

2002

Cognitive model of new data on human problem solving

Eric Fleischman
Colby College

Follow this and additional works at: <https://digitalcommons.colby.edu/seniorscholars>



Part of the [Psychology Commons](#)

Colby College theses are protected by copyright. They may be viewed or downloaded from this site for the purposes of research and scholarship. Reproduction or distribution for commercial purposes is prohibited without written permission of the author.

Recommended Citation

Fleischman, Eric, "Cognitive model of new data on human problem solving" (2002). *Senior Scholar Papers*. Paper 85.

<https://digitalcommons.colby.edu/seniorscholars/85>

This Senior Scholars Paper (Open Access) is brought to you for free and open access by the Student Research at Digital Commons @ Colby. It has been accepted for inclusion in Senior Scholar Papers by an authorized administrator of Digital Commons @ Colby.

A COGNITIVE MODEL OF NEW DATA ON
HUMAN PROBLEM SOLVING

by

ERIC S. FLEISCHMAN

Submitted in Partial Fulfillment of the Requirements of the
Senior Scholars Program

COLBY COLLEGE
2002

Acknowledgements

First and foremost, I would like to thank my advisor, Randy Jones. Without Randy none of this work would have been possible. Over the past two years Randy has been patient enough to work with me and teach me how cognitive models work and why we should build them. He has also given me the opportunity to publish and present this work in a variety of forums, including the Cognitive Science Society and The Consortium for Computing in Small Colleges.

Additionally, I would like to thank both Clare Congdon of Computer Science and Nicholas Rohrman of Psychology for being readers of my paper. I appreciate all of the time and feedback you have given me.

Finally, thanks to all of the other organizations that have supported this work, especially the Senior Scholars Program at Colby College. I would also like to extend a special thank you to Alexander Renkl and Robert Atkinson (and their research groups), who provided me with valuable data on the fading effect, as well as Colleen Burnham and the rest of the Colby Psychology Department, who helped me run my own experiments. This work would never have been possible without your support and assistance.

ABSTRACT

During the educational process there are a multitude of strategies that educators may employ in order to maximize learning among their pupils. For symbolic problem-solving skills (such as mathematics), a particularly effective technique is to have students study worked-out example problems and unworked practice problems. Research shows that students who explain parts of worked examples to themselves learn more effectively than students who do not. This is called the *self-explanation effect* (Chi et al., 1989; Fergusson-Hessler & de Jong, 1990; Pirolli & Bielaczyc, 1989). In summary, this research proved that students learn most effectively by studying examples when they are careful to explain to themselves as many steps of the example as they can. This theory further states that students who do not carefully explain worked out example steps do not perform as well on subsequent problems. To explain this result, VanLehn and Jones (VanLehn, Jones & Chi, 1991) developed Cascade, a cognitive model that posits particular memory, problem-solving, and learning mechanisms to account for the *self-explanation effect*.

Subsequent psychological research has concluded that *fading* completely worked examples can further improve learning (Renkl, Atkinson & Maier, 2000). Fading consists of removing some of the solution steps in an example, forcing students to solve those portions themselves. To explain this new result, Jones and Fleischman (2001) used Cascade to model the basic cognitive processes of subjects given faded examples. This explanation relied on a small set of assumptions, to be verified by future experiments on human subjects.

My primary work this year has been in collaboration with psychologists who have run further detailed experiments, in part to test the predictions we made last year. The new data provide detailed behavior traces of students studying a variety of different sets of example problems, in order to learn some basic principles of mathematical probabilities. The detailed nature of the data allows me to use Cascade to do a very precise analysis of the errors the subjects generate, the learning episodes they experience, and the knowledge they acquire. Part of my task has been to engineer Cascade's knowledge base from classical physics to the probability principles in the current experiments. The rest of my work involves coding the subject behavior traces and tuning Cascade's parameters to fit them. The upshot of this work is that the new data are consistent with the predictions we made last year, providing further evidence that Cascade is a useful model of human learning and problem solving, and further insight into effective teaching procedures for problem-solving skills.

Contents

CHAPTER 1	Introduction	1
1.1	Research Goals	1
1.2	Symbolic Problem-Solving Skills	1
1.3	Overview of the Results to be Modeled	2
1.4	Using Cognitive Modeling as the Research Approach	3
1.5	Contributions of the Current Research	5
CHAPTER 2	Improving Learning in Problem Solving	8
2.1	Studying Examples	8
2.2	Self-Explaining Examples	9
2.3	Fading Examples	10
CHAPTER 3	What Cascade Is and How it Works	12
CHAPTER 4	Cascade and Fading	16
4.1	Cascade Provides a Potential Explanation of the Fading Effect	16
4.2	Initial Results	17
4.3	Cascade's Predictions about Fading	18
CHAPTER 5	Modeling Methods	20
5.1	Engineering Knowledge for Probability Problems	20
5.2	Representing Faded Examples	22
5.3	Knowledge as Parameters	23
5.4	Tuning the Knowledge Base to Model Individual Subject Behavior	24
CHAPTER 6	Modeling the Data to the Predictions	25
6.1	Initial Results and Predictions off of this Data	26
CHAPTER 7	Ongoing Psychological Experimentation	28

CHAPTER 8	Conclusions	29
8.1	A Review of Cascade's Predictions	29
8.2	Results	30
8.3	What does this tell us about Learning and Teaching	30
CHAPTER 9	Future Work	33
9.1	Improving Cascade	33
9.2	Further Data Analysis	34
APPENDIX A	Outline of Cascade's Algorithm	35
APPENDIX B	Cascade Knowledge Base for Probability	37
APPENDIX C	Probability Problems Encoded in Cascade	40
APPENDIX D	Sample Protocol Encodings	46
APPENDIX E	Sample Task Analysis	54
APPENDIX F	How to Model Subjects	57
BIBLIOGRAPHY	References	62

1 Introduction

1.1 Research Goals

Over the past decade there has been a substantial amount of psychological research that aimed to better understand the mechanisms involved in student learning of quantitative problem solving techniques (Chi et al., 1989; Fergusson-Hessler & de Jong, 1990; Pirolli & Bielaczyc, 1989; Renkl, Atkinson & Maier, 2000). In that time, researchers have identified some significant factors that correlate with improvement in certain types of symbolic problem-solving skills. One goal of this sort of work has been to help educators build more effective teaching materials by understanding the mechanisms behind the observed influences on learning.

This work has several goals, one of which is to gain a better understanding of the mechanisms that make students learn more effectively in certain situations. With this sort of insight one could do many things including more effectively designing teaching materials and making other sorts of modifications to the curriculum. An additional goal is to see whether one can apply an existing model of learning to a new psychological result. If one can make such an application it provides additional evidence for the theory. That is, we can be more confident in using the theory to understand human learning and education.

1.2 Symbolic Problem-Solving Skills

There are many different types of problems that people learn to solve. This research focuses on symbolic problem solving. Symbolic problems involve the

manipulation of symbols in order to ascertain an answer. This is clearly the sort of problem solving involved in, for example, mathematical calculations, but it also appears in other domains such as logic and other sorts of proofs, as well as the manipulation of chemical formulas.

These types of problems are well suited to be attacked with modern artificial intelligence techniques and systems. Traditional AI systems have the ability to process systematically many different combinations of equations, which provides an excellent framework for implementing a symbolic problem-solving system. Such systems can also be useful for other types of symbolic tasks, such as word problems, historical analysis and paper writing. Although these tasks are difficult due to their abstract and ambiguous nature, they share many of the same concepts one finds in the more formal types of symbolic problems described above.

1.3 Overview of the Results to be Modeled

Previous psychological research has studied a variety of methods to aid learning in human problem solving. Early work demonstrated that solving practice problems helps people learn. By solving such problems subjects are able to learn the rules required as well as rehearse standard solution approaches for the problems.

Subsequent research found that students who study completely solved example problems and then practice some unworked problems learn even more effectively than practicing the unworked problems alone (e.g., Chi et al., 1989; Pirolli & Anderson, 1985; Renkl, 1997, VanLehn, 1996). Among other advantages, by studying examples and then solving problems subjects are able to make connections between solving strategies, thus

allowing for solutions by analogy. After this result further work found that example studying is even more effective if students employ a technique called *self-explanation*. (Chi et al., 1989; Fergusson-Hessler & de Jong, 1990; Pirolli & Bielaczyc, 1989) Self-explanation is the process of thoroughly understanding each solution step provided by an example, as opposed to simply reading the example lines and accepting them without any significant analysis.

Some of the most recent research studies the process of gradually *fading* steps from the worked examples (Renkl et al., 2000). During faded instruction a subject is presented with a fully worked out example and is then presented with subsequent examples in which parts of the solution are omitted. The subject is then asked to complete these missing portions of the example, forming a hybrid between a completely worked example and a completely unsolved practice problem. Ongoing research is attempting to discover whether particular patterns of fading (such as fading steps from top to bottom or bottom to top) are more effective than others.

1.4 Using Cognitive Modeling as the Research Approach

Cognitive modeling involves the attempt to formalize hypothetical psychological processes to a fine enough level of detail that they can be implemented in a running computer program. Typically, evidence for various cognitive processes is gathered from experimental subject data. The cognitive model can then be validated by replicating the data that it is intended to explain. Further validation comes from replicating data that was not involved in the original design of the model. Generally the best type of validation occurs when the model is demonstrated to replicate entirely different classes of behavior

from those for which the model was originally developed. Success in such an endeavor suggests that the underlying mechanisms built into the model are general and accurate.

Cognitive modeling provides insight into psychological processes that may otherwise be difficult to attain. It allow us to examine the processes in detail and gain a deeper understanding for what the subject is thinking, not just what they are doing. As a research approach, this type of modeling helps to guide subsequent psychological research by making specific, testable predictions. Together with such predictions, a cognitive model allows a researcher to inspect data with a particular set of assumptions and possibly find tendencies within a given pool of subjects that might otherwise be overlooked.

As mentioned above, building such models requires a formal theory that has adequate detail to allow it to be executed. Executing the model on complete tasks helps ensure that the model does not leave out or gloss over significant details that may otherwise go unnoticed. Additionally, building a model makes assumptions explicit because without making them explicit the program will not be complete. Most importantly, cognitive modeling works within a fundamental paradigm for viewing intelligent behavior as information processing.

However, it should be noted that there are some assumptions that these results rely upon. Most importantly, this assumes that your models are cognitively plausible. If models can not be deemed plausible then any results which are ascertained from subsequent analysis can and should be disputed by the community at large. While complete plausibility may be unattainable for a variety of reasons, they should be accurate within the given problem domain.

1.5 Contributions of the Current Research

In understanding the work presented here, it is important to understand the groundwork laid by last year's project. Jones and Fleischman (2001) discovered and demonstrated a potential explanation for why fading improves learning. Their work focused on the fact that students generally do not self-explain examples as much as they could, so their potential learning opportunities suffer.

Because self-explaining has been shown repeatedly to improve learning, Jones and Fleischman posited that fading works because it encourages students to self-explain pieces of the example. Fading also focuses attention on small portions of the example, which may make learning easier (because it is easier for students to learn one small piece of knowledge at a time). According to this explanation, fading appears to combine the advantages of giving students completely worked problems (which guides their problem solving so they can more easily identify what they know and don't know) with the advantages of giving them unsolved practice problems (which encourages them to practice using their own knowledge instead of simply reading examples).

The recently discovered fading phenomenon provided an opportunity for explanation as well as to explore further an existing cognitive model of learning and problem solving. Thus, this work combines a number of objectives. First, it is generally useful for a cognitive model to make a clear distinction between the mechanisms it models and the background knowledge that represents the varied levels of expertise of different students. The current work provides an opportunity to apply Cascade to a new problem domain and see if that can be accomplished without changing Cascade's

underlying mechanisms. This also supports the goal of determining whether Cascade's principles extend to provide a consistent account of new psychological data. That is, does the theory extend beyond the empirical data of self-explanation that it was originally designed to explain? VanLehn and Jones (1993a, 1993b; VanLehn et al., 1991) posited that Cascade is an accurate account of human problem solving and knowledge acquisition in the area of symbolic problem-solving, but until this work it had only been demonstrated rigorously on the self-explanation effect. But if the Cascade theory is accurate, then it's underlying cognitive processes should also explain (or at least be consistent with) future research that identifies additional regularities about how students learn.

