ERIK ROLF WEITNAUER

INTERACTIONS BETWEEN PERCEPTION AND RULE-CONSTRUCTION IN HUMAN AND MACHINE CONCEPT LEARNING

INTERACTIONS BETWEEN PERCEPTION AND RULE-CONSTRUCTION IN HUMAN AND MACHINE CONCEPT LEARNING

ERIK ROLF WEITNAUER

Vorgelegt zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften

Technische Fakultät, Universität Bielefeld

Oct 2015

Erik Rolf Weitnauer: *Interactions between Perception and Rule-Construction in Human and Machine Concept Learning*, Oct 2015

SUPERVISORS:
Prof. Helge Ritter
Prof. Robert Goldstone

LOCATION:
Bielefeld

TIME:
Oct 2015

Gedruckt auf alterungsbeständigem Papier (ISO 9706)

ABSTRACT

Concepts are central to human cognition and one important type of concepts can be represented naturally with symbolic rules. The learning of such rule-based concepts from examples relies both on a process of perception, which extracts information from the presented examples, and a process of concept construction, which leads to a rule that matches the given examples and can be applied to categorize new ones. This thesis introduces PATHS, a novel cognitive process model that learns structured, rule-based concepts and takes the active and explorative nature of perception into account. In contrast to existing models, the PATHS model tightly integrates perception and rule construction. The model is applied to a challenging problem domain, the physical Bongard problems, and its performance under different learning conditions is analyzed and compared to that of human solvers.

RELATED PUBLICATIONS BY THE AUTHOR

Samuel John, Erik Weitnauer, and Hendrik Koesling. Entropy-based correction of eye tracking data for static scenes. In *Proceedings of the symposium on eye tracking research and applications*, pages 297–300. ACM, 2012.

E. Weitnauer and H. Ritter. Physical bongard problems. *Artificial Intelligence Applications and Innovations*, pages 157–163, 2012.

Erik Weitnauer, Robert Haschke, and Helge Ritter. Evaluating a physics engine as an ingredient for physical reasoning. In *Simulation, Modeling, and Programming for Autonomous Robots*, pages 144–155. Springer, 2010.

Erik Weitnauer, Paulo F Carvalho, Robert L Goldstone, and Helge Ritter. Grouping by similarity helps concept learning. In M. Knauff, M. Pauen, N. Sebanz, and I. Wachsmuth, editors, *35th Annual Conference of the Cognitive Science Society*, 2013.

Erik Weitnauer, Paulo F Carvalho, Robert L Goldstone, and Helge Ritter. Similarity-based ordering of instances for efficient concept learning. In *36th Annual Conference of the Cognitive Science Society*, pages 1760–1765, Quebec City, Canada, 2014. Cognitive Science Society.

Erik Weitnauer, David H Landy, Robert L Goldstone, and Helge Ritter. A computational model for learning structured concepts from physical scenes. In *37th Annual Conference of the Cognitive Science Society*, pages 2631–2636. Cognitive Science Society, 2015.

ACKNOWLEDGMENTS

In 2010, on a research retreat with Helge Ritter's research group from Bielefeld University, the topic of Bongard problems and a clever model that solved some of them, written by Harry Foundalis – a graduate student of Douglas Hofstadter, came up during one of our dinner conversations. This was at a time when I had been searching for a PhD topic for many months, and I was ready to jump into the task of exploring this fascinating problem domain. Just a few weeks later, Helge and I decided that my dissertation topic would center on a computer program solving physical Bongard problems – a new flavor of Bongard problems which I had begun to carve out.

The initial work on my thesis was very playful: designing PBPs, showing them to anyone interested or not-so-interested, and talking about them. I also read more of Douglas Hofstadter's work, getting sucked into his fascinating fluid analogy framework. My natural inclination towards programming combined with Hofstadter's limited interest in relating his work to the rest of the field, other than in stark contrasts, were an uncanny combination. An earlier solid literature review and discussions with fellow researchers in my area would certainly have saved me time. Instead, I slipped into treating the dissertation as a solitary programming project for a while, following the elusive light of fluid analogies.

A turning point for my dissertation was a three month stay in Bloomington, Indiana, at the lab of Robert Goldstone. I ended up extending my stay to over two years, for which Rob and David Landy are responsible. Rob, together with Paulo Carvalho, a fellow graduate student, helped me to situate my work in existing research, discover applications in cognitive science, and contributed to studies I ran with human participants solving PBPs. I finally started scientific collaborations and conversations about my dissertation work.

I have come a long way since the beginning of this journey and have learned, though certainly not by following a straight path, much about what it means to be a researcher and to do research. Here are some of the people I want to thank for accompanying me on my journey.

First of all I want to thank Helge Ritter, my main supervisor. I both loved and feared his never-ending stream of deep and fascinating new ideas (depending on how much time I had left for a particular aspect of my work). Thank you for creating a great space for research in Bielefeld, for always finding time for me, for connecting me with the people at Indiana University, and for your ongoing support!

