Learning Domain Theories via Analogical Transfer

Matthew Klenk and Kenneth D. Forbus

Qualitative Reasoning Group, Northwestern University
2133 Sheridan Road, Evanston, IL 60208 USA
{m-klenk, forbus}@northwesterrn.edu

Abstract

Learning domain theories is an important challenge for qualitative reasoning. We describe a method for learning new domain theories by analogy. We use analogies between pairs of problems and worked solutions to create a mapping between the familiar and the new domains, and use this mapping to conjecture general knowledge about the new domain. After some knowledge has been learned about the new domain, another analogy is made between the domain theories themselves providing conjectures about the new domain. An experiment is described where the system learns to solve rotational kinematics problems by analogy with translational kinematics problems, outperforming a version of the system that is incrementally given the correct domain theory.

Introduction

Progress in qualitative reasoning has led to a variety of techniques for model formulation, making predictions, performing diagnosis, and handling other tasks. However, little effort has focused on the process of learning domain theories. To be sure, in some cases hand-engineering domain theories is sufficient. However, this can be a very time-consuming process, requiring considerable effort. Being able to re-use this investment by automatically constructing theories for similar domains could be of great practical value. Furthermore, there is ample evidence that people heavily use analogy to learn new domains (Gentner & Gentner 1983; Gentner 2003). Systems that learn domain theories by analogy could be used to model human learning.

Falkenhainer's (1988) PHINEAS system was the first QR system to address this problem. Based on the hypothesis that diagnosis, explanation and theory formation are all intertwined, PHINEAS used *similarity-driven* explanation to show how analogy can be used to develop new theories about specific situations. As a learning agent works in a new domain, it should be able to transfer knowledge from previous well understood domains. Falkenhainer called the inability to offer a best guess or apply knowledge across domains the *adaptability problem*.

Textbook authors routinely exploit human adaptability (Shive & Weber 1982). In the linear kinematics section of the textbook used for this study (Giancoli 1991), there are eight worked out examples, *worked solutions*, which show all of the different ways in which the four linear kinematics equations can be used. But in the later rotational kinematics section, there are only two worked solutions. Furthermore, two of the rotational kinematics equations are not part of any worked solutions in the book. The summary section of rotational motion chapter invites the learner to use analogy to fill in the details: "The dynamics of rotation is analogous to the dynamics of linear motion" (p. 197, Giancoli 1991). This is common practice in textbooks, and analogies between domains form the basis of system dynamics (Olson 1966; Shearer *et al.* 1967).

This paper describes how analogies between worked solutions can be used to learn domain theories. Our strategy is itself analogous to that used in PHINEAS, which used comparisons of (simulated) behavior to create an initial cross-domain mapping that was subsequently used to create a partial theory for the new domain. It differs, however, in several significant ways: (1) We use analogies between worked solution pairs to drive the process, (2) We are learning quantitative, rather than qualitative, domain theories, which requires very different verification work, and (3) We are using a more psychologically plausible retrieval mechanism. While our current work focuses on quantitative domain theories, our method should also be usable for qualitative domain theories as well.

We start by describing our representations and problem-solver. Next we review the ideas of structure-mapping theory and our computational models which are used in this work. Then we describe our learning method, and present an experiment showing that it can learn rotational kinematics by analogy with translational kinematics, and do so faster than a system that is told the laws of the domain incrementally. We close with a discussion of related work and future plans.

Representation and Problem Solving

Representing physics problems requires a broad background of everyday knowledge, including the object and event types found in such problems. We use the ResearchCyc[1] knowledge base contents, augmented with our own extensions, as our starting point. Our extensions

[1] http://research.cyc.com/

```
(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 Distance) Car-2-6 Acc-2-6)
  (Meter 30))
...
(query (valueOf ((QPQuantityFn Time-Quantity)
  Acc-2-6) Duration-2-6))
```

Figure 1: Problem 2-6 Representation (sample)

concern QP 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.