In the long run, we hope this research will provide further insight into effective teaching procedures for problem-solving skills. As has been stated previously, if the underlying theory behind Cascade is accurate then this could be used as a tool to aid in the development of teaching materials. In order to state that this sort of model can in fact obtain valuable results, one must determine whether the model explains the data it is run on as well as whether the underlying mechanisms are in fact cognitively plausible. While this work does not answer these claims fully, it does provide supporting evidence, and hopefully leads us further along the path to answering these critical questions.

The remainder of this report first provides a review of a variety of psychological regularities that have been identified in humans learning to solve symbolic problems, in Chapter 2. Chapter 3 presents Cascade, the computer model that was first developed in the early 1990s to account for some of these psychological results. Chapter 4 essentially summarizes the work I complete before the senior scholars project, using Cascade to

demonstrate a potential explanation for why fading works, as well as generating a number of predictions to be tested by subsequent psychological studies. These predictions drove new experiments that provided the data that I spent most of my time modeling for the senior scholars project. Chapters 5 and 6 describe the methods I used to model the new data with Cascade, together with the results of my modeling efforts. Chapter 7 briefly discusses further experiments we are running on human subjects, inspired by this work. Chapters 8 and 9 summarize our conclusions and provide an outline for additional related research for the near future.

2 Improving Learning in Problem Solving

2.1 *Studying Examples*

There are many ways in which instructors might present educational material to students. In the area of symbolic problem solving, a common approach is to present a series of concepts, equations, and rules, and then a set of practice problems that require the students to apply the concepts correctly. In such a presentation the student must do most of the work to figure out how to apply the given rules within the context of a problem. If the problems are complicated, students can often founder because it is difficult to learn more than one thing at a time by practice (VanLehn, 1987; Jones & VanLehn, 1992; VanLehn et al., 1991; VanLehn & Jones, 1993c). An alternative teaching method is to introduce some fully worked out examples that illustrate how to apply the target concepts to problem solutions. A fully worked out example usually contains a series of solution steps that guide the student from the problem's premises to the solution (see Figure 1). However, it is up to the student to look through the problem and ensure that they understand each and every solution step. In the best case, a student will go through each problem step and ensure that they understand it before proceeding to the next step. Additionally, they will learn how the steps fit together in sequence to solve the problem. Typically, after studying a set of fully worked examples, students are also given further practice problems to test and refine their knowledge.

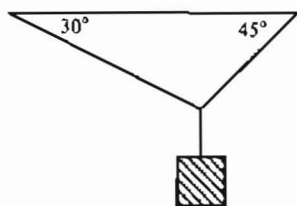


Figure a

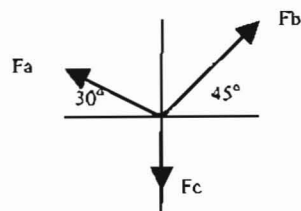


Figure b

Problem:

Figure a shows an object of weight W hung by strings. Consider the knot at the junction of the three strings to be "the body." The body remains at rest under the action of the three forces shown in figure b. Suppose we are given the magnitude of one of these forces. How can we find the magnitude of the other forces?

Solution:

F_a , F_b and F_c are all the forces acting on the body. Since the body is unaccelerated:

$$F_a + F_b + F_c = 0$$

Choosing the x - and y -axes as shown, we can write this vector equation as three scalar equations:

$$F_{ax} + F_{bx} = 0$$

$$F_{ay} + F_{by} + F_{cy} = 0$$

Using equation 5-2. The third scalar equation for the z -axis is simply:

$$F_{ax} = F_{bx} = F_{cx} = 0$$

That is, the vectors all lie in the x - y plane, so that they have no z -components. From the figure we see that:

$$F_{ax} = -F_a \cos 30^\circ = -0.866F_a$$

$$F_{ay} = F_a \sin 30^\circ = 0.500F_a$$

and:

$$F_{bx} = F_b \cos 45^\circ = 0.707F_b$$

$$F_{by} = F_b \sin 45^\circ = 0.707F_b$$

Also:

$$F_{cy} = -F_c = -W$$

Because the string C merely serves to transmit the force on one end to the junction at its other end.

Substituting these results into our original equations, we obtain:

$$-0.866F_a + 0.707F_b = 0$$

$$0.500F_a + 0.707F_b - W = 0$$

If we are given the magnitude of any one of these three forces, we can solve these equations for the other two. For example, if $W=100\text{N}$, we obtain $F_a=73.3\text{N}$ and $F_b=89.6\text{N}$

Figure 1 A physics example (taken from VanLehn, Jones, & Chi, 1991).

2.2 Self-Explaining Examples

During the learning of symbolic problem-solving skills, it has been found that *self-explanation* is one key to learning. Self-explanation occurs when a subject goes through the necessary steps in order to ensure that they understand a concept used at a given point in an example. Studies have shown that subjects who self-explain typically

score higher on subsequent examinations than those who do not (Chi et al., 1989; Pirolli & Anderson, 1985). Consequently, if educators were able to force students into situations in which they would self explain the concepts at hand, then students ought to learn more and score higher on subsequent problems.

While students can learn more effectively by thoroughly self-explaining fully worked out examples, the reality is that students do not always self explain when it would be beneficial for them to do so. Thus, studies so far demonstrate that self-explanation correlates with improved learning, but they have not determined which factors might motivate students to employ this productive strategy.

2.3 Fading Examples

Some of the most recent work on learning in problem solving (Renkl & Atkinson, 2000) examines the effect of *example fading* on learning. In example fading, students first receive a fully worked out example and then subsequently study similar problems, each with an increasing number of solution steps removed from the explanation. After several iterations of this process the students are solving the problems entirely on their own. Renkl and Atkinson (2000) found that a curriculum of gradually faded examples promotes better learning than a curriculum that consists only of fully worked examples and completely unworked practice problems.

The fading result is intriguing, because it suggests new ways to improve educational materials. However, the initial research left several questions about fading unanswered. Most notably, what cognitive processes cause fading to be effective? Additionally, is it sufficient to fade examples arbitrarily, or are some approaches to

fading more effective than others? Which parts of a given problem should be faded in order to maximize learning? Renkl & Atkinson did not provide answers to these questions, but noted discovering the answers is paramount to the development of better curricula.

3 What Cascade Is and How it Works

Cascade is a computer program originally developed in the early 90's to model the self-explanation effect and to study various learning and problem-solving strategies observed in humans (VanLehn, Jones, & Chi, 1991; VanLehn & Jones, 1993a). In modeling the self-explanation effect, the Cascade creators aimed to identify mechanisms that would explain the observed data by Chi et al. (1989)

Cascade has two basic modes of operation that share most of their mechanisms. These methods are explaining and problem solving. As mentioned previously, examples consist of a problem followed by a fully worked-out solution. Referring back to Figure 1, the solution of an example is made up of a series of lines, each of which represents a step in the solution of the problem. Cascade models example explaining as the generation of a proof. Given a set of premises (the problem description), and a set of equations (background knowledge about the task), Cascade must find a way to prove that each line in the example solution can be derived from the premises. If Cascade cannot derive a line, the assumption is that it must be missing one of the necessary pieces of background knowledge. In this case, we say the Cascade has encountered an *impasse* in problem solving. Impasses are an indication of a gap in knowledge, and are an opportunity to learn.

Generating example explanations is not always trivial, because the examples generally do not show *all* the reasoning behind each step, possibly leaving some important material and assumptions implicit. In the example shown in Figure 1, many students are confused as to why the sign of the projection of F on the x axis is negative. It is up to Cascade to explain the *why* each solution step is correct. In doing so, Cascade can

model different types of subjects based upon the way in which it derives reasons for using a particular rule.

Cascade explains the self-explanation effect by exploiting one of the effect's key correlations. The self-explanation effect notes that good learners turn out to be students who generate many self-explaining statements when studying examples. Thus, Cascade simulates a "perfectly good" learner by attempting to explain each and every line of an example. In doing so it is able to learn in two general ways: by identifying potentially missing knowledge necessary for explaining the example, and by integrating the various necessary questions together into a coherent solution (rather than a disjoint set of example lines). Both types of learning hopefully allow Cascade to apply the appropriate knowledge to subsequent problems. When simulating a "perfectly poor" learner, Cascade is programmed not to explain *any* of the example lines, but rather accept them as true and simply store them in memory. This clearly diminishes the amount of learning in a given subject; by not explaining each line, Cascade has no opportunity to encounter any impasses that may exist, meaning no learning can occur. Additionally, Cascade is not able to synthesize the goals and subgoals of the problem into a coherent, whole solution (VanLehn, Jones & Chi, 1991).

In Cascade, deriving each line involves two basic steps. The first step matches the line to equations stored in memory. During this process, Cascade matches values in the examples to variables in the equations, in order to determine which sought quantities are known and which need to be filled in. Secondly, Cascade attempts to prove recursively why each of the variables ought to have the value with which it is paired. These comparisons may be trivial (perhaps they are given by the problem statement, or have

already been solved) or may involve multiple steps that require further proof to verify the match. If Cascade is unable to prove a given quantity-value assertion, it does not initially assume that it has a knowledge gap. Rather, it assumes that it has simply made a mistake in its proof derivation (Cascade and students must generally search through a variety of potential proof paths to find one that works). So Cascade first tries to backtrack and find another way to derive the given line. Only if Cascade can find no way to explain a particular line does it assume that there is an impasse indicating a knowledge gap. At this point Cascade will fall back on a secondary area of long-term knowledge, containing common-sense knowledge, rules of thumb, or other pieces of knowledge that do not apply directly to the type of problem being solved (and so are not used when doing the initial proofs). This reflects observations in human subjects, who will use such types of knowledge to try to overcome impasses, but are (correctly) hesitant to use such knowledge normally during problem solving and example explaining. In the face of an impasse, a student will often make an educated guess that relies on such “risky” knowledge. Sometimes this yields a complete proof, and the student forms a newly understood rule about the problem domain (although sometimes these rules are correct and sometimes these rules are incorrect). Cascade uses the secondary knowledge base in a similar way. (VanLehn, Jones & Chi, 1991)

As mentioned earlier, the other activity that Cascade engages in is the solving of a given problem. For the most part, problem solving is similar to example explaining. The main difference is that, in example-explaining, Cascade knows what answers it needs to prove, and this limits the amount of search it needs to do to find a solution. Unsolved problems are harder because the answer is not known ahead of time, so there is a much

broader space of potential solution paths to try. There is also no feedback if Cascade generates an incorrect answer, meaning there are fewer opportunities to be made aware of potential impasses.

4 Cascade and Fading

4.1 Cascade Provides a Potential Explanation of the Fading Effect

In 2001 Jones & Fleischman began experiments to see whether the mechanisms originally built into Cascade (to model the self-explanation effect) might also be able to explain the new observations associated with the fading effect (Renkl et al., 2000). If the Cascade model is in fact an accurate model of the cognitive processes at hand, then it should model these processes without modification. Cascade posits that the primary source of learning arises from deliberate self-explanation of example lines (VanLehn et al., 1991; VanLehn & Jones, 1993b). Thus, if Cascade is also an accurate account of the fading effect, it predicts that fading must somehow be associated with increased self-explanation. As mentioned previously, Cascade can model ideal learning behavior by self-explaining every piece of every example solution, but real students rarely do that. Therefore, if fading has some beneficial influence on example explaining, it seems that Cascade would predict that fading techniques have the potential to yield increased knowledge acquisition in cases where students would not normally encounter and patch all existing knowledge gaps. Because Cascade already has mechanisms that account for why self-explanation is effective, our initial hypothesis in this work was that Cascade would need no additional mechanisms in order to account for the fading effect.

