parameterization: Target problem has different parameter values, but the qualitative outcome is the same.

   A ball is released from rest from the top of a 500m tall building on Earth and falls to the ground. If air resistance is negligible, which of the following is most nearly equal to the distance the ball falls during the first 3s after it is released?

2. Extrapolation: Target problem has parameter values that are so different that the qualitative outcome changes as well.

   A ball is released from rest from the top of an 80m tall building on Earth and falls to the ground. If air resistance is negligible, which of the following is most nearly equal to the distance the ball falls during the first 5s after it is released?

3. Restructuring: The target problem asks for a different parameter.

   A ball is released from rest from the top of a 200m tall building on Earth and falls to the ground. If air resistance is negligible, how long does it take to fall 80m?

4. Extending: The target problem includes distracting information.

[4] These levels are from a 10-level catalog of transfer tasks used in DARPA's Transfer Learning Program.

A ball with density 5Nm/kg is released from rest from the top of a 100m tall building on Earth and falls to the ground. If air resistance is negligible, which of the following is most nearly equal to the distance the ball falls during the first 4s after it is released?

5. Restyling: The target problem involves different types of everyday objects.

   A crate is dropped off the edge of a 100m cliff on Earth and falls to the ground. If air resistance is negligible, which of the following is most nearly equal to the distance the crate falls during the first 4s after it is released?

6. Composing: The target problem requires combining concepts from two different base problems.

   An astronaut on a planet with no atmosphere throws a ball upward from near ground level with an initial speed of 4.0m/s. The ball rises to a maximum height of 5.0m before returning to the astronaut who then drops the ball from the top of a 100m tall building. If air resistance is negligible, which of the following is most nearly equal to the distance the ball falls during the first 4s after it is released? (Composed the Source Problem with Problem 2 from Figure 15)

The rest of this chapter describes how a Companion using analogical model formulation solves AP Physics style problems, across these six transfer levels. While in the previous chapter example sketches formed the basis of the explanations for analogical model formulation, here I introduce *worked solutions*. These are worked through example problems at the level of detail a student would encounter in a textbook. A central hypothesis in analogical model formulation is that it allows for robust problem-solving in everyday scenarios. One measure of robustness comes directly from the transfer levels themselves. Using worked solutions as explanations to source problems, performance on problems from each transfer level provides a systematic empirical measurement of the ability of adapt explanations to new situations. Additionally, the system was initially evaluated externally by the Educational Testing Service. This hands-off evaluation provides additional evidence regarding the robustness of the system.

This chapter is organized as follows. After a brief summary of the Companions agents employed here, the representations used in this work and how they were created is presented. This is followed by characterizing the model formulation challenges in AP Physics and highlighting how analogy can solve these problems. Then the analogical problem-solving method is described in detail. Next, the results of an experiment administered by ETS are presented in which a Companion using analogical model formulation achieved a 63.8% initial improvement across the six transfer levels. This is followed by a detailed analysis indicating that most problem-solving failures were caused by human errors in the implementation and representations, and not due to analogical model formulation. After addressing these issues, a second experiment was performed in which the Companion achieved an initial improvement of 95.8% averaged across the six transfer levels. A final analysis is presented which evaluates this problem-solving method as a cognitive model. After the experiments, there is a discussion of future directions based upon these results.

5.1 Companions Cognitive Architecture

Once again, analogical model formulation is implemented on the Companions Cognitive Architecture. Because this work involved an external evaluation and did not involve sketching, a slightly different configuration of agents was employed. For these experiments, the following agents were used:

- Session Manager: Provides facilities for user interaction
- Facilitator: Manages sessions and directs communications between agents
- Executive: Monitors the Companion's responsibilities and delegates work to the appropriate agents (e.g. follows scripts describing experiments, records quiz results, checkpoints KBs, etc.)
- Session Reasoner: Performs domain reasoning, in this case physics problem-solving
- Similarity-based Retriever: Provides examples to the Session Reasoner based upon similarity to the current problem.

The primary addition for this work is the Executive. In this configuration, new problems can be given either incrementally through the Session Manager or through a script describing an experiment uploaded to the Executive. This allowed the external evaluation team at the Educational Testing Service to initiate batch experiments. When the executive finished with the experiments, it would send the results back to the Session Manager running on a machine at ETS.

5.2 Representations of Problems and Worked Solutions

When students study for the AP Physics exam, one important way they learn is by doing problem sets. For feedback, students often get *worked solutions*. These step-by-step explanations are always used in textbooks to illustrate problem-solving. Worked solutions are typically incomplete, outlining steps abstractly rather than at the level of detail found in a proof. The system is designed to use worked solutions as examples to formulate models of new problems. As in the BMCT experiment, all of the representations used in this work are in CycL, the predicate calculus language of the ResearchCyc KB (Matuszek *et al.* 2006). These representations factored out the problem of natural language understanding, while maintaining the incomplete nature of worked solutions and the everyday scenarios in the physics problems. The ResearchCyc ontology is especially useful for representing physics problems and worked solutions because it includes over 30,000 distinct types of entities, over 8,000 relationships and functions, and 1.2 million facts constraining them. Thus, the everyday concepts that appear in physics problems like "astronaut" and "dropping" are already defined for us, rather than us generating them specifically for this project.

In addition to the templates used to create the problems in Figure 15, ETS and Cycorp developed templates to generate problems and worked solutions representing each transfer level. Consequently, all the problems and worked solutions in this evaluation were created externally. The representations of the 460 physics problems used in this evaluation contained 4,973 instances from 108 conceptual types and 103 unique relations. When including the worked solutions, the representations include 11,230 instances from 110 types and 144 relations. After describing the worked solution conventions, an example problem and worked solution are presented for concreteness.

5.2.1 Constructing Worked Solutions

Using physics textbooks as a guide, a classification of physics example step types was developed in collaboration with ontologists at Cycorp. While this list is unlikely to be exhaustive, it was sufficient to cover the worked solutions to the problems in this work, as well as the worked solutions to the physics problems in Chapter 7 of this dissertation.

- Categorize the physics problem: Given that examples appear in certain sections of physics textbooks, this is the first step of each worked solution. The results of this step are statements placing the problem in an ontology of physics problem types. (e.g. Recognizing a problem as a linear kinematics problem and an incline plane problem.
- Instantiate a physics equation: These steps were for when an equation was written out in an example. The result was an equation tied directly to entities in the problem. (e.g. the acceleration of a ball is equal to its change in velocity over the duration of an interval.)
- Determine a quantity value from the situation: These steps involve assumptions based upon the situation. (e.g. Assuming a falling ball is accelerating at 10 m/s^2)
- Divide a whole into the sum of its parts: This step involves taking a compound quantity and breaking it into its component pieces. (e.g. Dividing the net force on an object its component forces)
- Solving a physics equation: This step computes the value of a particular quantity using an instantiated equation. (e.g. Given $f=10N$ and $m=1kg$, solve for a in $f=ma$)
- Sanity check a solution: This step checks to make sure a computed answer is consistent with the rest of the scenario. (e.g. If the computed distance a ball falls off a 100m building is 150m, then this step would say that the ball actually fell only 100m)
- Select the appropriate multiple choice answer: This is the final step of each worked

solution. Here the computed answers are used to select the correct multiple choice answer. (e.g. given a computed answer of 79.5m the answer choices a) 40m, b) 60m, c) 80m and d) 100m, the correct answer would be 'c'.)

Worked solutions are sequences of steps. In addition to type information, each step has preconditions and results. The preconditions are a collection of facts from the problem and background knowledge used in the step. These provide the context of the solution step. The results are facts which are known as a result of the step.

5.2.2 Example Problem and Worked Solution

The problem representations are intended to be direct translations into predicate calculus from natural language problem statements, without any abstraction or reasoning. Figure 16 shows a subset of the 37 facts used to represent Problem 2 from Figure 15. The facts in Figure 16 define the planet with no atmosphere, the astronaut throwing the bat and the question asking for the gravitational force of the planet. The average number of facts for each problem is 44.

```
...
(isa Throwing-1 ThrowingAnObject)
(isa Astronaut-1 Astronaut)
(isa Planet-1 Planet)
(isa BaseballBat-1 BaseballBat)
(groundOf Planet-1 Ground-1)
(performedBy Throwing-1 Astronaut-1)
(no-GenQuantRelnFrom in-ImmersedFully Planet-1 Atmosphere)
(eventOccursNear Throwing-1 Ground-1)
(objectThrown Throwing-1 BaseballBat-1)
(querySentenceOfQuery Query-1 (valueOf (AccGravityFn Planet-1) Acc-1))
```

Figure 16: Part of the representation of Problem 2 (simplified for readability)

The worked solutions are predicate calculus representations of the example problems found in textbooks. They are not deductive proofs, nor problem-solving traces of the operations of the

Companion. They leave out many steps and characterize problem-solving operations in very general ways. Here is an English rendering of the worked solution for Problem 2 from Figure 15:

1. Categorize the physics problem: The problem is a distance-velocity problem under constant acceleration
2. Instantiate a physics equation: Instantiate the distance-velocity equation specific to the quantities of this problem (e.g. the bat and the upward motion event) ($V_f^2 = V_i^2 + 2ad$)
3. Determine a quantity value from the situation: Given the projectile motion of the bat, lack of atmosphere, and the maximum altitude of bat, infer that the acceleration of the bat is equal to the acceleration due to the gravity of the planet ($a = g$), the distance the bat travels during the upward motion event ($d = 5m$) and that the bat is not moving at the maximum height ($V_f = 0m/s$)
4. Solving a physics equation: Use the assumed values and the given parameters to solve the equation for the acceleration due to gravity ($g = -1.6m/s^2$)
5. Sanity check a solution: Determine if there is a relevant boundary condition, i.e., ascertain that the answer is consistent the scenario. ($g = -1.6m/s^2$)
6. Select the appropriate multiple choice answer: Select the appropriate multiple choice answer ("c")

The predicate calculus version of this worked solution consists of 104 facts.

```
(stepType Step3 DeterminingValueFromContext)
(stepUses Step3 (isa Throwing-1 ThrowingAnObject))
(stepUses Step3 (occursNear Throwing-1 Ground-1))
(stepUses Step3
      (no-GenQuantRelnFrom in-ImmersedFully Planet-1 Atmosphere))
(stepUses Step3 (objectMoving Upward-1 BaseballBat-1))
(stepUses Step3 (direction Upward-1 Up-Directly))
...
(stepResult Step3
   (valueOf
      (AtFn ((QPQuantityFn Speed) BaseballBat-1) (EndFn Upward-1))
      (MetersPerSecond 0)))
(stepResult Step3
   (valueOf ((QPQuantityFn DistanceTravelled) BaseballBat-1 Upward-1)
      (Meter 3)))
(stepResult Step3
   (valueOf (AtFn ((QPQuantityFn Acceleration) BaseballBat-1) Upward-1)
      ((QPQuantityFn AccelerationDueToGravity) Planet-1)))
```

Figure 17: Problem 2 worked solution Step 3 (simplified for readability)

Figure 17 shows part of the representation for Step 3. The first fact indicates that this is determining value from the situation step. The `stepUses` statements give the preconditions for the step. The subset of `stepUses` statements displayed here state that there is no atmosphere on the planet, the throwing event occurs near the ground and that the bat is the object moving in the upward movement event. The last three facts contain the results of this step, which are values for specific parameters: the speed of the bat at the end of the upward movement event, the distance that bat travels during this event and the bat's acceleration during this event. Figure 16 and Figure 17 as well as the rest of the figures in this section use simplified entity, predicate and function names to improve the readability of the predicate calculus. The complete representations for the problem and worked solution of Problem 2 appear in Appendix A. The average number of facts across all the worked solutions is 163.

5.3 Analogical Model Formulation for AP Physics

Solving physics problems typically requires four types of modeling decisions: relevance reasoning, quantity value assumptions, default circumstances, and modeling abstractions. This section describes each of these in turn and how analogical model formulation uses worked solutions to make modeling decisions in new problems, without a complete domain theory.

Relevance reasoning in physics problem-solving determines which equations are applicable for a given situation. Even in a relatively constrained domain like AP Physics, the number of potentially relevant equations can be quite large, due to specialized forms. For example, while solving Problem 2, it would be a mistake for a system to consider magnetic forces on the baseball bat. Efficient problem-solvers must first determine which equations are relevant. Analogical model formulation uses the insight that similar problems are likely to involve similar equations. All the equations for physics phenomena applied to a problem are found by searching the candidate inferences produced by the analogy between the new problem and worked solution(s).

Quantity value assumptions occur when the problem-solver infers a parameter value from the scenario. For instance, in Problem 2, the problem-solver must recognize that the distance the baseball bat travels is 5m and that its velocity at the end of the upward motion event is 0m/s. Neither of these facts is given explicitly in the problem. While the velocity at the end of the upward motion event could be derived via calculus or a qualitative model, the distance the bat travels is necessarily an approximation, because the scenario description states that the throwing event occurs “near ground level” and the maximum altitude of the bat is 5m. Analogical model formulation uses candidate inferences to suggest quantity values via analogy.

Physics problems frequently require problem-solvers to assume certain circumstances by default. The most common of these in AP Physics is to assume that events happen on Earth and are subject to Earth's gravity. For example, Problem 3, the lifting box problem, requires this assumption to determine the net force on the box. Again, analogical model formulation relies on candidate inferences to find such default circumstances.

The last type of modeling decision involves categorizing everyday objects as abstractions. When reasoning with a domain theory defined in abstract terms, it is necessary to move from the everyday objects and events to this abstract vocabulary. This is another form of relevance reasoning, because abstractions are a way of framing the problem in terms of what phenomena should be considered. Given the problem of a ball falling off the building, a problem-solver would likely abstract the ball into a point mass and not an electrical particle, thus, pruning the search space to the appropriate equations and relevant assumptions. As indicated above, the relevant equations and assumptions are suggested via analogy. Therefore abstraction modeling decisions are implicit in the other modeling decisions made by analogical model formulation

5.3.1 Example of Modeling Decisions via Analogy

This section shows how a Companion uses analogical model formulation to make these modeling decisions for the following restyling variation of Problem 2 from Figure 15:

> "A physicist on an asteroid with no atmosphere throws a spoon upward from near ground level with an initial speed of 4.0 m/s. If the spoon rises to a maximum height of 5.0 m, what is the acceleration due to gravity on this asteroid? (a) 0.8 m/s^2; (b) 1.2 m/s^2;

(c) 1.6 m/s^2; (d) 20 m/s^2"

First, the Similarity-based Retriever, using MAC/FAC, provides the worked solution to Problem 2 (outlined in Section 2.1) as a reminding. Then, the Session Reasoner uses SME to create an analogy with this reminding as the base and the new problem as the target. The most relevant correspondences from the best mapping are summarized in Table 6. Recall that candidate inferences are expressions from the base (here, the worked solution) that are conjectured to hold in the target, by virtue of the mapping's correspondences. A number of these are `stepUses` or `stepResult` statements representing worked solution steps and their contexts. These suggest modeling decisions applicable for the problem. Analogical model formulation draws upon these candidate inferences to incrementally build a scenario model for the problem.

Worked Solution Item	Problem Scenario Item
`Planet-1`	`Asteroid-5`
`BaseballBat-1`	`Spoon-5`
`Astronaut-1`	`Physicist-5`
`Upward-1`	`Upward-5`
`(StartFn Upward-1)`	`(StartFn Upward-5)`
`(EndFn Upward-1)`	`(EndFn Upward-5)`
...	...

Table 6: Correspondences between the worked solution and the problem scenario

As an example of relevance reasoning, Step 2 of the worked solution contains the equation $V_f^2 = V_i^2 + 2ad$ in terms of the baseball bat and its upward movement event. The candidate inferences generated from this step include a corresponding equation with the quantities V_f, V_i, a, and d in terms of the problem entities: `Spoon-5` and `Upward-5`. Analogical model formulation includes this equation in the model for the problem.

Analogical model formulation handles decisions concerning quantity value assumptions and default circumstances in the same manner. The determining value from situation worked solution step type indicates one of these assumptions. As a result of the analogical mapping, these steps appear as candidate inferences in the problem. These candidate inferences suggest quantity value and default circumstance assumptions in the problem. In this mapping, the candidate inferences suggest that the velocity of the spoon at the end of its upward movement is zero, the distance the spoon travels during the upward movement event is 3 meters, and the acceleration of spoon during the upward movement event is the acceleration due to gravity of the asteroid. While default circumstances do not occur in this example, they are represented in the same way in worked solutions and handled by analogical model formulation in the same manner.

To provide a stringent test of analogical model formulation from examples, the only modeling knowledge the system has concerns heuristics for evaluating candidate inferences. Importantly, the system has no general knowledge of physics equations, quantity values, or default circumstances. Without a reminding, it cannot solve any problems. This resulted in a useful simplification: The system does not explicitly categorize everyday objects in terms of abstractions. Making such abstractions explicit is useful only when there is abstract domain knowledge that will trigger on it. Such information is implicit in the choice of equations, quantity value assumptions, and default circumstances. As the experiments below indicate, this works well when the analogous problems are sufficiently similar. It is likely that explicit categorization and abstract domain theories are required for more distant transfer. Consequently, an important future direction involves learning such knowledge. Preliminary results are reported in (Klenk *et al.* 2008).

5.4 Problem-solving Algorithm

This section describes the algorithm for solving AP Physics style problems using analogical model formulation. Figure 18 outlines the algorithm. After describing each step in detail, the implementation in the Companion cognitive architecture is presented along with an example.

```
Given: a problem, $P$, and a case library, $C$, of worked solutions, $\{ws_1 \ldots ws_n\}$
1.  Retrieve analog $ws_i$ using MAC/FAC with probe: $P$ and case library: $C$
    1.1.  While there is unmapped event structure,
          1.1.1. retrieve analog $ws_j$ using MAC/FAC with probe: ($P$ – {facts covered by the currently mapped
                 structure}) and case library: ($C$ – {already retrieved worked solutions})
2.  Analyze $P$ to determine the sought after quantity, $q$, and problem type, $t$.
3.  Using analogical model formulation, solve for $q$ by one of the following methods, using $t$ to determine when
    the answer is appropriate:
    3.1.  If the value of $q$ is given in $P$, use that value.
    3.2.  If the value of $q$ is mentioned in a candidate inference, use that candidate inference.
    3.3.  Search candidate inferences generated by analogs for an equation, $e$, mentioning $q$
          3.3.1. Recursively solve for each of the other quantities, $q_1 \ldots q_i$, mentioned in $e$
          3.3.2. Solve $e$ for $q$
4.  If boundary condition check exists in candidate inferences,
    4.1.  Compare mapped values in $P$ to determine correct value for $q$
5.  Use the computed value for $q$ to select the multiple choice answer for $P$
```

Figure 18: Solving AP Physics problems via analogical model formulation

5.4.1 Step 1: Retrieve analogous worked solutions

The process of solving a physics problem begins by generating an analogy with one or more relevant worked solutions. With the problem as the probe, MAC/FAC is used to retrieve a relevant example from a case library of worked solutions. The mapping between the worked solution (as the base) and the problem (as the target) is evaluated for adequacy by the loop in Step 1.1. Fundamentally, physics problems are about events. The *event structure* of a problem consists of the events that occur in it. If the analogy does not map all of the event structure, additional analogues must be retrieved. Otherwise, there would be no knowledge from which to formulate a model for the unmapped events[5]. For each iteration in Step 1.1, the already-matched parts of the probe are removed, so that retrievals are focused only on cases that are similar to the unmatched aspects of the problem. This was essential for handling the Composing transfer condition, since multiple source analogues are needed to solve a single problem. For example, the following retrievals would be made while solving the Composing example in Section 5:

> *An astronaut on a planet with no atmosphere throws a ball upward from near ground level with an initial speed of 4.0m/s. The ball rises to a maximum height of 5.0m before returning to the astronaut who then drops the ball from the top of a 100m tall building. If air resistance is negligible, which of the following is most nearly equal to the distance the ball falls during the first 4s after it is released?*

A possible first retrieval could be the worked solution for Problem 1 of Figure 15. The resulting mapping would include the release and falling events, but not the throwing and projectile motion events. Because there are unmapped events, another retrieval would be made, with only the facts pertaining to the throwing and projectile motion events as the probe to MAC/FAC. This may result in the retrieval of the worked solution for Problem 2 from Figure 15. The resulting mapping would include the throwing and projectile motion events. Because the entire event structure of the problem is accounted by the two analogies to the worked solutions, this step is complete.

5.4.2 Step 2: Problem analysis

Solving most physics problems eventually boils down to finding the value for some quantity.

[5] The distracters added in the Extending transfer condition never included events, only quantities and entities.

But which quantity, and what form of description is appropriate for the value, must be ascertained by analyzing the problem. There are several different broad types of problems on the AP Physics exam. The subset of the exam used in this work contains the following problem types:

- Numeric value problems: Determining the numeric value of a specific parameter

  A ball is released from rest from the top of a 200m tall building on Earth and falls to the ground. If air resistance is negligible, which of the following is most nearly equal to the distance the ball falls during the first 4s after it is released? (a) 20m; (b) 40m; (c) 80m; (d) 160m.

- Symbolic value problems: Determining the symbolic value of a specific parameter

  A block of mass M is released from rest at the top of an inclined plane, which has length L and makes an angle q with the horizontal. Although there is friction between the block and the plane, the block slides with increasing speed. If the block has speed v when it reaches the bottom of the plane, what is the magnitude of the frictional force on the block as it slides? (a) f = Mgsin(q); (b) f = Mgcos(q); (c) f = MgLsin(q)- ½Mv^2 ;(d) f = [MgLsin(q)- ½Mv^2]/L.

- State elaboration problems: Determining which parameter value will produce a described outcome

  Which of the following tensions is required to move a box of mass 8kg from rest on the floor upward with constant acceleration when it is pulled vertically upward by a cord attached to the box? (a) 40N; (b) 60N; (c) 70N; (d) 120N.

- Qualitative behavior problems: Determining the qualitative outcome of a situation

  A box of mass 8kg is at rest on the floor when it is pulled vertically upward by a cord attached to the box. If the tension in the cord is 104N, which of the following describes the motion, if any, of the box? (a) It does not move; (b) It moves upward with constant velocity; (c) It moves upward with increasing velocity but constant acceleration; (d) It moves upward with increasing velocity and increasing acceleration.

This step identifies the problem type and sought quantity by analyzing the facts describing the query of the problem and the multiple choice answers. If the query concerns a quantity, then that is considered to be the sought quantity. In that case, the problem type is determined to be numeric or symbolic based on the kinds of expressions found in the possible answers. Instead of asking for specific quantity values, the query can concern a qualitative state. In these cases, if the possible answers are quantity values then the problem is a state elaboration problem, otherwise the problem is a qualitative behavior problem. For state elaboration problems, the sought quantity is determined by analyzing the event structure in the problem. In the example above, the acceleration of the box is the sought parameter. For qualitative behavior problems, the sought quantity is found by domain-specific rules that determine what value(s) are needed to distinguish between the possible answers. In the example above, for instance, the acceleration and velocity of the box during the pulling event would be sought.

The basic problem-solving operation in physics problems is solving equations to find values. For numeric and symbolic value problems, this is sufficient. For qualitative behavior problems, the values of the sought quantities are tested to see which of the qualitative descriptions they satisfy. For state elaboration problems, an *assumption case* is created for each answer choice. The assumption case includes all the facts of the problem and an assumption of the value for the parameter given in the answer choice (e.g., the tension in the example problem above). Then, for each assumption case, the system solves for the sought quantity and determines if it is consistent with the problem description. If it is, then that assumption case is the correct answer.

5.4.3 Step 3: Solve for *q* via analogical model formulation

This step creates a scenario model incrementally, based on the analogy with the worked solution(s). The process starts by trying to find an appropriate value for the sought quantity *q*. In general, this is a recursive process, so the general strategy is describe below.

Given a quantity *q* to be solved for, its value can be determined in one of three ways:

1. It is already known as part of the problem. That is, there is a `valueOf` statement in the problem representation that provides an appropriate value for *q*. For symbolic problems, the `valueOf` statement must be expressed in symbolic terms compatible with the possible answers, as ascertained in the previous section. When a numeric answer is sought, the `valueOf` statement must provide a numeric value. In these cases, the value from the statement is used.
2. It is assumable. That is, there is a candidate inference containing a `stepResult` statement which provides a value for *q*. In this case the value from the analogy is assumed.

3. It is mentioned in a relevant equation. That is, there is a candidate inference which contains an equation that mentions q. In this case, recursive subgoals are spawned to solve for the other quantities in the equation, and once their values are found, a value is derived for q.

While the first case is straightforward, the second and third cases make important modeling decisions via analogy. The second case handles quantity value assumptions and default circumstances. The third case is a form of relevance reasoning, since analogous situations are assumed to be governed by similar equations.