Example Problem and Worked Solution

All problems and worked solutions used in this work were taken from the same physics textbook (Giancoli 1991). Problems are defined as cases. 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 2-6, p. 26). This problem is represented in our system as a case of 10 facts, a subset of which appears in Figure 1.

Worked solutions are represented at the level of examples found in textbooks, which is more abstract than a proof or problem-solving trace. For example, the worked solution for problem 2-6 consisted of four steps:

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)

Figure 2 shows how step 3 is represented.

```
(isa Gia-2-7-Step-3 WorkedSolutionStep)
(hasSteps Gia-2-7-WS Gia-2-7-Step-3)
(priorStep Gia-2-7-Step-3 Gia-2-7-Step-2)
(stepType Gia-2-7-Step-3 AssumingValue)
(stepUses Gia-2-6-WS-Step-3
 (objectStationary (StartFn Acc-2-6) Car-2-6))
(stepResult Gia-2-6-WS-Step-3
 (valueOf
    (AtFn ((QPQuantityFn Speed) Car-2-6)
    (StartFn Acc-2-6))
  (MetersPerSecond 0)))
```

Figure 2: Problem 2-6 worked solution step 3

Domain Theories

Our domain theories consist of *encapsulated histories* (Forbus 1984) representing equations. Encapsulated histories are templates describing pieces of histories (Hayes 1978). They were motivated by two concerns. First, some phenomena are best described by discontinuous

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

Figure 3: Example Encapsulated History

patterns of events (e.g., collisions). Second, they permit constraints to be placed on time itself, which is not possible for model fragments, given their semantics (i.e., time is implicit, and their consequences hold throughout whatever period they are active). Equations like the velocity/time law above hold over events (e.g., translational motion under constant acceleration), and hence encapsulated histories are the appropriate mechanism for describing the conditions under which they hold.

Figure 3 illustrates the encapsulated history representing the equation of velocity as a function of time ($v_f=v_i+at$). 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 compound form shown in Figure 3 is automatically translated into a set of predicate calculus facts. While the consequence of this encapsulated history is a quantitative equation, the same representation could be used to represent qualitative relationships. Similarly, this technique should be adaptable to learning model fragments as well.

Solving a Problem

Our system solves for quantities in three ways. First, the quantity may already be known as part of the problem. Second, rules can be used to apply modeling assumptions, i.e., "Objects at rest have no velocity". Third, an encapsulated history may be instantiated that results in an equation containing the sought after quantity. This is done by satisfying the participant constraints and the encapsulated history conditions statements in the problem. Once the encapsulated history has been instantiated, the system solves for the other quantities in the equation, and then attempts to solve the equation for the original parameter. The algebra routines are based upon the system in Forbus and de Kleer (1993). Both the problem-solving strategies and the mathematics knowledge are fixed in the current system, and cannot be extended via learning.

Structure-mapping and Analogy

We use Gentner's (1983) structure-mapping theory, which postulates that analogy and similarity are based on structural alignment between two structured representations (the *base* and *target*) to find the maximal structurally consistent match between them. A structurally consistent match must satisfy the constraints of *tiered-identicality*, *parallel connectivity*, and *one-to-one mapping*. Tiered-identicality constraint provides a strong preference for only allowing identical predicates to match, but allows for exceptions, when doing so would enable a much larger structure to match. The parallel connectivity constraint says that if two statements are matched then their arguments must also match. 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*: a preference for mappings that are highly interconnected and contain deep chains of higher order relations.

The Structure Matching Engine (SME) simulates the process of analogical matching between a base and target (Falkenhainer *et al.* 1989). The output of this process is one or more *mappings*. A mapping is a set of *correspondences* representing a construal of what items (*entities* and *expressions*) in the base go with what items in the target. Mappings include a *structural evaluation score* indicating the strength of the match, and *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. SME operates in polynomial time, using a greedy algorithm (Forbus & Oblinger, 1990).