4.2 Initial Results

Jones and Fleischman's studies found that Cascade can indeed demonstrate improved learning when it simulates processing faded examples over processing normal examples by not self-explaining them. With simulated fading, the system was forced to self-explain only the faded portions of the examples. Our predictions were that this would lead to improved problem solving only in those cases where the faded portion of the example appeared in a place that would force Cascade to encounter a knowledge gap. We ran experiments on Cascade to verify that this was true. The knowledge analysis we performed for the experiments was also able to tell us exactly which pieces of certain examples should be faded in order to allow the system to solve each particular problem. (Jones & Fleischman, 2001)

Our study showed that Cascade can be mapped well to graded fading of examples, with the key assumption that fading corresponds to "forced" self-explanation of portions of each example. Given that assumption, but without any modifications to the Cascade model on any level, the mechanisms of fading seem clear and consistent.

These experiments also demonstrated that arbitrary fading of sections of a given example is unlikely to yield increased performance on subsequent problems. Rather, only those portions of an example that emphasize the most important knowledge chunks should be faded. When Cascade was given an example that faded only a portion that Cascade already knew how to solve well, the benefits were generally negligible and did *not* improve subsequent problem solving.

4.3 Cascade's Predictions about Fading

This early work with Cascade demonstrated that the model tells a possibly accurate story about fading that is at least internally consistent. However, showing that Cascade works a certain way does not necessarily tell us that human students also work that same way. Thus, Jones & Fleischman's (2001) report culminated in several hypotheses and predictions that required subsequent experimentation to verify (or possibly to invalidate the model, depending on the outcome of the experiments). These predictions included:

1. "Faded examples cause effective learning by forcing the student to encounter and overcome an impasse."
2. There is likely "...at least some benefit to example fading from the learning of search control knowledge."
3. "The primary benefit of a faded example is that it forces the student to process parts of the example that they might otherwise ignore."

To test these predictions, Jones and Fleischman recommended acquiring much more detailed subject data on the fading effect. Early reports of the effect provided statistical summaries of aggregate subject behavior. But the predictions made by Cascade predict different benefits to fading for different individual subjects, requiring a fine-grained analysis of each subject's behavior. The most appropriate technique for such data is *protocol analysis*, which involves forcing subjects to talk aloud while solving problems (and studying examples), transcribing what they say, and then coding the transcripts to look for different types of cognitive events. The coded events within each subject protocol can then be checked for consistency with the predictions made by Cascade.

In response to our predictions and recommendations for further experimentation, Renkl and Atkinson agreed to run talk-aloud experiments that would provide data with which we could test Cascade's predictions. The results of those experiments provided us with several transcribed protocols that we were able to model with Cascade and match against Cascade's predictions. We are also running an additional set of human subjects at Colby College, with experiments also designed around Cascade's predictions.

5 Modeling Methods

In order to analyze the new human data with Cascade, there were a number of jobs to accomplish. This chapter describes the primary activities I engaged in during the course of this year. Essentially, I had to adapt Cascade to model the specific problems that the human subjects solved, tune the Cascade model for each individual subject to accurately represent initial knowledge, code the subject protocols to look for evidence of impasses, learning, errors, and successful problem-solving, and run each individual Cascade model to match its behavior to the behavior of the subjects.

5.1 *Engineering Knowledge for Probability Problems*

Prior work with Cascade involved building a detailed knowledge base so that the system could solve the same physics problems that were given to subjects in the self-explanation experiments. Rather than using physics, the subjects in the fading experiments were solving simple probability problems. In particular, they focused on learning and using three basic formulas for probability computations: the addition rule ($P(E \text{ or } F) = P(E) + P(F) - P(E \text{ and } F)$), the subtraction rule ($P(E) = 1 - P(\text{not } E)$), and the special multiplication rule ($P(E \text{ and } F) = P(E) \times P(F)$). Thus, one of my first tasks was to replace Cascade's knowledge base with these and other equations required to solve the probability problems the subjects were presented with. In making these modifications, it is important to stress that no changes were made to the underlying model itself. However, modifications were made in a few other areas. First and foremost, the "knowledge base" file was modified to reflect knowledge required in this new area. This file contains the

equations that Cascade uses both to solve problems and self-explain examples. This also required building a formal, symbolic representation of the word problems presented to the subjects. An example appears in Figure 2. Once these pieces were in place, Cascade was able to solve all of the problems and examples given to the subjects, essentially modeling a student who already knows all of the probability principles perfectly, and does not have to do any learning to use them. We refer to this set of equations as the *target knowledge base*, because it is what we hope all students will know after learning the subject.

Wording of problem 2: Mrs. Zinfandel purchased 12 bottles of her favorite vintage red wine. Unfortunately, due to improper storage, 4 bottles have turned to vinegar and are undrinkable. What is the probability that the first bottle that Mrs. Zinfandel opens is vinegar but the second one is drinkable?

Cascade representation of problem 2:

```
scene(p2,[
  current_situation(p2),
  given_inst(p2,situation),
  given_inst(redwine1,collection),
  given_inst(event1a,event),
  eventtype(event1a,choose_no_replacement),
  eventpool(event1a,redwine1),
  eventgoal(event1a,vinegar1),
  given_inst(event1b,event),
  eventtype(event1b,choose_no_replacement),
  eventpool(event1b,redwine1),
  eventgoal(event1b,goodwine1),
  after(event1a,event1b),
  given_independent(event1a,event1b)
]).
```

```
givens(p2,[
  given(f(initial_cardinality,redwine1),12),
  given(f(initial_cardinality,vinegar1),4),
  given(f(initial_cardinality,goodwine1),8),
  given(f(number_chosen,event1a),1),
  given(f(number_chosen,event1b),1)
]).
```

```
soughts(p2,[
  f(probability,and(event1a,event1b))
]).
```

Figure 2

5.2 Representing Faded Examples

During these experiments different subjects worked with different sorts of experimental materials. In the control condition, subjects were presented with fully worked out examples and subsequently were asked to solve similar sorts of problems. In other cases subjects were first given fully worked out examples and then were given partially worked out examples and asked to complete the missing solution steps. Finally, these same subjects were given unsolved problems and asked to solve them from scratch.

Problem: Xing Computers offers a very cheap computer plus printer package in order to get well known in the US. The quality of Xing products is not, however, the very best. Right out of the box, only 80% ($p=0.80$) of the computers function properly. The printers are damaged in 20% ($p=0.20$) of the cases. In 4% ($p=0.04$) of the cases both the computer and printer malfunction. If a company orders two packages, what is the probability that both packages will be flawed in some way?

Step 1: Probability of malfunction computer: $1-0.8 = 0.2$

Step 2: Probability of malfunctioning computer and/or printer: $0.2+0.2-0.04 = 0.36$

Step 3: ?

Figure 3

Figure 3 provides an example of a faded problem. In this example the subject is expected to solve step 3 of the problem. By this point, however, they have seen at least one fully worked out example of a similar nature. If they did in fact self-explain on those examples, this should be easy to solve. However, if they did not then they will probably solve this incorrectly. After working the step for themselves, the subject is presented with the correct answer for that step of the example. This gives the subject the opportunity to reexamine their answer and attempt to correct it if necessary. During this process, the subject may reach an impasse. If the subject successfully learns and patches the knowledge gap that causes the impasse, then they should be able to solve subsequent problems that use the same (previously missing) piece of knowledge.

5.3 Knowledge as Parameters

In running these experiments an area of importance is the knowledge students have upon the start of the experiments. During the experiments of Renkl and Atkinson, the researcher always began with a pre-test to evaluate what the subject knows before being given instructional materials. After filling out this part of the examination, the subject was given instructional materials that informed them about the target probability principles. Each subject was then given some example problems (some were given faded examples, and some were given only completely worked examples) and then asked to solve some practice problems. Finally, the subject was given a post-test to determine what they had learned.

One of the underlying assumptions in this work is that all students have the same problem-solving and learning mechanisms at their disposal. This is obviously not the case in all situations, as there are a variety of factors that may affect these mechanisms (such as IQ, memory capacity, motivation and personality). However, the Cascade model assumes that, for most people, background knowledge has the largest impact on performance. Additionally, different subjects will study different portions of a given example. This work does not aim to explain why it is that subjects study different parts of a given example. Rather the model is forced to study the same portions that a subject explains (determined by analyzing the subject protocol transcript). As one might imagine, Cascade is also forced to ignore the same portions of a given example that a subject ignored during testing.

5.4 Tuning the Knowledge Base to Model Individual Subject

Behavior

As suggested above, the goal here is not just to model knowledgeable problem solvers but rather *each* of the individual solvers involved in these studies. This requires adjusting the target knowledge base so that it approximates the initial knowledge of each subject. To accomplish this, I examined the subject protocols (discussed below) to find evidence that they were missing any of the knowledge required to solve some of the problems. If I located such an equation, I removed it from that individual's Cascade model. After tuning the knowledge base, I had Cascade study the same examples and solve the same problems that the subject did. Much like the students, if Cascade were missing some knowledge, it fumbled and was unable to correctly solve some of the problems. However, when given a corrected example step from a faded example, Cascade would back up and search for a correct explanation, learning some missing knowledge if possible.

Thus, fading forced the model to address the knowledge gaps, reach impasses and resolve them. Then in subsequent examples Cascade would quickly run through the problems and solve them correctly using the newly obtained knowledge. This again demonstrates that Cascade can hypothetically reproduce the fading effect. But the question that remains is whether the types of behavior exhibited by the Cascade models (which reflect Cascade's predictions about fading) match the types of behavior exhibited by the human subjects.

6 Modeling the Data to the Predictions

In order to match Cascade's behavior and predictions to the data we received from Renkl and Atkinson, we needed to complete a thorough *protocol analysis* of the data. As mentioned previously, this involved analyzing the transcripts of each subject's behavior and categorizing sections that provided evidence of certain types of significant events. During this phase we looked for any and all episodes of learning and analyzed what prompted the subject to come to an impasse and then whether or not they were able to resolve it. More often than not, subjects in faded examples were able to successfully resolve impasses. However, subjects who did not receive faded examples did not resolve impasses as frequently.

The coding categories we created were in line with the predictions we are trying to test. We coded excerpts of the transcripts if they fell into any of the following categories:

1. Reading the example without self-explaining.
2. Self-explaining part of an example.
3. Encountering an impasse (reflecting missing knowledge).
4. Generating an incorrect answer (reflecting incorrect or missing knowledge).
5. Detecting an incorrect answer due to fading.
6. Self-explaining in response to fading.
7. Generating a correct answer.

I also used the analysis from Cascade to determine which of the probability principles are necessary to solve each problem. Correct solutions of problems suggested knowledge of the corresponding principles, and I used that to determine whether the subject had

successfully learned. For now, I have done some qualitative analysis of the data, checked for general trends and compared them to the predictions Cascade makes. More thorough statistical analysis of the data awaits the completion of more human subject data, which we are collecting right now (described below). However, I have modeled the initial data thoroughly with Cascade, and the results are promising.

6.1 Initial Results and Predictions off of this Data

From our work with this data thus far, a few things are clear. First and foremost, the data we have analyzed thus far is consistent with the predictions regarding the effects of fading on impasse resolution. We are hesitant to call it conclusive until we have been able to encode and model even more data. However, the data we have analyzed so far shows that fading techniques result in subjects reaching new impasses they would otherwise have missed, and that subjects are subsequently able to resolve those impasses (although they sometimes do not resolve the impasses, and apparently do not learn anything). Such impasse resolution is the goal in instructional materials as this is the way that learning takes place according to the self-explanation effect.

This analysis found that subjects rarely engaged in self-explanation on fully worked examples whereas they did self-explain during faded examples. This is probably due to the fact that faded examples forced the subjects to generate an answer of some form. While fading did not always ensure self-explanation (some subjects even skipped it during faded examples and simply said they were unable to generate an answer of any kind) it did, more often than not, result in self-explanations that did not occur in other scenarios.