Analogical modeling decisions, like all non-deductive inferences, should be verified if possible (Falkenhainer 1988). The `stepUses` statements in the worked solution provide context for the worked solution step. These statements can be thought of as preconditions for the analogical modeling decision. Currently, these preconditions are used in only one situation. If these statements mention a planetary body, which is not included in the mapping, and there is a different planetary body in the problem, the analogical modeling decisions based upon this solution step are deemed unusable. This is a useful heuristic for this domain because decisions based on planetary bodies typically involve assumptions involving gravitational constants, which of course vary across planets. Currently, the rules that do these verifications are hand coded. An important aspect of future work is to enable Companions to learn and refine these rules with experience.

In addition to verifying the inference, it is important to understand how numbers are handled in the analogical mapping. Because these are all within-domain analogies, when there is a correspondence between number entities, it is likely spurious. For example, if the problem includes a ball moving at 1m/s and the worked solution includes a ball moving at 2m/s, then 1 could be placed in correspondence with 2. This is a spurious correspondence because there is no reason to believe that all 2's in the worked solution should be considered 1's in the problem. Therefore when candidate inferences for equations include numeric values, the number from the worked solution is used. When the candidate inference concerns an assumed value, the target value is used if units match; otherwise, the base value is used. Returning to the example, if the worked solution includes the distance-velocity equation, $V_f^2 = V_i^2 - 2ad$, then the resulting mapping would include a candidate inference suggesting $V_f^2 = V_i^2 - 1ad$ as an appropriate equation for the problem. Because the number from the worked solution is always used for equations, even with the spurious correspondence, the correct equation $V_f^2 = V_i^2 - 2ad$ is instantiated in the problem.

The equation solving and algebraic simplification routines are straightforward, based on (Forbus & de Kleer 1993).

5.4.4 Step 4: Checking boundary conditions

Doing "sanity checks" of answers is always a good problem-solving practice. In physics, this is involves testing boundary conditions. For example, if a problem asked, "How far a ball would fall off a 200m building in 4s?", its worked solution would include a sanity checking step in which the computed answer, 80m, was compared to the height of the building, 200m. Since this is less, the computed answer is okay. If the computed answer were larger than the height of the building, it means that the boundary conditions of the equations are violated. Since one ignores the impact crater in these problems, the answer would then be the height of the building, because that is the point at which the behavior captured by the falling event ends.

This aspect of the scenario model is also dependant on the analogy. Boundary conditions are recognized by candidate inferences involving ordinal relationships (i.e., `greaterThan` or `lessThan`) between parameters in the problem. Currently only boundary condition tests involving the sought quantity are processed. This is because it is clear how to resolve such a failure, i.e. use the value compared against it instead, because it constitutes a limit point (Forbus 1984) for that quantity[6].

5.4.5 Step 5: Selecting the multiple choice answer

Finally, the appropriate multiple choice answer is selected. For numeric and symbolic value problems, the computed answer is compared against each of the answer choices and the closest answer is selected. The closest answer is determined as follows. For numeric value problems, the closest answer is the simply the answer choice with the minimum absolute value of the difference between the computed answer and the answer choice with one exception. If all the answer choices have the same sign but the computed answer is the opposite sign, then the closest answer to the opposite of the answer choice is selected. This simple heuristic corrects when the system orients the positive and negative directions differently than the problem designer. For

[6] This heuristic is reasonable for mechanics but would not be appropriate for other domains, such as thermodynamics.

symbolic value problems, the closest possible answer would be identical to the computed answer. When none of the answer choices satisfy this criterion, the system identifies answers with the same quantities as the computed answer. By substituting random values for each of the quantities, the system solves each equation and compares the results. If they are within a threshold the answer is selected. This heuristic alleviates limitations in the algebra manipulation aspects of system.

For qualitative behavior problems, qualitative arithmetic is used to select the consistent answer choice. For instance, if there is a computed positive vertical velocity, the object must be moving upwards. In the example qualitative behavior problem of Section 5.4.2, the Companion determines that answer ‘c’, the box moves upward with constant acceleration and increasing velocity, is the only consistent choice. This is because the box’s velocity at the beginning of the event is 0m/s and its computed acceleration during the event is 3m/s^2. For state elaboration problems, the first assumed value that is consistent with the computed answer is selected. In the state elaboration example from Section 5.4.2, the problem states that the box is moving upward with constant acceleration, therefore, the consistent assumption case results in a positive acceleration for the box. The answer ‘d’ contains the only tension, 120N, which results in a positive acceleration, 5m/s^2.

Importantly, the system is not allowed to guess if it cannot compute the answer for a problem.

5.4.6 Implementation in a Companion

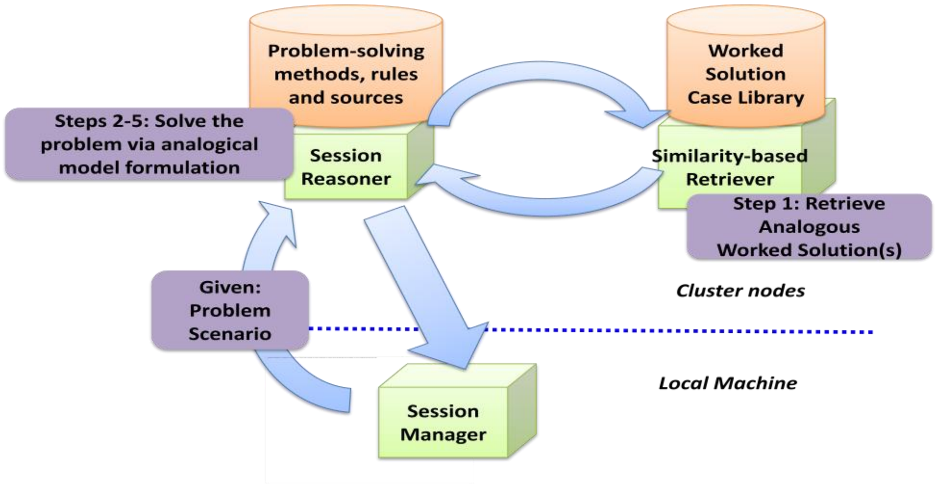

Problem-solving methods, rules and sources
Worked Solution Case Library
Steps 2-5: Solve the problem via analogical model formulation
Session Reasoner
Similarity-based Retriever
Step 1: Retrieve Analogous Worked Solution(s)
Cluster nodes
Given: Problem Scenario
Local Machine
Session Manager

Figure 19: Analogical model formulation implemented on the Companion cognitive architecture

Figure 19 shows how the steps of the algorithm are divided among various Companion's agents. Aside from retrieving worked solutions, the entire process takes place on the Session Reasoner. The process begins with the Session Reasoner requesting relevant worked solution(s) from the Similarity-based Retriever. After the Session Reasoner solves the problem and selects an answer, it is sent to the Session Manger for display on the user's machine. An alternative method of interaction uses the Executive to run batch experiments. This works in the same manner except the user interactions with the Session Manager are scripted and the results are recorded by the Executive. The algorithm from Figure 18 is implemented using an AND/OR problem-solver drawn from (Forbus & de Kleer 1993). The problem-solving knowledge consists of 27 methods, 169 backchaining rules, and two reasoning sources, which are procedural attachments efficiently implementing analogical processing and algebraic operations.

For illustration, here is how a Companion employs the above algorithm to solve the following restyling problem:

> A box is dropped from the top of a 300m cliff on Earth and falls to the ground. If air resistance is negligible, which of the following is most nearly equal to the distance the cliff falls during the first 7.3s after it is released? (a) 36.5m; (b) 73m; (c) 266.45m; (d) 532.9m

The problem is presented to the Companion as a case of 28 predicate calculus facts. The Companion begins by asking the Similarity-based Retriever for a relevant example, which in this case is the worked solution for Problem 1 from Figure 15. The Session Reasoner uses SME create a mapping between the retrieved worked solution and the problem. The event structure of the problem contains three events: the initial situation, the dropping, and the falling. All three events are included in the correspondences of this mapping; therefore the Companion does not retrieve additional analogues. Next, it determines that this is a numeric value problem based upon the

```
Base expressions:
   1. (valueOf ((QPQuantityFn DistanceTravelled) Ball-1 Interval-1) Distance-1)
   2. (valueOf ((QPQuantityFn Time-Quantity) Interval-1) (SecondsDuration 4))
   3. (temporallyCooriginating Fall-1 Interval-1)
   4. (TimeInterval Interval-1)
   5. (objectActedOn Release-1 Ball-1)
   6. (primaryObjectMoving Fall-1 Ball-1)
   7. (objectStationary Initial-Situation-1 Ball-1)
Target Expressions:
   1. (valueOf ((QPQuantityFn DistanceTravelled) Box-5 Interval-5) Distance-5)
   2. (valueOf ((QPQuantityFn Time-Quantity) Interval-5) (SecondsDuration 7.3))
   3. (temporallyCooriginating Fall-5 Interval-5)
   4. (TimeInterval Interval-5)
   5. (objectActedOn Release-5 Box-5)
   6. (primaryObjectMoving Fall-5 Box-5)
   7. (objectStationary Initial-Situation-5 Box-5)
```

Figure 20: Expressions in alignment based upon identical predicates

query statement and the answer choices. The query statement for this problem indicates that the distance the box travels over the 7.3 second interval is the sought quantity.

Next, the Companion proceeds to solve for the sought quantity, the distance the box travels. Because there are neither `valueOf` statements concerning the distance the box travels nor candidate inferences suggesting a quantity value or default circumstance modeling assumption, the Companion searches for a relevant equation mentioning the sought quantity. This is done by searching the candidate inferences of the analogy. Figure 20 contains the aligned expressions which result in the entities `Ball-1` and `Interval-1` from the worked solution corresponding with `Box-5` and `Interval-5` from the problem. These correspondences result in a candidate inference for the applicable equation mentioning the distance the distance `Box-5` travels during `Interval-5`, $d=v_it+at^2$, shown in Figure 21.

Figure 21: Candidate inference (dashed lines) projected from the base into the target based upon correspondences

In order to solve this equation for the distance the box travels, the Companion first solves for each of the other parameters in the equation: the duration of the interval, the speed of the box at the start of the interval, and the acceleration of the box during the interval. The duration of `Interval-5`, 7.3s, is given directly in the problem. Using the analogy to make a quantity value assumption, the Companion infers the speed of the `Box-5` at the start of the interval, 0m/s, based on Step 5 of the worked solution. Step 5 states that `Ball-1` at the beginning of `Interval-1` has a speed of 0m/s because it starts at rest. A subset of the candidate inferences referring to this step are shown in Figure 22. Recall from Section 2.1 that `AnalogySkolemFn` expressions represent entities which appear in the base (i.e., worked solution) representation and do not have a corresponding entity in the target (i.e., problem description). Because the value from the base description is used for equations, the `AnalogySkolemFn` is removed from the expression when making this inference. This is reasonable because these are performing within-domain analogies; for cross-domain analogies, more work would be required to resolve the analogy skolem into an appropriate constant for the target domain. Next, this inference must be verified. The Companion makes sure that the step does not rely upon the worked solution occurring on a different planetary body. The `stepUses` statements, the context of the worked solution step, do not include references to any planetary bodies as defined by the ResearchCyc ontology. Therefore, the Companion assumes 0m/s as the speed of the box at the start of the interval.

```
(stepUses (AnalogySkolemFn ETS-WorkedSolution-1-0-1-Step5)
       (primaryObjectMoving Fall-5 Box-5))
(stepUses (AnalogySkolemFn ETS-WorkedSolution-1-0-1-Step5)
       (objectStationary Initial-Situation-5 Box-5))
(stepUses (AnalogySkolemFn ETS-WorkedSolution-1-0-1-Step5)
       (contiguousAfter Fall-5 Initial-Situation-5))
(stepUses (AnalogySkolemFn ETS-WorkedSolution-1-0-1-Step5)
       (temporallyCooriginating Fall-5 Interval-5))
(stepResult (AnalogySkolemFn ETS-WorkedSolution-1-0-1-Step5)
       (valueOf
              (MeasurementAtFn ((QPQuantityFn Speed) Box-5)
                     (StartFn Interval-5))
              (AnalogySkolemFn (MetersPerSecond 0))))
```

Figure 22: Candidate inferences permitting the quantity value modeling assumption for the speed of the box

To solve for the acceleration of the box during the falling event, 10m/s^2, the Companion makes a default circumstance modeling decision. This decision is made in the same manner as the quantity value decision for the speed of the box at the beginning of the falling event, with one exception. The `stepUses` statements for the worked solution step suggesting this decision mention `PlanetEarth`, a planetary body. Because `PlanetEarth` is unmapped in the analogy, the Companion searches for any planetary bodies in the problem. The Companion accepts the inference because the problem does not mention any planetary bodies. This is an example of the Companion making a default circumstance modeling assumption. After recursively solving for these three parameters, the Companion solves the equation for the distance `Box-5` traveled during `Fall-5`, 266.45m.

Next, the Companion checks for a candidate inference concerning a boundary condition check. In this case, step 7 from the worked solution is a boundary condition check and results in candidate inferences, shown in Figure 23. Recall that these candidate inferences are produced by the correspondences between the analogous worked solution and the problem. The answer in the worked solution, 161.312m, has no correspondence at the time of the analogy as it was just computed in the previous step. The first fact states that the boundary condition should be greater than the computed answer. The next two facts indicate the boundary condition and computed answer quantities for this comparison. The computed answer, 266.45m, is less than the height of `Cliff-5`, 300m, and therefore the Companion uses the computed answer when selecting the multiple choice option.

```
(stepUses (AnalogySkolemFn ETS-WorkedSolution-1-Step7)
       (greaterThanOrEqualTo (Meter 300) (AnalogySkolemFn (Meter 161.312))))
(stepUses (AnalogySkolemFn ETS-WorkedSolution-1-Step7)
       (valueOf (MeasurementAtFn ((QPQuantityFn Height) Cliff-5) Fall-5)
              (Meter 300)))
(stepUses (AnalogySkolemFn ETS-WorkedSolution-1-Step7)
       (valueOf ((QPQuantityFn DistanceTravelled) Box-5 Interval-5)
              (AnalogySkolemFn (Meter 161.312))))
```

Figure 23: Candidate inferences indicating a boundary check condition

In the final phase of problem-solving, the Companion uses the computed answer to select the appropriate multiple choice option. Because this is a numeric problem, the answer choice closest to the computed answer, 266.45m, is selected. In this case, answer 'c' is 266.45m exactly, and, therefore, the Companion selects it. Solving this problem takes the Companion approximately 35 seconds.[7]

5.5 Evaluation

A series of experiments was conducted to evaluate a Companion's ability to transfer knowledge across physics problems via analogical model formulation and its fidelity as a cognitive model. The initial experiment was an external evaluation conducted by the Educational Testing Service on largely unseen problems. The timing of this evaluation was determined by an external, funder-mandated timetable and included a code freeze. The results of this evaluation were summarized in (Klenk & Forbus 2007a). Although the results, as described below, showed statistically significant transfer across all six levels, there were a number of irregularities. An analysis of these results led to the discovery of numerous representation and implementation errors. After fixing these bugs, a second experiment, with the same design as the first, was run. This experiment confirms that the problems were not with analogical model formulation, and

[7] The Companion was running on 4 cluster nodes, each with two 3.2 Ghz Pentium Xeon processors and 3 GB of RAM.

provides evidence about the asymptotic performance of analogical model formulation. This section discusses both experiments in turn, including the post-analysis and changes made to the system and representations before the second experiment. In addition to these performance metrics, the operational data from the first experiment is compared to human data presented by van Lehn (1998) to evaluate this algorithm as a cognitive model.

5.5.1 Experiment 1

The first evaluation was carried out by the Educational Testing Service, who remotely accessed a Companion running on a cluster at Northwestern University. The problems and worked solutions in this experiment were generated from templates. Examples, in English, appear in Figure 15. These templates represent approximately 20% of the typical Mechanics portion of the AP Physics exam. The system designers saw examples from less than 50% of the templates before the evaluation, and none of the templates themselves.

ETS measured the amount of transfer learning in the system. Transfer learning occurs when an agent learns faster in the target domain given learning in a related source domain. This is most naturally measured as a difference in learning curves, which motivates this experiment's design. The experiment was designed without knowledge of the analogical model formulation approach, reducing the level of tailorability.

5.5.1.1 Methodology

To define transfer levels, a *source set* is required. The source set consisted of 20 problems and worked solutions, 5 from each of the problem types illustrated in Figure 15. To generate learning curves for each of the 6 transfer levels, five *target training sets* were created for each of the transfer levels. Each target training set consisted of 4 quizzes, with one problem from each problem type in each quiz. Each problem represented a systematic transformation, based upon the transfer level, from a problem in the source set. The Companion was given each target training set as a sequence of quizzes. After a quiz was administered, the Companion was given the worked solutions for the problems on that quiz. Thus, the worked solutions from earlier quizzes were available for use in solving later quizzes within the target training set. After each target training set, the Companion's memory was reset. The Companion was administered each target training set twice, one for each experimental condition. First, the Companion is given each target training set without access to the problems and worked solutions of the source set (the *no-transfer* condition). Then, to measure transfer learning, the Companion is given each target training set with access to the source set's problems and worked solutions (the *transfer* condition). Comparing the learning curves for these two conditions provides a measure of how much was learned via transfer from the source set.

There are three ways for transfer to manifest itself. (1) The system may get a jump start, i.e., the learning curve in the transfer condition has a higher y-intercept. (2) The system may learn faster. (3) The system may reach a higher level of performance. These are not mutually exclusive. Given the direct application of examples in analogical model formulation, the jump start transfer results are central to the performance of the system.

5.5.1.2 Results

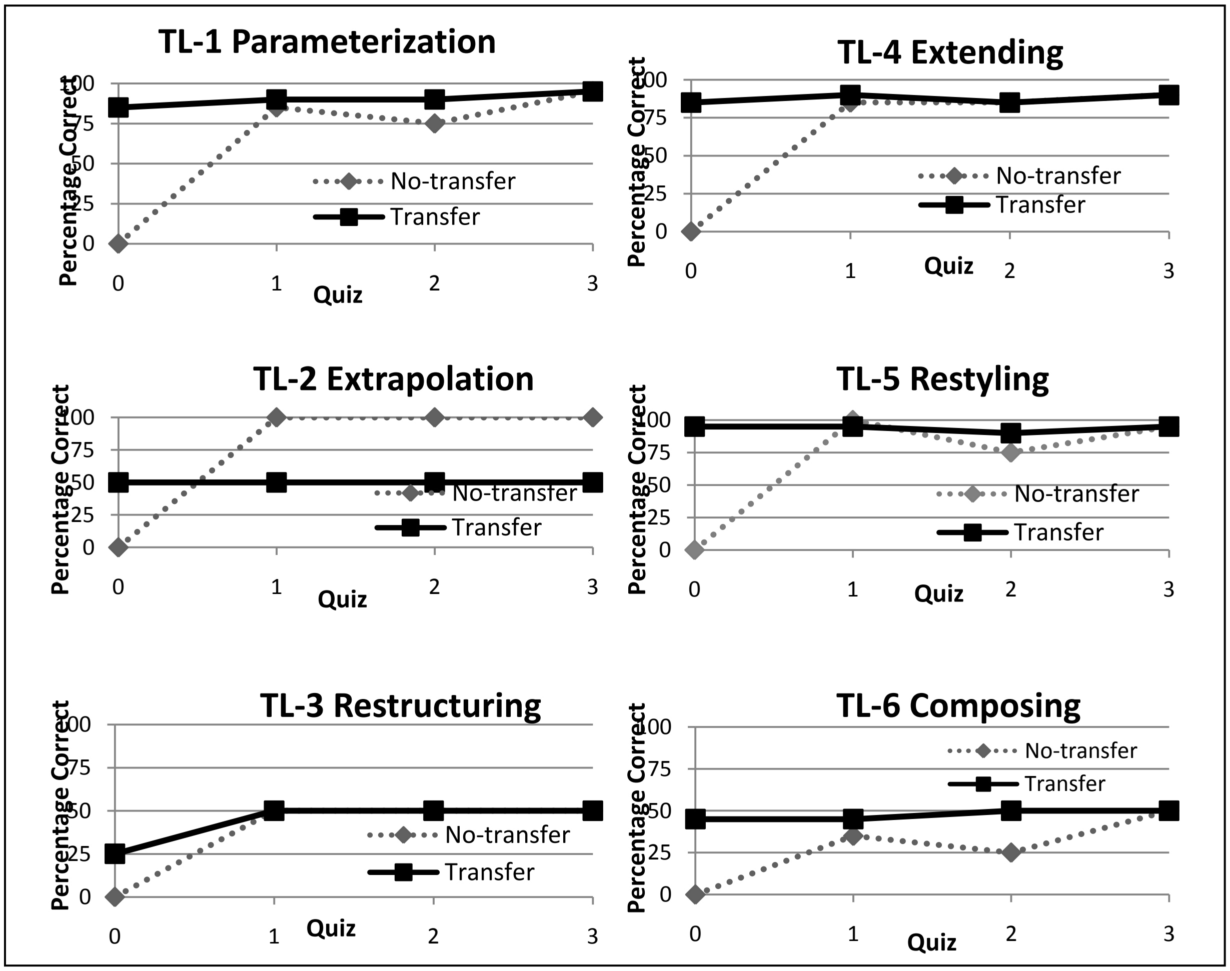


Figure 24: Experiment 1 results, administered by the Educational Testing Service

Figure 24 shows the learning curves for both the transfer and no-transfer conditions for each transfer level. TL-1, TL-3, TL-4, TL-5, and TL-6 all had 80 problems. TL-2 only had 40 problems as it was impossible to change the numeric parameters in such a way to qualitatively change the outcome for two Problem Types, 2 and 4. Recall that in order to more precisely measure the effectiveness of analogical learning, the Companion's KB contained no equations of physics nor any other modeling knowledge about the domain. This means whenever a problem is solved, it is solved by analogical model formulation. Given that analogical model formulation requires worked solutions to solve problems, the no-transfer condition always begins at 0% and improves as the Companion accumulates worked solutions. All transfer levels showed a statistically significant jump start ($p<.01$). For TL-1, TL-4, and TL-5, the jump start was 88%. Other levels were not as high: TL-2 was 50%, TL-3 was 25%, and TL-6 was 44%. This provides an average of 63.8% in jump-start performance, supporting the hypothesis that analogical model formulation can be used to solve AP Physics style problems, including handling these kinds of transfer.

5.5.2 Post-analysis and system modifications

While the jump start results support the hypothesis, it is important to understand whether the performance failures are due to analogical model formulation, or due to some other factor. Theoretically, there are many ways analogical model formulation can fail: The system can fail to find a precedent when one exists, mappings could be incorrect, or candidate inferences could be incorrectly analyzed. Surprisingly, these problems accounted for only a small minority of the problem-solving failures. As illustrated below, the vast majority of the failures were due to human error in representing the problems and implementing the system.

In TL-2 (extrapolation), there is negative transfer, in that the no-transfer condition outperformed the transfer condition in later quizzes. This occurred because the Companion was repeatedly getting a problem correct for the wrong reasons in the no-transfer condition. An error in the worked solution representations for the target training set worked solutions of Problem Type 3 caused the Companion to incorrectly assume a value for acceleration, which coincidently led to the correct answer. The failure of the system to detect this and other recurring problems is leading us to focus effort on improving the Executive.

The low ceilings in the transfer condition in TL-2, and in both conditions in TL-3 (restructuring) and TL-6 (composing), are due to a combination of three limitations in the fixed components of the Companion's problem-solving strategies and a number of representation errors. The problem-solving strategy limitations were

1. The internal resource limit (i.e., maximum number of and/or graph nodes) was about 5% too low for some of the composing problems.
2. The algebra system was unable to correctly handle all of the necessary algebraic manipulation and equation comparison (e.g. trigonometry and composing symbolic and numeric problems).
3. The strategy of trying each value in turn for state elaboration problems, which make up 25% of TL-3, was grossly inefficient.

None of these problems concerns analogical model formulation, and the system was changed in the following manner. To solve the first problem, the internal resource limit was increased by 5%. Recall that learning modeling decisions is the focus of this work, not equation solving strategies. Therefore the second problem was solved by simply extending the algebra system to handle the necessary cases. The third problem was solved by developing a more efficient strategy for state elaboration problems. This involved altering the problem analysis and answer selection steps from the algorithm. Recall this example of a state elaboration problem:

Which of the following tensions is required to move a box of mass 8kg from rest on the floor upward with constant acceleration when it is pulled vertically upward by a cord attached to the box? (a) 40N; (b) 60N; (c) 70N; (d) 120N.