I also want to thank Robert Goldstone for being so open-minded and open-hearted, for breaking his maybe-not-so-strict-after-all rule of not hiring postdocs that don't have their doctoral degree yet, for all the practical advice

on how to apply my work to fascinating questions in cognitive science, and for always pushing me gently towards finishing my dissertation. I would probably be far from done right now without you.

The person that I had the pleasure to work with the closest in the last two years here in Bloomington is David Landy. Thank you so much, David, for all your support! It is done, and I am finally able to focus entirely on the amazing new things we are working on.

One of the things that helps to keep spending years and years on a dissertation enjoyable are great colleagues – and I had the fortune to have quite a few of them, both in Bielefeld and in Bloomington. Christof, René, Alex, Steffi, Jan, Ulf, Jonathan, Paulo, Josh, Seth, David BW, David B, Christian – I'll always remember these times fondly!

While I was still in Germany my parents, siblings and grandmas would get to see me at least once or twice a year, but after going to the U.S. and declaring that the next time I would come back would be for my disputation, they probably were wondering whether they had lost me to science for good. Thank you for pretending to believe in my always moving deadlines! I'll actually be back to visit now!

It was my two sons, Nils and Sven, that made it easier for me to put the importance of my dissertation in relation to what is truly valuable in life. They also hugely motivated me to come home at reasonable times so that I could enjoy playing with them. Finally, I want to thank you, Xinrui, for being so strong and for working so hard to provide me with the space and time that I needed to write this dissertation. I love you!

CONTENTS

1 INTRODUCTION

1.1 MOTIVATION

Concepts are central to cognition. They are the mental representations of the kinds of objects, situations and relationships we encounter in a dynamic and uncertain world. Concepts group these entities into distinct categories, they act as building blocks that can be combined in endless ways to form new thoughts and concepts, and when shared between individuals, concepts form the basis for language and efficient communication. The ubiquity of concepts in cognition makes the study of how they are represented, used, and learned a compelling choice for advancing scientific knowledge of the human mind.

A frequent and important type of concept, and category[1], in human cognition is structured or relational (Gentner and Kurtz [2005]). For these categories, the membership of instances cannot easily be defined through a list of their properties but instead relies on relationships and the inner structure of the instances. For example, concepts like *bridge* or *enemy* are characterized by relationships between external entities, *positive* and *negative feedback systems* are defined by the relations between their components and, to give another example, the concept of *offside position* in soccer relies on spatial relationships within game situations. A natural way to represent such categories is to use symbolic, verbal rules, like "a position behind the last defense player". Humans are able to learn such verbal rules from examples and counterexamples, and often from very few of them. This ability is based both on a process of perception, which extracts information from the presented examples, and a process of concept construction, which leads to a rule that matches the given examples and can be applied to categorize new ones. My goal in this thesis is to explore how perception and concept construction unfold and interact during concept learning.

While there are several successful cognitive models of rule-based concept learning, none of them adequately captures the iterative, interactive nature of perception and rule construction. This is important to capture for several reasons. First, it addresses an inherent property of human perception – its iterative, incremental and explorative nature – as well as a real challenge that human learners face: the necessity to choose in which order to perceptually explore any given scene. This is especially relevant when learning from scenes with an inner structure, where the number of perceptual

1 The term concept is typically used to refer to a mental notion while the term category refers to the set of instances in the world that is grouped by the concept. The distinction between the two is subtle and of little relevance here, and I will use both terms interchangeably.

descriptors that could be read off the scene, like relationships between elements, increases rapidly with the number of elements. Perceiving any of these features comes at a cognitive cost but most existing algorithms treat it as free. Second, an iterative perception process that works concurrently with a rule construction process makes it possible to constrain what rules are constructed next based on the features perceived so far and to constrain what should be perceived next based on the rules constructed so far. What one sees influences what hypotheses about underlying concepts one builds and those hypotheses in turn influence what further aspects of the instances one pays attention to (Goldstone [2003]). This mutual restriction of perceptual and conceptual search spaces has the potential to significantly increase the efficiency of the learning process. Third, the positive feedback between the two processes might offer a natural explanation for a range of effects in human learning like order effects, where the order in which examples are presented influence learning performance, as well as "garden-path" or functional fixedness effects, where a learner is led down a wrong direction by an initially coherent but ultimately superficial pattern in the examples.

This thesis explores the consequences of treating perception as an iterative and active process that is tightly integrated with the process of concept learning. It thereby connects perception, structured concepts, and learning from examples – three elements that have been central both in research on natural and artificial systems throughout recent decades. My approach to this topic is building a computational cognitive model of the process of perceiving and learning verbal rules from small sets of structured examples.