Additionally, it was found that subjects encountered a far greater number of impasses during faded examples than during fully worked out examples. This is a logical conclusion, as one cannot reach an impasse without first self-explaining a given portion of an example. Since the data indicates that fewer self-explanations occurred, it follows that fewer impasses would be reached.

This data analysis has confirmed our hypotheses regarding the accuracy of the Cascade model. That is, we believe that Cascade is an accurate account of the learning phenomena found within these new subjects within the realm of the new predictions made by Renkl and Atkinson. This work has also found that Cascade can easily be modified to solve problems within a new problem domain, providing valuable insight into the flexibility of the model with the area of symbolic problem solving.

7 Ongoing Psychological Experimentation

To further extend this work, we are currently running a number of subjects with the Department of Psychology at Colby College. Utilizing the same materials used by Renkl and Atkinson in their work, these experiments aim to collect further data regarding fading among subjects in similar learning environments. A major focus of these new experiments is the comparison between those subjects who received faded examples versus those who did not receive such examples. With this data we can clearly test whether the subjects in the fading condition behave differently from the subjects in the control condition.

As of this writing we are nearly complete in running 48 subjects doing a full talk-aloud protocol for each subject. During these sorts of experiments a subject is tested individually while being recorded. They have been instructed to say everything they are thinking out loud and when they are quiet the experimenter prompts them for further information. After the completion of the experiment their voice is transcribed and then one can begin a detailed analysis of the transcription alongside the materials that the subject used. Additionally, we plan to run additional subjects without collecting audio data. For these subjects we will only have their written materials for analysis.

8 Conclusions

8.1 *A Review of Cascade's Predictions*

As outlined earlier, VanLehn, Jones & Chi (1991) developed Cascade in order to identify the cognitive mechanisms that contribute to the self-explanation effect. Jones & Fleischman (2001) demonstrated that, in initial testing, Cascade can use these same principles to account for the fading effects found by Renkl et al. (2000). However, this account depends on the assumption that fading is effective because it encourages students to self-explain where they otherwise might not. This is an assumption that can be tested by checking whether faded portions of examples correlate with the effects of self-explanation. These effects include increased numbers of impasses (meaning increased opportunities to learn) and increased resolution of impasses (meaning learning actually occurred). Additionally, we can tell if learning occurred during a faded example by checking whether the subject does not correctly apply a particular probability principle before working the example, but does correctly apply it after working the example. Cascade predicts that this will only happen for principles that are contained in the “proof” of the faded portion of the example, and that we will see evidence of this learning by seeing the subject encounter an impasse and successfully resolving it.

A further goal of this work was to test the assumption that Cascade can be modified to solve problems in the area of Probability, which is a switch from the problem domain in which it was originally designed to solve (Newtonian Physics). It is important that this modification occurs only by changing Cascade's knowledge base and not any of its underlying mechanisms for problem solving, example explaining, or learning.

8.2 Results

In this new data it is clear that Cascade is able to model subjects in this new problem domain. As of this writing the model has successfully been modified so as to solve probability problems. Additionally, these modifications were made with *no* changes to Cascade's core mechanisms but rather only with changes to the knowledge base that powers the model, and by creating a "Cascade-like" representation of the examples and problems that Renkl and Atkinson used in their experiments.

In terms of self-explanation, the protocol encodings show that some of Renkl and Atkinson's subjects engaged in this activity, and they usually learned when they did. This is also consistent with Cascade's predictions. Students who self explain more often, in either Probability or Newtonian Physics, reach and resolve more impasses and attain higher scores on subsequent problems.

Finally, Cascade's account of fading as "encouraged self-explanation" is borne out in the data. The subjects who were given faded examples reached and resolved more impasses than those who were not presented with faded examples. We were able to replicate this effect with individual Cascade models of each subject. In the Cascade model, faded examples resulted in greater amounts of learning and an increase in the number of impasses resolved during execution.

8.3 What does this tell us about Learning and Teaching?

In terms of learning, the benefits are clear. Psychological research has shown that fading examples yields improved learning, and Cascade fully concurs with this premise.

By utilizing this powerful model we simply have gathered evidence for *why* fading works, and bolstered the notion that fading is a teaching technique that should be used in the classroom.

Another interesting result lies in Cascade's potential ability to help in the development of educational materials. Renkl et al. (2000) believed that fading should be used in education but noted that while this is an effective technique, when not used properly the results will be non-existent. Cascade has taken this logic a step further by suggesting that fading will not be effective in cases where students may already be self-explaining and encountering impasses for other reasons. However, Cascade makes even further claims by also providing an analytical tool to determine where in the curriculum fading should be used. In examining the behavior traces generated by Cascade, one may determine where it is that a student is and is not learning a concept, and then may test the results of fading in these areas simply by making minor tweaks to the background knowledge or the examples and problems. Thus Cascade is not simply a reflection of psychological data but it can also be used as an aid in curriculum development.

What's more, Cascade may answer other questions related to learning. In the case of fading, Cascade gives us insight into why subjects learn more than in non-faded examples. This is clearly a step forward from the initial identification of the correlation between fading and improved learning. However, Cascade also addresses a question originally asked by VanLehn, Jones & Chi (1991). Their development of Cascade provided a potential explanation for why self-explanation works, but they were unable to explain how instructors might motivate students to self-explain. This most recent work with Cascade helps us understand at least one of the mechanisms that can be used to

encourage students to self-explain the most beneficial parts of an example so as to maximize their opportunities to learn. If Cascade's explanation for the effectiveness of example fading is correct, this may be the single most important result of this work.

9 Future work

The work completed so far has increased our understanding of how to improve some types of education. Certainly, there are many possible directions to go from here. This final chapter presents some of most fruitful paths to follow in the near future.

9.1 *Improving Cascade*

Although we accomplished this work without changing any of Cascade's key cognitive mechanisms, there are some aspects of human problem solving and learning that the model does not simulate or explain well. Clearly one such phenomenon was the case in which a subject would get stuck in a given example and then begin to try random combinations of equations in an effort to obtain an answer. While some effort was made to model the more general cases that subjects would try, a substantial effort was not made to model each and every attempt by the individual subjects. More effort could be made to do this, however it is not clear how useful this would be. If there are predictable regularities in how students respond to hard problems, we would certainly like to understand them. But we will eventually reach a level of detail where individual differences cannot easily be captured in a computer program.

There are also other sorts of conditions that Cascade is unable to model. For example, when a subject "slips" and makes a simple addition error in a computation, they ascertain the incorrect answer yet their logic may have been correct. Thus far Cascade is largely unable to model this phenomenon. Additionally, Cascade has no mechanisms for gradual knowledge acquisition. That is, knowledge that is acquired slowly when rehearsed over the course of multiple problems. This does not, however, invalidate the

results at hand. The data thus far indicated that analogy is more often used during the faded examples than during the study of completely worked examples.

9.2 Further Data Analysis

With the new data being collected at this time, it would be fruitful to do a more thorough data analysis. This analysis should focus on whether impasses are really statistically correlated with fading. In doing such an analysis we can be more precise about determining whether there's possibly some other form of learning going on. It is certainly within the realm of possibility that Cascade is only partially correct and given our view of the cognitive processes at hand there may be additional phenomena which we are unable to see thus far. This sort of statistical correlation tests for these sorts of conditions.

Appendix A: Outline of Cascade's Algorithm

(from VanLehn, Jones & Chi, 1991)

The Main Loop of Cascade's Rule Interpreter

In problem P, to find a value V for quantity G or to show that V is the value of quantity G, try these methods in order until once succeeds:

1. Analogical search control.

Do the following five steps in order, failing if any one fails and the failure can't be handled:

- a. Retrieve an example E that is similar to P.
If retrieval fails, then flip pages looking for an example with a diagram that is similar to P's diagram.
- b. Retrieve a mapping between E and P. If retrieval fails, then
reread problem statements of E and P, and create a mapping.
- c. Using the mapping, substitute terms in G to form a target goal T.
- d. Retrieve a triple (E T R), where R is bound by retrieval to a rule.
If retrieval fails, then
reread lines of E's solution to stimulate recall.
If rereading lines stimulates only partial recall, then redo the derivation of the line that stimulated partial recall, and retrieve a triple from the new derivation.
If rereading lines fails to stimulate recall, then redo the whole derivation, and retrieve a triple from the new derivation.
- e. Show that R's conditions are met.
- f. Apply R's equation to G and V.
- g. Create a triple (P G R).
- h. Return whatever Step f returned.

2. Regular rule selection and application.

Do the following steps in order, failing if any one fails and the failure cannot be handled:

- a. Retrieve a domain rule (or any rule if this is not pass 1) whose equation contains a quantity unifying with G and whose condition is met by the current situation. Call the rule R.
- b. Plant a backup point so that a different rule can be retrieved if R leads to failure.
- c. Apply R to G and V.
- d. If R is an overly general rule, then create a specific version of the rule by instantiating R and substituting variables for problem-specific constants. Call this new rule R and mark it as a domain rule of P's task domain.
- e. Create a triple (P G R).
- f. Return whatever Step c returned.

3. Transformational analogy.

If a problem is being solved, then do the following steps in order, failing if any one fails and the failure cannot be handled:

- a. Retrieve an example (as in Step 1a).
- b. Retrieve a map (as in step 1b).
- c. Create a target goal T via mapping G (as in step 1c).
- d. Retrieve a line of the example that contains T .
If retrieval fails, then reread each line to see if it contains T .
- e. Substitute terms in the line via the map to put it in terms of P .
- f. Apply the line's equation to G .
- g. Return whatever Step f returned.

4. Analogy abduction.

If this is the third pass, and an example I being explained and a value V for G is known, then do the following steps in order, failing if any one fails and the failure cannot be handled:

- a. Create an analogy rule R (see test), and
- b. Mark it as a domain rule of P 's task domain, and
- c. Create a triple $(P\ G\ R)$.
- d. Return Success.

5. Impasse: No rules apply to G .

If there are backup points, then resume one,
else if this is Pass 1, then start over with Pass 2,
else if this is Pass 2 and an example is being explained, then start over with Pass 3,
else fail utterly. This problem/example cannot be solved/explained.

To apply an equation E to a quantity G when the value is unknown:

1. Let S be all quantities in E except G .
2. Recurse to find the values of each quantity in S .
3. Substitute values for quantities in E .
4. Solve E for G .
5. Return the result as G 's value.

To apply an equation E to a quantity G when the value V is given:

1. Solve E for G , obtaining expression X .
2. Match X to V , obtaining a set S of quantity-value pairs.
3. Recurse to show that each quantity in S has the value with which it is paired.
4. Return success.