During problem analysis (Step 2), the new strategy identifies a *limit point quantity* whose value determines the consistency of a scenario. Here, the acceleration of the box is a limit point quantity. Using qualitative mechanics, the scenario is consistent only if the acceleration is greater than $0m/s^{2.}$. Therefore, $0m/s^2$ is selected as the limit point. Using the query, the sought quantity is the tension of the cord. Next, the Companion assumes the limit point quantity value and proceeds with the algorithm to solve for the sought after quantity. After assuming an acceleration of $0m/s^2$ for the box, the Companion uses analogical model formulation to solve for the tension of the cord, 80N. Instead of solving for the acceleration four different times, once for each assumption case, the new strategy is considerably more efficient, solving for the tension only once.

To select a multiple choice answer, the new strategy uses the scenario model to determine the qualitative relationship between the limit point quantity and the sought quantity. In this case, the acceleration of the box is qualitatively proportional to the tension in the cord (i.e., if the acceleration of the box is increased then the tension of the cord is increased). Since the problem indicates a consistent solution involves a positive acceleration, the Companion selects the multiple choice answer that is greater than 80N, in this case, choice ‘d’, 120N. This new strategy only applies to state elaboration problems, which make up 25% of TL-3, Restructuring.

The only change to the analogical model formulation portion of the algorithm was the addition of one rule for verifying modeling decisions. When inferring a numeric parameter in quantity value assumptions, the Companion verifies that the units of the assumed value are applicable to the quantity type. For instance, the Companion will not assume 5m/s for the acceleration of an object. Enabling Companions to learn such verification rules is an important element of future work on this project.

A close examination of the entire set of problems and worked solutions revealed two systematic kinds of representation errors.

1. Facts were sometimes omitted from the problem representations. For example, in some of the original representations, the correct answer was not listed as one of the answer choices, or the direction of a pulling event was not mentioned.
2. The agreed-upon conventions for representing worked solution steps were not always employed.

An example of the second type of error comes from the erroneous worked solutions which caused the negative transfer for TL-2. Given the question:

> What will happen to a 10kg crate being pulled upwards by a string with a tension of 38N? a) remain stationary; b) move upwards with constant speed; c) move upwards with constant acceleration; d) move upwards with increasing

acceleration

The worked solution for this problem calculates the acceleration of the crate to be -6.2m/s^2. In sanity checking step for this answer, the comparison with the boundary condition was omitted. The result of this step is that -6.2m/s^2 is not the acceleration of the crate. This step is followed by a determining a value from the situation step, in which this fact is used to infer that the acceleration of the Crate is 0m/s^2. Both of these steps violate the agreed upon conventions. First, because the sanity checking step represents a check against a boundary condition, there should have been a comparison between -6.2m/s^2 and 0m/s^2. The result of this step should have been that the acceleration of the crate was 0m/s^2. Second, the determination of a value from the situation step is wrong, because the step does not depend upon the problem scenario but is actually a sanity checking step. Because this inference is part of the sanity checking step, it was simply removed to create a worked solution which followed the conventions.

Recall that the problems and worked solutions were automatically generated from templates, so these template-level bugs led to errors in all instances of that problem type at that transfer level. These systematic errors were corrected to generate an improved corpus of problems and worked solutions.

5.5.3 Experiment 2

To test these explanations for the results of Experiment 1, another experiment was conducted on the same sets of problems after fixing the representation and implementation errors.

5.5.3.1 Method

The same the experimental procedure, problems and worked solutions as Experiment 1 were used with the following changes. First, the experiment was conducted internally, instead of the Educational Testing Service, due to budget cuts. Second, the corrected corpus of problems and worked solutions was used. Third, the system changes described in Section 5.5.2 were implemented.

As in Experiment 1, the experimental hypothesis is that analogical model formulation transfers modeling knowledge across these six transfer levels. As before, learning curves for each of the six transfer levels were collected and the analysis focused on the jump starts in the transfer condition.

5.5.3.2 Results of Experiment 2

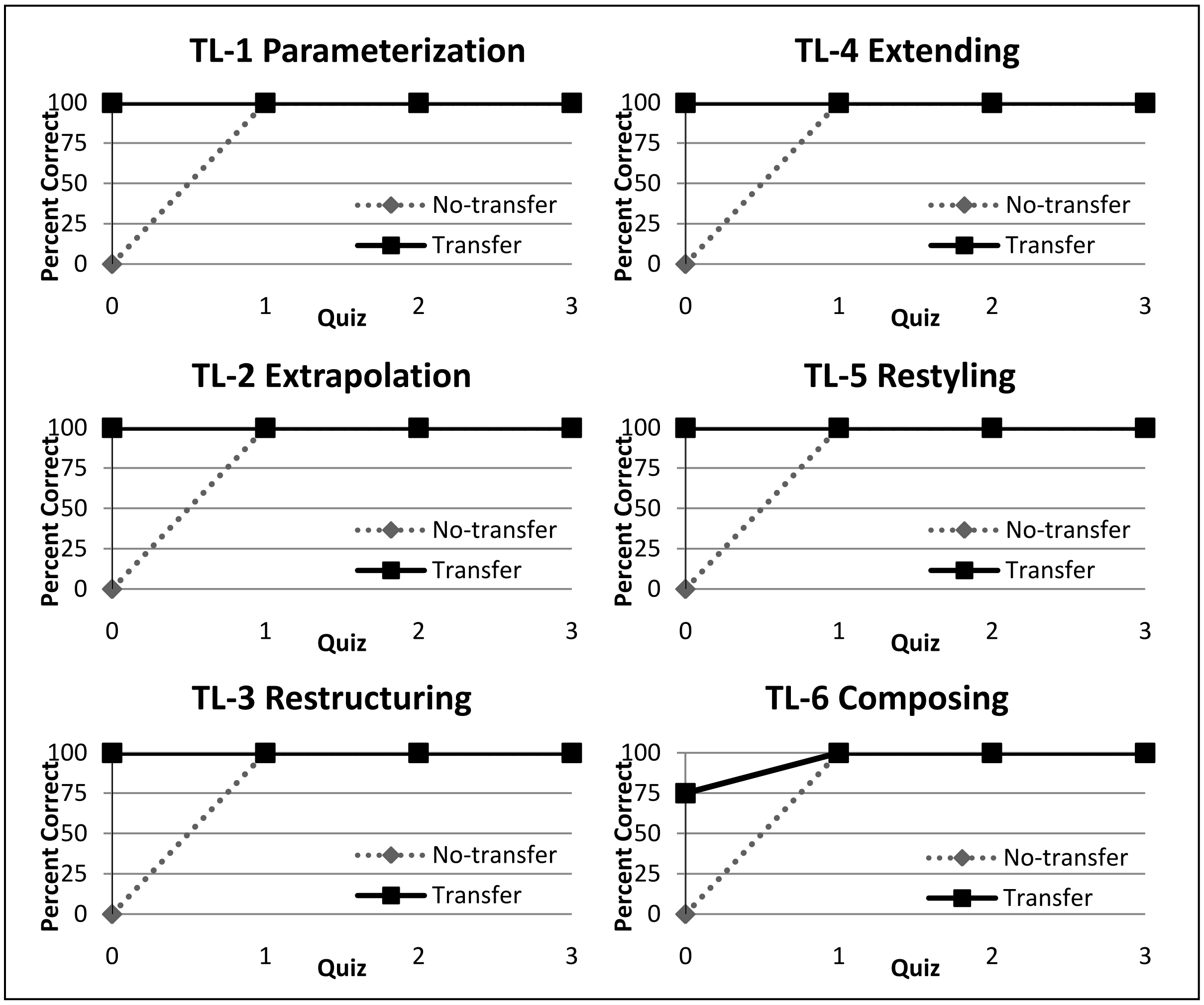


Figure 25: Final Evaluation Results

Figure 25 contains the learning curves for each of the six transfer levels. Across all the transfer levels, the Companion achieved a 95.8% jump start due to the source problems and worked solutions. TLs 1-5 all exhibited perfect transfer. The Companion performed at ceiling (100%) given just the source set worked solutions. On TL-6, the Companion recorded a jump start of 75%. Once again, all of the jump starts are statistically significant ($p < .01$).

These results illustrate that analogy can be used to formulate models in the domain of AP Physics problem-solving over a broad range of problems. The retrieval rates and mapping rates were both 100%. That is, MAC/FAC always selected appropriate analogous worked solution(s) if they were available and SME always created a useful set of correspondences and candidate inferences between the worked solutions and the problems. The only failures of transfer involved limitations in the rules for verifying analogical inferences. In particular, the verification rules prevented a necessary modeling decision from being made on a set of difficult problems (25% of the composing problems). These composing problems involved source problems occurring on different planets and quantities referencing the planetary body of the problem explicitly (e.g., the work done by the gravitational force of an asteroid on a sliding block). Therefore, the heuristic which prevents analogical inferences involving unmapped planetary bodies fails because this inference concerns a definition of net work equation and is not dependant on the gravitational constant in the problem.

5.5.4 Evaluation as a Cognitive Model

The previous two experiments illustrate the Companion's capability to solve AP Physics problems using analogical model formulation. Because much of the motivation for this work comes from cognitive science, it is important to evaluate how analogical model formulation performs as a cognitive model. For this section, I refer to this as the AMF model. While capability is certainly an important benchmark for cognitive models, comparisons against existing psychological data are important as well. In the case of Newtonian physics-problem solving, there has been extensive work studying aspects of the human performance. This section contains a favorable qualitative comparison between the AMF model of physics problem-solving with the most relevant previously published human data.

This work was originally published in (Klenk & Forbus 2007b). The human data comes from a collection of verbal protocols presented first by Chi *et al.* (1989). These protocols were collected from 9 subjects learning Newtonian Mechanics. The subjects were told to speak aloud while they studied worked examples and solved a series of problems. These protocols were used to investigate the differences between good and poor problem-solvers. One key difference in the protocols was that good problem-solvers would explain the worked solutions to themselves whereas poor problem-solvers would simply read the examples. The researchers called this the *self-explanation effect* and created the Cascade system to model it (VanLehn *et al.* 1992). When attempting to fit Cascade to this data, VanLehn and Jones (1993) observed that people sometimes used analogical reasoning even in situations when they had valid first-principles knowledge. Later, VanLehn (1998) reanalyzed the original protocols, leading to a taxonomy of analogy events:

- Initialization events – the subject sets up a mapping between the examples and the problems.
- Transfer events – the subject infers something about the solution from an example. These events were further divided by the type of inference made:

1. <u>Line:</u> The subject transferred a whole equation, vector, or diagram.
2. <u>Part of a line:</u> The subject transferred a detail from a line, such as whether a projection function was sine or cosine, or whether a vector went up or down.
3. <u>Search control:</u> The subject made the decision on what steps to do by consulting the

example and seeing what steps it did.

4. Checking: The subject decided whether their most recent action or decision was correct by consulting the example.
5. Failure: The subject failed to find anything useful during this transfer event.

Initialization events were indicated by the subject flipping the book to a worked solution, reading some of the example and deciding if it will be useful to solve the current problem. VanLehn found that initialization events usually occur at the beginning of the problem-solving, consistent with existing research (e.g. Bassok & Holyoak 1989; Faries & Reiser 1988; Ross 1989). Cascade did not model subjects' retrieval process and used a heuristic mapping algorithm between the problem and the example (VanLehn & Jones 1993). The AMF model differs by taking existing computational models of analogical mapping and similarity-based retrieval as starting points, and building the problem-solving system on top of them. In the AMF model, initialization events occur at the beginning of problem-solving when the Companion uses MAC/FAC to retrieve a similar example(s) from memory. The use of recursive retrievals for complex analogs is consistent with VanLehn's finding that for complex problems, there could be multiple initialization events.

Regarding transfer events, Cascade only accounted for line transfer and search control events with its separate analogical mechanism and memory. Their approach is based on the notion that analogy is invoked to resolve impasses reached by rule-based reasoning. The AMF model uses analogy to inform all modeling decisions, as well as some other important inferences such as checking boundary conditions. Most of the transfer events can easily be mapped to parts of the AMF model.

Because the AMF model answers all problems using analogy, it relies heavily on the transfer events from VanLehn's analysis. Line transfers are indicated in the AMF model in two ways: (1) Using a candidate inference to map an equation from the worked solution onto the problem and (2) inferring parameter values from the problem situation. Consistent with VanLehn's findings, these events occur throughout the problem-solving process. The other transfer events are modeled incompletely. The AMF model does not have anything corresponding to part-of-line transfers. These events are extremely rare, accounting for just 3% of all transfer events in the protocols. Analogical search control is implicit in the analogical model formulation process. Therefore, it is difficult to identify these as individual events in the AMF model. Checking transfer events corresponds to employing a boundary condition check in the AMF model. Transfer failures are indicated when a precondition test fails, blocking the use of a candidate inference.

Given these differences, a quantitative comparison of the number of different analogy events would not be informative. However, in addition to the consistencies noted above, there should be a qualitative pattern of consistency between the AMF model and the human protocol data. That is, the AMF model predicts more line transfer events than observed in protocols, since it is at the extreme poverty end of assumptions about initial knowledge. Second, analogical search control is done implicitly; therefore the AMF model has no explicit search control events. Third, given the incompleteness of the AMF model in regards to modeling checking and transfer failures, it should have fewer such events than in protocols.

Table 7: Analogy Events per Problem

Event Type	Companion's Problems			Protocols (n=24)
	Correct (n=64)	Failed (n=36)	Total (n=90)	
Initialization	1.12	1.16	1.13	1.2
Transfer	6.72	9.39	6.72	4.9
Line	6.13 (91%)	9.33 (99%)	7.18 (95%)	2.6 (54%)
Checking	.47 (7%)	.03 (<1%)	.33 (4%)	.75 (15%)
Failure	.12 (2%)	.03 (<1%)	.09 (1%)	.38 (8%)
Other	N/A	N/A	N/A	1.17 (24%)

Using the results first experiment, analogy events were categorized using reasoning traces from the first quiz in each transfer level. The first quiz only, or jump start, was used as these are central to the AMF model and its claims. These events are displayed in Table 7 and broken down by the result of the problem-solving episode. The "Other" event type contains the analogy events not modeled, including search control and part of line transfer as well as the miscellaneous events from the protocols. The Companion's analogy events are further divided depending on if the problem in which the event occurred was solved correctly. Also, percentages of total transfer events are supplied next to each transfer event in each condition.

These results have a reasonable qualitative fit with VanLehn's human protocols. VanLehn summarized the results concerning type of information transferred in the protocols by saying "The basic result is simply that most students, both Good and Poor, transferred whole lines from the example to the problem" (1998, p. 364). The AMF model was even more dependent on line transfers for problem solving due to the fact that the vast majority of the domain knowledge used to solve these problems came from examples. The majority of the "Other" transfers were search control events, which motivates future work developing methods for search control. While the AMF model checking and failed transfer events are incomplete, both of these occurred more frequently on correctly solved problems. This, in addition to the fact that the protocols noted even more of these types of events per problem, indicates that a more complete model of these events could lead to more robust problem-solving.

5.6 Discussion and Future Directions

These experiments support the claim that analogical model formulation is a robust method for transferring knowledge across these six transfer levels and a plausible cognitive model of human problem-solving. First, the breadth of the materials and methods for evaluation are noteworthy. Drawn from four problem types, the 460 problems and worked solutions created for this evaluation included entities of 110 conceptual types and 144 unique relations. Second, in an experiment externally administered by ETS, a Companion achieved a significant jump start on all transfer levels (63.8% averaged across transfer levels) consisting of largely unseen problems. Finally, the second experiment provides a better understanding of the effectiveness of analogical model formulation across these six transfer levels. In this experiment, the Companion demonstrated perfect transfer across transfer levels 1-5. The 25% transfer failures on composing problems demonstrate a weakness of analogical model formulation. Specifically, while the hand-coded verification rules were effective across a broad range of scenarios, they do not handle all situations. The combination of these failures and the fact that the additional verification rule was added prior to Experiment 2 indicates that learning and refining these rules automatically is an important direction for future work.

As in the previous chapter, this chapter's emphasis is on system performance on a human-level test. While achievement is an important benchmark, the method is of particular importance to understanding cognition. The AMF model's favorable comparison with human protocol data illustrates the cognitive plausibility of this approach. People are able to overcome the challenges posed by model formulation in this domain. Traditional model formulation depends on having a complete domain theory and abstract input descriptions. In addition, there is no account for how new domain knowledge is acquired. Analogical model formulation provides a solution to these problems. First, model formulation by example does not require a complete domain theory and operates over descriptions containing broad ranges of entities drawn from everyday scenarios. Second, additional domain knowledge can be added to the system through examples. As the results from these experiments show, analogical model formulation can enable a system to solve hard problems (i.e., AP Physics style problems) without a complete domain theory. I know of no other problem-solving experiments which demonstrate performance of analogical learning over systematic variations of relationships between problems at this scale.

Clearly there is much work remaining to realize the full potential of analogical model formulation, and the potential for analogical learning in problem-solving more generally. Even in the realm of AP Physics, recall that this corpus of problems is drawn from roughly 20% of the Mechanics portion of the AP Physics exam, of which Mechanics is only one section. Expanding the evaluation would further provide additional information regarding some of the underlying assumptions of analogical model formulation: How does retrieval scale with additional distracters? And where is the limit in differences between source and target problems which produces mapping failures?

As mentioned earlier, one limitation that must be overcome is relying on hand-coded rules for verifying analogical inferences and algebraic operations. There were ample signals to the system that something was wrong with its knowledge (i.e., multiple repeated failures in some of the transfer conditions in the first experiment), but the current version of the architecture was unable to exploit this information. Consequently, there are plans to expand the capabilities of the Executive in the Companion architecture, enabling the system to take responsibility for learning and refining its verification rules. By keeping track of successful and unsuccessful analogical modeling decisions, a Companion could learn in what context different types of modeling decisions are effective. Keeping and analyzing records of its problem-solving successes and failures should also provide the information needed for formulating its own learning goals (Ram & Leake 1995). To improve this work as a cognitive model, additional work on analogically checking boundary conditions is required. Instead of checking for boundary conditions at the end of each problem-solving episode, the Companion should check the analogy to see if one applies after solving each equation. These boundary conditions are part of the modeling knowledge associated the mathematical equations.

In addition to testing the system on more problem types and extending analogical model formulation through learning verification rules and boundary conditions, other methods of knowledge transfer are required for a more complete account of physics problem-solving. First, as in the BMCT, analogical model formulation does not provide an account for how examples become more abstract domain theories. This is especially important in AP Physics as equations may be applicable in highly dissimilar situations. One promising direction is to construct generalizations of the physical phenomena as *encapsulated histories* (Forbus 1984) using SEQL (Kuehne *et al.* 2000). These abstract domain theories would require the system to make explicit modeling abstraction decisions. Preliminary results from linear and rotational mechanics indicate that SEQL generalization can be effective for learning to make modeling abstraction decisions (Klenk *et al.* 2008). Integrating these techniques to learn abstract domain theories would enable a Companion to transfer what it learns even more broadly.

Second, as a Companion accumulates generalizations in one area of physics, this expertise should be useful for learning new related domains. The second part of this thesis discussed in Chapter 7 presents a model for cross-domain analogy. Cognitive scientists have shown how cross-domain analogies are useful for learning new areas of physics (Gentner & Gentner 1983; Falkenhainer 1988). Chapter 7 describes the domain transfer via analogy model and presents results from linear and rotational kinematics as well as the dynamical analogies between mechanical, electrical and thermal systems (Olsen 1943).

6 Analogical Model Formulation General Discussion

The last three chapters have discussed analogical model formulation and provided evidence its utility in two complex reasoning domains: problems from the Bennett Mechanical Comprehension Test and the AP Physics exam. In this chapter, I will review how different techniques from Artificial Intelligence and models from Cognitive Science relate to analogical model formulation. This chapter closes with a summary of my contributions and a review of important open questions.

6.1 Related Work

From a theoretical perspective, the two closest areas of research to analogical model formulation are model formulation and case-based reasoning. It is also important to compare analogical model formulation to other approaches for reasoning about the domains presented here: sketch-based and AP Physics problem-solving. Finally, this research is comparable to other transfer learning techniques.

6.1.1 Model Formulation

An important contribution of the qualitative reasoning community has been the formalization of the model formulation process (Falkenhainer & Forbus 1991). Methods have been developed to efficiently identify what levels of detail should be included and which perspectives should be taken in a scenario model (Nayak 1994; Rickel & Porter 1994; Levy *et al.* 1997). These approaches have focused primarily on engineering and scientific domains.

The motivation for analogical model formulation comes from three challenges posed to these methods when using them to account for human common sense reasoning. First, they require a complete and correct domain theory. This is a result of the focus on industrial applications where models of the systems components are known in advance. On the other hand, humans maintain contradictory domain theories, arrive at incorrect answers, and are able to generate models outside their areas of expertise. Second, traditional model formulation approaches do very little reasoning about everyday descriptions. A typical example is Flores & Cerda (2000), who formalized equivalent circuit configurations as rewrite rules, to simplify circuit schematics in a humanlike way in order to make analyses more tractable. These hand-coded approaches will be difficult to scale to human-level knowledge, which includes tens of thousands of entity types. Third, traditional approaches provide no account for learning new domain knowledge. While encoding complete and correct human commonsense knowledge is extremely difficult, people are proficient at providing examples. Analogical model formulation provides the capability to formulate models of everyday scenarios without complete domain knowledge. Furthermore, additional domain knowledge can be added to the system through examples.

6.1.2 Case-based Reasoning/Analogical Problem Solving

Analogical problem-solving systems take an approach similar to analogical model formulation, e.g. (Veleso & Carbonell 1993; Melis & Whittle 1999; Ouyang & Forbus 2006). These systems solve new problems by transferring plans, rather than modeling knowledge, from previous problem-solving episodes. Thus analogy is used as a means of guiding the problem-solver, but it could, with more effort, solve the problems without analogy. By contrast, the systems described here cannot solve anything without a prior example. Another difference is that the analogues for our system are worked examples, which are at a more abstract level than the problem-solver's internals, whereas the analogues for these prior systems were plans constructed by the problem-solvers themselves. In regards to the AP Physics work, it is unclear if plan-based analogical problem-solvers would do well on restructuring or composing problems. Restructuring problems require a different sequence of operations to solve for a new parameter; our method of only mining modeling information is agnostic with regard to the order in which information was used in worked solutions. Composing problems require combining concepts from multiple problems, which makes choosing plan steps more complex.

As described in Section 5.5.4, Cascade is perhaps most similar due to its focus on learning from examples and its domain of Newtonian physics (VanLehn 1999). Cascade was evaluated for problem-solving performance on handful of Newtonian physics problems. In addition to analogical search control knowledge, Cascade learns through resolving impasses while studying examples and solving problems. Primarily, Cascade uses overly general rules to resolve an impasse and learn new domain knowledge. If the impasse occurs during example explanation and the overly general rules fail, Cascade allows this example to be used as a base for making analogical modeling decisions. In Cascade's evaluation, this use of analogy was employed specifically for one particularly difficult problem (VanLehn *et al.* 1992). Instead of using analogy for modeling decisions as a last resort, analogical model formulation demonstrates that analogy can play a primary role in model formulation. Additionally, It is intriguing that analogical model formulation does as well as it does, and can even be used to explain a number of analogy event types found by VanLehn (1998) in protocol studies (Klenk & Forbus 2007b). It does seem likely that the best model of human reasoning (and most practical solution in engineering terms) is to both use analogical model formulation and learning that goes beyond accumulating examples.

In Case-Based Reasoning (CBR) systems, inferences are made about problems based upon previous cases. Most of today's CBR systems are based on feature-vectors, and hence lack the representational capacity to handle the types of problems represented here. In CBR systems which use relational representations (as used here), typically a heavy emphasis is placed on adapting the known case to the current problem (Kolodner 1993). This frequently requires domain specific heuristics. In analogical model formulation, the adaptation is almost completely handled via structure mapping. Structure mapping theory uses common relational structure to constrain the inference process. The only domain specific heuristics for adaptation are the rules for evaluating the inferences and not for the matching process itself. In the BMCT work, the class of problem determined what types of inferences to make via analogy. The Companion would infer modeling abstractions and qualitative mechanics relations for outcome problems, and causal models and techniques for measuring spatial quantities for DQA problems. In the AP Physics system, adaptation occurs when interpreting numbers in inferences and through the verification of analogical modeling decisions, which can result in the rejection of inferences. Like most CBR systems, these adaptation methods are currently hand-generated and consequently limited in their scope. Another important difference between CBR and analogical model formulation is in regards to the how cases are retrieved. Retrieval systems for relational CBR tend to use indexing schemes that are carefully designed on a domain-specific and task-specific basis. By contrast, MAC/FAC, which is used in both the BMCT and AP Physics systems, is domain-independent and task-independent, and has been used to explain a number of psychological results (Forbus 2001).