In the next section, I introduce a novel category learning domain, the physical Bongard problems (PBPs), that requires the learner to discover a rule that correctly sorts a set of structured physical scenes into two categories. In the remaining two sections, I review literature related to rule-based learning algorithms and models across the fields of artificial intelligence, machine learning, psychology and cognitive science. In the main part of the thesis, I develop the computational cognitive model PATHS ("Perceiving and Testing Hypotheses on Structured data"), which gives a process-level account of the cognitive processes involved in learning structured, dynamic concepts, specifically the physical Bongard problems. The PATHS model takes inspiration from the work of Douglas Hofstadter and Harry Foundalis on perception-oriented computational models of analogy-making and especially shares their focus on the iterative aspect of perception and its interaction with higher-level cognitive processes (Hofstadter [1996], Foundalis [2006]). Instead of casting PBPs as an analogy and structure-mapping problem, the PATHS model uses a hypothesis testing account, treating PBPs as a categorization problem. Another key property is that the PATHs model makes use of ideas from rational models (e.g., Goodman et al. [2008a]), using a Bayesian estimation of hypothesis probability to drive rational decision making by the model. This marrying of process-level modeling with mean-

ingful probabilistic underpinnings for the local decisions the PATH model makes is one of the model's particular strengths.

Ideally, this computational modeling approach can benefit both psychology and artificial intelligence research. On the side of psychology, it may lead to a deeper understanding of the cognitive mechanisms involved in learning rule-based concepts. On the side of artificial intelligence, the computational model may help to realize natural interactions between human and artificial cognitive systems by endowing the latter with an understanding of, and means to simulate, human learning mechanisms.

Chapter 2 discusses how the PATHS model perceives various aspects of physical scenes, using the initial outlines and positions of the objects as input data. Chapter 3 describes the process of creating and testing hypotheses in the model and its integration with perception. Chapter 4 reports on a series of studies with human subjects solving PBPs and Chapter 5 compares their performance to that of the model. The last chapter summarizes the results and describes potential applications of the PATHS model.

1.2 PHYSICAL BONGARD PROBLEMS (PBPS)

In 1970, the English translation of a book on pattern recognition by Mikhail Moiseevitch Bongard was published in the U.S., and caught the attention of Douglas Hofstadter who was working on a book himself. Hofstadter was impressed mainly by the 100 visual categorization problems in the appendix of the book and decided to feature them in "Gödel, Escher, Bach: an Eternal Golden Braid", 1979, using the term *Bongard Problems* (BPs). In this way, Bongard problems were introduced to a larger audience. The same problems were picked up again in the late 90s by Harry Foundalis, a graduate student at Hofstadter's group, who eventually in 2006 graduated with a thesis titled: "Phaeaco: A Cognitive Architecture inspired by Bongard's Problems". He had developed a cognitive architecture that could work on BPs and actually solved a small subset of them. Impressively, his architecture resembles a vertical slice through a cognitive system, containing low-level perceptual processes working at the pixel-level and high-level symbolic analogical reasoning processes. Foundalis' work motivated and influenced the development of the PATHS model presented here.

So what are Bongard problems? They are a set of visual pattern recognition and categorization tasks where each problem is a set of twelve scenes. Six of the scenes are positioned on the left and are members of one category; the other six scenes are on the right and belong to a second category. The task is to find the underlying rule that discriminates between instances of the two categories. Figure 1 shows two examples of Bongard problems.

Many of the existing Bongard problems require some creative thinking to solve them and while some of them are quick to solve, others are quite hard. One aspect that makes them tricky to solve is that there are usually many ways to encode and interpret each scene and only particular interpretations

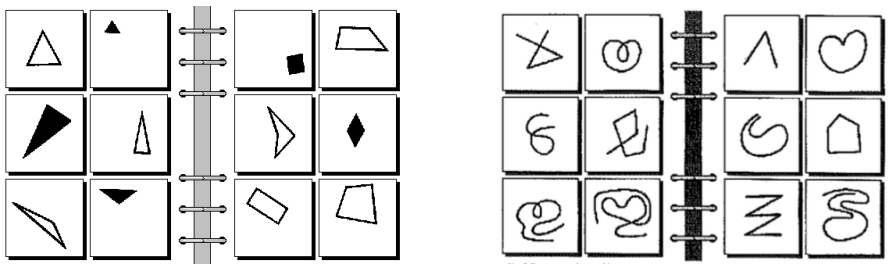


Figure 1: Two of the classical Bongard problems. On the left is BP06 "triangle vs. quadrangle", on the right is BP30 "seft-crossing line vs. no self-crossing line".

will be consistent across scenes and lead to a solution. For example, even a scene as seemingly simple as the first scene on the left side in BP06 has many ways in which it can be interpreted: as a white triangle, as three black lines of the same length, or as a multi-segment line with particular slopes and position. Any one of these interpretations might be "right", depending on the context set by other scenes. This means that in a Bongard problem, one has to solve two problems at the same time: finding the right way to look at and encode the individual scenes and finding the rule based on those perceptions that discriminates between scenes on the left and scenes on the right.

There is no formal definition of what counts or does not count as a BP other than having two sets of example scenes that are sufficient for humans to identify the two underlying concepts. In this sense, the Physical Bongard Problems I am about to introduce, can be considered an extension to the Bongard problem domain. In any case, they inherit most of the properties that make BPs a promising domain for exploring questions of cognitive psychology.