Appendix B: Cascade Knowledge Base for Probability

```
constraint(v(f(probability, and(E1, E2)))=v(f(probability, E1))*v(f(probability, E2)),
  p(e1, e2)=p(e1)*p(e2)) :-
  inst(E1, event),
  inst(E2, event),
  independent(E1, E2).
```

```
constraint(v(f(probability, E))=v(f(cardinality, Y, E))/v(f(cardinality, Z, E)),
  p(e1)=x/y) :-
  inst(E, event),
  eventpool(E, Z),
  eventgoal(E, Y).
```

```
constraint(v(f(cardinality, X, E))=v(f(initial_cardinality, X)),
  goalx=initial(x)) :-
  inst(E, event),
  eventgoal(E, X),
  not(after(_, E)).
```

```
constraint(v(f(cardinality, X, E))=v(f(initial_cardinality, X)),
  poolx=initial(x)) :-
  inst(E, event),
  eventpool(E, X),
  not(after(_, E)).
```

```
constraint(v(f(cardinality, X, E2))=v(f(initial_cardinality, X)),
  goalx2=initial(x)) :-
  inst(E1, event),
  inst(E2, event),
  eventgoal(E2, X),
  not(eventgoal(E1, X)),
  eventtype(E1, choose_no_replacement),
  after(E1, E2).
```

```
constraint(v(f(cardinality, X, E2))=v(f(initial_cardinality, X))-v(f(number_chosen, E1)),
  goalx=initial(x)-number_chosen(e)) :-
  inst(E1, event),
  inst(E2, event),
  eventgoal(E1, X),
  eventgoal(E2, X),
  eventtype(E1, choose_no_replacement),
  after(E1, E2).
```

```
constraint(v(f(cardinality, X, E2))=v(f(initial_cardinality, X))-v(f(number_chosen, E1)),
```



```

poolx=initial(x)-number_chosen(e)) :-
    inst(E1,event),
    inst(E2,event),
    eventpool(E1,X),
    eventpool(E2,X),
    eventtype(E1,choose_no_replacement),
    after(E1,E2).

constraint(v(f(cardinality,X,E2))=v(f(initial_cardinality,X)),
    goalx_with_replacement=initial(x)) :-
    inst(E1,event),
    inst(E2,event),
    eventgoal(E2,X),
    eventtype(E1,choose_with_replacement),
    after(E1,E2).

constraint(v(f(cardinality,X,E2))=v(f(initial_cardinality,X)),
    poolx_with_replacement=initial(x)) :-
    inst(E1,event),
    inst(E2,event),
    eventpool(E2,X),
    eventtype(E1,choose_with_replacement),
    after(E1,E2).

/* ESF allows for: not(eventa) = 1-p(eventa) */
constraint(v(f(probability,not(E)))=1-v(f(probability,E)),
    not(e1)=1-p(e1)) :-
    inst(E,event),
    inst(not(E),event).

/* ESF This is our 'or' rule */
constraint(v(f(probability,or(E1,E2)))=v(f(probability,E1))+v(f(probability,E2))-
v(f(probability,and(E1,E2))),
    or(e1,e2)=p(e1)+p(e2)-and(e1,e2)) :-
    inst(E1,event),
    inst(E2,event),
    inst(or(E1,E2),event),
    inst(and(E1,E2),event),
    independent(E1,E2).

/* ESF This is the 'double not' rule */
constraint(v(f(probability,and(not(E1),not(E2))))=v(f(probability,not(or(E1,E2)))),
    p(and(not(e1),not(e2)))=p(not(or(e2,e2)))) :-
    inst(E1,event),
    inst(E2,event),
    inst(and(not(E1),not(E2)),event),

```

```
inst(not(or(E1,E2)),event),  
independent(E1,E2).
```

```
/* We have to hack these to just say a few combinations exist. Otherwise  
Prolog will search for all possible combinations, which gives an  
infinite recursion. RMJ 12-14-01 */
```

```
exists(not(E)) :- given_inst(E,event).  
exists(or(E1,E2)) :- given_inst(E1,event), given_inst(E2,event).  
exists(and(E1,E2)) :- given_inst(E1,event), given_inst(E2,event).  
exists(and(not(E1),not(E2))) :- given_inst(E1,event), given_inst(E2,event).  
exists(not(or(E1,E2))) :- given_inst(E1,event), given_inst(E2,event).
```

```
type(not(E),T) :- given_inst(E,T).  
type(or(E1,E2),T) :- given_inst(E1,T),given_inst(E2,T).  
type(and(E1,E2),T) :- given_inst(E1,T),given_inst(E2,T).  
type(and(not(E1),not(E2)),T) :- given_inst(E1,T),given_inst(E2,T).  
type(not(or(E1,E2)),T) :- given_inst(E1,T),given_inst(E2,T).
```

Appendix C: Probability Problems Encoded in Cascade

```
/* problems */

problem_order([common_sense,math,txt,
p1,
hack]).

/* Problem p1 */

scene(p1,[
current_situation(p1),
given_inst(p1,situation),
given_inst(ballset1,collection),
given_inst(event1a,event),
eventtype(event1a,choose_no_replacement),
eventpool(event1a,ballset1),
eventgoal(event1a,redballset1),
given_inst(event1b,event),
eventtype(event1b,choose_no_replacement),
eventpool(event1b,ballset1),
eventgoal(event1b,whiteballset1),
after(event1a,event1b),
given_independent(event1a,event1b)
]).

givens(p1,[
given(f(initial_cardinality,ballset1),5),
given(f(initial_cardinality,redballset1),3),
given(f(initial_cardinality,whiteballset1),2),
given(f(number_chosen,event1a),1),
given(f(number_chosen,event1b),1)
]).

soughts(p1,[
f(probability,and(event1a,event1b))
]).

/* Problem p2 */

scene(p2,[
current_situation(p2),
given_inst(p2,situation),
```

```

given_inst(redwine1, collection),
given_inst(event1a, event),
eventtype(event1a, choose_no_replacement),
eventpool(event1a, redwine1),
eventgoal(event1a, vinegar1),
given_inst(event1b, event),
eventtype(event1b, choose_no_replacement),
eventpool(event1b, redwine1),
eventgoal(event1b, goodwine1),
after(event1a, event1b),
given_independent(event1a, event1b)
]).

```

```

givens(p2, [
given(f(initial_cardinality, redwine1), 12),
given(f(initial_cardinality, vinegar1), 4),
given(f(initial_cardinality, goodwine1), 8),
given(f(number_chosen, event1a), 1),
given(f(number_chosen, event1b), 1)
]).

```

```

soughts(p2, [
f(probability, and(event1a, event1b))
]).

```

/* Problem p3 */

```

scene(p3, [
current_situation(p3),
given_inst(p3, situation),
given_inst(cases1, collection),
given_inst(event1a, event),
eventtype(event1a, choose_no_replacement),
eventpool(event1a, cases1),
eventgoal(event1a, whiskey1),
given_inst(event1b, event),
eventtype(event1b, choose_no_replacement),
eventpool(event1b, cases1),
eventgoal(event1b, bricks1),
after(event1a, event1b),
given_independent(event1a, event1b)
]).

```

```

givens(p3, [
given(f(initial_cardinality, cases1), 50),

```

```

given(f(initial_cardinality,whiskey1),45),
given(f(initial_cardinality,bricks1),5),
given(f(number_chosen,event1a),1),
given(f(number_chosen,event1b),1)
]).

```

```

soughts(p3,[
f(probability,and(event1a,event1b))
]).

```

/* Problem p4 */

```

scene(p4,[
current_situation(p4),
given_inst(p4,situation),
given_inst(bulbs1,collection),
given_inst(event1a,event),
eventtype(event1a,choose_no_replacement),
eventpool(event1a,bulbs1),
eventgoal(event1a,deadbulbs1),
given_inst(event1b,event),
eventtype(event1b,choose_no_replacement),
eventpool(event1b,bulbs1),
eventgoal(event1b,goodbulbs1),
after(event1a,event1b),
given_independent(event1a,event1b)
]).

```

```

givens(p4,[
given(f(initial_cardinality,bulbs1),6),
given(f(initial_cardinality,deadbulbs1),3),
given(f(initial_cardinality,goodbulbs1),3),
given(f(number_chosen,event1a),1),
given(f(number_chosen,event1b),1)
]).

```

```

soughts(p4,[
f(probability,and(event1a,event1b))
]).

```

/* Problem p5 */

```

scene(p5,[
current_situation(p5),

```

```

given_inst(p5,situation),
given_inst(shirts1,collection),
given_inst(event1a,event),
eventtype(event1a, choose_with_replacement),
eventpool(event1a, shirts1),
eventgoal(event1a, badstitching1),
given_inst(event1b,event),
eventtype(event1b, choose_with_replacement),
eventpool(event1b, shirts1),
eventgoal(event1b, badcolor1),
after(event1a, event1b),
given_independent(event1a, event1b)
)).

```

```

givens(p5, [
given(f(initial_cardinality,shirts1),100),
given(f(initial_cardinality,badstitching1),10),
given(f(initial_cardinality,badcolor1),20),
given(f(number_chosen,event1a),1),
given(f(number_chosen,event1b),1)
]).

```

```

soughts(p5,[
f(probability,not(or(event1a,event1b)))
]).

```

/* Problem p6 */

```

scene(p6,[
current_situation(p6),
given_inst(p6,situation),
given_inst(pictures1,collection),
given_inst(event1a,event),
eventtype(event1a, choose_with_replacement),
eventpool(event1a, pictures1),
eventgoal(event1a, blur1),
given_inst(event1b,event),
eventtype(event1b, choose_with_replacement),
eventpool(event1b, pictures1),
eventgoal(event1b, dark1),
after(event1a, event1b),
given_independent(event1a, event1b)
]).

```

```

givens(p6, [

```

```

given(f(initial_cardinality,pictures1),100),
given(f(initial_cardinality,blur1),40),
given(f(initial_cardinality,dark1),10),
given(f(number_chosen,event1a),1),
given(f(number_chosen,event1b),1)
)).

```

```

soughts(p6,[
f(probability,not(or(event1a,event1b)))
]).

```

/* Problem p7 */

```

scene(p7,[
current_situation(p7),
given_inst(p7,situation),
given_inst(computers1,collection),
given_inst(event1a,event),
eventtype(event1a,choose_with_replacement),
eventpool(event1a,computers1),
eventgoal(event1a,cpul),
given_inst(event1b,event),
eventtype(event1b,choose_with_replacement),
eventpool(event1b,computers1),
eventgoal(event1b,printer1),
after(event1a,event1b),
given_independent(event1a,event1b)
]).

```

```

givens(p7,[
given(f(initial_cardinality,computers1),100),
given(f(initial_cardinality,cpul),20),
given(f(initial_cardinality,printer1),20),
given(f(number_chosen,event1a),1),
given(f(number_chosen,event1b),1)
]).

```

```

soughts(p7,[
f(probability,not(or(event1a,event1b)))
]).

```

/* Problem p8 */

```

scene(p8,[

```

```

current_situation(p8),
given_inst(p8,situation),
given_inst(device1,collection),
given_inst(event1a,event),
eventtype(event1a, choose_with_replacement),
eventpool(event1a, device1),
eventgoal(event1a, x1),
given_inst(event1b,event),
eventtype(event1b, choose_with_replacement),
eventpool(event1b, device1),
eventgoal(event1b, y1),
after(event1a, event1b),
given_independent(event1a, event1b)
]).

```

```

givens(p8, [
given(f(initial_cardinality,device1),100),
given(f(initial_cardinality,x1),10),
given(f(initial_cardinality,y1),10),
given(f(number_chosen,event1a),1),
given(f(number_chosen,event1b),1)
]).

```

```

soughts(p8,[
f(probability,not(or(event1a,event1b)))
]).

```


Appendix D: Sample Protocol Encodings

Participant 1005 Example-Pair
Condition

Vp liest Problem 1: ballot box
Schritt 1: Vp liest vor
Schritt 2: (Vp überlegt)... okay.
Schritt 3: Vp liest vor

Vp liest Zinfandel Beispiel.
Schritt 1: liest vor
Schritt 2: liest vor
Schritt 3: please enter the appropriate
answer. Boy...summsumm...I don't
know this right now.
Intervention VL (Vp bekommt
Taschenrechner, Hinweis, dass Vp
Antwort auf Papier schreiben soll).
4/12, 8/11, 0.24, alright.
(vergleicht evtl. die Antwort, spricht
aber nicht)

Sought: $p(\text{and}(\text{event1a}, \text{event1b}))$
S: $\text{value}(p(\text{and}(\text{event1a}, \text{event1b})))$
S: $\text{solve}(p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b}))$
S: $\text{value}(p(\text{event1a}))$
S: $\text{solve}(p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool}))$
S: $\text{value}(\text{size}(\text{selectionpool}))$
F: $\text{value}(\text{size}(\text{selectionpool})) = 4$
S: $\text{value}(\text{size}(\text{totalpool}))$
F: $\text{value}(\text{size}(\text{totalpool})) = 12$
F: $\text{solve}(p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool})): 1/3$
F: $\text{value}(p(\text{event1a})) = 1/3$

S: $\text{value}(p(\text{event1b}))$
S: $\text{solve}(p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool}))$
S: $\text{value}(\text{size}(\text{selectionpool}))$
F: $\text{value}(\text{size}(\text{selectionpool})) = 8$
S: $\text{value}(\text{size}(\text{totalpool}))$
S: $\text{solve}(\text{size}(\text{totalpool}) = \text{initial}(\text{totalpool}) - \# \text{ previous pulls})$
S: $\text{value}(\text{initial}(\text{totalpool}))$
F: $\text{value}(\text{initial}(\text{totalpool})) = 12$
S: $\text{value}(\# \text{ previous pulls})$
F: $\text{value}(\# \text{ previous pulls}) = 1$
F: $\text{solve}(\text{size}(\text{totalpool}) = \text{initial}(\text{totalpool}) - \# \text{ previous pulls}): 11$
F: $\text{value}(\text{size}(\text{totalpool})) = 11$
F: $\text{solve}(p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool})): 8/11$
F: $\text{value}(p(\text{event1b})) = 8/11$
F: $\text{solve}(p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b})): 8/33$
F: $\text{value}(p(\text{and}(\text{event1a}, \text{event1b}))) = 8/33$

VP liest Untrustworthy Beispiel.