An important direction for intelligent agent research involves integrating episodic memory (Tulving 1972). Nuxoll (2007) argues that episodic memories (Tulving 1972) a number important cognitive capabilities including detecting significant input and sense of identity. Specifically, Nuxoll implemented a domain independent episodic memory in the SOAR cognitive architecture (Nuxoll & Laird 2007). The episodic memory enabled SOAR agents to use virtual sensing, perform action modeling, record previous successes and failures, learn retroactively, and boost other learning mechanisms in three different domains. Analogical model formulation differs by using a cognitive model of the retrieval process, MAC/FAC, and focusing solely on the broad task of model formulation. Unlike the retrieval mechanism used in SOAR, MAC/FAC uses automatically computed feature vectors from relational representations allowing for richer case representations. Both of these works take place within cognitive architectures and attempt to provide domain-independent integrated solutions.

6.1.3 Sketch-based Problem-Solving and Visual Analogy

Several other researchers use diagrams and analogy in problem-solving. SketchIt (Stahovich *et al.* 2000) used sketched input to compute qualitative rigid-body dynamic simulations. SketchIt only allows a handful of idealized components (e.g., rigid objects and springs) as inputs. Archytis (Yaner & Goel 2008) uses a method called compositional analogy to construct structural models from unlabeled two dimensional line drawings. Their system recognizes shapes based upon similarities between the lines and intersections in the problem and the labeled example. These shapes then guide the transfer of component types and connections from the example to the problem. Our system differs in a number of important ways. First, they use vector-graphics inputs, which are easier to process than the hand-drawn sketches we use with sKEA. On the other hand, we require users to segment their glyphs. While both systems rely to some degree on hand-labeled conceptual information, we use automatic model formulation during example creation to automatically add information, thereby reducing tailorability. Archytis does not address retrieval, and their analogies focus only on similarities in depiction, unlike our use of MAC/FAC for retrieval and the use of both visual and conceptual information in mapping.

6.1.4 Solving Textbook Problems

Textbook problem-solving has proven a frequent measuring stick for AI systems over the last 30 years. This research can be divided into roughly three different goals. The initial purpose of this work has been to first determine if computers are capable of solving textbook problems at all. After it was established that computers were capable of this task, researchers sought to understand the knowledge required to solve textbook problems. Given knowledge bases of textbook problem-solving knowledge, researchers have recently focused on evaluating how robustly their systems can perform on unseen problems.

Initial work on textbook problem solving includes with De Kleer's NEWTON (1977), MECHO (Bundy 1979) and ISAAC (Novak 1977). The primary result of NEWTON was the importance in qualitative reasoning when determining which equations were applicable for a given scenario. MECHO and ISAAC sought to understand equation solving strategies and techniques. MECHO and ISAAC take natural language input and move to structural abstractions via collections of hand-coded rules to solve the problems. The connection between the handful of everyday entities each system knew about and the abstractions of physics were hand-coded, as was all of the domain knowledge. These systems were aimed at exploring how computers could solve physics problems at all.

These textbook problem-solving systems were successful because of their knowledge about their domains. Batali approached the problem of automatically extracting knowledge from textbooks (1991). He found that textbooks include much more than just the logical presentation of the domain laws. In particular, textbooks include numerous examples to illustrate equations and concepts. Perhaps the most thorough investigation of complex textbook problem-solving comes from Pisan's TPS system (1998). TPS integrated the expert qualitative, quantitative and diagrammatic knowledge necessary to solve 150 Problems from a number of common thermodynamics textbooks.

Up to this point, these systems were all evaluated on problems known to the system architects. More recently, the HALO project (Barker *et al.* 2004) built knowledge-based systems that contained a few pages of hand-encoded textbook knowledge, to solve a small subset of AP Chemistry style problems. The three HALO teams were evaluated by an external team on new problems across a range of problem types: multiple choice and short answer. The teams were evaluated not just on the ability to produce the correct answer but also on producing human readable explanations. The evaluations of analogical model formulation in this thesis draw upon previous research projects. First, ETS evaluated the AP Physics problem-solving system on largely unseen problems. Second, in the AP Physics work, the transfer levels provide an additional measure of the robustness of the reasoning system instead of pure performance metrics. Third, the major point of departure between previous works on textbook problem-solving and the analogical model formulation approach is the focus on learning. Analogical model formulation expects new domain knowledge to be added through example models provided in these evaluations as sketches and worked solutions.

6.1.5 Transfer Learning

The transfer learning framework is explicit attempt to characterize how well learned knowledge transfers to new domains. There has been an increasing interest in transfer learning within the AI community. Hinrichs and Forbus (2007) describe how analogy can be used to transfer learned qualitative models between scenarios in a turn based strategy game. The cognitive architecture ICARUS achieves near transfer through goal decomposition (Choi *et al.* 2007) and has been augmented with a representation mapping algorithm (Shapiro *et al.* 2008) to handle more distant types of transfer. In analogical model formulation, finding domain mappings is critical to successful transfer. ICARUS requires abstracted domain theories in both the source and target tasks in order to do transfer. By contrast, analogical model formulation operates directly on the problem and a specific example without abstract modeling knowledge of the source or target task, making it more robust.

Within the machine learning community, transfer learning efforts focus have also focused on identifying mappings between scenarios and domains. This work is more relevant to the Domain Transfer via Analogy method discussed next. Therefore, it is discussed in detail in Chapter 8.

6.2 General Discussion

This dissertation seeks to provide AI systems with the flexibility and robustness of human reasoning. People are able to construct useful models of a broad range of scenarios. While an astrophysicist's model of the Earth orbiting the sun would be quite different from a layperson's, both people are able to create a model, use it to make predictions, and communicate their results. This task of model formulation has been formalized in the Qualitative Reasoning community. In this dissertation, I introduce the analogical model formulation method. This method enables model formulation in broad scenarios without complete and correct domain theories. Furthermore, because the majority of the domain knowledge used in analogical model formulation exists as examples, learning can be done by accumulating examples.

This method was evaluated on problems from two complex reasoning tasks: the Bennett Mechanical Comprehension Test and AP Physics-style questions. First and foremost, these tests represent difficult reasoning tasks, and analogical model formulation provided the necessary scenario models to enable a Companion to solve problems. Second, analogical model formulation enabled Companions to learn by accumulating examples in each of these domains. Third, the evaluations of each included measures of robustness. In the BMCT evaluation, three different sets of example sketches were used. This provided variability in the representations of the examples from which the Companion formulated models of new problems. Analogical model formulation successfully reasoned about scenarios drawn by a different person than the creators of the examples that formed the basis of the domain knowledge. Three aspects of the AP Physics evaluation provide measures of the robustness of the underlying system. 1) The first evaluation was designed and administered by the Educational Testing Service without knowledge of the system's problem-solving approach. 2) The evaluation included unseen problems generated from templates. The system designers saw examples from less than 50% of the templates used in the evaluation. 3) The evaluation systematically explored robustness by evaluating transfer across individual transfer levels.

These evaluations demonstrate that analogical model formulation is a promising method for robustly creating the models necessary for complex reasoning tasks from everyday scenarios. Furthermore, unlike traditional model formulation algorithms, systems using analogical model formulation are capable of expanding their domain knowledge by accumulating examples.

6.2.1 Open Questions and Future Directions

There are three important open questions regarding analogical model formulation and its relationship to cognition. The first two directions involve research into the hand coded rules which guide the retrieval process and inference evaluation. The final section discusses moving beyond example based reasoning through generalization.

6.2.1.1 Pragmatic Retrieval Constraints

Analogical model formulation creates scenario models for new situations based upon similar understood examples. This places a heavy burden on selecting the analogous example. As described in Chapter 3, this retrieval process allows for pragmatic retrieval constraints. On comparative analysis problems from the Bennett Mechanical Comprehension Test, the mapping of the retrieved example was evaluated to ensure that the resulting model would constrain the sought after parameters. For example, if the problem is asking for a comparison between the stability of two ladders, and the model resulting from a retrieved example includes no information concerning the stability of the two ladders, then this retrieval is not considered. On outcome questions from the same evaluation, incorrect retrievals were the cause of all problem-solving failures. In the AP Physics evaluation, the Companion would retrieve additional analogues in situations when the entire event structure of the problem was not included in the mapping. This allowed the Companion to make use of multiple example problems when reasoning about a new situation. These constraints improved the Companion's performance in both domains by allowing it to correctly answer questions it would have otherwise answered wrong or been unable to answer.

These pragmatic retrieval constraints were developed for each domain independently. Ideally, a system should be able to introduce new rules for evaluating retrieval based upon its experiences within the domain. First, to understand the space of possible retrieval constraints, a taxonomy should be created. The retrieval constraint in the AP Physics system focused on the mappings relationship to the event structure of the problem. This is a domain dependent heuristic which takes advantage of the fact that Physics problems center on events. This constraint does not apply to solving BMCT problems. Applying analogical model formulation to new domains will improve our understanding of pragmatic retrieval constraints and provide inspiration for methods which learn these constraints automatically.

6.2.1.2 Analogical Inference Evaluation and Adaptation

Another hand-coded aspect of analogical model formulation is the rules for analogical inference evaluation and adaptation. These rules are essential to the performance of analogical model formulation and are currently not a subject of learning in these systems. They are used for three purposes: to determine which types of modeling decisions should be made by the analogy, to verify analogical modeling decisions, and to adapt analogically suggested modeling decisions. Which types of modeling decisions are made via analogy is based upon the amount of domain knowledge available to the problem-solver. On outcome questions from the BMCT, the Companion had a qualitative mechanics domain theory. Therefore, the only analogical modeling decisions it made pertained to abstraction types for the scenario objects and abstract relationships between them. The Companion did not have an abstract domain theory for the comparative analysis or AP Physics problems. Consequently, analogical model formulation relied completely upon the analogy to construct the scenario model.

In regards to individual modeling decisions, the rules for verifying and adapting them are domain dependant. These rules represent important aspects of domain knowledge available to the problem-solver. While no evaluation or adaptation rules were used on the BMCT evaluation, they certainly would have helped. On certain outcome problems, the incorrect abstract qualitative mechanics relationship, `enmeshedGears-Adjacent`, would be inferred for two gears side by side. A rule which recognized the spatial context of the objects involved in these two relationships would be able to verify when these analogical modeling decisions applied. On comparative analysis problems, the Companion failed to arrive at answers when the causal model involved unknown entities or surfaces. Rules which recognized these situations could provide the problem-solver with three different options. The problem-solver could employ rerepresentation (Yan *et al.* 2003) on the current analogue, retrieve a different analogue, or use the spatial relationships from the base to posit where the surface or entity would exist in the target.

Verification and adaptation rules were used in the AP Physics evaluation. There were two verification rules. First, if the context of the modeling decision involved an unmapped planetary body and the problem included a different planetary body, then the inference was rejected. Second, when a value was assumed for a quantity, the units of the assumed value had to be applicable for the quantity type (e.g., m/s are applicable units for a velocity quantity). While these verification rules were rejected a number of incorrect inferences in the evaluation from Chapter 5, they incorrectly rejected a set of valid modeling decisions on 25% of Composing problems. An adaptation rule was employed when the candidate inferences suggesting equations included numbers. In these cases, the problem-solver used the corresponding number from the base. While these rules were necessary to produce the results contained in this dissertation, they are specific to this methods which propose and tweak these rules would be incredibly important for analogical model formulation systems which operate over a range of domains and for an extended period of time.

The application of domain knowledge during at this stage of analogical model formulation involves a number of open questions. Endowing the problem-solver with significant self-modeling capabilities could allow it to determine what types of modeling decisions should be made via analogical model formulation. Regarding verification rules, a straightforward context based generalization approach using SEQL appears very promising. For each analogical inference, construct a case by gathering all of the facts which represent the context of the inference. If the inference leads to positive results, then add the case to generalization of positive inferences. If not, the case would be added to a negative generalization. Then when faced with new analogically suggested modeling decisions, the problem-solver could use its similarity to the positive and negative generalizations as a measure of confidence. The most promising direction for learning adaptation rules involves posing general heuristics which could be programmed for each domain via example. For example, the general version of the heuristic used in AP Physics is to use the entity from the base in certain situations.

6.2.1.3 Generalization of Examples

Analogical model formulation illustrates the flexibility of reasoning and learning directly from examples. Many theories of learning and expertise note the importance of extracting general information from specific examples (e.g. Elio & Anderson 1984). In analogical model formulation, the context of the example is used to verify the analogical model decision. As indicated throughout this work, an intelligent agent can improve upon by generalizing across similar modeling decisions. This will extract the necessary aspects of the scenarios which enable the modeling decision away from the specifics of each scenario resulting in improved verification rules. Using SEQL generalization (Kuehne *et al.* 2000), I have completed initial research on learning participant abstraction modeling decisions from multiple examples in Physics problem solving (Klenk *et al.* 2008).

While using generalization to improving verification rules is an important direction for analogical model formulation, additional theoretical work is required to understand how abstract domain theories are acquired. One approach would look at directly applying generalization techniques to examples to extract schemas for the domain (e.g., physics equations or causal models). The goal of this approach would be to learn abstract domain theories. While analogical model formulation allows for knowledge intensive reasoning about a domain from examples, it would represent a significant step forward in knowledge engineering research to create abstract domain theories directly from examples.

In the next chapter, I introduce domain transfer via analogy which uses cross-domain analogy to reuse known abstract domain theories in learning novel but similar domains.

7 Domain Transfer via Analogy model of Cross-domain Analogy

While within-domain analogy is useful for understanding new situations based on examples, cross-domain analogy allows for the understanding of new domains based on similar, well understood domains. Cognitive science research has shown how cross-domain analogies are useful to scientists in producing paradigm shifts (Gentner *et al.* 1997) (Holyoak & Thagard 1989) (Falkenhainer 1987) and how students use cross-domain analogies learn new domains (Gentner & Gentner 1983). Textbook authors routinely exploit this particular aspect of human adaptability when introducing new concepts. For example, a common physics textbook has 11 example problems in linear kinematics and only two examples in rotational kinematics. To assist the learner the author explains that "the dynamics of rotation is analogous to the dynamics of linear motion" (Giancoli 1991). Students are expected to leverage this analogy to understand the modeling decisions and phenomena in rotational mechanics.

An analogy is a *cross-domain analogy* if the correspondences between the domains include different object types, quantities, and relations. The two requirements for effective cross-domain analogies are a known base domain and a *domain mapping*, which maps the object types, quantities, and relationships between the base and the target domains. This chapter presents the *Domain Transfer via Analogy* (DTA) model of cross-domain analogy. Using domain general retrieval and matching algorithms, DTA constructs a domain mapping from pairs of examples and transfers schemas and control knowledge from a known base domain to create a new target domain theory. Successful cross-domain analogies result in *persistent mappings*. Persistent mappings are correspondences between the base and target domains, enabling DTA to construct complex analogies incrementally from successful matches.

The next section discusses previous cross-domain analogy research in more detail and motivates the need for larger scale process models such as DTA. Next, an overview of the DTA algorithm is presented. This is followed by a description of a kinematics experiment demonstrating how DTA can enable faster learning than a baseline generalization system between the linear and rotational kinematics domains. Next is a second evaluation between the dynamical analogy domains (Olsen 1943) of linear, rotational, electrical and thermal systems. This chapter closes with a discussion of related and future work.

7.1 Previous Cognitive Science Research

7.1.1 Evidence for Cross-Domain Analogy

Cognitive scientists have long argued that cross-domain analogy is an important element in people's adaptability to new situations such as learning new domains quickly (Gentner 2003). Gentner & Gentner (1983) analyzed student's mental models of electricity, identifying many commonalities with models of flowing water and teeming crowds. Consistent with other psychological findings (Spiro *et al.* 1989), these analogies affect future inference and learning in the electrical domain.

In addition to student learning, a number of researchers have studied the role of cross-domain analogy in scientific discovery. Nersessian (1992) found extensive use of analogy in the scientific discoveries of Maxwell and Faraday. Gentner and her colleagues (1997) studied writings from Kepler, tracing his extensive use of analogy while he was developing his model of the solar system. Analogy is an important aspect of contemporary science as well; Dunbar (1995) found analogies prevalent among practicing biologists.

While cross-domain analogy allows for powerful inferences and accelerated learning, the spontaneous generation of useful cross-domain analogies is quite rare. In a series of experiments conducted by Gick and Holyoak (1983), human subjects were unable to transfer a solution schema between variations of the radiation problem. Not only did the subjects fail to spontaneously generate and exploit the analogy, they failed even after performing a variety of exercises to enhance the base scenarios accessibility (e.g. summarization, abstraction and visual depiction).

Given these findings, computational models of cross-domain analogy should encompass the following phenomena. First, cross-domain analogy supports complex inference. This goes beyond answering questions about a given scenario. The cross-domain analogies studied above result in new theories which people use to shape their reasoning about a domain. Second, spontaneous retrieval, generation and application of successful cross-domain analogies is rare. Third, a provided base domain, feedback and advice often facilitate cross-domain analogy.

7.1.2 Previous computational models

Drawing upon the evidence from cognitive psychology, a number of researchers have created computational models of cross-domain analogy. The most closely related work on cross-domain analogical learning is Falkenhainer's PHINEAS (1988). PHINEAS used comparisons of (simulated) behavior to create an initial cross-domain mapping that was subsequently used to create an explanation for the behavior in the new domain. DTA differs from PHINEAS in four important ways. First, DTA transfers equation schemas and control knowledge. Control knowledge guides the application of equation schemas, allowing the system to solve complex quantitative problems. Second, DTA uses domain general models of analogical retrieval and matching. While both PHINEAS and DTA use SME for analogical matching, PHINEAS did not have a psychologically plausible model of memory retrieval. DTA employs the MAC/FAC model of similarity-based retrieval, which has been used to account for several psychological phenomena and make new predictions concerning memory retrieval (Forbus 2001). Third, DTA is evaluated across 32 problems from four different domains, representing a significant increase in evaluation scale over previous systems. Finally, DTA uses persistent mappings to incrementally construct complex cross-domain analogies as it encounters new examples from the target domain.

This chapter, and this thesis, argues for the integration and evaluation of analogical processes within larger reasoning tasks. Here, DTA is integrated in the Companions cognitive architecture and evaluated on a range of quantitative physics problems. Problems drawn from kinematics, mechanics, electricity, and thermodynamics provide the rich set of domains necessary to better understand the effectiveness and implications of DTA for artificial intelligence and as a cognitive model.

7.2 Domain Transfer via Analogy Process Model

7.2.1 Overview

Domain Transfer via Analogy is a model of cross-domain analogical learning. DTA assumes a known base domain and an example from the new domain. The base domain consists of the abstract schemas and control knowledge used in reasoning and examples, or worked solutions, which include instantiations of the domain theory schemas. The process begins after the system fails to solve a problem in a new domain. Using a worked solution to this failed problem, DTA performs a cross-domain analogy to learn equation schemas and control knowledge in the target domain. The process is depicted in Figure 26 and consists of four steps: learning the domain mapping, initializing the target domain theory, extending the target domain via cross-domain analogy, and verifying the learned knowledge.

Figure 26: DTA algorithm

DTA assumes domain theories consisting of structured predicate calculus representations.

Therefore, any type of domain knowledge which can be stored in this form should be transferrable. In this thesis, the knowledge transferred consists of equation schemas, stored as *encapsulated histories* (Forbus 1984), and control knowledge.

This section begins with a discussion of the representations and problem-solving system. This is followed by a detailed description of the algorithm and an example.

7.2.2 Representations

Once again, the representations used are in CycL, the predicate calculus language of the ResearchCyc knowledge base (KB) (Matuszek *et al.* 2006). The representations use the ontology of ResearchCyc, plus our own extensions. These concern Qualitative Process theory (Forbus 1984) and problem-solving strategies, and are small compared to the 30,000+ concepts and 8,000+ predicates already defined in the KB. Thus, objects, relations, and events that appear in physics problems such as “rotor”, “car”, and “driving” are already defined in the ontology for us, rather than being created specifically for this project. This reduces the degree of tailorability in the experiments.

7.2.2.1 Example Problem and Worked Solution

The problems were selected from the following physics resources: (Shearer *et al.* 1971), (Giancoli 1991), (Ogata 1997), (Fogiel 1994) and ("Hooke's Law, Work and Elastic Potential Energy" 2009). Unlike the AP Physics transfer learning experiment described in Chapter 5, these representations were created internally by hand. The same principles were applied to create the problem and worked solution representations. Specifically, the entities in the problems were to be represented at the level described in the problem without abstraction. The problem representations are intended to be direct translations into predicate calculus from natural language problem statements, without any abstraction or reasoning. Consider the problem of "How long does it take a car to travel 30m if it accelerates from rest at a rate of 2 m/s^2?" (Example problem 2-6, p. 26, Giancoli 1991). This problem is represented in our system as a case of nine facts, shown in Figure 27. The first two facts define the entities in the problem, a transportation with land vehicle event, `Acc-2-6`, and an automobile, `Car-2-6`. The next 4 facts describe the motion of `Car-2-6` during `Acc-2-6`. The last 3 facts describe the question of the problem and how these facts are grouped together in a case.

```
(isa Car-2-6 Automobile)
(isa Acc-2-6 TransportWithMotorizedLandVehicle)
(objectStationary (StartFn Acc-2-6) Car-2-6)
(primaryObjectMoving Acc-2-6 Car-2-6)
(valueOf ((QPQuantityFn DistanceTravelled) Car-2-6 Acc-2-6)
  (Meter 30))
(valueOf (MeasurementAtFn ((QPQuantityFn Acceleration) Car-2-6) Acc-2-6)
  (MetersPerSecondPerSecond 2))
(isa Gia-Query-2-6 PhysicsQuery)
(hypotheticalMicrotheoryOfTest Gia-Query-2-6 Gia-2-6)
(querySentenceOfQuery Gia-Query-2-6
   (valueOf ((QPQuantityFn Time-Quantity) Acc-2-6) Duration-2-6)))
```

Figure 27: Example problem 2-6 representation

The worked solutions for these experiments use the same ontology of steps as the AP Physics worked solutions in Chapter 5. Once again, the worked solutions are represented at the level of explained examples found in textbooks. They are neither deductive proofs nor problem-solving traces produced by our solver. In instances where the problems did not have worked solutions, they were created to conform to existing worked solutions. Below is an English rendering of the worked solution for example problem 2-6:

1. Categorize the problem as a constant acceleration linear mechanics problem
2. Instantiate the distance by velocity time equation ($d = v_i t + .5at^2$)
3. Because the car is stationary at the start of the event infer that its velocity is zero ($v_i = 0$ m/s)
4. Solve the equation for t ($t = 5.8s$)

```
(isa Gia-2-6-WS-Step-2 SubstitutingBindingsForVariables)
(hasSolutionSteps Gia-2-6-WorkedSolution Gia-2-6-WS-Step-2)
(priorSolutionStep Gia-2-6-WS-Step-2 Gia-2-6-WS-Step-1)
(stepUses Gia-2-6-WS-Step-2 (primaryObjectMoving Acc-2-6 Car-2-6))
(stepUses Gia-2-6-WS-Step-2 (abstractionForObject Car-2-6 PointMass))
(stepUses Gia-2-6-WS-Step-2
          (abstractionForObject Acc-2-6 ConstantTranslationAccelerationEvent))
(stepUses Gia-2-6-WS-Step-2 (isa Gia-Query-2-6 PhysicsProblem-ConstantAcceleration))
(stepUses Gia-2-6-WS-Step-2
  (equationFormFor DistanceByVelocityTime-1DConstantAcceleration
   (mathEquals ((QPQuantityFn DistanceTravelled) ?OBJ ?INTERVAL)
    (PlusFn
     (TimesFn (MeasurementAtFn ((QPQuantityFn Speed) ?OBJ) (StartFn ?INTERVAL))
              ((QPQuantityFn Time-Quantity) ?INTERVAL))
     (TimesFn (MeasurementAtFn ((QPQuantityFn Acceleration) ?OBJ) ?INTERVAL)
              (SquaredFn ((QPQuantityFn Time-Quantity) ?INTERVAL)) 0.5)))))
 (stepResult Gia-2-6-WS-Step-2
  (equationForSolution Gia-Query-2-6
   (mathEquals ((QPQuantityFn DistanceTravelled) Car-2-6 Acc-2-6)
    (PlusFn
     (TimesFn (MeasurementAtFn ((QPQuantityFn Speed) Car-2-6) (StartFn Acc-2-6))
              ((QPQuantityFn Time-Quantity) Acc-2-6))
     (TimesFn (MeasurementAtFn ((QPQuantityFn Acceleration) Car-2-6) Acc-2-6)
              (SquaredFn ((QPQuantityFn Time-Quantity) Acc-2-6)) 0.5)))))
```

Figure 28: Representation for worked solution step 3 from example problem 2-6

The entire worked solution consists of 38 facts. Figure 28 shows the predicate calculus representation for the second worked solution step. There are two differences from the worked solutions in Chapter 5. The `abstractionForObject` statements relate entities in the problem with participant abstraction types in the domain theory. In this case, `Car-2-6` is modeled as a `PointMass` and `Acc-2-6` is considered a `ConstantTranslationAccelerationEvent`. Also, the first argument to `equationFormFor` is the encapsulated history type from which the equation comes. In this case, the encapsulated history representing the equation being instantiated in this step is `DistanceByVelocityTime-1DConstantAcceleration`. While these are additional constraints on the worked solution representations, these connections between the worked solutions and domain theories function simply as an internal representation for the explanations of physics problem solutions. Recall that DTA uses analogies between worked solutions from different domains to learn a domain mapping. This additional information helps structure the domain theory for the target during transfer.