BONGARD PROBLEMS + PHYSICS. Physical Bongard Problems (PBPs) pose the same task as the classical Bongard problems: to find the rule that discriminates the examples on the left from the examples on the right. What changes is that constraints on the content of the scenes are introduced in order to shift the focus from low-level visual processing towards dynamics and interaction: PBPs are BPs with physics at their heart. Instead of arbitrary static patterns, the images contain snapshots of 2D physical scenes depicted from a side perspective. The scenes may contain arbitrary non-overlapping rigid objects which could stand stably on the ground, be positioned in mid-air or be placed at the side of a steep hill. The objects are understood to not be moving at the time of the snapshot and no hidden joints or self-propelled objects are allowed.

There is no formal definition of what constitutes a PBP beyond those constraints and in principle the concept described by a PBP could be arbitrarily complex logical rules. This would, however, not be in the spirit of PBPs. I am mainly interested in natural physical situations, i.e., PBPs with solutions

that are immediately convincing to humans once discovered, although the discovery itself might be difficult. The author of this thesis has sketched out 36 physical Bongard problems, which explore different feature and relation types as well as different solution structures. They are based on static or dynamic object properties like "shape" or "stability", on spatial and physical relationships between objects like "left of", "close to" or "supports", as well as properties of groups of objects or whole scenes like numerosity or predictability. Some problems focus on events that happen at a particular time during the imagined unfolding of events like collisions between objects, while others are based on the reaction of objects to a simple kind of imagined interaction, like pushing or lifting an object. Figures 2, 3 and 4 show examples of PBPs. In the appendix is a complete list of PBPs and their solutions. There are likely many more features beyond the ones explored in the listed PBPs on which new PBPs could be based. Additionally, features can be combined to create new PBPs.

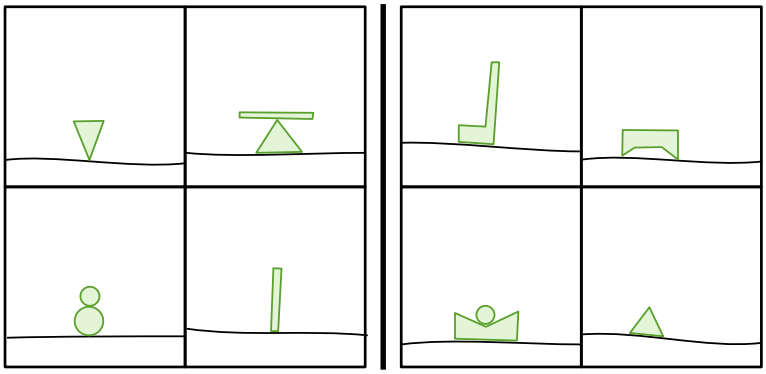


Figure 2: PBP 08. Unstable versus stable object configurations.

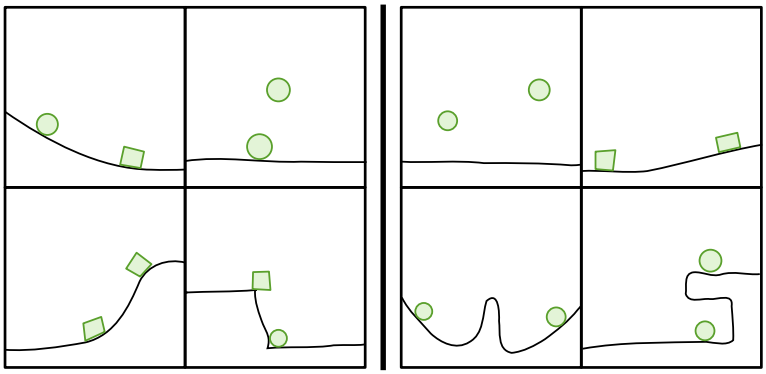


Figure 3: PBP 22. The two objects collide or don't collide.

Relative to the classical Bongard problems, the content of PBPs is more restricted – scenes can only contain physical objects. Interestingly, this restriction opens up a large space of new puzzles that rely on the intuitive instantiation of physical rules and turn static visual patterns into imagined dynamic situations and imagined interactions with them.

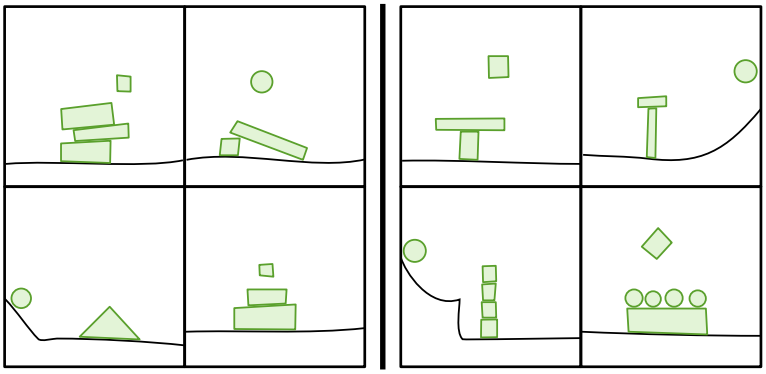


Figure 4: PBP 33. The small object destroys the construction or not.