VP liest ersten Schritt vor: Total number of cases is 50, for whisky is 45, so 9/10.

Schritt 2: Found whisky in the first case and bricks in the second.

Total number of cases containing bricks is 5/50, or 1/10. So, 0.1, getting closer, ...whisky in the first and bricks in the second... (Vi: keep talking) OK, I am just trying to figure out how to do this, you just multiply this, $0.9 \cdot 0.1$ is 0.09. (Vp liest die Antwort am Computer und freut sich, obwohl ihre Lsg eigentlich gar nicht stimmt)

Vp liest Problem 4: Mrs. Dark okay. First replace the original defective dining room bulb, that will be 3/6. (schreibt auf Papier). 3/6 defective. So 1/2. Enter. Thank you. Okay. Now second part. Before the replacement with the functioning one, there are 5 left, and 2 out of 5, ..., no, 3/5 again (radert und schreibt neu), (which is) .6. (kontrolliert am PC) yeah. We got .5 times .6, .3, and ohh, here I have the probability, and this is a

Sought: $p(\text{and}(\text{event1a}, \text{event1b}))$

S: $\text{value}(p(\text{and}(\text{event1a}, \text{event1b})))$

S: $\text{solve}(p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b}))$

S: $\text{value}(p(\text{event1a}))$

S: $\text{solve}(p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool}))$

S: $\text{value}(\text{size}(\text{selectionpool}))$

F: $\text{value}(\text{size}(\text{selectionpool})) = 45$

S: $\text{value}(\text{size}(\text{totalpool}))$

F: $\text{value}(\text{size}(\text{totalpool})) = 50$

F: $\text{solve}(p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool})): 45/50$

F: $\text{value}(p(\text{event1a})) = 45/50$

S: $\text{value}(p(\text{event1b}))$

S: $\text{solve}(p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool}))$

S: $\text{value}(\text{size}(\text{selectionpool}))$

F: $\text{value}(\text{size}(\text{selectionpool})) = 5$

S: $\text{value}(\text{size}(\text{totalpool}))$

S: $\text{solve}(\text{size}(\text{totalpool}) = \text{initial}(\text{totalpool}) - \# \text{ previous pulls})$

S: $\text{value}(\text{initial}(\text{totalpool}))$

F: $\text{value}(\text{initial}(\text{totalpool})) = 50$

S: $\text{value}(\# \text{ previous pulls})$

F: $\text{value}(\# \text{ previous pulls}) = 1$

F: $\text{solve}(\text{size}(\text{totalpool}) = \text{initial}(\text{totalpool}) - \# \text{ previous pulls}): 49$

F: $\text{value}(\text{size}(\text{totalpool})) = 49$

F: $\text{solve}(p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool})): 5/49$

F: $\text{value}(p(\text{event1b})) = 5/49$

F: $\text{solve}(p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b})): 9/98$

F: $\text{value}(p(\text{and}(\text{event1a}, \text{event1b}))) = 9/98$

Sought: $p(\text{and}(\text{event1a}, \text{event1b}))$

S: $\text{value}(p(\text{and}(\text{event1a}, \text{event1b})))$

S: $\text{solve}(p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b}))$

S: $\text{value}(p(\text{event1a}))$

S: $\text{solve}(p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool}))$

S: $\text{value}(\text{size}(\text{selectionpool}))$

F: $\text{value}(\text{size}(\text{selectionpool})) = 3$

S: $\text{value}(\text{size}(\text{totalpool}))$

F: $\text{value}(\text{size}(\text{totalpool})) = 6$

F: $\text{solve}(p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool})): 3/6$

F: $\text{value}(p(\text{event1a})) = 3/6$

small one, okay.

S: value($p(\text{event1b})$)
S: solve($p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool})$)
S: value($\text{size}(\text{selectionpool})$)
F: value($\text{size}(\text{selectionpool}) = 3$)
S: value($\text{size}(\text{totalpool})$)
S: solve($\text{size}(\text{totalpool}) = \text{initial}(\text{totalpool}) - \# \text{ previous pulls}$)
S: value($\text{initial}(\text{totalpool})$)
F: value($\text{initial}(\text{totalpool}) = 6$)
S: value($\# \text{ previous pulls}$)
F: value($\# \text{ previous pulls} = 1$)
F: solve($\text{size}(\text{totalpool}) = \text{initial}(\text{totalpool}) - \# \text{ previous pulls}$): 5
F: value($\text{size}(\text{totalpool}) = 5$)
F: solve($p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool})$): 3/5
F: value($p(\text{event1b}) = 3/5$)
F: solve($p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b})$): 3/10
F: value($p(\text{and}(\text{event1a}, \text{event1b})) = 3/10$)

Vp liest Problem 5: T-Shirt

Schritt 1: liest vor

Schritt 2: liest ersten Term,

wiederholt: probability of stitching and / or color defects occurring, probability of $1/10 + 1/5 = 120$, I don't understand this, of stitching and / or color defects occurring, okay, that's coming no later, it's an incredibly, how did they get this problem, -1/50, oh, okay, okay. Gotcha.

Schritt 3: liest vor: probability of selecting shirts correctly made shirt, okay, that is a trap, that's it, gochta.

Vp liest Problem 6: camera

Schritt 1: liest vor

Schritt 2: liest ersten Term vor, kurze Pause, okay, clicking, dang it, $2/5 + 1/10 = 2/50$, probability of blurring, äh, it changes 1 and you don't need the flaws, probability of blurring the image and / or forgetting the flash, I have no idea, you really chose one challenge me, 10 percent, too dark,

VI: you read louder

Oh, sorry. He manages to blur the

Sought: $p(\text{not}(\text{or}(\text{event1a}, \text{event1b})))$
S: value($p(\text{not}(\text{or}(\text{event1a}, \text{event1b})))$)
S: solve($p(\text{not}(\text{or}(\text{event1a}, \text{event1b}))) = 1 - p(\text{or}(\text{event1a}, \text{event1b}))$)
S: value($p(\text{or}(\text{event1a}, \text{event1b}))$)
S: solve($p(\text{or}(\text{event1a}, \text{event1b})) = p(\text{event1a}) + p(\text{event1b}) - p(\text{and}(\text{event1a}, \text{event1b}))$)
S: value($p(\text{event1a})$)
F: value($p(\text{event1a}) = 2/5$)
S: value($p(\text{event1b})$)
F: value($p(\text{event1b}) = 1/10$)
S: value($p(\text{and}(\text{event1a}, \text{event1b}))$)
S: solve($p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b})$)

image in 40 % of his photos, $2/5$, and he forgets to activate the flash in $1/10$ so that the pictures end up too dark. If you randomly chose one of Jonathan's developed pictures, what is the probability that it will be flawless?

Okay, that is both of them, that is one of those, so multiply them together and we have both of them, *schreibt was und radiert es wieder weg*, it is worth a try, oh not supposed to erase, oh, this ain't right.04 Times .46 ... oh that's way off, oh my gosh, I am such a dumb butt

Vp liest Problem 7: Xing Computers

$1/5$ out of $1/4$ is, okay so it means you take $1/5 + 1/5 -$ pause 4 okay $- 1/25$ okay, $1/5$ plus $1/5$ minus $4/25$, okay what do you ..., for what you can say, $.2 + .2 - .04$... worst develop is .36 alright, okay, now, probability of defective computer and / or defective printer just to receive a well-functioning package $1 - .36$, .64 gives a chance of a good computer or whatever you call it, alright moving along

S: value(p(event1a))
 F: value(p(event1a)) = $2/5$
 S: value(p(event1b))
 F: value(p(event1b)) = $1/10$
 F: solve(p(and(event1a,event1b))=p(event1a)*p(event1b)): $2/50$
 F: value(p(and(event1a,event1b))) = $2/50$
 F: solve(p(or(event1a,event1b)) = p(event1a) + p(event1b) - (p(and(event1a,event1b)))) : $23/50$
 F: value(p(or(event1a,event1b))) = $23/50$
 S: solve(p(or(event1a,event1b))*((p(event1a))*p(event1b))))
 S: solve(p(or(event1a,event1b)))
 F: value(p(or(event1a,event1b))) = $23/50$
 S: solve((p(event1a))*p(event1b)))
 S: solve(p(event1a))
 F: value(p(event1a)) = $2/5$
 S: solve(p(event1b))
 F: value(p(event1b)) = $1/10$
 F: value((p(event1a))*p(event1b))) = $2/50$
 F: value(p(or(event1a,event1b))*((p(event1a))*p(event1b)))) = $2/50$
 $* 23/50 = 23/1250$

Sought: p(not(or(event1a,event1b)))
 S: value(p(not(or(event1a,event1b))))
 S: solve(p(not(or(event1a,event1b))) = $1 - p(or(event1a,event1b))$)
 S: value(p(or(event1a,event1b)))
 S: solve(p(or(event1a,event1b)) = p(event1a) + p(event1b) - p(and(event1a,event1b)))
 S: value(p(event1a))
 F: value(p(event1a)) = $1/5$
 S: value(p(event1b))
 F: value(p(event1b)) = $1/5$
 S: value(p(and(event1a,event1b)))
 S: solve(p(and(event1a,event1b))=p(event1a)*p(event1b))
 S: value(p(event1a))
 F: value(p(event1a)) = $1/5$
 S: value(p(event1b))
 F: value(p(event1b)) = $1/5$
 F: solve(p(and(event1a,event1b))=p(event1a)*p(event1b)): $1/25$
 F: value(p(and(event1a,event1b))) = $1/25$
 F: solve(p(or(event1a,event1b)) = p(event1a) + p(event1b) - (p(and(event1a,event1b)))) : $9/25$
 F: value(p(or(event1a,event1b))) = $9/25$

Vp liest Problem 8: 2 components
 oh goodness. A problem in the production of both components leads to one being defective in 10% of the cases. What is the probability that a randomly chosen. Two components of an electronic device are produced indepently of each other. 1/10 . Two components are produced indepently. 1/10 times 1/10, .1 times .1, okay .01 what are you saying (*redet mit Taschenrechner*) okay now you take 1/10 and add that 1/10 to it and then you subtract both which is 1 /100 which would make the decimals, this is easier, .1 + .1 - .01 because .01 ... which is the answer to that one .19 enter.
 The next one is 1 - .19 to get a well functioning one and that is .81. And that's all. Next problem.
 You have just completed...