7.2.2.2 Domain Theories for Problem-solving

The domain theories consist of encapsulated histories representing equations and control knowledge to guide problem-solving. Encapsulated histories act as equation schemas with participants, conditions, and consequences. During problem solving, the system instantiates applicable encapsulated histories to determine which equations are available.

```
(def-encapsulated-history VelocityByTime-1DConstantAcceleration
  :participants
  ((theObject :type PointMass)
   (theEvent  :type Constant1DAccelerationEvent))
  :conditions
  ((primaryObjectMoving theEvent theObject))
  :consequences
  ((equationFormFor VelocityByTime
   (mathEquals
    (AtFn (Speed theObject) (EndFn theEvent)
    (PlusFn (AtFn (Speed theObject) (StartFn theEvent))
            (TimesFn (AtFn (Acceleration theObject) theEvent)
                     (Time-Quantity theEvent)))))))
```

Figure 29: Encapsulated history for $v_f=v_i+at$ in condensed form

Figure 29 shows the definition for the encapsulated history representing the equation $v_f=v_i+at$, velocity as a function of time. There are two participants, `theObject` and `theEvent`, which must satisfy their type constraints, the abstractions `PointMass` and `Constant1DAccelerationEvent`, respectively. Furthermore, the conditions of the encapsulated history must be satisfied in order to instantiate it and conclude its consequences. In this case, it is necessary that `theObject` be the object moving in `theEvent`. The compositional modeling language (Bobrow *et al.* 1996) form shown in Figure 29 is automatically translated into a set of predicate calculus facts for use in the system. This knowledge is necessary for the physics problem-solver to successfully answer questions.

DTA enables the learning of domain theories that are represented with schema-like knowledge. In these experiments, DTA learns the encapsulated histories of a new domain via cross-domain analogy, in terms of participants, conditions and consequences. The validity of the learned domain theories is evaluated by using it to solve physics problems. Consequently, it is necessary to understand how the problem-solving system works. The physics problems in this work all ask for the values of specific quantities. If there is an equation in which the only unknown parameter is the sought quantity, the system simply solves the equation to determine the answer. For more complicated problems, the problem solver makes use of control knowledge.

Pisan's work on engineering thermodynamics problem-solving (1998) demonstrated the importance of control knowledge for producing expert-like solutions, which are necessary for intelligent tutoring systems and cognitive modeling experiments. Drawing upon distinctions from Pisan, the following equation selection and manipulation strategies were used: frame equations, decomposition of composite quantities, and symmetric substitution.

The problem-solving strategies use *frame equations* to set up problem-solving. Frame equations are central to their domains and serve as starting points for problem-solving. For example, the conservation of momentum, $p_{before} = p_{after}$, is a frame equation for mechanics. For this evaluation, domain specific control knowledge concerning frame equations and preferences between different frame equations has been added to the domain theories. In addition to the encapsulated histories representing equations, this control knowledge must be transferred as well to solve problems in the target domain theory. Figure 30 shows the facts indicating that conservation of momentum and the linear mechanics work energy theorem are frame equations, and conservation of momentum should be tried first in scenarios where both are applicable.

```
(frameEquationType ConservationOfLinearMomentum)
(frameEquationType WorkEnergyTheorem-LM)

(preferFrameTypeOver ConservationOfLinearMomentum WorkEnergyTheorem-LM)
```

Figure 30: Control knowledge statements indicating frame equations and preferences

After selecting a frame equation, the problem-solver uses strategies of symmetric substitution (e.g., given the work energy equation, $W=Ke_{final} - Ke_{initial}$, substitute each of the Ke's with $.5mv^2$) and decomposition (e.g., substitute F_{net} with the sum of the component forces). Once the only unknown in the equation is the sought quantity, the system invokes algebra routines based on the system described in Forbus & De Kleer (1993).

As described in detail in Chapter 5, in addition to equations, solving physics problems requires a number of modeling decisions. While the preceding chapters described how analogical model formulation enables a system to make these decisions in broad domains, the systems in these experiments use hand coded rules to make approximations and assumptions. Therefore, the system's performance depends solely on the encapsulated histories and control knowledge of the domain theories. While this is sufficient for the goals of this experiment, analogical model formulation and DTA could operate complementarily within a single intelligent agent. Integrating analogical model formulation and DTA is an important direction for future work.

7.2.3 Algorithm

Using worked solutions, DTA learns a domain mapping and transfers equation schemas and control knowledge from a known base domain to a new target domain theory. After the system fails to solve a problem from a new domain, DTA is invoked by providing the worked solution to that problem. When DTA is successful, new encapsulated histories and control knowledge are added to the target domain theory along with a set of persistent mappings between the domains.

7.2.3.1 Step 1: Learn Domain Mapping

Because different domains are represented with different predicates and conceptual types, a domain mapping is essential to successful cross-domain analogies. DTA learns the domain mapping through an analogical mapping between worked solutions from the two domains. Using the worked solution for the failed problem as a probe, MAC/FAC selects an analogous worked solution from a case library containing worked solutions from the known domain. Next, SME creates an analogy between the retrieved worked solution and the worked solution to the failed problem. If there has already been a successful cross-domain analogy between these domains, there will be persistent mappings. The persistent mappings are required correspondences between the base and target worked solutions during the retrieval and analogical matching process.

The analogy between the worked solutions results in up to three mappings. Each mapping contains a set of correspondences, which are used to form the domain mapping. DTA sorts the mappings by their structural evaluation scores. Beginning with the best mapping, each correspondence is added to the domain mapping when the base entity (e.g., numbers, types, quantities, and relations) is mentioned in the base domain theory. This process continues with the rest of the mappings by adding correspondences to the domain mapping that do not violate the one-to-one constraint. For example, if the best mapping had a correspondence between `PointMass` and `RigidObject` and the second mapping had a correspondence between `PointMass` and `Ball`, the domain mapping would only include the mapping between `PointMass` and `RigidObject`. The reason for combining multiple mappings is that the global mapping constraints in SME may preclude useful local correspondences.

7.2.3.2 Step 2: Initialize the Target Domain

DTA uses the domain mapping to initialize new domain theory elements. For each encapsulated history from the base domain mentioned in the domain mapping, DTA attempts to create a corresponding encapsulated history in the target domain. Before transferring the encapsulated history, all of the quantities and types mentioned in the encapsulated history must appear in the domain mapping. If they do, then using the domain mapping, DTA proceeds by substituting concepts in the base encapsulated history with the corresponding concepts from the target domain. The resulting encapsulated history is then added to the target domain theory.

7.2.3.3 Step 3: Extend the Target Domain Theory

After initializing the target domain theory, DTA extends the domain theory through a second cross-domain analogy. This time the analogy is between the base and target domain theories themselves. The domain theories consist of the facts representing the encapsulated histories and the control knowledge. DTA constrains this analogy with two sets of mapping constraints. To ensure consistency, each element of the domain mapping becomes a required correspondences constraint. To prevent non-analogous target items from interfering with the mapping, each encapsulated history in the target which does not participate in the domain mapping is prevented from mapping to any of the base encapsulated histories.

This analogy results in a set of candidate inferences, i.e., conjectures about the target, using partially mapped expressions from the base. Because the base domain theory is made up of encapsulated histories, these candidate inferences describe corresponding encapsulated histories in the target domain theory. The corresponding encapsulated histories in the target are represented by *skolem entities* in the candidate inferences. Recall from Chapter 2 that SME creates skolem entities for entities appearing in the base which have no correspondence in the mapping. To complete the cross-domain analogy, DTA adds corresponding entities to the target and extends the mapping with these new target entities and correspondences.

Using the resulting candidate inferences, DTA constructs new encapsulated histories and control knowledge for the target domain theory. Before adding the encapsulated histories to the target domain theory, it is necessary to resolve other skolem entities mentioned in the candidate inferences concerning new encapsulated histories. If all the encapsulated history's participant types and quantities are included in the domain mapping, then there will not be any other skolems, and DTA adds these candidate inferences into the target domain theory. The control knowledge from the base domain also results in candidate inferences. Each candidate inference without any skolem entities containing control knowledge is assumed into the target domain theory as well.

7.2.3.4 Step 4: Verification

While powerful, cross-domain analogies are risky and frequently contain invalid inferences. Therefore, DTA verifies the newly proposed knowledge by retrying the problem whose failure began the entire process. If this problem is solved correctly, DTA assumes that the newly acquired domain theory is correct. Otherwise, DTA forgets both the new domain theory and the domain mapping resulting from the worked solution comparison. At this point, the system using DTA could invoke the process again after removing the base worked solution from the case library. This number of retrievals an agent should perform with DTA depends in principle on a self-model of its own knowledge and a model of its interactions with the world. Currently, this number is hardcoded for each system.

After a successful transfer, the encapsulated histories and control knowledge are available for reasoning about future problems in the target domain. Also, the mapping from the cross-domain analogy is added to the domain mapping and stored as a persistent mapping. This will assist in future cross-domain analogies between the base and target domains.

7.2.4 Example

To better understand how DTA uses multiple cross-domain analogies to transfer domain theories, we describe an example of how it learns rotational kinematics through an analogy to linear kinematics. The system begins with a linear kinematics domain theory and worked solutions. Because the system has no encapsulated histories of rotational kinematics, it fails to solve the following problem, "Assuming constant angular acceleration, through how many turns does a centrifuge rotor make when accelerating from rest to 20,000 rpm in 5 min?" Because the system failed, DTA is invoked and the worked solution to this problem is provided.

In order to create a domain mapping between linear and rotational kinematics, DTA creates an analogy between a linear kinematics worked solution and the provided rotational kinematics worked solution. Using the provided rotational kinematics worked solution as a probe, DTA uses MAC/FAC to retrieve an analogue from the systems case library of linear kinematics worked solutions. In this case, the analogous worked solution retrieved is for the problem discussed previously, "How long does it take a car to travel 30m if it accelerates from rest at a rate of $2m/s^2$?" DTA uses an analogy between these two worked solutions to produce the domain mapping necessary for cross-domain analogy. In this case, the mathematical relationships are isomorphic, $d = v_i t + .5at^2$ and $\theta=\omega_i t + .5\alpha t^2$, which places the quantities between the domains into correspondence. It should be noted that SME handles partial matches, allowing correspondences to be created even when the mathematical relationships in the worked solutions being compared are not completely isomorphic. Making use of the KB's ontology, the minimal ascension places `primaryObjectMoving` in correspondence with objectRotating based upon the shared common ancestor of objectMoving. Next, the mapping's correspondences are extracted to create the domain mapping, a subset of which appears in Table 8.

Base Item	Target Item
PointMass	RigidObject
ConstantLinear-AccelerationEvent	ConstantRotational-AccelerationEvent
primaryObjectMoving	objectRotating
Acceleration	AngularAcceleration
Speed	RateOfRotation
DistanceTravelled	AngularDistTravelled
Time-Quantity	Time-Quantity
DistanceByVelocityTime-1DConstantAcceleration	DistanceTime-Rotational

Table 8: Resulting Domain Mapping

After learning the domain mapping, the next step is to initialize the target domain theory. This is done by searching the domain mapping for encapsulated histories from the base domain. In this example, DistanceByVelocityTime-1DConstantAcceleration is found. In order to transfer this encapsulated history to the rotational kinematics domain theory, all of its participant types and quantities must participate in the domain mapping. This encapsulated history contains two participant types, PointMass and ConstantLinear-AccelerationEvent and four quantities, Acceleration, Speed, Time-Quantity and DistanceTravelled. All of these appear in the domain mapping, allowing DTA to transfer this encapsulated history to the target domain. For each fact in the base domain theory mentioning DistanceByVelocityTime-1DConstantAcceleration, DTA substitutes all subexpressions based upon the domain mapping. This results in a new encapsulated history, `DistanceTime-Rotational`, which represent the rotational kinematics equation, $\theta=\omega_i t + .5\alpha t^2$, and was mentioned in the rotational kinematics worked solution. DTA initializes the target domain theory by adding this new encapsulated history.

Next, DTA extends the new domain theory with a cross-domain analogy between the base and target domain theories themselves. To maintain consistency, this analogy is constrained with the domain mapping acting as required correspondence constraints. The 41 facts describing the linear mechanics encapsulated histories make up the base, and the 6 facts of the newly initialized rotational mechanics domain theory are the target. As expected, the sole target encapsulated history maps to the corresponding linear mechanics encapsulated history. This mapping includes the quantities, conditions and types in these encapsulated histories. These correspondences result in candidate inferences involving the facts of the other encapsulated histories from the base. DTA uses these candidate inferences to infer rotational kinematics encapsulated histories. For example, the candidate inference shown in Figure 31 suggests that there is an encapsulated history in the target analogous to the `VelocityByTime-1DConstantAcceleration` linear mechanics encapsulated history. This candidate inference states that the suggested encapsulated history has the operating condition that the object must be rotating during the event. The `AnalogySkolemFn` expression indicates that there was no corresponding entity in the target. Therefore, to extend the target domain theory, DTA creates entities for all the analogy skolems, i.e. turning `(AnalogySkolemFn VelocityByTime-1DConstantAcceleration)` into `EHType-1523`, and assumes these facts into the rotational mechanics domain theory. During this step, three new encapsulated histories join the initialized encapsulated history in the rotational kinematics domain theory.

```
(qpConditionOfType
 (AnalogySkolemFn VelocityByTime-1DConstantAcceleration)
 (objectRotating :theEvent :theObject))
```

Figure 31: Candidate inference suggesting a condition of an encapsulated history type

The final step is to verify the validity of the learned knowledge. This is done by attempting to solve the original problem again. Since the worked solution contains the answer, it is simply compared against the computed answer. If they match, then the learned knowledge is assumed to be valid. If the system gets the problem wrong, it takes two steps. First, it erases the domain mapping and the inferred encapsulated histories in the rotational mechanics domain theory. Second, it repeats the entire process one more time, with the next-best worked solution from the case library. Abandoning, rather than debugging, a mapping may be unrealistic. One aspect of future work involves exploring diagnosis and repair of faults in learned domain theories and domain mappings. In this case, the learned encapsulated histories allow the system to answer the problem correctly.

Recall at the beginning the system had zero encapsulated histories in its rotational kinematics domain theory. While the rotational kinematics worked solution contained an example of one encapsulated history, after employing DTA, the system learned schemas for four rotational kinematics equations. This knowledge is now available for future problem-solving episodes. The next two sections describe evaluations of DTA across a variety of physics domains.

7.3 Kinematics Experiment

The first experiment was conducted using the rotational and linear kinematics domains. This experiment compares the learning rates of a system using DTA to a baseline *spoon-fed* system. Each system uses the exact same problem-solver. At the time of this experiment, the DTA algorithm had not been implemented on the Companions cognitive architecture.

Instead of providing the spoon-fed system with the worked solution after failing a problem, it is provided with the general encapsulated histories needed to solve that specific problem. Therefore, after failing a problem, the baseline system can solve that problem and any other problems which use the same equations. Both systems begin with the necessary rules for problem-solving strategies and modeling decisions. The DTA system was allowed to try an additional worked solution from the base domain theory, if it failed to verify domain knowledge resulting from the first iteration of the algorithm.

The experiment consists of two parts: one to evaluate learning of rotational kinematics by an analogy with linear kinematics, and the other to evaluate learning of linear kinematics based upon rotational kinematics.

7.3.1 Materials

Linear Kinematics

a) How long does it take a car to travel 30m if it accelerates from rest at a rate of 2m/s^2?

b) We consider the stopping distances from a car, which are important for traffic safety and traffic design. The problem is best deal with in two parts: (1) the time between the decision to apply the brakes and their actual application (the "reaction time"), during which we assume a=0; and (2) the actual braking period when the vehicle decelerates. Assuming the starting velocity is 28m/s, the acceleration is -6.0m/s^2, and a reaction time of .5s, What is the stopping distance?

c) A baseball pitcher throws a fastball with a speed of 44m/s. It has been observed that pitchers accelerate the ball through a distance of 3.5m. What is the average acceleration during the throwing motion?

d) Suppose a ball is dropped from a 70m tower how far will it have fallen after 3 seconds?

e) A jetliner must reach a speed of 80m/s for takeoff. The runway is 1500m long, what is the constant acceleration required?

Rotational Kinematics

a) Through how many turns does a centrifuge rotor make when accelerating from rest to 20,000 rpm in 5 min? Assume constant angular acceleration

b) A phonograph turntable reaches its rated speed of 33 rpm after making 2.5 revolutions, what is its angular acceleration?

c) Through how many turns does a centrifuge rotor make when accelerating from rest to 10,000 rpm in 270 seconds? Assume constant angular acceleration

d) An automobile engine slows down from 3600 rpm to 1000 rpm in 5 seconds, how many radians does the engine turn in this time?

e) A centrifuge rotor is accelerated from rest to 20,000 rpm in 5 min, what is the averaged angular acceleration?

Figure 32: Kinematics evaluation materials

The problems for both domains are listed in Figure 32. The problem and worked solution representations were created in the manner described in Section 7.2.2.1. For linear kinematics, the representations for the problems and worked solutions had means of 15.4 and 52 facts respectively. These representations included 38 different types and 52 unique relations. For rotational kinematics, the problem and worked solution representations had a mean of 9.8 and 46.2 facts respectively. These representations included 21 types and 33 relations. 9 types and 26 relations appear in problems and worked solutions from both domains. Each domain theory consists of four encapsulated histories each representing a different kinematics equation.

The design of this evaluation seeks to answer the following questions. First, can DTA transfer the encapsulated histories to solve problems in new domains? Second, can DTA learn better than a baseline system which is incrementally given the correct domain theory? Third, how important is the verification of the learned domain theory and the ability to retrieve additional analogues affect performance?

7.3.2 Learning Rotational Kinematics from Linear Kinematics

In the first part, linear kinematics is the known base domain and rotational kinematics is the target domain. The problems from the target domain were presented a series 120 of trials representing every possible ordering of the five rotational kinematics problems. Because rotational kinematics is the target domain, both systems begin the trial without any rotational kinematics encapsulated histories. Therefore, neither system should be able to solve the first problem on any of the trials. During each trial, when the DTA system fails to solve a problem, the DTA method is invoked with the worked solution to that problem. If DTA fails to verify the learned knowledge on the first iteration, DTA is invoked again with the second retrieval from the case library.

After each problem in the baseline condition, the system is given the necessary encapsulated histories to solve the problem. The encapsulated histories, either learned by cross-domain analogy in the DTA system or provided to the baseline system, allow the system to answer future rotational kinematics questions. At the end of each trial, the system's knowledge was reset.

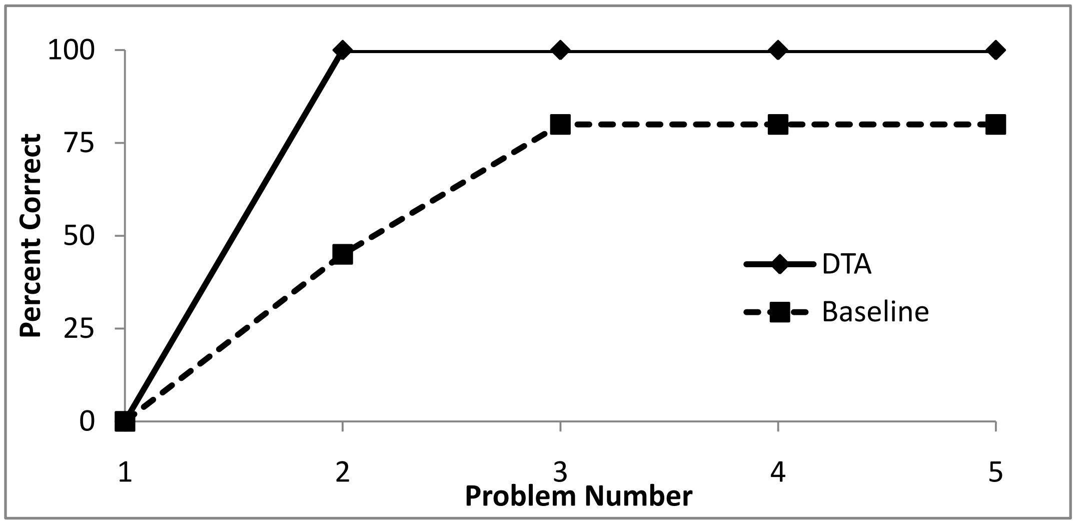


Figure 33: Rotational mechanics learning curves

Figure 33 compares the rotational kinematics learning rates for the DTA and baseline conditions averaged across 120 trials. The DTA system exhibited perfect transfer. That is, after studying just one worked solution, the analogy system was able to score 100% on the rest of the problems. Because the DTA system performed perfectly, there were only 120 transfer attempts, all of which succeeded. Of these, 72 attempts (60%) required retrieving a second worked solution to generate a successful domain mapping. This highlights the importance of verifying the knowledge learned from the cross-domain analogy. The performance in the baseline condition was markedly worse than DTA. After one problem, the baseline system was only able to solve the next problem 45 percent of the time. Also, the baseline system's ceiling was at 80 percent. This was due to the fact it was unable to solve rotational kinematics problem 'b' from Figure 32 regardless of what problems it had already seen, because none of the other problems use the same equation. DTA overcomes this through the analogy between the domain theories themselves. This allows DTA to infer equations not mentioned explicitly in the worked solution from the target domain. After the first problem in each of the DTA trails, all four linear kinematics encapsulated histories were transferred to the rotational kinematics. Therefore, the DTA system scored 100% on each of the subsequent problems.

7.3.3 Learning Linear Kinematics from Rotational Kinematics

This experiment followed the same form as the previous experiment but with the opposite base and target domains. Here, rotational kinematics is the base domain and linear kinematics is the target domain. Once again, the learning rates between the baseline system and the DTA system are compared.

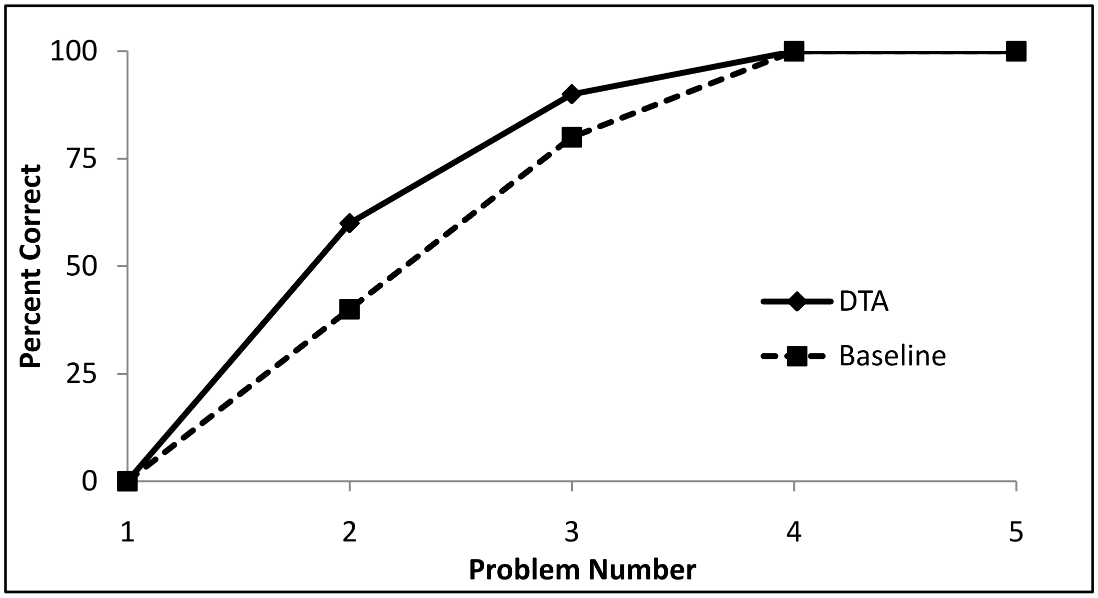


Figure 34: Linear mechanics learning curves

Once again, the DTA system outperformed the baseline system. Figure 34 graphs the learning curves of the two conditions. After the first problem, the DTA system got the next problem correct 60% of the time, compared with the 40% performance of the baseline. After seeing two worked solutions, the analogy system scored 90% on the third problem where the baseline system scored 80%. On the fourth and fifth problems of each trial, both systems performed at a ceiling of 100%. The baseline condition was able to achieve a ceiling of 100% as every equation required was used by at least two problems. In the analogy condition, DTA was unable to transfer the correct domain theory for two linear kinematics problems. This led to 180 transfer attempts, 120 after the first problem of the trial, 48 after failures on the second problem and 12 after the third problem. Out of the 180 attempts, 120 (66%) were successful. Of the successful attempts, none of them required using an additional retrieval.