The interpretation of each one of the physical scenes is like constructing a story of what is going on there. There are many possible stories the same scene could tell, though, and thus an important task is to come up with one that is consistent with the context provided by the other scenes. What constitutes a useful representation of what happens inside a scene depends on the solution to PBP, and the solution depends on the representations.

PBPs for Cognitive Research

In the process of figuring out the concepts presented in PBPs, a learner faces several deep challenges. These challenges are not specific to the domain of PBPs but are challenges that natural and artificial cognitive systems face in a dynamic, physical world. This makes PBPs both intricate to solve and, more importantly, an interesting domain for research on human cognition.

In the following list of challenges aspects of PBPs, the first set of three aspects is unique to PBPs, while the further ones are shared by PBPs and classical BPs.

PHYSICS, TIME & INTERACTION. Although PBPs are sets of static images, they require the learner to perceive and predict the dynamics in the depicted physical scenes. This predictive aspect of perception is essential for embodied agents that interact with a dynamic world in general, and is present in human cognition (Clark [2013], Hubbard [2005]). The need to invoke implicit physical knowledge of how the depicted object configuration will evolve (or respond to imagined physical interventions) for solving a problem is the main distinguishing characteristic of a PBP. This involves natural assumptions, such as the association of some mass with each object and the presence of a downward directed gravity force. Using these assumptions, we can make physical judgments about the stability of a configuration or predict likely states of motion, such as a ball accelerating on a ramp.

To see a scene as physical allows us to see it as a snapshot of a dynamical process. This connection generates a rich set of additional features strongly

related to time and arising from forward predictions of the changes expected in the depicted scene. A mental simulation of unfolding actions can augment the scene with events that themselves are not depicted, like the collision of objects. The timing of such events itself could also be an ingredient to the solution.

Physical understanding includes judgments about how objects might respond to imaginary interventions. This is important in many situations in life, such as to judge whether some location can support my body, or how objects can be moved in a scene without causing unwanted interference with other objects.

SELECTIVE PERCEPTION. Learning situations in the real world are often messy – typically the relevant information is embedded into irrelevant and possibly distracting information. A central skill in learning from a given situation is to figure out what parts in it are relevant for a given task. This includes perception: in physical scenes with many objects, it is inefficient to perceive all possible relationships between all possible object pairs. Instead, both in the perception and interpretation of a scene, typically a few important objects and features have to be picked out. An efficient solution to this challenge requires an iterative perception process on a feature level, which current rule-based concept learning systems are not capable of.

STRUCTURED INSTANCES & OPEN FEATURE SPACE. Beside the features of individual objects, many PBPs are based on the spatial and physical relationships between objects. Furthermore, several objects of one scene might have to be interpreted as a group based on common features or roles to find a solution.

When attempting to solve a PBP, it is neither a priori clear which features are relevant for the solution, nor is a complete list of potential features given. Instead, when solving a PBP one uses basic feature types associated with physical situations to construct a large set of concepts tailored to the specific problem at hand. Relationship features can be applied to any of the object pairs in the scene, and objects can be grouped in various ways which allows perceiving properties on the formed groups. Another way to go beyond a fixed list of features that describe a scene is to combine several features. The "direction in which the small circle moves" is an example for a concept that is composed of lower-level features like size, shape, and movement. These concepts can be applied to subsequent scenes and act as high-level features.

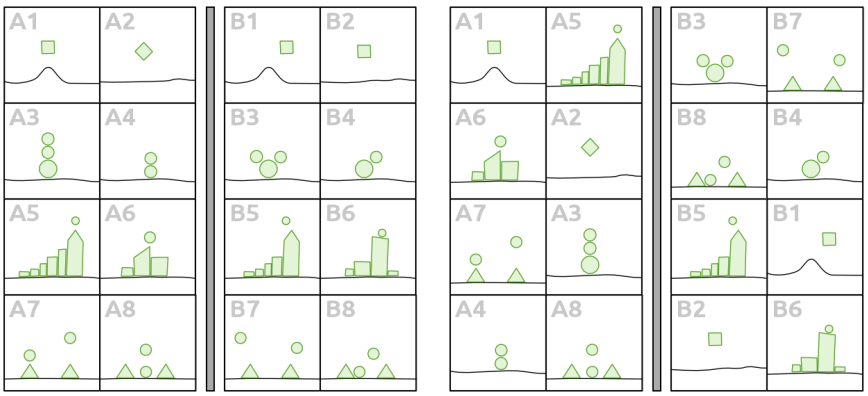
STRUCTURAL ALIGNMENT. One challenge in working with structured instances like PBP scenes is that any meaningful comparison or abstraction of two scenes requires an alignment of the scenes that clarifies which elements and relationships in one scene correspond to which elements and relationships in the other scene (Gentner and Markman [1994]). The dependency between mappings and abstractions is, however, more symmet-

ric than this characterization might suggests; mappings and abstractions mutually inform and supplement each other. The knowledge of consistent mappings between elements of two scenes supports the construction of a shared representation, but also does knowledge of a shared representation of two scenes support the construction of consistent mappings.