F: solve($p(\text{not}(\text{or}(\text{event1a}, \text{event1b}))) = 1 - p(\text{or}(\text{event1a}, \text{event1b}))$) : 16/25
 F: value $p(\text{not}(\text{or}(\text{event1a}, \text{event1b})))$: 16/25

Sought: $p(\text{not}(\text{or}(\text{event1a}, \text{event1b})))$
 S: value($p(\text{not}(\text{or}(\text{event1a}, \text{event1b})))$)
 S: solve($p(\text{not}(\text{or}(\text{event1a}, \text{event1b}))) = 1 - p(\text{or}(\text{event1a}, \text{event1b}))$)
 S: value($p(\text{or}(\text{event1a}, \text{event1b}))$)
 S: solve($p(\text{or}(\text{event1a}, \text{event1b})) = p(\text{event1a}) + p(\text{event1b}) - p(\text{and}(\text{event1a}, \text{event1b}))$)
 S: value($p(\text{event1a})$)
 F: value($p(\text{event1a})$) = 1/10
 S: value($p(\text{event1b})$)
 F: value($p(\text{event1b})$) = 1/10
 S: value($p(\text{and}(\text{event1a}, \text{event1b}))$)
 S: solve($p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b})$)
 S: value($p(\text{event1a})$)
 F: value($p(\text{event1a})$) = 1/10
 S: value($p(\text{event1b})$)
 F: value($p(\text{event1b})$) = 1/10
 F: solve($p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b})$): 1/100
 F: value($p(\text{and}(\text{event1a}, \text{event1b}))$) = 1/100
 F: solve($p(\text{or}(\text{event1a}, \text{event1b})) = p(\text{event1a}) + p(\text{event1b}) - p(\text{and}(\text{event1a}, \text{event1b}))$) : 19/100
 F: value($p(\text{or}(\text{event1a}, \text{event1b}))$) = 19/100
 F: solve($p(\text{not}(\text{or}(\text{event1a}, \text{event1b}))) = 1 - p(\text{or}(\text{event1a}, \text{event1b}))$) : 81/100
 F: value $p(\text{not}(\text{or}(\text{event1a}, \text{event1b})))$: 81/100

Participant 2003 Example-Pair Condition

Problem 2

So I have a total of 12 bottles and there are 4 that are turned into vinegar. So a total of 4 vinegar and 8 drinkable. Probability of vinegar is 1/3 and drinkable is 2/3. Now if we take one then we're left with... There is a 1 in 3 chance that that will be vinegar...

Sought: $p(\text{and}(\text{event1a}, \text{event1b}))$
 S: value($p(\text{and}(\text{event1a}, \text{event1b}))$)
 S: solve($p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b})$)
 S: value($p(\text{event1a})$)
 S: solve($p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool})$)
 S: value($\text{size}(\text{selectionpool})$)
 F: value($\text{size}(\text{selectionpool})$) = 4
 S: value($\text{size}(\text{totalpool})$)
 F: value($\text{size}(\text{totalpool})$) = 12
 F: solve($p(\text{event}) = \text{size}(\text{selectionpool}) / \text{size}(\text{totalpool})$): 1/3

since there are 11 left there's 3 that are vinegar and 8 that are drinkable. The chance of it being drinkable is 8 to 11 so the probability of her drinking...probability that the first bottle is vinegar but the second is drinkable...2 red balls and 2 white balls is 4, probability is 1/2 so if we multiply 1/3 times 8 over 11 that will be 8 over 33 so that will be the probability, 8 divided by 33 is .24. (Participant keys in answers) One is .33, the other .73, the other is .24 so I missed the second section by .01.

Problem 4

So we have 6 bulbs , 3 defective so there is a 3 over six probability of them being defective, meaning $\frac{1}{2}$.

The second solution step is that we have 5 left and we have the choice of 2 that are defective and 3 that are functioning so there is a probability of 3 over 5 that equals .6 of the overall probability of getting first a defective bulb and replacing it with a functioning one is $\frac{1}{2}$ times $\frac{3}{5}$ that's .5 times .6 that's a .3 or $\frac{3}{10}$ probability. (Participant keys in

F: value(p(event1a)) = 1/3

S: value(p(event1b))

S: solve(p(event) = size(selectionpool) / size(totalpool))

S: value(size(selectionpool))

F: value(size(selectionpool)) = 8

S: value(size(totalpool))

S: solve(size(totalpool) = initial(totalpool) - # previous pulls)

S: value(initial(totalpool))

F: value(initial(totalpool)) = 12

S: value(# previous pulls)

F: value(# previous pulls) = 1

F: solve(size(totalpool) = initial(totalpool) - # previous pulls): 11

F: value(size(totalpool)) = 11

F: solve(p(event) = size(selectionpool) / size(totalpool)): 8/11

F: value(p(event1b)) = 8/11

F: solve(p(and(event1a,event1b))=p(event1a)*p(event1b)): 8/33

F: value(p(and(event1a,event1b)) = 8/33

Sought: p(and(event1a,event1b))

S: value(p(and(event1a,event1b)))

S: solve(p(and(event1a,event1b))=p(event1a)*p(event1b))

S: value(p(event1a))

S:solve(p(event) = size(selectionpool) / size(totalpool))

S: value(size(selectionpool))

F: value(size(selectionpool)) = 3

S: value(size(totalpool))

F: value(size(totalpool)) = 6

F:solve(p(event) = size(selectionpool) / size(totalpool)): 3/6

F: value(p(event1a)) = 3/6

S: value(p(event1b))

S: solve(p(event) = size(selectionpool) / size(totalpool))

S: value(size(selectionpool))

F: value(size(selectionpool)) = 3

S: value(size(totalpool))

S: solve(size(totalpool) = initial(totalpool) - # previous pulls)

S: value(initial(totalpool))

F: value(initial(totalpool)) = 6

S: value(# previous pulls)

F: value(# previous pulls) = 1

answers.)

F: solve(size(totalpool) = initial(totalpool) - # previous pulls): 5
 F: value(size(totalpool)) = 5
 F: solve(p(event) = size(selectionpool) / size(totalpool)): 3/5
 F: value(p(event1b)) = 3/5
 F: solve(p(and(event1a,event1b))=p(event1a)*p(event1b)): 3/10
 F: value(p(and(event1a,event1b)) = 3/10

Problem 6

First solution is 2 divided by 5 times 1 over 10...so .4 times .1 equals .04 that means there is a...now what we have is .4 plus .1 minus .04...equals .46 then 1 minus .46 equals .64 so there is a .64 probability that the pictures will be flawless. (Participant keys in answers.)

Sought: p(not(or(event1a,event1b)))
 S: value(p(not(or(event1a,event1b))))
 S: solve(p(not(or(event1a,event1b))) = 1- p(or(event1a,event1b)))
 S: value(p(or(event1a,event1b)))
 S: solve(p(or(event1a,event1b)) = p(event1a) + p(event1b) - p(and(event1a,event1b)))
 S: value(p(event1a))
 F: value(p(event1a)) = 2/5
 S: value(p(event1b))
 F: value(p(event1b)) = 1/10
 S: value(p(and(event1a,event1b)))
 S: solve(p(and(event1a,event1b))=p(event1a)*p(event1b))
 S: value(p(event1a))
 F: value(p(event1a)) = 2/5
 S: value(p(event1b))
 F: value(p(event1b)) = 1/10
 F: solve(p(and(event1a,event1b))=p(event1a)*p(event1b)): 2/50
 F: value(p(and(event1a,event1b))) = 2/50
 F: solve(p(or(event1a,event1b)) = p(event1a) + p(event1b) - p(and(event1a,event1b))) : 23/50
 F: value(p(or(event1a,event1b))) = 23/50
 F: solve(p(not(or(event1a,event1b))) = 1- p(or(event1a,event1b))) : 27/50
 F: value p(not(or(event1a,event1b))) : 27/50

Problem 8

Both components have a probability of 1 over 10 so .1 times .1 equals .01 so the second step is 1 over 10 times 1 over 10 minus 1 over 100 that's going to be 19 over 100 which equals .19 so

Sought: p(not(or(event1a,event1b)))
 S: value(p(not(or(event1a,event1b))))
 S: solve(p(not(or(event1a,event1b))) = 1- p(or(event1a,event1b)))
 S: value(p(or(event1a,event1b)))
 S: solve(p(or(event1a,event1b)) = p(event1a) + p(event1b) -

1 minus 10 over 100 will equal .81.
(Participant keys in answers.)

$p(\text{and}(\text{event1a}, \text{event1b}))$
 S: $\text{value}(p(\text{event1a}))$
 F: $\text{value}(p(\text{event1a})) = 1/10$
 S: $\text{value}(p(\text{event1b}))$
 F: $\text{value}(p(\text{event1b})) = 1/10$
 S: $\text{value}(p(\text{and}(\text{event1a}, \text{event1b})))$
 S: $\text{solve}(p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b}))$
 S: $\text{value}(p(\text{event1a}))$
 F: $\text{value}(p(\text{event1a})) = 1/10$
 S: $\text{value}(p(\text{event1b}))$
 F: $\text{value}(p(\text{event1b})) = 1/10$
 F: $\text{solve}(p(\text{and}(\text{event1a}, \text{event1b})) = p(\text{event1a}) * p(\text{event1b})) : 1/100$
 F: $\text{value}(p(\text{and}(\text{event1a}, \text{event1b}))) = 1/100$
 F: $\text{solve}(p(\text{or}(\text{event1a}, \text{event1b})) = p(\text{event1a}) + p(\text{event1b}) -$
 $(p(\text{and}(\text{event1a}, \text{event1b})))) : 19/100$
 F: $\text{value}(p(\text{or}(\text{event1a}, \text{event1b}))) = 19/100$
 F: $\text{solve}(p(\text{not}(\text{or}(\text{event1a}, \text{event1b}))) = 1 - p(\text{or}(\text{event1a}, \text{event1b}))) :$
 $81/100$
 F: $\text{value } p(\text{not}(\text{or}(\text{event1a}, \text{event1b}))) : 81/100$

Appendix E: Sample Task Analysis

Problem 1:

Givens:

Collection(ballset1)
Initial_cardinality(ballset1, 5)
Subcollection(ballset1, redballset1)
Subcollection(ballset1, whiteballset1)
Initial_cardinality(redballset1, 3)
Initial_cardinality(whiteballset1, 2)
Event(event1a)
type(event1a, choose_no_replacement)
pool(event1a, ballset1)
eventgoal(event1a, redballset1)
numberchosen(event1a, 1)
event(event1b)
type(event1b, choose_no_replacement)
pool(event1b, ballset1)
eventgoal(event1b, whiteballset1)
numberchosen(event1b, 1)
after(event1a, event1b)

Sought: $P(\text{event1a and event1b})$

Use equation $P(X \text{ and } Y) = P(X) * P(Y)$

Conditions: event(X), event(Y)

Soughts: $P(\text{event1a})$, $P(\text{event1b})$

Sought: $P(\text{event1a})$

Usage equation: $P(X) = \text{cardinality}(Y) / \text{cardinality}(Z)$

Conditions: event(X), type(X, choose_no_replacement), pool(X,Z), eventgoal(X,Y)

Soughts: $\text{cardinality}(\text{redballset1})$, $\text{cardinality}(\text{ballset1})$

Sought: $\text{cardinality}(\text{redballset1})$

Use equation: $\text{cardinality}(Y) = \text{initial_cardinality}(Y)$

Conditions: event(X), type(X, choose_no_replacement), eventgoal(X, Y),

not(after(_,X))

\Rightarrow *that line means "if there is no cardinality defined, then cardinality = initial cardinality"*

Sought: $\text{cardinality}(\text{ballset1})$

Use equation: $\text{cardinality}(Y) = \text{initial_cardinality}(Y)$

Conditions: event(X), type(X, choose_no_replacement), pool(X, Y), not(after(_,X))

Sought: $P(\text{event1b})$

Usage equation: $P(X) = \text{cardinality}(Y) / \text{cardinality}(Z)$

Conditions: event(X), type(X, choose_no_replacement), pool(X,Z), eventgoal(X,Y)

Soughts: cardinality(whiteballset1), cardinality(ballset1)
 Sought: cardinality(whiteballset1)
 Use equation: $\text{cardinality}(Y) = \text{initial_cardinality}(Y)$
 Conditions: event(X), type(X, choose_no_replacement), event(E), eventgoal(X, Y),
 after(E,X), numberchosen(E,Z)
 Sought: cardinality(ballset1)
 Use equation: $\text{cardinality}(Y) = \text{initial_cardinality}(Y)$
 Conditions: event(X), type(X, choose_no_replacement), event(E), pool(X, Y),
 after(E,X), numberchosen(E,Z)