7.3.4 Discussion

In both parts of the experiment, the domain transfer via analogy system outperformed the spoon-fed baseline system. DTA learned faster and achieved a ceiling that was the same or higher than the baseline. An analysis of the linear kinematics transfer failures indicates that increasing the complexity of sub-event structures and time intervals increases the difficulty in generating an appropriate domain mapping. Linear kinematics problems 'b' and 'd' both contained such structures. One requirement for successful transfer between these domains is that an `objectRotating` statement in the rotational kinematics worked solution must correspond with a `primaryObjectMoving` statement in the linear kinematics worked solution. In order for this to occur, the entities which make up these expressions must already be in alignment. Given the structure of linear mechanics problems 'b' and 'd', the events listed in the step uses statements may differ from the events referenced in the quantities and equations. Therefore, one aspect of future work is to incorporate *rerepresentation* strategies (Yan *et al.* 2003) to bring these worked solutions into better alignment with the analogous rotational kinematics worked solutions. The added complexity of the linear kinematics problems also slowed the baseline learning system.

Another interesting phenomena illustrated by these experiments is the difference in the utility of retrieving an additional worked solution if the first one fails. In these experiments, more than one of the base worked solutions could potentially serve as an analogue to create the domain mapping. In the learning rotational kinematics experiment, retrieving additional worked solutions was critical to the DTA system's performance. On the other hand, in the experiment learning linear kinematics, the additional retrievals never produced an adequate domain mapping. This provides evidence that the decision to retrieve additional analogues should be controlled by the system based upon its goals. For example, if the system did not have other tasks requiring attention, then it could continue retrieving additional analogues until it achieved a successful transfer.

7.3.4.1 Limitations of the Kinematics experiment

While these results are very encouraging, there are a number of weaknesses in this experiment:

1. Linear and rotational kinematics are very similar domains
2. The domains were limited in depth to only kinematics
3. Only encapsulated histories were transferred
4. The domains did not contain any non-analogous aspects

These simplifications informed the investigation of applying DTA to a broader range of phenomena in the next section.

7.4 Dynamical Analogies

To address the four limitations of the kinematics experiment, a new corpus of domains was created use Olsen's *Dynamical Analogies* (1943) as a starting point. The following four domains were used: linear mechanical, rotational mechanical, electrical, and thermal systems. The dynamical analogy domains differ from the kinematics domains used in the previous experiment in several important dimensions. First, the new domains include superficially dissimilar domains such as mechanical and electrical systems. Second, the dynamical analogy domains cover more phenomena than the kinematics scenarios. Therefore, a single cross-domain analogy between two worked solutions does not include all of the entities from the base and target domain theories. Third, the complexity of the problems was increased, requiring control knowledge to be included in the domain theories. Fourth, each of these domains has non-analogous elements (e.g., equations to calculate the moment of inertia of a point in rotational mechanics and nothing corresponds to kinetic energy in thermal systems). Table 9 aligns the analogous quantities from the domains. This will stress the verification aspects of DTA.

Table 9: Dynamical analogy domains aligned by analogous quantities

Linear	Rotational	Electrical	Thermal
Force [F]	Torque [T]	Voltage across [V]	Temperature difference[T]
Speed [v]	Rate of rotation [ω]	Electrical current level [i]	Heat flow rate [q]
Linear deflection [x]	Rotational deflection [β]	Electrical charge [q]	Thermal energy [H]
Mass [F=ma]	Moment of inertia [T=Jα]	Inductance [V=Ldi/dt]	n/a
Linear momentum [p=mv]	Rotational momentum [p=J ω]	n/a	n/a
Linear kinetic energy [Ke=.5mv^2]	Rotational kinetic energy [Ke=.5 J ω^2]	Inductance energy [Energy=.5Li^2]	n/a
Linear compliance [F=x/C]	Rotational compliance [T= β/C]	Electrical capacitance [V=q/C]	Thermal capacitance [T=H/C]
Translational elastic potential [EPE=.5(x^2)/C]	Rotational elastic potential [EPE=.5(ω ^{2})/C]	Capacitance energy [Energy=.5(q^2)/C]	n/a
Linear damping [F=bv]	Rotational damping [T=Dω]	Electrical resistance [V=q/R]	Thermal resistance [q=T/R]
Power [P=Fv]	Power [P=Tω]	Power [P=Vi]	n/a

7.4.1 Changes to the underlying system

The challenges introduced by the dynamical analogy domains required a number of alterations to the system.

7.4.1.1 Companions Cognitive Architecture

For this experiment, the DTA algorithm was implemented on the Companions Cognitive Architecture (Forbus *et al.* 2008). Implementing DTA on a Companion provides two benefits. First, as in the previous experiments, the separation of MAC/FAC from problem-solving operations saves memory space allowing for additional debugging information during development. Second, this implementation allowed for the reuse of the experimental infrastructure created for the AP Physics evaluation. The experimental scripts provide a concise way of defining experiments and collecting data.

7.4.1.2 Control Knowledge

The problems from dynamical analogies domains require more complex algebraic manipulations than those from the kinematics experiment. For example, the problems include more entities, leading to substantially larger models. More critically, the straightforward recursive equation solving strategies used in kinematics experiment are insufficient for these problems. For example, the rotational mechanics problem Gia-8-12 requires the problem-solver to cancel the unspecified mass quantity from the momentum before and after. Therefore, a number of the problems in this experiment require control knowledge for the problem-solver to arrive at the correct solution.

7.4.1.3 Persistent Mappings

In the kinematics experiment, a successful interaction of DTA resulted in the entire kinematics domain theory being learned. Given the breadth of the dynamical domains, no single worked solution will include all of the quantities and abstraction types mentioned in that domain. This evaluation employs persistent mappings to incrementally build the cross-domain analogy over multiple worked solutions. Persistent mappings consist of correspondences between the base and target domains which have proven useful in previous transfer episodes. These mappings have implications for future learning in the target domain. For concreteness, consider use the problems in Figure 35 to learn about electrical systems via cross-domain analogy with linear mechanics.

Linear Mechanics	Electrical Systems
Gia-4-1: Estimate the net force needed to accelerate a 1500kg race car at -5 m/s^2?	Gia-21-37-P: A 75-V emf is induced in a .3 H coil by a current that rises uniformly from 0 to I in 2ms. What is the value of I?
Phy-1: When a 13.2-kg mass is placed on top of a vertical spring, the spring compresses 5.93 cm. Find the force constant of the spring.	Gia-17-24-P: How much charge flows from a 12-V battery when it is connected to a 7.5 micro-F capacitor?

Figure 35: Example problems from the linear mechanics and electrical system domains.

The Companion is provided a case library of worked solutions to the linear mechanics problems. DTA begins after the Companion fails to solve the first problem, Gia-21-37-P, because it does not have any encapsulated histories about electrical systems. After retrieving the worked solution to problem Gia-4-1 from memory and transferring the equation schema for the definition of self inductance, V=L*di/dt, the Companion verifies the knowledge by successfully solving Gia-21-37-P. After a successful transfer, DTA stores the mappings from the analogy between the domain theories as persistent mappings. In this case, the persistent mappings include correspondences between the domain theories including quantities (e.g., `ForceQuantity` and `VoltageAcross`), abstraction types (e.g., `PointMass` and `Inductor-Idealized`), relations (e.g., `objectTranslating` and `objectActedOn`) and encapsulated history types (e.g. `DefinitionOfNetForce` and `DefinitionOfSelfInductance`).

After this successful iteration of DTA, the Companion is presented with a new problem from electrical systems, Gia-17-24-P. The Companion fails to arrive at the correct solution and invokes DTA. The first step of DTA is to retrieve an analogous worked solution from memory and extract a domain mapping. DTA uses the persistent mappings as required correspondence constraints to SME during the retrieval process. This guides the retrieval process to select the worked solution to problem Phy-1 and to construct a mapping building upon its knowledge about the domains. The resulting domain mapping includes the persistent mappings and the correspondences between the worked solutions to Phy-1 and Gia-17-24-P. This domain mapping constrains the analogy between the linear mechanics and the electrical systems domain theories. At this point, the electrical system domain theory consists of the equation schema for self inductance and control knowledge indicating it is a frame equation. DTA uses the candidate inferences from the analogy between the domain theories to extend the target domain theory with the equation schema for the definition of charge on a capacitor (i.e., the charge on a capacitor is equal to the voltage across the capacitor multiplied by its capacitance). This knowledge is verified by the Companion successfully solving Gia-17-24-P, and the persistent mappings are extended with the correspondences of the analogy between the domain theories.

As the example above illustrates, persistent mappings allow complex cross-domain analogies to be incrementally constructed as additional knowledge about the target become as available.

7.4.1.4 Non-analogous Base and Target Knowledge

The dynamical analogy domains include a number of non-analogous equations. These are items for which there is no corresponding item in the other domain. There is no such thing as thermal inductance, nor are there analogous items for the equations to compute moment of inertia. Including these elements in the analogy provides a number of challenges for a cross-domain analogy system. A non-analogous item in the base will be hypothesized in the target. A non-analogous target item could potentially match with a base item. Given the one-to-one constraint of analogical matching, this would prevent the analogous target item from being suggested in analogical inferences.

Three aspects of DTA prevent the non-analogous items in the base and target from adversely affecting the cross-domain analogy. First, encapsulated histories are transferred only if all the facts concerning participant types and quantities are mentioned in the domain mapping. Because non-analogous base encapsulated histories will include quantities and participant types, they will only be transferred if there is a target worked solution which causes all of the quantities to appear in the domain mapping. For example, a cross-domain analogy from electrical to thermal mechanics will not transfer an encapsulated history for inductance because no worked solutions from the target domain will include this non-existent quantity. Second, the verification step provides additional protection against non-analogous base knowledge being transferred into the target. If a non-analogous base encapsulated history is transferred to the target domain theory, it may affect the problem-solving during the verification step. If it does, then the newly inferred target items and the domain mapping which produced them are forgotten. Third, because of the assumption that knowledge about analogous encapsulated histories in the target comes only from the analogy, target encapsulated histories which are not included in the domain mapping are excluded from the cross-domain analogy.

7.4.1.5 Interactive Cross-Domain Analogies

The scope of the dynamical analogies domains requires that the target domain be constructed over a number of iterations of cross-domain analogy. The analogy is introduced in pieces and separated by non-analogous aspects of the domain. For example, Giancoli (1991) introduces rotational motion over an entire chapter. The chapter begins with discussing kinematics equations and their analogues with linear motion. Next, moment of inertia is introduced, which leads to a discussion of the non-analogous elements concerning the computation of moment of inertia and radius of gyration for various idealized objects. Giancoli invokes the analogy again to introduce rotational kinetic energy and angular momentum. Text books and teachers present cross-domain analogies iteratively and provide students with correspondences between the domains.

Therefore, it is useful to view cross-domain analogical learning as an interactive process. Adapting DTA to take advice was done in two ways. One, the Companion can be instructed to invoke DTA with a given set of correspondences serving as additional persistent mappings, constraining the retrieval and matching process between the worked solutions. Two, the Companion can be instructed to invoke DTA with a given base worked solution. Retrieval is a similarity based process and in cross-domain analogies there is very little surface similarity. Therefore, one would not expect MAC/FAC to be particularly effective as the number of non-analogous base worked solutions grows. These pieces of advice can be invoked together by specifying a base worked solution and a set of correspondences.

7.4.2 Experiment

This experiment evaluates DTA's performance on the dynamical analogy domains addressing the following questions:

- Can DTA transfer the analogous knowledge necessary to solve problems in dynamical analogy domains?
- When retrieval fails, does providing the Companion with the analogous base solution lead to successful transfer?
- What are the effects of persistent mappings in learning domain mappings and aiding retrieval?

7.4.2.1 Materials

The problems were selected from the physics resources with the following goals. First, each of the dynamical analogy equations from Table 9 should be included in the problem set. Second, the problems were limited in complexity to requiring at most 3 different physics equations. This is consistent with using more basic problems when introducing subjects. Third, problems were favored which provided a worked solution. The problems which did not have a worked solution had one created in the same manner as the proceeding experiment. Fourth, problems were selected based upon an ability to solve them using our existing algebra system. Problems which involved calculus were simplified to satisfy this requirement. Table 10 contains natural language representations of the problems aligned by the underlying analogy.

Table 10: Dynamical analogy problems aligned by central equations[8]

Linear	Rotational	Electrical	Thermal
Gia-4-1: Estimate the net force needed to accelerate a 1500kg race car at -5 m/s^2?	Gia-8-9: A 15N force is applied to a cord wrapped around a 4kb wheel with a radius of 33cm. The wheel is observed to accelerate uniformly from rest to reach an angular speed of 30 rad/s in 3s, if there is a frictional torque of 1.1 Nm, determine the moment of inertia of the wheel.	Gia-21-37-P: A 75-V emf is induced in a .3 H coil by a current that rises uniformly from 0 to I in 2 ms. What is the value of I?	N/A
Gia-7-2: A 10,000kg railroad car traveling at a speed of 24 m/s strikes an identical car at rest. If the cars lock together as a result of the collision, what is their common speed afterwards?	Gia-8-12: A mass attached to the end of a string revolves in a circle on a frictionless tabletop. The other end of the string passes through a hole in the table. Initially, the ball rotates with a speed of 2.4m/s in a circle of radius .8m. The string is then pulled through the hole so that the radius is reduced to .48m. What is the speed of the mass now?	N/A	N/A
Gia-6-3: A 145g baseball is thrown with a speed of 25 m/s. What is the work done if it is starting from rest?	Gia-8-40-P: A centrifuge rotor has a moment of inertia of 4e10^{-2} kgm^2. How much energy is required to bring it from rest to 10,000 rpm?	Gia-21-46-P: How much energy is stored in 40mH inductor at an instant when the current is 12 A?	N/A
Phy-1: When a 13.2-kg mass is placed on top of a vertical spring, the spring compresses 5.93 cm. Find the force constant of the spring.	She-2-16: A simple torsion bar is formed by a cylindrical steel rod has a diameter of .25 inch and a length of 3 inches and is fixed to a rigid support at one end. The end rotations 1 radian when a 20 N force is applied, Compute the inertia of the rod?	Gia-17-24-P: How much charge flows from a 12-V battery when it is connected to a 7.5 micro-F capacitor?	Gia-14-1: How much heat is required to raise the temperature of an empty 20-kg vat made of iron from 10C to 90C?
Gia-6-28-P: A spring has a spring constant of 380 N/m. How much must this spring be compressed to store 60J?	She-2-18: An ideal torsion spring is attached to the shaft of a gear. A torque, T, is applied to the gear twisting the spring. Assuming that K=11 Nm/rad. Compute the energy stored in the spring when it is displaced by 2 rads.	Gia-17-7: A 12-V battery is connected to a 20 micro-F capacitor with uses a paper dielectric, how much electric energy can be stored in the capacitor?	N/A
Oga-3-8: Given a speed of 20 m/s what is the force applied by a series of dampers with damping constants of 7 Ns/m and 5 Ns/m?	Oga-6-12: A Cylinder is rotating with a speed of 15 rad/s what is the force applied from a dashpot (b = 2 Ns/rad)?	Gia-18-2: A plate on the bottom of a small tape recorder specifies that it should be connected to that it should be connected to a 6V and will draw 300 mA, what is the resistance of the recorder?	Gia-14-8: A major source of heat loss from a house is through the windows. Calculate the rate of heat flow through a glass window 2m x 1.5m and 3.2mm thick if the temperatures at the inner and out surfaces are 15C and 14C.
Gia-6-47-P: If a car generates 15hp when traveling at a steady 100 km/h, what must be the average force exerted on the car due to friction and air resistance?	Rea-286: The drive shaft of an automobile rotates at 377rad/s and transmits 59.6W from the engine to the rear wheels. Compute the torque developed by the engine.	Gia-18-6: An electric heater draws 15 A on a 120 V line. How much power does it use and how much does it costs per month 30 days if it is operated 3 hrs/day and the electric company charges $.08 per Kwh?	N/A

[8] Problem sources are indicated by their prefixes. The first number indicates the chapter and the second problem indicates the example within that chapter. The 'P' designation is used for problems from the end of the chapter. Gia are from (Giancoli 1991), Oga are from (Ogata 1997), She are from (Shearer *et al.* 1971), Rea are from (Fogiel 1994) and Phy are from ("Hooke's Law, Work and Elastic Potential Energy" 2009)

Predicate calculus representations for problems and worked solutions were created for the 22 problems: 7 from linear mechanics, 7 from rotational mechanics, 6 from electrical systems, 2 from thermal systems.

As in the previous experiment, the rules for the problem-solving system are hand coded.

7.4.2.2 Method

The base domain for this experiment is linear mechanics, because it is the dynamical analogy domain most students learn first. The linear mechanics 7 problems and worked solutions were added to the case library. The base domain theory consists of the encapsulated histories and control knowledge necessary to solve these 7 problems. The target domains for this experiment are rotational, electrical, and thermal systems. The initial target domain theory includes only the non-analogous encapsulated histories necessary to solve the problems (e.g. equations for computing moment of inertia).

Instead of learning curves, the design of this experiment seeks to evaluate DTA's performance on a per problem basis. Therefore each problem was tested independently in four conditions. First, the entire DTA algorithm was run on each problem. To isolate the retrieval mechanism, the Companion was only allowed one attempt to retrieve a worked solution from the base domain. In the second condition, each problem was presented with the correct retrieval from linear mechanics. The third and fourth conditions address the effects of persistent mappings. the third condition provides each problem with the persistent mappings resulting from a successful run of DTA on the most similar problem. For example, in condition three for Gia-21-37-P, the persistent mappings were provided from a successful transfer between Gia-6-3 and Gia-21-46-P. In the fourth condition, the persistent mappings and the correct retrieval were provided.

For each problem, DTA was scored correct if the Companion was able to solve the problem after the transfer and retrievals were scored using Table 10. Therefore each problem has only one correct retrieval out of 7 worked solutions in the case library.

The goal of this experiment is to assess DTA's ability to learn rotational mechanical, electrical, and thermal systems through cross-domain analogy with linear mechanics. Using the four conditions, the retrieval, mapping and persistent mapping components are evaluated independently.

7.4.2.3 Results

Table 11 shows the results of the four trials for each of the three transfer domains.

Condition	Rotational Mechanical (7)		Electrical Systems (6)		Thermal Systems (2)	
	Correct(%)	Retrieval(%)	Correct(%)	Retrieval(%)	Correct(%)	Retrieval(%)
1: Complete DTA	2(29%)	3(43%)	1(17%)	2(33%)	0(0%)	0(0%)
2: Given base worked solution	6(86%)	n/a	4(67%)	n/a	2(100%)	n/a
3: Given persistent mapping	3(43%)	3(43%)	1(17%)	3(50%)	0(0%)	0(0%)
4: Given persistent mapping and base worked solution	7(100%)	n/a	4(67%)	n/a	2(100%)	n/a

Table 11: DTA results by successful transfers and number of retrievals using linear mechanics as the base domain

In condition 1, DTA transferred useful equation schemas and control knowledge to 3 out of 15 problems from the target domains (20%). While the Companion solved problems from rotational mechanical and electrical domains, it failed to solve either of the thermal system problems. These results are not as bad as they seem because, of the worked solutions retrieved, only 5 out of the 15 (33%) were analogous to the target worked solution. Compared against random retrieval, this result is not statistically significant (p<.19). In condition 2, where DTA was provided the correct retrieval from memory, the Companion successfully solved problems from all three of the domains. DTA transferred useful equation schemas and control knowledge for 12 out of the 15 problems (80%) of the problems. In the third condition, the Companion was provided with the persistent mappings resulting from a successful transfer of the closest target problem. After retrieving 6 out of 15 (40%) correctly, DTA successfully transferred knowledge to solve 4 (27%) of the problems. Unlike condition 1, the retrieval result for condition 3 is statistically significant from random retrieval (p<.04). In the final condition, DTA was provided with both the persistent mappings and correct retrieval and the Companion correctly solved 13 of the 15 (87%) problems.

7.4.3 Discussion

First, these results indicate DTA is able to transfer knowledge from linear mechanical systems in order to solve problems from related domains. In addition to successful transfer to rotational mechanics, DTA was able to transfer linear mechanical knowledge to the superficially dissimilar domains of electrical and thermal systems under certain conditions. These cross-domain analogies include non-analogous base and target elements. In addition, these cross-domain analogies required the transfer of not only equation schemas but also control knowledge regarding how to employ the equations to solve problems in the target domain. While the analogous portion of these domains contains 7 equations relating 11 quantities, there are 33 total equations and 56 total quantities used in the 4 domain theories and 22 worked solutions. The next sections discuss the retrieval, mapping, verification and persistent mapping aspects of the algorithm in detail.

7.4.3.1 Retreival

The primary cause for DTA's failures was the inability to retrieve an analogous worked solution. This is consistent with psychological findings regarding the difficulties in the spontaneous retrieval of cross-domain analogies (Gick and Holyoak 1983). Given a known base domain, DTA retrieved the analogous worked solution only 33% of the time. By providing persistent mappings, this performance improved to 40% which is statistically significant compared to chance.

An analysis of the retrieval failures indicates that every failure occurred during the 1st stage of

MAC/FAC. Recall that the MAC stage consists of a fast non-structural comparison between feature vectors for each of the cases in the case library and the probe case. The feature vectors are automatically computed and each component has a strength proportional to the number of occurrences of individual predicates. Looking at the feature vectors more closely, the majority of the highest weighted entries are relations concerning the definition of worked solution steps. Therefore, the MAC stage is largely determined by the size of the worked solution probe. Of the rest of the entries, few are shared across the domains. Rotational mechanics shares some motion predicates with the base domain, contributing to improved retrieval. Also, persistent mappings improved retrieval slightly by altering the probe's feature vector to include additional predicates from the base domain. Every time the analogous worked solution was selected in the MAC stage, it was returned as the retrieval.

Combining the results from condition 1 and 3, retrieval performed statistically better than chance (37% $p<.05$). For this experiment, the Companion was only allowed one retrieval attempt. One method to improve performance would be by allowing multiple retrievals. The rigorous verification of transferred knowledge prevents non-analogous retrievals from tainting the target domain theory. As discussed earlier, the number of retrievals should be under the control of the agent. If the agent using DTA does not have other tasks requiring attention, then it would be free to make multiple retrievals. Otherwise, the agent should entertain another course of action, such as interactive cross-domain analogy. In this case, the agent asks the user for an analogous worked solution from the base domain.

7.4.3.2 Mapping and Transfer

Given the cross-domain nature of these analogies, the mapping of worked solutions to learn the domain theory followed by the analogy between the domain theories was very successful. Of the 41 problems in which DTA used the correct retrieval, transfer was successful on 32 (78%) of them. Through analyzing the failures, we can better understand the limits of DTA. The 9 transfer failures can be divided into three types of failures: merge failures, one-to-one constraint failures, and incomplete mapping failures.

The first cause of mapping failures occurs when the mapping fails to include a necessary correspondence because another correspondence was already in the mapping. Recall that during SME, local match hypothesis are merged into structurally consistent global mappings. For example, the worked solution of Gia-6-3 includes two relations connecting the ball to the throwing event: `objectThrown` and `objectTranslating`. The analogous rotational mechanics problem, Gia-8-40-P, has one relationship linking the rotor to its rotation event: `objectRotating`. Due to the one-to-one constraint, `objectRotating` can only map to `objectThrown` or `objectTranslating`. If it maps to `objectThrown`, the subsequent cross-domain analogy will fail because `objectTranslating` is condition in the analogous equation schema.

The second cause of mapping failures is one-to-one constraint violations in the analogous equations. For example, the worked solution to problem Gia-6-3 uses the equation schema for linear kinetic energy (Ke = .5mv^2) which includes 3 participants: the object, the movement event, and the time point of the measurement. The three participants are `Ball-6-3`, `Throwing-6-3` and `(EndFn Throwing-6-3)` respectively. The analogous electrical problem, Gia-21-46-P, uses the equation schema for inductance energy ($E = .5\ Li^2$) which also includes three participants: the inductor, the induction event, and the time point of the measurement. In this case, the induction event is the same entity as the time point. Therefore, the one-to-one constraint prevents the participant types for the events and the time points for appearing in the same mapping. Recall, if all of an equation schema's participant types are not in the domain mapping, then the equation schema cannot be transferred via DTA.

The third cause of mapping failures is an incomplete mapping. This occurs when the mapping with the analogous worked solution transfers some of the equation schemas required to solve the target problem. This error occurred during the analogy between electrical problem Gia-17-7 and the linear mechanics problem Gia-6-28-P. Gia-17-7 requires equation schemas for both the definition of electrical capacitance, $V=q/C$, and the definition of capacitor energy, $Ce=.5(q^2)/C$. The worked solution analogy between Gia-6-28-P and Gia-17-7 successfully transfers the equation schema for electrical capacitance, but not capacitor energy. Therefore, the verification step fails, and the transferred knowledge and domain mapping is forgotten.