MAC/FAC (Forbus *et. al.* 1994) models similarity-based retrieval. The inputs are a case, the *probe*, and a library of cases. The first stage (MAC) uses a computationally cheap, non-structural matcher to filter candidates from a pool of memory items, returning up to three if they are very close. The second stage (FAC) uses SME to compare the cases retruned by MAC to the probe and returns the best candidate (or candidates, if they are very similar). Both SME and MAC/FAC have been used as performance systems in a variety of domains and as cognitive models to account for a variety of psychological results (Forbus 2001).

Different domains are often represented using different predicates, especially when they are first being learned and underlying commonalities with previous knowledge have not yet been found. *Minimal ascension* (Falkenhainer 1988) is one method for matching non-identical predicates. If two predicates are part of a larger aligned structure and share a close common ancestor in the taxonomic hierarchy, then SME can include them in the mapping. For example, given the statements in Figure 4, if the `stepUses` statements are aligned as well as the `Step-Base` and `Step-Target`, `Obj-Base` and `Obj-Target`, and `Event-Base` and `Event-Target`, then SME will attempt to match `primaryObjectMoving` with `objectRotating`. They are siblings in the ResearchCyc ontology, and hence minimal ascension allows them to be placed into correspondence.

Base Expression:

```
(stepUses Step-Base
  (primaryObjectMoving Event-Base Obj-Base))
```

Target Expression:

```
(stepUses Step-Target
  (objectRotating Event-Target Obj-Target))
```

Figure 4: Minimal Ascension maps `primaryObjectMoving` to `objectRotating`

Analogical Learning of Domain Theories

Our system learns a domain theory by using multiple analogies. Learning is invoked when it fails to solve a problem. After failing to solve a problem, the system is given a worked solution for that problem, as a student might get out of a textbook. It uses this worked solution to create conjectures about knowledge in the new domain, using the algorithm outlined in Figure 5. The case library contains a set of worked solutions from the known domain. First, the worked solution for the failed problem is used as a probe to MAC/FAC, to retrieve an analogous worked solution from memory. A comparison is made using SME, with the retrieved worked solution constituting the base and the worked solution for the failed problem as the target. The mappings SME produces are then combined to create a *domain mapping*. The reason for combining multiple mappings is that each mapping often covers only some aspects of the solution. The best mapping is used as a starting point, with correspondences drawn from the others included only if they do not violate the one-to-one

1. Retrieve analog using the target worked solution as a probe in MAC/FAC
2. Use SME to create a match between the analog and the worked solution
3. Retrieve correspondences from resulting mappings
4. Create domain mapping by selecting correspondences in which the base element appears in the base domain theory
5. Initialize target domain theory using these correspondences
6. Use SME to create a match between the base and the target theories constrained by the domain mapping
7. Transfer domain theory using the candidate inferences
8. Verify learned domain theory by attempting the failed problem again
9. If failure, go once more to step 1. Otherwise, accept new target domain knowledge as correct

Figure 5: Analogical Domain Learning

constraint.

When the system gets the first problem in a new domain, its theory for that domain is empty. The candidate inferences for the domain mapping thus become the basis for a new domain theory. We currently require that every concept in the encapsulated history is mentioned in the domain mapping, i.e., there are no analogy skolems where we must postulate a new predicate or category of entity. If there is enough similar structure between the worked solutions, at least one encapsulated history will be created. If no encapsulated histories can be created due to an inability to find a satisfactory domain mapping, the system does not try to learn anything from this particular failure.

The system also extends a partially learned, or just initialized, domain theory with another analogy. The domain mapping becomes required correspondence constraints of a new analogy between the base and target domain theories themselves, ensuring that the overall domain theory is consistent. As before, any encapsulated history imported into the target becomes a conjecture about the new domain theory.

While powerful, analogies are not guaranteed to be sound. Consequently, we verify the newly proposed domain knowledge by trying again to solve the problem whose failure motivated the learning. If this problem is solved correctly, our system assumes that the new domain theory constructs are correct. Otherwise, it deletes both the new domain theory constructs and the domain mapping. Then, it tries one more time, considering the next best worked solution retrieved from memory.