Problem 5:

Givens:

Collection(shirts1)
 Initial_cardinality(shirts1, 100)
 Subcollection(shirts1, stitchingdefect1)
 Subcollection(shirts1, colordefect1)
 Initial_cardinality(stitchingdefect1, 10)
 Initial_cardinality(colordefect1, 20)
 Event(event1a)
 type(event1a, choose_with_replacement)
 pool(event1a, stitchingdefect1)
 eventgoal(event1a, stitchingdefect1)
 numberchosen(event1a, 1)
 event(event1b)
 type(event1b, choose_with_replacement)
 pool(event1b, shirts1)
 eventgoal(event1b, colordefect1)
 numberchosen(event1b, 1)
 after(event1a, event1b)
 given_independent(event1a, event1b)

Sought: $P(\text{not}((\text{event1a}) + P(\text{event1b}) - P(\text{event1a and event1b})))$

Use equation $P(\text{not}(X)) = 1 - P(X)$

Conditions: event(X)

Soughts: $P(\text{event1a and event1b})$, $P(\text{event1a})$, $P(\text{event1b})$

Use equation $P(X \text{ and } Y) = P(X) * P(Y)$

Conditions: event(X), event(Y)

Soughts: $P(\text{event1a})$, $P(\text{event1b})$

Sought: $P(\text{event1a})$

Usage equation: $P(X) = \text{cardinality}(Y) / \text{cardinality}(Z)$

Conditions: event(X), type(X, choose_with_replacement), pool(X,Z), eventgoal(X,Y)

Soughts: cardinality(deadbulbs1), cardinality(bulbs1)

Sought: cardinality(deadbuls1)

Use equation: $\text{cardinality}(Y) = \text{initial_cardinality}(Y)$

Conditions: event(X), type(X, choose_with_replacement), eventgoal(X, Y),
not(after(_,X))

⇒ *that line means "if there is no cardinality defined, then cardinality = initial cardinality"*

Sought: cardinality(ballset1)

Use equation: $\text{cardinality}(Y) = \text{initial_cardinality}(Y)$

Conditions: event(X), type(X, choose_with_replacement), pool(X, Y), not(after(_,X))

Sought: P(event1b)

Usage equation: $P(X) = \text{cardinality}(Y) / \text{cardinality}(Z)$

Conditions: event(X), type(X, choose_with_replacement), pool(X,Z), eventgoal(X,Y)

Soughts: cardinality(goodbulbs1), cardinality(bulbs1)

Sought: cardinality(goodbulbs1)

Use equation: $\text{cardinality}(Y) = \text{initial_cardinality}(Y)$

Conditions: event(X), type(X, choose_with_replacement), event(E), eventgoal(X, Y),
after(E,X), numberchosen(E,Z)

Sought: cardinality(bulbs1)

Use equation: $\text{cardinality}(Y) = \text{initial_cardinality}(Y)$

Conditions: event(X), type(X, choose_with_replacement), event(E), pool(X, Y),
after(E,X), numberchosen(E,Z)

Appendix F: How to Model Subjects

A major portion of this project involved modifying the system to model the subjects at hand. That is, modifying the knowledge base to replicate a given subject and then comparing the way in which Cascade solves the problems to the way that subjects solve the same problems. This appendix aims to give an overview as to how one would go about modifying the knowledge base and testing to see the accuracy of the given modifications as compared to the protocol data.

First, there are several files one must be familiar with in order to model a given subject:

- 1 `cascadeb.pl` – This file is the main file which is called when the system is started. This file serves only as a way to organize calls to other important files. If you make custom versions of other files you must modify this file in order to ensure you are using the correct version of a given file.
- 2 `constraintsb.pl` – In this file you will find the list of constraints that Cascade is using. That is, the list of rules that the system knows and can then apply to solve problems. These rules are typically labeled as to what they do with standard Prolog syntax for commenting so you should be able to quickly find the rules you are searching for.
- 3 `problemsb.pl` – This file serves as the place where the problems are stored for the model. Each problem is labeled by a single comment prior to the start of the problem and then you know a given problem is over as the next problem has the header comment right below the previous problem.

- 4 `og-constraintsb.pl` – This file serves as the repository for `og-constrains`, otherwise known as *overly general rules*. This work did not focus on the manipulation of these rules, but one could modify them by making changes to this file.

It should be noted that all files end with a “b” in their names as that indicates that these files are for the probability domain. Files without a “b” (such as `problems.pl`) are part of the original Cascade application and solve problems in Newtonian Physics.

When one goes about modeling a given subject within this system, the procedure is as follows:

- 1 First one must begin with a detailed protocol analysis. Other portions of this appendix give examples of this sort of analysis, but one must first do an analysis of the protocol data so as to identify the important rules which need to be added/removed from the model.
- 2 Once the protocol data has been analyzed one can look at the `constraintsb.pl` file and make sure that all of the appropriate knowledge has been added to this file. Typically modeling data consists of little more than removing rules (IE commenting them out) but at times you may need to add incorrect rules to the knowledge base. One can do this simply by modifying these files. My suggestion would be to make a copy of `constraintsb.pl` with the subject number in it. So for subject 1005, I would call this new file `constraintsb_1005.pl`. Then make a copy of `cascadeb.pl` as `cascadeb_1005.pl`.

and modify that new version of the cascade file to use the new constraints file rather than the default constraints.

- 3 Typically changes will not need to be made to the problemsb.pl file.

However, if you wish to add new problems to the system you can do that at this point. I would use a similar setup for the problemsb.pl file renaming that you use for the constraintsb.pl file as described in step 2.

At this point you are ready to begin running data and inspecting the results.

In order to run examples and problems within Cascade there are really only two commands that you would need, `exp()` and `sexpl()`. In modeling this data I relied heavily on `exp()`, however `sexpl()` is important and should be explained:

- 1 `exp()` – The `exp` command tells the model to attempt to solve the given example. Within the parentheses you simply supply the problem you would like the model to solve. So if you wanted to solve problem `p1` the command would be `exp(p1)`. Note the period after the command. In Prolog it is required that you end each line with a period as that tells the interpreter to execute the command.
- 2 `sexpl()` – This command tells the model to self-explain a given problem. The argument for this command is the problem which you would like the model to self-explain. Once again after you close the parentheses you must add a period. So an example of one of these commands would be `sexpl(p1)`. This tells the model to self-explain problem `p1`.

In my work this year, I used `exp()` almost exclusively. However, I used this command in one of two different ways:

- As outlined above, issue the command `exp(p1)`. and it will solve problem p1.
- If you issue the command `exp(p1, 1/10)`. it will try and solve problem p1 but will not complete the solution until it obtains the answer 1/10. This mode is comparable to forcing the model to solve until it obtains the correct answer and it will learn many rules along the way.

Once you begin to issue these commands you will find that the system will be spitting out full traces of what it is trying to do. Depending upon what you are trying to model you will be search for many different sorts of things within the trace. One of the most important things you will be looking for is the case in which Cascade learns a new rule. Here is an example of one such learning event:

```
*****Learn new equation constraint(v(f(probability,not
or(_835,_848)))=v(f(number_chosen,_835))-v(f(probability,or(_835,_848))),eblc(($
e1)=v(f(number_chosen,_835))-p(e1,p6,-1)):-inst(or(_835,_848),event),inst(not
or(_835,_848),event))
```

In this example you can see how the model has been learning. Typically lines in which learning takes place start with a few stars, such as the one above (*****). If you see the stars at the beginning of a line then you know that some learning has taken place.

One other thing to look for in a trace is a line that looks like this:

```
belief_revision_ok
```


When you see that line it typically means that the model has run out of equations to attempt to use and is now going to begin trying to learn by using overly-general rules as well as by using other means. It may learn incorrect rules along the way, but it will try to learn so as to provide a correct solution.

There are a couple of other files in the Cascade directory that I created which may come in handy:

- 1 The file `correct_answersb` contains all of the correct answers for the 8 primary problems used in the probability studies.
- 2 `subject*-analysis` are a series of files that I created for each subject that I studied. Look through some for an example of an analysis of the modeling of a given subject.

At this point you have an overview as to what things mean within the system. Here is an example of how to start the system, ask a subject to solve an example with a provided solution and then solve a problem. This starts from the command shell, assuming your current directory is the directory in which Cascade resides:

- 1 `gprolog` – starts the Prolog environment
- 2 `[cascadeb].` – Loads the cascadeb environment from the file `cascadeb.pl`
- 3 `[problemsb].` – Loads the problems from the file `problemsb.pl`
- 4 `exp(p6, [1-(40/100+10/100-40/100*(10/100))]).` – Ask the system to solve problem p6 and get an answer of `[1-(40/100+10/100-40/100*(10/100))].`
- 5 `exp(p7).` – solve the problem p7 with no specified correct solution.

References

- Chi, M. T. H., Bassok, M., Lewis, M. W., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science*, 13, 145–182.
- Ferguson-Hessler, M. G. M., & de Jong, T. (1990). Studying physics texts: Differences in study processes between good and poor solvers. *Cognition and Instruction*, 7, 41–54.
- Jones, R. M., & Fleischman, E. S. (2001). Cascade explains and informs the utility of fading examples to problems. *Proceedings of the Twenty-Third Annual Conference of the Cognitive Science Society* (pp.459–464). Mahwah, NJ: Lawrence Erlbaum.
- Jones, R. M., & VanLehn, K. (1992). A fine-grained model of skill acquisition: Fitting Cascade to individual subjects. *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society* (pp. 873–878). Hillsdale, NJ: Lawrence Erlbaum.
- Pirolli, P., & Anderson, J. R. (1985). The role of learning from examples in the acquisition of recursive programming skills. *Canadian Journal of Psychology*, 39, 240–272.
- Pirolli, P., & Bielaczyc, K. (1989). Empirical analyses of self-explanation and transfer in learning to program. In G. M. Olson & E. E. Smith (Eds.), *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society* (pp. 450–457). Hillsdale, NJ: Lawrence Erlbaum.
- Renkl, A. (1997). Learning from worked-out examples: A study on individual differences. *Cognitive Science*, 21, 1–29.
- Renkl, A., Atkinson, R. K., & Maier, U. H. (2000). From studying examples to solving problems: Fading worked-out solution steps helps learning. In L. R. Gleitman & A. K. Joshi (Eds.), *Proceedings of the Twenty-Second Annual Conference of the Cognitive Science Society* (pp. 393–398). Mahwah, NJ: Lawrence Erlbaum.
- VanLehn, K. (1987). Learning one subprocedure per lesson. *Artificial Intelligence*, 31(1), 1–40.
- VanLehn, K. (1996). Cognitive skill acquisition. *Annual Review of Psychology*, 47, 513–539.
- VanLehn, K., & Jones, R. M. (1993a). Learning by explaining examples to oneself: A computational model. In S. Chipman & A. L. Meyrowitz (Eds.), *Foundations of*

knowledge acquisition: Cognitive models of complex learning. Boston: Kluwer Academic.

VanLehn, K., & Jones, R. M. (1993b). Integration of explanation-based learning of correctness and analogical search control. In S. Minton (Ed.), *Machine learning methods for planning*. Los Altos, CA: Morgan Kaufmann.

VanLehn, K., & Jones, R. M. (1993c). Better learners use analogical problem solving sparingly. *Machine Learning: Proceedings of the Tenth International Conference* (pp. 338–345). San Mateo, CA: Morgan Kaufmann.

VanLehn, K., & Jones, R. M. (1993d). What mediates the self-explanation effect? Knowledge gaps, schemas or analogies? *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society* (pp. 1034-1039). Hillsdale, NJ: Lawrence Erlbaum.

VanLehn, K., Jones, R. M., & Chi, M. T. H. (1991). A model of the self-explanation effect. *Journal of the Learning Sciences*, 2, 1–59.