The three types of mapping failures demonstrate the limits of DTA's ability to learn domain mappings from the analogy between two worked solutions. As the next section describes, persistent mappings allow DTA to overcome merge failures and incomplete mapping failures.

7.4.3.3 Persistent mappings

The results from conditions 3 and 4 of the dynamical analogies experiment demonstrate that persistent mappings support incremental learning of the target domain theory through multiple cross-domain analogies. In condition three, persistent mappings enabled an additional correct retrieval. Persistent mappings prevent worked solution mapping failures. In the merge failure between worked solutions Gia-6-3 and Gia-8-40-P, the persistent mappings included a required correspondence between `objectTranslating` and `objectRotating`. This shows how successful matches can assist future analogies between worked solutions. Also, persistent mappings prevent incomplete mappings by already including aspects of the cross-domain analogy. In the incomplete failure between worked solution Gia-6-28-P and Gia-17-7, the persistent mapping and target domain theory already includes the definition of capacitance. This was the result of the analogy between Phy-1 and Gia-17-24-P. Therefore, when analogy between Gia-17-7 and Gia-6-28-P transfers the definition of capacitance energy, the Companion is able to use both equations to successfully solve Gia-17-7. DTA is able to incrementally construct complex cross-domain analogies by storing persistent mappings from successful iterations.

An underlying assumption of persistent mappings is that the entire cross-domain analogy satisfies the one-to-one constraint. That is, each element of the base domain theory corresponds to at most one element in the target domain theory. This assumption was not valid in this experiment. This caused DTA to fail on the electrical power problem, Gia-18-6, which it succeeded on in previous conditions. The electrical power equation, P=Vi, involves three participants, a power supply, an electrical component, and an electrical conduction event. In the first two conditions, the analogy between the worked solutions for Gia-6-47-P and Gia-18-6 resulted in a domain mapping including a correspondence between point mass and electrical component. Using the resulting domain mapping, the power equation is transferred and the Companion solves the problem. In conditions 3 and 4, the persistent mapping resulting from a successful transfer for problem Gia-21-37-P includes a correspondence between point mass and inductor. This is a required correspondence during the worked solution analogy between Gia-6-47-P and Gia-18-6 preventing the necessary domain mapping between point mass and electrical component. Therefore an important direction for future work involves relaxing persistent mappings. Successful transfers would support persistent mappings and transfer failures involving persistent mappings would be evidence against them.

The dynamical analogy evaluation and the kinematic evaluation demonstrate that DTA enables the reuse of knowledge from an understood domain in new domains. Using analogies between worked solutions, DTA learns a domain mapping which is used to constrain a cross-domain analogy. This cross-domain analogy transfers equation schemas and control knowledge. Verified transfers result in persistent mappings which guide future analogies between the domains. The next chapter presents related work on cross-domain analogy and directions for future research.

8 Related and Future Work on Cross-domain Analogy

This chapter compares DTA with other methods not mentioned in previous chapters and discusses future directions.

8.1 Related Work

The major threads of related work concern cognitive science models of analogy, AI transfer learning, and other forms of example-based reasoning.

8.1.1 Cross-domain Analogy simulations from Cognitive Science

The differences between DTA and PHINEAS (Falkenhainer 1987) are discussed in Section 7.1.2. Another analogical problem-solving and learning system is Holyoak and Thagard's PI (1989). PI used a pragmatic theory of analogy to model solving a variation of the radiation problem through schema induction. After a successful analogical problem-solving episode, PI induces a schema which is treated as a general rule which applies to the two analogous situations. PI only used analogy during problem-solving, and its retrieval model was never extensively tested. On the other hand, DTA makes analogies between both example problems as well as analogies between domains themselves. The domain theory analogy enables DTA to transfer domain knowledge not explicitly referenced in the particular worked solutions used in creating the domain mapping (e.g., the rest of the domain theory in the kinematics experiment and the control knowledge in the dynamical analogy experiment). Moreover, DTA tests its learned knowledge, and uses it to solve new problems from the target domain, whereas PI did neither.

Kühnberger *et al.*'s I-Cog (2007) explores the trade-offs between analogy and two other reasoning modules, all of which operate over noisy data, using the Heuristic-Driven Theory Projection (Gust *et al.* 2006) model of analogy. Schwering *et al*. (2008) identifies the importance of combining analogy with deductive and inductive techniques for achieving human level reasoning. These approaches emphasize ubiquity of analogy in reasoning and learning as well as the integration of analogy with other reasoning processes. A major point of departure between the above simulations and DTA is the scale of the tasks. While the above systems have explored similar domains to those explored here, they have not been systematically evaluated on sets of problems and domains. Scaling up is a major challenge for analogical learning systems (Forbus 2001). By learning physics domain theories, the results of the cross-domain analogy are used in a large-scale reasoning task, solving physics problems.

8.1.2 Transfer Learning

Hinrichs & Forbus (2007) describe how analogy can be used to transfer learned qualitative models between scenarios in a turn based strategy game. As in DTA, examples are used to find the domain mapping between source and target domains. From a cognitive architecture perspective, ICARUS achieves near transfer through goal decomposition (Choi *et al.* 2007) and has been augmented with a representation mapping algorithm (Shapiro *et al.* 2008) to handle more distant types of transfer. The methods employed by ICARUS require abstracted domain theories in both the source and target tasks. As demonstrated in the kinematics experiment, DTA can transfer abstract knowledge from the source domain to the target domain using a domain mapping learned from specific examples.

Within the reinforcement learning (Sutton and Barto 1998) community, transfer learning efforts have focused on using mappings between scenarios and domains. In these RL efforts, the transfer consists of learned state action policies. Lui and Stone (2006) use a version of SME to accelerate learning of state action policies in novel but similar tasks within the keep-away soccer domain. Taylor (2008) emphasizes the importance of the mapping between the states and actions of the source and target domains. In these systems the mappings are one shot processes to jump start learning in the target domain. After the initial transfer, the mapping with the base domain is ignored. By using persistent mappings, DTA is an iterative process in which the transferred target knowledge and the domain mapping are incrementally verified and extended. Molineaux *et al.* (2008) use a Case-based Reasoning technique to use examples to find mappings for use by a reinforcement learning system playing a real-time strategy game. Unlike the above reinforcement learning methods which use feature vectors, DTA operates over predicate calculus descriptions which facilitates the representation of episodes, explanations and plans. These structures provide additional context for the transfer and enable the transferred knowledge to be more broadly applicable. A transfer failure can be quickly identified by failing to solve the target problem. The kinematics and dynamical analogy experiments demonstrate how DTA can transfer schemas and control knowledge using one example from the target domain. For a more direct comparison, it is necessary to integrate DTA with established domain learning techniques. This is an important direction for future research.

8.2 Conclusions and Future Directions

Using domain general methods of similarity-based retrieval and analogical matching, Domain

Transfer via Analogy (DTA) enables the transfer of equation schemas and control knowledge from an understood domain to a new domain. Persistent mappings support this process by building up a complex cross-domain analogy from successful local mappings.

These claims are supported by experimental results from cross-domain analogies between linear and rotation kinematics as well as the dynamical analogy domains of linear, rotational, electrical, and thermal mechanical systems. Given a known base domain and a worked solution from the target domain, DTA transferred schemas representing domain specific equations and control knowledge concerning preferences between equations useful for problem-solving. This knowledge was then applied successfully to solve problems from the new domain.

These experiments open a number of interesting directions for future research. Here, I focus on four directions: integration with domain learning techniques, application to new domains, extending DTA through diagnosis of errors in transferred knowledge, and interactive cross-domain analogy.

8.2.1 Integration with Domain Learning Techniques

As discussed in the related work section, DTA has not been used with learned domain theories. To perform transfer learning, DTA should be integrated with existing domain learning techniques. For Physics, a promising direction involves learning generalizations for participant abstraction modeling decisions within each domain (Klenk *et al.* 2008), then applying DTA to transfer these generalizations to new domains. Because DTA transfers schemas and predicate calculus representations, it should be possible to transfer knowledge produced by a variety of learning algorithms. In physics, it would be interesting to use reinforcement learning to learn a policy for the selection of operators during problem-solving within a domain. DTA should be able to transfer this state action policy to new domains in addition to the existing encapsulated histories and control knowledge.

8.2.2 Application to New Domains

Another direction for future work is to apply DTA to new domains. For example, simulation game domains have a number of different properties than physics problems solving (e.g., incomplete knowledge, the integration of planning and action). While cross-domain analogies between strategy games are not as well understood as the physics analogies, it is a reasonable assumption that an agent which performs well in one strategy game or situation could benefit from transferring some aspects of its knowledge to a new game or situation. Integrating with existing learning techniques and applying DTA in new domains opens the question of how an agent adapts the newly acquired domain knowledge.

8.2.3 Debugging the Learned Knowledge

An underlying assumption of this work is that by identifying similarities between pairs of domains, the domain mapping, DTA can transfer abstract domain knowledge between them. The existence of these underlying similarities does not necessarily imply that the situations are completely analogous. The cross-domain analogy may provide a useful starting point for exploration of the new domain, but it may also introduce a number of misconceptions which could prevent future learning in a domain. This is consistent with psychological findings in student learning (Burnstein 1986). Thus far, persistent mappings have been considered hard constraints for future analogies between the domains. As illustrated in the dynamical analogy evaluation, this can have a negative effect of performance. Applying DTA in new domains and over a greater period of learning will likely require relaxing this constraint. This provides an opportunity to exploit either models of conceptual change (Friedman & Forbus 2008) or model-based diagnosis techniques (de Koning 1997) to identify and repair errors in the persistent mappings or the target domain theory.

8.2.4 Interactive Cross-domain Analogy

The fact that cross-domain analogies are used so frequently in textbooks and explanatory discourse implies that these are important communicative devices. Most cross-domain analogy research views analogy as an isolated process within a cognitive agent. The analogy occurs, and then the agent, or the research, evaluates the result. DTA represents an important step toward interactive analogy. Persistent mappings enable the reuse of the analogy and its expansion as additional information becomes available. DTA currently accepts two pieces of advice: user supplied mappings and worked solutions. User supplied mappings are treated in the same manner as persistent mappings, informing both the retrieval and mapping process. User supplied worked solutions are a particularly effective form of feedback, as shown by conditions 2 and 4 in the dynamical analogies experiment. Within this framework, a simple extension involves enabling the user to provide negative feedback regarding individual persistent mappings.

Another potential advance in interactive cross-domain analogy would involve the Companion suggesting insights to the user. This would take the form of suggested correspondences, from which the user could draw new insights, and suggested predictions about aspects of the target domain. For example, in critiquing a course of action in a strategy game, the Companion could express concerns about a particular plan. The ensuing discussion could upon similarities with other games identifying correspondences between units (e.g., riflemen ↔ elves) or strategies (e.g., blitzkrieg ↔ scorched earth). Also of importance would be the predictions resulting from the cross-domain analogies. If I consider my tanks in game A as analogous to the cavalry from game B, then I should watch out for over extending them, a frequent occurrence in my experiences in game B. In interactive cross-domain analogy, the system informs the user, and, as previously described, the user informs the system.

Each of these directions contain a number of exciting research questions, addressing them all would require many more theses.

9 Closing Thoughts

Let us return to the original problem of alleviating the brittleness of AI systems. A primary cause of brittleness in knowledge based systems involves their lack of ability to reuse their existing knowledge in new situations or domains. The approach taken in this dissertation focuses on two methods, analogical model formulation and domain transfer via analogy, which employ domain general analogical processes into robustly reuse knowledge.

9.1 Analogical Model Formulation

Analogical model formulation enables the understanding of new situations based on previous examples. Through analogy with an understood example, analogical model formulation infers the assumptions, approximations, causal models, and equations necessary for reasoning about the new situation.

Analogical model formulation addresses the following shortcomings of current model formulation methods. First, by relying on examples, analogical model formulation does not require a complete and correct domain theory. Second, analogical model formulation can build scenario models of everyday situations. The breadth of entity types is limited only by the underlying ontology. Third, systems using analogical model formulation learn incrementally by accumulating new examples.

This method has been evaluated on the Bennett Mechanical Comprehension Test and problems of the style found on the AP Physics exam. To construct models for BMCT problems, analogical model formulation transferred the following kinds of modeling knowledge from examples: qualitative mechanics abstractions, causal models, and instructions for visual quantity measurements. In AP Physics, an external evaluation created transfer levels representing systematic differences between examples and problems. Using examples, analogical model formulation successfully applied equations, modeling assumptions and default values to create the models necessary to solve the new problems.

9.2 Domain Transfer via Analogy

Domain transfer via analogy (DTA) allows the reuse of abstract domain theories through cross-domain analogy. Using domain general retrieval and matching algorithms, DTA constructs a domain mapping from pairs of explanations. This domain mapping is used to initialize and extend the target domain theory. The learned aspects of the target domain theory are verified by using them to solve new problems. When successful, DTA stores persistent mappings to enable the identified similarities between the domains to assist in future cross domain analogies.

The DTA method makes the following contributions as a cognitive simulation of cross-domain analogy. First, DTA transfers the equation schemas and control knowledge necessary to solve quantitative physics problems. Second, DTA uses persistent mappings to incrementally construct complex cross-domain analogies as it encounters new examples from the target domain.

This model has been evaluated across a variety of physics domains. In kinematics, the system using DTA out performed a spoon-fed baseline. In dynamical analogies, DTA transferred equation schemas and control knowledge between the superficially dissimilar domains of mechanical, electrical and thermal systems. Consistent with psychological findings, retrieving analogous examples between domains was difficult. Persistent mappings enable the cross-domain analogy to incrementally extend, based upon successful transfers, to address the depth of the evaluation domains which required multiple cross-domain analogies.

9.3 Final Thoughts

To overcome brittleness, AI systems must be able to identify and reuse related knowledge. Analogical processing seems central to this goal. By using domain general cognitive simulations of analogical matching and retrieval, the methods described in this work are two new ways of reusing knowledge from examples and domains.

Once free from brittleness, AI systems will be able to exist over a long period of time, adapting to new tasks through analogies with their existing knowledge.

10 References

Adams, J. (2003). *Mathematics in Nature: Modeling Patterns in the Natural World*. Princeton, NJ: Princeton University Press

Barker, K., Chaw, S., Fan, J., Porter, B., Tecuci, D., Yeh, P., Chaudhri, V., Israel, D., Misha, S., Romero, P., and Clark, P. (2004). A question-answering system for AP Chemistry: Assessing KR&R Technologies. *Proceedings of the Ninth International Conference on the Principles of Knowledge Representation and Reasoning (KR 2004)*. Whistler, 488-497.

Bennett, G.K. (1969). *Bennett Mechanical Comprehension Test*. The Psychological Corporation: San Antonio.

Batali, J. (1991). *Automatic Acquisition and Use of Some of the Knowledge in Physics Texts*. Massechutes Institute of Technology: Department of Computer Science (PhD thesis)

Bassok, M., and Holyoak, K. J. (1989). Interdomain transfer between isomorphic topics in algebra and physics. *Journal of Experimental Psychology: Learning, Memory, & Cognition*. 15.

Bringsjord, S., and Schimanski, B. (2003). What is Artificial Intelligence? Psychometric AI as an Answer. In *Proceedings of the International Joint Coneference on Artificial Intelligence (IJCAI-03)*.

Burnstien, M. (1986). Concept formation by incremental analogical reasoning and debugging. In R. S. Michalski, J. B. Carbonell and T. M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach* (Vol. 2). Los Altos, CA: Kaufmann.

Bundy, A. (1979). Solving mechanics problems using meta-level inference. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-79).*

Chi, M., M. Bassok, M. Lewis, P. Reimann, and Glaser, R. . "Self-explanations: How students study and use examples in learning to solve problems." *Cognitive Science*, 1989.

Choi, D., Konik, T., Nejati, N., Park, C., and Langley, P. (2007). Structural transfer of cognitive skills. In *Proceedings of the Eighth International Conference on Cognitive Modeling*. Ann Arbor, MI.

Cohen, P., Schrag, R., Jones, E., Pease, A., Lin, A., Starr, B., Gunning, D. and Burke, M. (1999). The DARPA High-Performance Knowledge Bases Project. *AI Magazine*. 19(4): 25-49.

Cohn, A. (1996). Calculi for Qualitative Spatial Reasoning. *Artificial Intelligence and Symbolic Mathematical Computation*. Springer.

de Kleer, J. (1977). Multiple representations of knowledge in a mechanics problem solver. In

Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-77).

Dunbar, K. (1995). How scientists really reason: Scientific reasoning in real-world laboratories. In R. J. Sternberg & J. Davidson (Eds.), *Mechanisms of insight*. Cambridge, MA: MIT Press.

Falkenhainer, B. (1988). Learning from Physical Analogies. Technical Report No. UIUCDCS-R-88-1479, University of Illinios at Urbana-Champaign. (Ph.D. Thesis)

Falkenhainer, B. (1992). Modeling without amnesia: Making experience-sanctioned approximations. In *Proceedings of Qualitative Reasoning*.

Falkenhainer, B. and Forbus, K. (1991). Compositional modeling: Finding the right model for the job. *Artificial Intelligence*. 51.

Falkenhainer, B., Forbus, K. and Gentner, D. (1989). The Structure Mapping Engine: Algorithm and examples. *Artificial Intelligence*. 41.

Faries, J. M., and Reiser, B. J. (1988). Access and use of previous solutions in a problem solving situation. In V. L. Patel & G. L. Groen (Ed.), *Proceedings of the Tenth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Flores, J., and Cerda, J. (2000) Efficient modeling of linear circuits to perform qualitative reasoning tasks. *AI Communications*.

Fogiel, M., ed. (1994). *The Physics Problem Solver*. Research and Education Association.

Forbus, K. (1984). Qualitative process theory. *Artificial Intelligence*, 24.

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

Forbus, K. and de Kleer, J. (1993). *Building Problem Solvers*, MIT Press.

Forbus, K., Ferguson, R. and Gentner, D. (1994). Incremental Structure-Mapping. In *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*.

Forbus, K., and Gentner, D. (1986). Learning physical domains: Toward a theoretical framework. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach*. Los Altos, CA: Kaufmann.

Forbus, K., and Gentner, D. (1997). Qualitative mental models: Simulations or memories?" *Proceedings of the 11th International Workshop on Qualitative Reasoning.* Cortona, Italy.

Forbus, K., Gentner, D. and Law, K. (1995). MAC/FAC: A model of similarity-based retrieval. *Cognitive Science*, 19(2), 141-205.

Forbus, K. and Hinrichs, T. (2004). Companion Cognitive Systems: A step towards human-level AI. In *AAAI Fall Symposium on Achieving Human-level Intelligence through Integrated Systems and Research*, October, Washington, DC.

Forbus, K., Klenk, M. and Hinrichs, T. (2008). Companion cognitive systems: Design goals and some lessons learned. In the *AAAI Fall Symposium on Naturally Inspired Artificial Intelligence*. Washington D.C.

Forbus, K., Lockwood, K., Klenk, M., Tomai, E., and Usher, J. (2004). Open-domain sketch understanding: The nuSketch approach. In *AAAI Fall Symposium on Making Pen-based Interaction Intelligent and Natural*, Washington, DC.

Forbus, K. and Oblinger, D. (1990). Making SME greedy and pragmatic. In *Proceedings of the Cognitive Science Society*.

Forbus, K., Tomai, E., and Usher, J. (2003). Qualitative spatial reasoning for visual grouping in sketches. In *Proceedings of the 17th International Workshop on Qualitative Reasoning*. Brasilia, Brazil.

Forbus, K. and Usher, J. (2002). Sketching for knowledge capture: A progress report. In *Proceedings of Intelligent User Interfaces*, San Francisco, California.

Forbus, K., Usher, J. and Tomai, E. (2005). Analogical learning of visual/conceptual relationships in sketches. In *Proceedings of Association for the Advancement of Artificial Intelligence (AAAI-05)*. Pittsburgh, PA.

Friedman, S. and Forbus, K. (2008). Learning Causal Models via Progressive Alignment & Qualitative Modeling: A Simulation. In *Proceedings of the 30th Annual Conference of the Cognitive Science Society (CogSc-08i)*. Washington, D.C.

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

Gentner, D. (2003). Why we're so smart. In *Language in mind: Advances in the study of language and thought*, by D. Gentner and D. Goldin-Meadow. Cambridge, MA: MIT Press.

Gentner, D., Brem, S., Ferguson, R. W., Wolff, P., Markman, A. B., & Forbus, K. D. (1997). Analogy and creativity in the works of Johannes Kepler. In T. B. Ward, S. M. Smith, & J. Vaid (Eds.), *Creative thought: An investigation of conceptual structures and processes*. Washington, DC: American Psychological Association.

Gentner, D., and Gentner, D. R. (1983). Flowing waters or teeming crowds: Mental models of electricity. In *Mental Models*, by D. Gentner and A.L. Stevens. Eltham, London: Greenwich University Press.

Giancoli, D. (1991). *Physics: Principles with Applications*. 3rd Edition. Prentice Hall.

Gick, M. and Holyoak, K. (1983). Schema induction and analogical transfer, Cognitive Psychology. 15.

Gust, H., Kühnberger, K.-U., and Schmid, U. (2006). Metaphors and Heuristic-Driven Theory Projection (HDTP). *Theoretical Computer Science*, 354(1), 98-117.

Hinrichs, T. and Forbus, K. (2007). Analogical learning in a turn-based strategy game. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*. Hyderabad, India.

"Hooke's Law, Work and Elastic Potential Energy" *Physics Homework Help from Experts*. 21 March 2009. <http://www.physics247.com/physics-homework-help/elastic-potential.php >.

Holyoak, K. and Simon, D. (1999). Bidirectional reasoning in decision making by constraint satisfaction. *Journal of Experimental Psychology—General*. 128.

Holyoak, K. J., and Thagard, P. (1989). A computational model of analogical problem solving. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning*. New York: Cambridge University Press.

Hummel, J. E., and Holyoak, K. J. (2003). A symbolic-connectionist theory of relational inference and generalization. *Psychological Review*, 110, 220-264.

Jones, R., Laird, J., Nielsen, P., Coulter, K., Kenny, P., and Koss, F. (1999). Automated Intelligent Pilots for Combat Flight Simulation. *AI Magazine*. 20(1)

Kim, H. (1993). Qualitative reasoning about fluids and mechanics. Tech. Rep. No. 47. The Institute for the Learning Sciences, Northwestern University. (Ph.D. Thesis)

Klenk, M. and Forbus, K. (2007a). Measuring the level of transfer learning by an AP Physics problem-solver. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI-07).* Vancouver, BC.

Klenk, M., and Forbus, K. (2007b). Cognitive modeling of analogy events in physics problem solving from examples. In *The Proceedings of the Cognitive Science Society (CogSci-07).* Nashville, TN.

Klenk, M., Friedman, S., and Forbus, K. (2008). Learning Modeling Abstractions via Generalization. In *Proceedings of the 22nd International Workshop on Qualitative Reasoning*. Boulder, CO.

Koff, R. M. (1961). How does it work? New York: Doubleday

Kokinov, B. and Petrov, A. (2001). Integrating memory and reasoning in analogy-making: The AMBR model. In D.Gentner, K.Holyoak, & B.Kokinov (Eds.), *The Analogical Mind:*

Perspectives from Cognitive Science. Cambridge, MA: MIT Press.

Kolodner, J. (1993). *Case-Based Reasoning*. San Mateo: Morgan Kaufmann.

Koning, K. de (1997), Model-Based Reasoning about Learner behaviour. *Frontiers in Artificial Intelligence and Applications*, Amsterdam: IOS Press.

Kuehne, S., Forbus, K., Gentner, D. and Quinn, B. (2000). SEQL: Category learning as progressive abstraction using structure mapping. In *Proceedings of the Cognitive Science Society (CogSci-2000).*

Kühnberger, K.-U., Geibel, P., Gust, H., Krumnack, U., Ovchinnikova, E., Schwering, A., and Wandmacher, T. (2008). Learning from Inconsistencies in an Integrated Cognitive Architecture. In *The Proceedings of the 1st Conference on Artificial General Intelligence (AGI-08),* Memphis, TN.

Labrou, Y. and Finin, T. (1997). Semantics and conversations for an agent communication language. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, Nagoya, Japan.

Lenat, D. (1995). A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11)

Lenat, D., and Feigenbaum, E. (1991). On the thresholds of knowledge. *Artificial Intelliegence*.

Levy, A., Iwasaki, Y., and Fikes, R. (1997). Automated model selection for simulation based on relevance reasoning. *Artificial Intelligence*. 96.

Lui, X., and Stone, P. (2006). Value-Function-Based transfer for reinforcement learning using structure mapping. *Proceedings of the 21st Association for the Advancement of Artificial Intelligence (AAAI-06).* Boston, MA.