Experiment

To examine how well this analogical learning method works, we need a baseline. Our baseline *spoon-fed* system consists of the same problem-solver, but with analogical learning turned off. When it receives a problem it cannot solve, it is given not just a worked solution, but whatever general encapsulated histories are needed to solve that specific target domain problem. In other words, it is given the correct knowledge, in its internal representations, ready for future use. This makes for a tough comparison, since our system in the analogy condition must figure out the encapsulated histories for itself.

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 6: Test Problem Set

Method

Both systems begin with a linear kinematics domain theory, two worked solutions of linear kinematics problems, and hard-coded rules for problem-solving strategies and making modeling decisions. The systems are then tested on how quickly they can learn rotational kinematics problems. The testing materials are 5 problems, listed in Figure 6, and worked solutions. Learning curves were created by running 120 trials representing every possible ordering of the test materials. In each trial, after each problem, the system was given either the worked solution or encapsulated histories for that problem, depending on the condition. After each trial, the system's knowledge was reset.

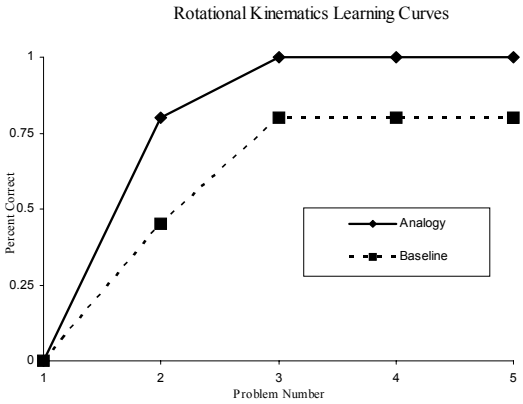

Figure 7: Experiment Results

Results

Figure 7 compares the learning curves for the analogy and baseline conditions. After studying just one worked solution, the analogy system was able to solve next problem correctly 80 percent of time. Furthermore, the analogy system has perfect performance after working on just two problems. The baseline system's ceiling was at 80 percent, and after one problem it was only able to get the next problem correct 45 percent of the time.

Further analysis of these results details the strength of the analogy approach. The baseline system failed to score above 80 percent of any of the conditions. The baseline system was unable to solve problem 'b' from Figure 6 regardless of what problems it has already seen, because none of the other problems use the same equation. The analogy system performed quite well, only in one situation

did the analogical domain transfer fail to learn the whole rotational kinematics domain after just one worked solution. This occurred when problem 'b' was the first problem. Problem 'b' makes no mention of a time quantity preventing a correspondence to be created for it. While a time quantity exists in both of these domains, it does not necessarily mean they should be aligned. The strength of the analogical approach is that transfer is guided by structural similarity. This is critical for broader application of this theory. For example, in linear and rotational dynamics, both domain theories have a mass quantity, but transfer is only possible when a domain mapping is made between mass, in linear dynamics, and moment of inertia, in rotational dynamics. (e.g. F=ma and $T=I\alpha$)

Related Work

As noted above, the closest work is Falkenhainer's PHINEAS (1988), which learned qualitative descriptions of processes based on analogies involving behaviors. PHINEAS used envisioning to verify its conjectures, whereas we use mathematical problem solving. Klenk & Forbus (2007) describe a system that learns by accumulating examples to solve AP physics problems within the same domain. Klenk *et al.* (2005) describe a system that learns causal models via analogies involving sketches annotated with causal knowledge. Both of these systems only learn within the same domain, and neither constructs general domain theories, unlike the system described here. Silver (1986) used explanation-based learning to acquire new mathematical skills, by contrast our system's mathematical knowledge is hard-wired.

In the QR community, de Kleer's work (1977) in reasoning on sliding motion problems demonstrated that qualitative reasoning was required for solving many quantitative physics problems. More recent AI work on transfer learning has recognized the importance of generating mappings between domains to allow for knowledge transfer, Liu and Stone (2006) use a version of SME to accelerate learning of state action policies in keep away soccer. Instead of using structure-mapping to accelerate learning, we use structure-mapping to learn new general domain concepts.