CONTEXT. A suitable representation of a physical scene cannot be given a priori, but depends on the context that is set by the other scenes. A single scene could be used in several PBPs and have a different interpretation in each of them, such as a different choice of what is the main object and what is the "background". Similarly, the very same object configuration could be described as stable or as unstable, depending on the reference frame set by the other scenes.

These challenging aspect, taken together, make PBPs a novel and promising domain for research in concept learning. However, the original 8-scene version of the PBPs that I designed first was not well suited for some important experimental setups. One important aspect to look at in concept learning experiments is the learner's ability to generalize, that is to transfer the learned concepts to new examples. The initial version of the PBPs contained only four scenes per side, which made it difficult to split them into a training and a test set. Another important aspect in concept learning are effects that the order in which the examples are presented has on the learning outcomes. Different ordering of examples does not only influence what is presented first and what is presented later, but also which of the examples are presented close to each other. When humans work on PBPs, they actively explore different possible rules by comparing one scene to neighboring scenes. When two scenes are next to each other, humans are more likely to compare them perceptually. Thus, an important question to ask is whether this spatial juxtaposition matters for human problem solvers, and whether a model can capture this consequence of active perception.

To better support these kinds of experiments, I created a second version of many of the original problems by extending them in two ways. First, I added more scenes to the problems, so that each of them contains 20 scenes in total. Second, I designed the twenty scenes to form five similarity groups, such that scenes within a group are more similar to each other than scenes across groups. Figure 5 shows the second version of PBP24. This setup allows researchers to manipulate the similarity of scenes that are presented spatially close to each other by reordering the scenes within the PBP. Alternatively, researchers can manipulate the similarity of scenes that are presented temporally close to each other when showing only a few of the scenes at a time.



(a) similar within pairs (b) dissimilar within pairs

Figure 5: In the extended version of PBPs, the scenes are organized in five similarity groups. This figure shows two different arrangement of the scenes of PBP 24 that place similar scenes close to each other (left) or far from each other (right). In the left arrangement, the scenes in each row belong to the same similarity group.

1.3 HOW CONCEPTS ARE LEARNED

Machine Learners

After looking at properties of the PBP domain, the following two sections take a step back and review the literature related to category learning. I will focus on learning abstractions from structured examples and on an iterative, active exploration (perception) of examples. I start with the artificial intelligence and machine learning field and look at psychology and cognitive science next, with the focus on cognitive models of rule-based category learning.

Learning from examples has been a cornerstone of artificial intelligence since its beginnings in the 1960s. Among the popular approaches were systems working with rich and often handcrafted knowledge structures. These "expert systems" were developed to utilize rich domain knowledge for making intelligent inferences and decisions. Some of those systems had the capability to learn from observations themselves, helping to address the bottleneck of time-intensive handcrafting of the systems' internal knowledge structures. Early structural learning systems include Evans' geometry analogy finder (Evans [1964]), Winston's concept learner learning the concept of an arch from carefully crafted positive and negative examples (Winston [1970]), and Fikes' robot action planning algorithms (Fikes et al. [1972]). Michalski summarized and compared influential systems of the time, including his own INDUCE, Hayes-Roth's SPROUTER, Vere's THOOTH and Buchanan's MetaDENDRAL (Dietterich and Michalski [1981]). By the mid-80s, the two major approaches in the structured learning field were inductive learning (Dietterich and Michalski [1985]), which was using inductive

learning techniques to come up with rule-based hypotheses based on a set of positive and negative examples, and explanation-based learning (Mitchell et al. [1986]), which used deductive reasoning techniques to explain one or very few examples based on given background knowledge. These early learning systems advanced our knowledge of how humans and machines can learn from structured data. In the case of MetaDENDRAL, the algorithms were even applied as a tool for making new scientific discoveries in the field of chemistry (Buchanan and Feigenbaum [1978]). Yet, many of the early structural learning systems used idiosyncratic representations and algorithms that made it hard to utilize them in settings different from the one the system was initially designed for. Additionally, perception and encoding of examples were typically assumed to happen outside of the algorithms, and none of the reviewed models tries to capture the incremental nature of example perception.

Inductive learning requires a leap from the given data to a possible generalization, and Tom Mitchell formalized generalization as a search in the, typically very large or infinite, space of generalizations (Mitchell [1982]). To choose one over another hypothesis given they both match the training data equally well requires biases in the learner (Mitchell [1980]). These biases can take the form of language biases influencing the space of generalization that is searched, or differences in the search heuristic influencing the order in which the space of generalizations is searched.