Markman, A.B., & Medin, D.L. (2002). Decision Making. In D.L. Medin & H. Pashler (Eds.) *Stevens Handbook of Experimental Psychology* (3rd Edition), Volume 2. New York: John Wiley and Sons.

Matuszek, C., Cabral, J., Witbrock, M., and DeOliveira, J. (2006). An introduction to the syntax and content of Cyc. In *Proceedings of the 2006 AAAI Spring Symposium on Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering*. Stanford, CA.

Melis, E., and Whittle, J. (1999). Analogy in inductive theorem proving. *Journal of Automated Reasoning*.

Molineaux, M., Aha, D., and Moore P. (2008). Learning continuous action models in a real-time strategy environment. In *FLAIRS Conference 2008*.

Nayak, P. (1994). Causal approximations. *Artificial Intelligence*.

Nersessian, N. J. (1992). How do scientists think? Capturing the dynamics of conceptual change in science. In Giere, R. N. (ed.) *Cognitive Models of Science*. Minneapolis, MN: University of Minnesota Press.

Newell, A., and Ernst, G. (1965). The search for generality. In *Proceedings of IFIP Congress* 65:195-208.

Nielsen, P.E. (1988). *A qualitative approach to rigid body mechanics*. Tech. Rep. No. UIUCDCS-R-88-1469; UILU-ENG-88-1775. Urbana, Illinois: University of Illinois at Urbana-Champaign, Department of Computer Science. (Ph.D. Thesis)

Novak, G. (1977). Representations of knowledge in a program for solving physics problems. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-1977).*

Nuxoll, A. (2007). *Enhancing intelligent agents with episodic memory*. The University of Michigan: Department of Computer Science and Engineering (Ph.D. Thesis)

Nuxoll, A. M. and Laird, J. E. (2007). Extending cognitive architecture with episodic memory. In *Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI-07)*. Vancouver, CA.

Ogata, K. (1997). *System Dynamics*. Prentice Hall.

Olson, H. (1943). *Dynamical Analogies.* D. Van Nostrand.

Ouyang, T. and Forbus, K. (2006). Strategy variations in analogical problem solving. In *Proceedings of Association for the Advancement of Artificial Intelligence (AAAI-06)*. Boston, MA.

Penn, D. C., Holyoak, K. J., and Povinelli, D. J. (2008). Darwin's mistake: Explaining the discontinuity between human and nonhuman minds. *Brain and Behavioral Sciences*, 31, 109-178.

Pisan, Y. (1998). *An integrated architecture for engineering problem solving*. Northwestern University: Department of Computer Science. (PhD Thesis)

Ram, A., and Leake, D. B. (1995). *Goal-Driven Learning*. The MIT Press.

Spiro, R., Feltovich, P., Coulson, R, and Anderson, D. (1989). Multiple analogies for complex concepts: Antidotes for analogy-induced misconception in advanced knowledge acquisition. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning*. New York: Cambridge University Press.

Rickel, J., and Porter, B. (1994). Automated modeling for answering prediction questions: Selecting the time scale and boundary. In *Proceedings of American Association for Artificial Intelligence (AAAI-94)*.

Ross, B. (1989). Remindings in learning and instruction. In S. Vosniadou & A. Ortony (Ed.), *Similarity and Analogical Reasoning*. Cambridge: Cambridge University Press.

Schwering, A.; Krumnack, U.; Kühnberger, K.-U., Gust, H. (2008). Analogy as integrating framework for human-level reasoning, In *The Proceedings of the 1st Conference on Artificial General Intelligence (AGI-08)*, Memphis, TN. IOS Press. 419-423.

Shapiro, D., Könik, T., and O'Rorke, P. (2008). Achieving far transfer in an integrated cognitive architecture. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*. Chicago, IL.

Sharma, M., Holmes, M., Santamaria, J., Irani, A., Isbell, C., and Ram, A. (2007). Transfer learning in real-time strategy games using hybrid CBR/RL. In *Proceedings of the International Joint Conference for Artificial Intelligence (IJCAI-07).*

Shearer, J., Murphy, A., and Richardson, H. (1971). *Introduction to System Dynamics*. Reading, MA: Addison-Wesley

Stahovich, T. F., Davis, R., and Shrobe, H. (2000). Qualitative rigid body mechanics. *Artificial Intelligence*. 119.

Sutton, R. and Barto, A. (1998). *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Taylor, M. (2008). *Autonomous Inter-Task Transfer in Reinforcement Learning Domains*. The University of Texas at Austin: Department of Computer Sciences (Ph.D. Thesis). Available as Technical Report UT-AI-TR-08-5.

Tulving, E. (1972). Episodic and semantic memory. In E. Tulving & W. Donaldson (Eds.), *Organization of memory*. New York: Academic Press.

VanLehn, K. (1998). Analogy events: How examples are used during problem solving. *Cognitive Science* 22(19): 347-388.

VanLehn, K. (1999). Rule learning events in the acquisition of a complex skill: An evaluation of Cascade. *Journal of the Learning Sciences*. 8(2).

VanLehn, K., Jones, R. M., and Chi, M. T. H. (1992). A model of the self- explanation effect. *Journal of the Learning Sciences,* 2(1).

VanLehn, K., and Jones, R. M. (1993). Better learners use analogical problem solving sparingly.

In *Proceedings of the Tenth International Conference on Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Veloso, M., and Carbonell, J. (1993). Derivational analogy in PRODIGY: Automating case acquisition, storage, and utilization. *Machine Learning*.

Weld, D. (1988). Comparative Analysis. *Artificial Intelligence,* 36:333-374.

Yan, J., Forbus, K. and Gentner, D. (2003). A theory of rerepresentation in analogical matching. In *Proceedings of the Cognitive Science Society (CogSci-03)*.

Yaner, P. and Goel, A. (2008). From design drawings to structural models by compositional analogy. *International Journal of AI in Engineering Design, Analysis and Manufacturing*, Special Issue on Multimodal Design, 22(2): 117-128.

11 Appendices

11.1 Online

An on-line appendix to this dissertation may be found at

http://www.cs.northwestern.edu/~mek802/dissertation.html

This website includes archives of the following predicate calculus representations:

- The ETS generated problem and worked solutions for AP Physics-style problems
- The corrected versions of the AP Physics-style problems
- The kinematics problems and worked solutions
- The dynamical analogy problem and worked solutions

11.2 Appendix A

Problem 2 Representation

```
(isa Hyp-MT-ETS-Query-2-0-1 Microtheory)
(genlMt Hyp-MT-ETS-Query-2-0-1 PhysicsTestTakingAssumptionsMt)
(isa Movement-2-0-1 ProjectileMotion)
(isa Upward-Movement-2-0-1 ProjectileMotion)
(isa Astronaut-2-0-1 Astronaut)
(isa Ground-2-0-1 SurfaceRegion-Tangible)
(isa Planet-2-0-1 Planet)
(isa Throwing-2-0-1 ThrowingAnObject)
(isa BaseballBat-2-0-1 BaseballBat)
(except    (ist    PhysicsTestTakingAssumptionsMt    (relationAllInstance    eventOccursAt    Event
PlanetEarth)))
(groundOf Planet-2-0-1 Ground-2-0-1)
(performedBy Throwing-2-0-1 Astronaut-2-0-1)
(no-GenQuantRelnFrom in-ImmersedFully Planet-2-0-1 Atmosphere)
(eventOccursNear Throwing-2-0-1 Ground-2-0-1)
(objectThrown Throwing-2-0-1 BaseballBat-2-0-1)
(valueOf (MeasurementAtFn ((QPQuantityFn Speed) BaseballBat-2-0-1) (StartFn Upward-Movement-2-0-
1))
 (MetersPerSecond 6.5))
(eventOutcomes Throwing-2-0-1 Movement-2-0-1)
(primaryObjectMoving Movement-2-0-1 BaseballBat-2-0-1)
(firstSubEvents Movement-2-0-1 Upward-Movement-2-0-1)
(maximumMotionInDirection Movement-2-0-1 Upward-Movement-2-0-1 Up-Directly)
(primaryObjectMoving Upward-Movement-2-0-1 BaseballBat-2-0-1)
(directionOfTranslation-Throughout Upward-Movement-2-0-1 Up-Directly)
(valueOf (MeasurementAtFn ((QPQuantityFn Altitude) BaseballBat-2-0-1) (EndFn Upward-Movement-2-0-
1)) (Meter 3))
(isa Acceleration-2-0-1 ScalarOrVectorInterval)
(isa NU-ETS-Query-2-0-1 KBContentTest-FullySpecified)
(isa NU-ETS-Query-2-0-1 ETSPhysicsSampleQuery)
(hypotheticalMicrotheoryOfTest NU-ETS-Query-2-0-1 Hyp-MT-ETS-Query-2-0-1)
(retainTerm (TestQueryFn NU-ETS-Query-2-0-1))
(testQuerySpecification NU-ETS-Query-2-0-1 (TestQueryFn NU-ETS-Query-2-0-1))
(querySentenceOfQuery (TestQueryFn NU-ETS-Query-2-0-1)
 (valueOf ((QPQuantityFn AccelerationDueToGravity) Planet-2-0-1) Acceleration-2-0-1))
(termToSolveFor (TestQueryFn NU-ETS-Query-2-0-1)
 (valueOf    ((QPQuantityFn    AccelerationDueToGravity)    Planet-2-0-1)    Acceleration-2-0-1)
Acceleration-2-0-1)
(isa Hyp-MT-ETS-Query-2-0-1-WS Microtheory)
(genlMt Hyp-MT-ETS-Query-2-0-1-WS Hyp-MT-ETS-Query-2-0-1)
(multipleChoiceSingleOptionList NU-ETS-Query-2-0-1 (TheList (MetersPerSecondPerSecond 2.2) "A"))
(multipleChoiceSingleOptionList NU-ETS-Query-2-0-1 (TheList (MetersPerSecondPerSecond 0.5) "B"))
(multipleChoiceSingleOptionList NU-ETS-Query-2-0-1 (TheList (MetersPerSecondPerSecond 7.0) "C"))
(multipleChoiceSingleOptionList NU-ETS-Query-2-0-1 (TheList (MetersPerSecondPerSecond 19.5) "D"))
```

Worked Solution for Problem 2 Representation

```
(isa ETS-WorkedSolution-2-0-1 Individual)
(isa ETS-WorkedSolution-2-0-1 PhysicsWorkedSolution)
(comment ETS-WorkedSolution-2-0-1
 "An  instance  of  PhysicsWorkedSolution.  ETS-WorkedSolution-2-0-1  is  a  worked  solution  for  the
question posed by NU-ETS-Query-2-0-1.")
(workedSolutionForKBContentTest NU-ETS-Query-2-0-1 ETS-WorkedSolution-2-0-1)
(workedSolutionMtForTestMt Hyp-MT-ETS-Query-2-0-1 Hyp-MT-ETS-Query-2-0-1-WS)
(isa ETS-WorkedSolution-2-0-1-Step1 WorkedSolutionStep)
(isa ETS-WorkedSolution-2-0-1-Step2 WorkedSolutionStep)
(isa ETS-WorkedSolution-2-0-1-Step3 WorkedSolutionStep)
(isa ETS-WorkedSolution-2-0-1-Step4 WorkedSolutionStep)
(isa ETS-WorkedSolution-2-0-1-Step5 WorkedSolutionStep)
(isa ETS-WorkedSolution-2-0-1-Step6 WorkedSolutionStep)
(isa ETS-WorkedSolution-2-0-1-Step7 WorkedSolutionStep)
(isa ETS-WorkedSolution-2-0-1-Step8 WorkedSolutionStep)
(firstSolutionStep ETS-WorkedSolution-2-0-1 ETS-WorkedSolution-2-0-1-Step1)
(hasSolutionSteps ETS-WorkedSolution-2-0-1 ETS-WorkedSolution-2-0-1-Step2)
(hasSolutionSteps ETS-WorkedSolution-2-0-1 ETS-WorkedSolution-2-0-1-Step3)
(hasSolutionSteps ETS-WorkedSolution-2-0-1 ETS-WorkedSolution-2-0-1-Step4)
(hasSolutionSteps ETS-WorkedSolution-2-0-1 ETS-WorkedSolution-2-0-1-Step5)
(lastSolutionStep ETS-WorkedSolution-2-0-1 ETS-WorkedSolution-2-0-1-Step6)
(priorSolutionStep ETS-WorkedSolution-2-0-1-Step2 ETS-WorkedSolution-2-0-1-Step1)
(priorSolutionStep ETS-WorkedSolution-2-0-1-Step3 ETS-WorkedSolution-2-0-1-Step2)
(priorSolutionStep ETS-WorkedSolution-2-0-1-Step4 ETS-WorkedSolution-2-0-1-Step3)
(priorSolutionStep ETS-WorkedSolution-2-0-1-Step5 ETS-WorkedSolution-2-0-1-Step4)
(priorSolutionStep ETS-WorkedSolution-2-0-1-Step6 ETS-WorkedSolution-2-0-1-Step5)
(solutionStepOperationType ETS-WorkedSolution-2-0-1-Step1 CategorizingAPhysicsProblem)
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1 (isa Throwing-2-0-1 ThrowingAnObject))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1 (eventOccursNear Throwing-2-0-1 Ground-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1 (groundOf Planet-2-0-1 Ground-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1 (isa Planet-2-0-1 Planet))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1
        (no-GenQuantRelnFrom in-ImmersedFully Planet-2-0-1 Atmosphere))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1 (objectThrown Throwing-2-0-1 BaseballBat-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1 (eventOutcomes Throwing-2-0-1 Movement-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1 (isa Movement-2-0-1 ProjectileMotion))
(solutionStepUses  ETS-WorkedSolution-2-0-1-Step1  (primaryObjectMoving  Movement-2-0-1  BaseballBat-
2-0-1))
(solutionStepUses  ETS-WorkedSolution-2-0-1-Step1  (firstSubEvents  Movement-2-0-1  Upward-Movement-
2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1 (isa Upward-Movement-2-0-1 ProjectileMotion))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1
 (maximumMotionInDirection Movement-2-0-1 Upward-Movement-2-0-1 Up-Directly))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1
 (primaryObjectMoving Upward-Movement-2-0-1 BaseballBat-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1
 (directionOfTranslation-Throughout Upward-Movement-2-0-1 Up-Directly))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1
 (valueOf (MeasurementAtFn ((QPQuantityFn Speed) BaseballBat-2-0-1) (StartFn Upward-Movement-2-0-
1))
  (MetersPerSecond 6.5)))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step1
 (valueOf (MeasurementAtFn ((QPQuantityFn Altitude) BaseballBat-2-0-1) (EndFn Upward-Movement-2-
0-1))
  (Meter 3)))
(solutionStepResult ETS-WorkedSolution-2-0-1-Step1
 (isa NU-ETS-Query-2-0-1 PhysicsProblem-DistanceVelocity))
(solutionStepResult ETS-WorkedSolution-2-0-1-Step1
 (isa NU-ETS-Query-2-0-1 PhysicsProblem-ConstantAcceleration))
(solutionStepOperationType ETS-WorkedSolution-2-0-1-Step2 SubstitutingBindingsForVariables)
(solutionStepUses ETS-WorkedSolution-2-0-1-Step2
 (isa NU-ETS-Query-2-0-1 PhysicsProblem-ConstantAcceleration))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step2
 (isa NU-ETS-Query-2-0-1 PhysicsProblem-DistanceVelocity))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step2
 (equationFormFor DistanceVelocityUnderConstantAcceleration
  (mathEquals (SquaredFn (MeasurementAtFn ((QPQuantityFn Speed) ?OBJ) (EndFn ?INTERVAL)))
   (PlusFn (SquaredFn (MeasurementAtFn ((QPQuantityFn Speed) ?OBJ) (StartFn ?INTERVAL)))
    (TimesFn 2 (MeasurementAtFn ((QPQuantityFn Acceleration) ?OBJ) ?INTERVAL)
     ((QPQuantityFn DistanceTravelled) ?OBJ ?INTERVAL))))))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step2 (isa Upward-Movement-2-0-1 ProjectileMotion))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step2 (isa BaseballBat-2-0-1 BaseballBat))
(solutionStepResult ETS-WorkedSolution-2-0-1-Step2
 (equationForSolution ETS-WorkedSolution-2-0-1
  (mathEquals (MeasurementAtFn ((QPQuantityFn Acceleration) BaseballBat-2-0-1) Upward-Movement-2-
0-1)
   (QuotientFn
    (DifferenceFn
     (SquaredFn (MeasurementAtFn ((QPQuantityFn Speed) BaseballBat-2-0-1)
      (EndFn Upward-Movement-2-0-1)))
     (SquaredFn (MeasurementAtFn ((QPQuantityFn Speed) BaseballBat-2-0-1)
      (StartFn Upward-Movement-2-0-1))))
    (TimesFn 2 ((QPQuantityFn DistanceTravelled) BaseballBat-2-0-1 Upward-Movement-2-0-1))))))
(solutionStepOperationType ETS-WorkedSolution-2-0-1-Step3
 DeterminingSpecificScalarOrVectorValuesFromContext)
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3 (isa Throwing-2-0-1 ThrowingAnObject))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3 (eventOccursNear Throwing-2-0-1 Ground-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3 (groundOf Planet-2-0-1 Ground-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3 (isa Planet-2-0-1 Planet))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3
 (no-GenQuantRelnFrom in-ImmersedFully Planet-2-0-1 Atmosphere))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3 (objectThrown Throwing-2-0-1 BaseballBat-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3 (eventOutcomes Throwing-2-0-1 Movement-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3 (isa Movement-2-0-1 ProjectileMotion))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3 (primaryObjectMoving Movement-2-0-1 BaseballBat-
2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3 (firstSubEvents Movement-2-0-1 Upward-Movement-
2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3 (isa Upward-Movement-2-0-1 ProjectileMotion))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3
 (maximumMotionInDirection Movement-2-0-1 Upward-Movement-2-0-1 Up-Directly))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3
 (primaryObjectMoving Upward-Movement-2-0-1 BaseballBat-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3
 (directionOfTranslation-Throughout Upward-Movement-2-0-1 Up-Directly))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3
 (valueOf (MeasurementAtFn ((QPQuantityFn Altitude) BaseballBat-2-0-1) (EndFn Upward-Movement-2-
0-1))
  (Meter 3)))
```

```
(solutionStepUses ETS-WorkedSolution-2-0-1-Step3
 (valueOf (MeasurementAtFn ((QPQuantityFn Speed) BaseballBat-2-0-1) (StartFn Upward-Movement-2-0-
1))
  (MetersPerSecond 6.5)))
(solutionStepResult ETS-WorkedSolution-2-0-1-Step3
 (valueOf ((QPQuantityFn DistanceTravelled) BaseballBat-2-0-1 Upward-Movement-2-0-1) (Meter 3)))
(solutionStepResult ETS-WorkedSolution-2-0-1-Step3
 (valueOf  (MeasurementAtFn  ((QPQuantityFn  Speed)  BaseballBat-2-0-1)  (EndFn  Upward-Movement-2-0-
1))
  (MetersPerSecond 0)))
(solutionStepResult ETS-WorkedSolution-2-0-1-Step3
 (valueOf (MeasurementAtFn ((QPQuantityFn Acceleration) BaseballBat-2-0-1) Upward-Movement-2-0-1)
  ((QPQuantityFn AccelerationDueToGravity) Planet-2-0-1)))
(solutionStepOperationType ETS-WorkedSolution-2-0-1-Step4 SolvingAMathematicalEquation)
(solutionStepUses ETS-WorkedSolution-2-0-1-Step4
 (equationForSolution ETS-WorkedSolution-2-0-1
  (mathEquals (MeasurementAtFn ((QPQuantityFn Acceleration) BaseballBat-2-0-1) Upward-Movement-2-
0-1)
   (QuotientFn
    (DifferenceFn
     (SquaredFn (MeasurementAtFn ((QPQuantityFn Speed) BaseballBat-2-0-1)
      (EndFn Upward-Movement-2-0-1)))
     (SquaredFn (MeasurementAtFn ((QPQuantityFn Speed) BaseballBat-2-0-1)
      (StartFn Upward-Movement-2-0-1))))
    (TimesFn 2 ((QPQuantityFn DistanceTravelled) BaseballBat-2-0-1 Upward-Movement-2-0-1))))))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step4
 (valueOf ((QPQuantityFn DistanceTravelled) BaseballBat-2-0-1 Upward-Movement-2-0-1) (Meter 3)))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step4
 (valueOf (MeasurementAtFn ((QPQuantityFn Speed) BaseballBat-2-0-1) (StartFn Upward-Movement-2-0-
1))
  (MetersPerSecond 6.5)))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step4
 (valueOf  (MeasurementAtFn  ((QPQuantityFn  Speed)  BaseballBat-2-0-1)  (EndFn  Upward-Movement-2-0-
1))
  (MetersPerSecond 0)))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step4
 (valueOf (MeasurementAtFn ((QPQuantityFn Acceleration) BaseballBat-2-0-1) Upward-Movement-2-0-1)
  ((QPQuantityFn AccelerationDueToGravity) Planet-2-0-1)))
(solutionStepResult ETS-WorkedSolution-2-0-1-Step4
 (valueOf ((QPQuantityFn AccelerationDueToGravity) Planet-2-0-1)
  (MetersPerSecondPerSecond (MinusFn 63.4))))
(solutionStepOperationType ETS-WorkedSolution-2-0-1-Step5 SanityCheckingPhysicsProblemSolution)
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5 (isa Throwing-2-0-1 ThrowingAnObject))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5 (eventOccursNear Throwing-2-0-1 Ground-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5 (groundOf Planet-2-0-1 Ground-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5 (isa Planet-2-0-1 Planet))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5
 (no-GenQuantRelnFrom in-ImmersedFully Planet-2-0-1 Atmosphere))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5 (objectThrown Throwing-2-0-1 BaseballBat-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5 (eventOutcomes Throwing-2-0-1 Movement-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5 (isa Movement-2-0-1 ProjectileMotion))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5 (primaryObjectMoving Movement-2-0-1 BaseballBat-
2-0-1))
(solutionStepUses  ETS-WorkedSolution-2-0-1-Step5  (firstSubEvents  Movement-2-0-1  Upward-Movement-
2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5 (isa Upward-Movement-2-0-1 ProjectileMotion))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5
 (maximumMotionInDirection Movement-2-0-1 Upward-Movement-2-0-1 Up-Directly))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5
 (primaryObjectMoving Upward-Movement-2-0-1 BaseballBat-2-0-1))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5
 (directionOfTranslation-Throughout Upward-Movement-2-0-1 Up-Directly))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5
 (valueOf (MeasurementAtFn ((QPQuantityFn Speed) BaseballBat-2-0-1) (StartFn Upward-Movement-2-0-
1))
  (MetersPerSecond 6.5)))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5
 (valueOf (MeasurementAtFn ((QPQuantityFn Altitude) BaseballBat-2-0-1) (EndFn Upward-Movement-2-
0-1))
  (Meter 3)))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step5
 (valueOf ((QPQuantityFn AccelerationDueToGravity) Planet-2-0-1)
  (MetersPerSecondPerSecond (MinusFn 63.4))))
(solutionStepResult ETS-WorkedSolution-2-0-1-Step5
 (consistent
  (valueOf ((QPQuantityFn AccelerationDueToGravity) Planet-2-0-1)
   (MetersPerSecondPerSecond (MinusFn 63.4)))))
(solutionStepOperationType                                          ETS-WorkedSolution-2-0-1-Step6
DeterminingTheBestAnswerFromASetOfChoices)
(solutionStepUses ETS-WorkedSolution-2-0-1-Step6
 (multipleChoiceSingleOptionList  NU-ETS-Query-2-0-1  (TheList  (MetersPerSecondPerSecond  2.2)
"A")))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step6
 (multipleChoiceSingleOptionList  NU-ETS-Query-2-0-1  (TheList  (MetersPerSecondPerSecond  0.5)
"B")))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step6
 (multipleChoiceSingleOptionList  NU-ETS-Query-2-0-1  (TheList  (MetersPerSecondPerSecond  7.0)
"C")))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step6
 (multipleChoiceSingleOptionList  NU-ETS-Query-2-0-1  (TheList  (MetersPerSecondPerSecond  19.5)
"D")))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step6
 (valueOf ((QPQuantityFn AccelerationDueToGravity) Planet-2-0-1)
  (MetersPerSecondPerSecond (MinusFn 63.4))))
(solutionStepUses ETS-WorkedSolution-2-0-1-Step6
 (consistent
  (valueOf ((QPQuantityFn AccelerationDueToGravity) Planet-2-0-1)
   (MetersPerSecondPerSecond (MinusFn 63.4)))))
(solutionStepResult ETS-WorkedSolution-2-0-1-Step6
 (testAnswers-SingleCorrectMultipleChoice NU-ETS-Query-2-0-1 "C"))
```