Discussion

We have shown that a domain theory for solving physics problems can be learned via cross-domain analogies. Our experiment shows furthermore that such analogical learning can be very efficient, when the two domains are sufficiently similar. The process of constructing domain mappings by exploiting similarities in worked solutions, and using that to import theories from one domain to another, is, we believe, a general and powerful process.

There are several directions we intend to pursue next. First, we have only tested this method with encapsulated histories, so we want to extend it to handle other types of domain knowledge. Based on experience in other analogical learning tasks, we believe that this will mainly involve figuring out the appropriate verification techniques. Second, we plan to integrate this algorithm into the Companion-based learning system of Klenk & Forbus (2007), so that we can combine both ways of analogical learning. We plan to explore a broader range of domain pairs, including domains which are quite distant, to explore better strategies for making use of weaker matches. We suspect that model-based diagnosis techniques could be used to debug analogically-derived domain theories, based on their success with diagnosing misconceptions in student models (de Koning *et al.* 2000).

We also expect that these techniques could be used more broadly in the QR community for accelerating the process of constructing domain theories. That is, given modeling environments designed to help domain experts create theories (cf. Bredeweg *et al.* 2006), there should be a growing library of domain theories to draw upon. An analogy-based assistant could help spot cross-domain connections, accelerating the process of constructing new domain theories.

Acknowledgements

This research was supported by the Cognitive Science Program of the Office of Naval Research.

References

Bredeweg, B., Salles, P., Bouwer, A., and Liem, J. 2006. Garp3 – A new Workbench for Qualitative Reasoning and Modelling. *Proceedings of the 20th International Workshop on Qualitative Reasoning*. Hanover, NH.

de Kleer, J. 1977. Multiple representations of knowledge in a mechanics problem solver. In *Proceedings of IJCAI-77*.

de Koning, K., Bredeweg, B., Breuker, J., and Wielinga, B. 2000. Model-Based Reasoning about Learner Behavior. *Artificial Intelligence*, 117(2).

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., Forbus, K. and Gentner, D. 1989. The Structure-Mapping Engine. *Artificial Intelligence.* 41.

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

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

Forbus, K. & de Kleer, J. 1993. *Building Problem Solvers*, MIT Press.

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

Forbus, K. & Oblinger, D. (1990). Making SME greedy and pragmatic. In *Proceedings of CogSci-1990*.

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

Gentner, D. 2003. Why we're so smart. In Gentner, D. and Goldin-Meadow, S. (Eds.), *Language in mind: Advances in the study of language and thought.* pp. 195-235. MIT Press.

Gentner, D. and Gentner, D. R. 1983. Flowing waters or teeming crowds: Mental models of electricity. In D. Gentner & A. Stevens (Eds.), *Mental Models*. Lawrence Erlbaum Associates.

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

Hayes, P. 1978. The Naive Physics Manifesto. In D. Michie (ed), *Expert Systems in the Microelectronic Age*. Edinburgh University Press, Edinburgh, Scotland.

Klenk, M., Forbus, K., Tomai, E., Kim,H., and Kyckelhahn, B. 2005. Solving Everyday Physical Reasoning Problems by Analogy using Sketches *Proceedings of 20th National Conference on Artificial Intelligence (AAAI-05)*.

Klenk, M., and Forbus, K. 2007. Measuring the level of transfer learning by an AP Physics problem-solver. In *Proceedings of AAAI-07*. Vancouver, CA.

Olson, H. 1966. *Solutions of Engineering Problems by Dynamical Analogies*, D. Van Nostrand.

Shearer, J., Murphy, A., and Richardson, H. 1967. *Introduction to Systems Dynamics*. Addison-Wesley Publishing Company.

Shive, J. and Weber, R. 1982. *Similarities in Physics*. Adam Hilger Ltd.

Silver, B. 1986. *Meta-Level Inference: Representing and Learning Control Information in Artificial Intelligence*. Elsevier.