Among the inductive learning algorithms, two high-level heuristics for how to search the hypothesis space crystallized: separate-and-conquer and divide-and-conquer techniques. Divide-and-conquer algorithms recursively split the dataset into disjunctive sets, which are then tackled independently. Work on learning structured concepts (Hunt et al. [1966]), discrimination nets (Simon and Feigenbaum [1964]) and decision trees like ID3 (Quinlan [1986]) use the divide-and-conquer approach. A more conservative technique of covering different parts of the data by logical rules are the separate-and-conquer techniques. Fürnkranz [1999] gives an excellent overview of 30+ years of algorithms using this technique. All separate-and-conquer algorithms use a similar top-level loop that searches for a rule that explains some of the positive examples, then separate these and recursively continue the search on the remaining examples. Fürnkranz compares these algorithms along the three different biases they introduce: language bias, search bias, and overfitting avoidance bias.

In these theoretical and algorithmic approaches to inductive learning, the process of iterative feature perception and its interaction with hypotheses construction is hardly considered. Even at the level of examples, many of the original inductive learning algorithms like INDUCE and ID3 were operating in a batch fashion, running a single time on all available data. Since then there has been active research on extending them to on-line, incremental algorithms on the example level (see Maloof and Michalski [2004] for an overview of this line of research). Another pocket of related work is the

constructive induction approach with algorithms capable of inventing new dimensions in the representation space (Wnek and Michalski [1994]). Yet here, too, the values of newly created features are typically immediately available to the algorithm for all learning instances.

In the 80s, the field of machine learning was established as a reaction to A.I. and cognitive science being mostly concerned with the use of knowledge, such as in expert systems, and paying less attention to how this knowledge could be learned (Langley [2011]). Quickly, the new field diverged from building and working with rich, structured knowledge representations and focused almost exclusively on learning from data represented as points in a high-dimensional space, typically $\mathbb{R}^N$. This simpler representation form allowed for a deeper theoretical treatment and led to impressive practical results with algorithms working on probabilistic, noisy or incomplete data, utilizing growing computational power and larger datasets. Important machine learning algorithms in the area of supervised classification include support vector machines, artificial neural networks and radial basis functions (Kotsiantis [2007]).

Despite the great success of these methods, their reliance on vector-based input representation makes the application to domains with dependencies between data instances or, as is the case with PBPs, data with inner structure highly non-trivial. It excludes the decision of which features in a scene are attended to and how to align related features across examples from the core learning mechanism and moves it into a preprocessing step. This is unsuited for our goal of modeling the selection and perception of features as an integral and tightly connected part of the learning process.

Within machine learning, the approach that is related closest to the interaction of perception and concept learning is active learning (see Settles [2010] for an overview). The general assumption underlying active learning approaches is that unlabeled training data is acquired much easier than the corresponding category labels and that algorithms should request labels only for those training instances where the label information provides the highest estimated utility. This corresponds to making choices about perceiving category labels and does not fit the PBP scenario in which all labels are given a priori.

While most active learning research focuses on decisions on the instance level, some of the algorithms make perceptual choices at the feature level. Under the term *active feature-value acquisition*, those approaches address situations in which only a subset of feature values of the available training instances is initially known. They employ heuristics to make good decisions about which of the instances should be perceived entirely (Melville et al. [2004]) or, making finer-grained decisions, which particular missing feature values of which instance should be perceived next (Saar-Tsechansky et al. [2009]). Viewing PBPs through this lens means treating the perception of features in PBP scenes as a costly operation and requires a PBP learning

model that minimizes these costs by attending only to such features that are most likely to be part of a correct categorization rule.

While this aspect is very well aligned with the requirements for a cognitive model that can solve PBPs, the existing active feature-value acquisition approaches are not. Precisely estimating the utility of a feature perception is a difficult task, as it requires computing the effect of perceiving any of the potential values that any of the missing features in any of the instances could take on the classifier that is induced. Due to the large number of necessary computations, a classifier that is fast to train and a strategy to sample only a small subset of the missing features and instances is typically used. This approach is most helpful when computation is cheap while the costs to acquiring feature values are high, as they might, for example, require running an experiment or a medical test. In the context of a psychologically plausible model of perception and concept learning, however, the costs of a detailed consideration of the potential effects of perceiving any of the potential features in a PBP scenes will be far bigger that the costs of the perception itself. For our scenario, a much simpler heuristic to decide what to perceive next will be needed, especially considering that the number of relational features in PBP scenes is large[2].

Though the field of machine learning grew very popular in A.I., a new community formed in the early 90s that kept working with structured representations using an approach called inductive logic programming. ILP combined the inductive and deductive approaches of earlier structured knowledge approaches and used horn clauses, a subset of 1st order logic, as a unified representation to replace the large number of idiosyncratic representations used before. A logical reasoning engine, such as PROLOG, was utilized to flexibly integrate background knowledge with training examples and utilize logical inferences as part of the learning process (Muggleton [1992], Muggleton and De Raedt [1994]). In the 2000's, researchers started to combine techniques and representations from the statistical learning field and ILP leading to the active new research areas of SRL (statistical relational learning) and probabilistic ILP (Getoor and Taskar [2007], Dietterich et al. [2008], Van Laer and De Raedt [2001]).

IPL algorithms can work with rich, structured representations and show impressive learning capabilities. Yet it is exactly the powerful symbol manipulation capabilities of the logic engines these systems rely on that make them a bad candidate for cognitive modeling. Additionally, the logic engines do not seem to lend themselves to a deep integration with an incremental perceptual process.

How can the discussed approaches and algorithms inform the design of a model that works on PBPs? Many of the reviewed systems would be able to

2 Assuming N_o object features and N_r types of relationships between objects, the number of potential relationship features is $F_r = N_o * N_r^k$, where k is the number of object features that may be combined to describe the reference object of the relationship. An example relationship feature is "an object is left of a small circle". Assuming $N_o = 20, N_r = 20, k = 2$, we get $F_r = 8000$ and for $k = 3$, the number of potential relationship features is $F_r = 16,000$.

find solutions to PBPs in principle, yet the perception and perceptual decision about which of the object and relationship features in a scene to look at would have to be made in a separate pre-processing step. In the reviewed literature, almost no attention was spent on the interaction of perception and concept learning I want to model in this thesis. Constructive induction algorithms allow re-encoding of instances but do not capture any incremental or costly perception of newly created features. Active feature-value acquisition algorithms make perceptual decisions on the feature level, but rely on vector representations and seem unsuited as the basis for a cognitive model.

I found many of the reviewed algorithms to be implausible choices for modeling human concept learning since they were not developed to take constraints of human memory and attention into account. Among the better candidates are iterative versions of decision tree algorithms and algorithms like INDUCE that generate and test hypotheses based on the observed data. In the next section, I will look at literature from cognitive science and models of human concept learning, including the INDUCE algorithm.

Human Learners and Cognitive Models

Concepts and categorization are central to human cognition. When people perceive something, they naturally perceive it *as* something; we interpret everything we see. This organization of perceptions into categories has various advantages. It allows to make predictions on properties of other category members, it allows for efficient communication with other individuals that have a shared understanding of the categories and existing concepts can act as building blocks for constructing new concepts. In fact, many cognitive acts can be seen as acts of categorization and naturally, much of psychology is concerned with concepts, categorization, and related issues (see Goldstone et al. [2012] for a great overview of the field). Physical Bongard Problems are instances of complex cognitive acts that come in the form of categorization tasks.

In psychology literature, the term concept is typically used to refer to the mental notion of a class or individual, while the term category refers to a set of external entities that are grouped together. Concepts exist in the mind, whereas categories – or at least their members – exist in the real world.

Much research has been done on models of how concepts are represented and used in the mind by studying a variety of categories and categorization tasks. Several different types of mental representations have been proposed and how well each of them is supported by experimental data is typically dependent on what kind of categorization tasks one looks at.

The idea that humans represent categories through simple logical rules was dominant in cognitive psychology in the 1950s and 1960s. In very influential work and in part as a response to behaviorist approaches, Bruner, Goodnow, and George [1956] proposed a hypothesis testing account of category learning, theorizing that concept learning involves active formation

of hypotheses and testing them on observed instances. A hypothesis would consist of a simple logical rule that defines the membership of instances. For instance, participants were asked to learn category memberships of geometrical figures that differed in some pre-defined binary attributes like shape, color or border type. In the following decade, this account was intensely explored, typically using artificially created Boolean categories with a small number of feature dimensions (Bourne [1970], Bower and Trabasso [1964], Feldman [2003]).

The 1970s saw a major shift of attention in the study of category learning away from rule-based accounts, due to researchers like Rosch and Mervis [1975], argued very convincingly that almost any natural category seems impossible to describe in terms of necessary and sufficient rules and is better described in terms of family resemblance using similarity measures. Another perceived shortcoming of the rule-based account was that they could not account for the graded responses to the degree of membership or typicality that people give. The two main types of similarity-based categorization theories that gained a lot of traction were prototype-based representations, which are structured around an average or typical instance of the category members (Rosch [1975]), and instance-based representations, which retain all seen instances (Medin and Schaffer [1978], Nosofsky [1986], Kruschke [1992], Logan [1988]). Both approaches rely on a similarity metric to compare new percepts to the prototype or to previously encountered instances and both are especially well suited for learning natural kinds or basic-level categories. It is possible to combine aspects of both approaches like in the SUSTAIN model of Love et al. [2004], which learns concepts by building several intermediate clusters – similar to having several prototypes.

Yet another fruitful approach to concept learning are connectionist models like artificial neural networks (ANNs). ANNs are very powerful in picking up pattern from presented data and are well suited for modeling perceptual learning processes (Bishop [1995], Goldstone [2003], Goldstone et al. [2009]). Yet, it is much harder to use ANNs for work at the symbolic level, which is the level at which the verbal rules are that humans come up with when solving PBPs. The benefits of this symbolic representation include the relative ease with which rules can express relationships between abstract elements, with which they can be recombined to form more complex rules and with which they are communicated between learners.

One important way in which categorization tasks differ is how strongly the to-be-learned category is driven by surface similarities. Natural kinds like bird species and man-made artifacts like tools typically share many surface similarities. Rosch coined the notion of "basic-level categories" for such categories that are at the most commonly used abstraction level and are characterized by a high within-category similarity (Rosch et al. [1976]). Other types of categories include ad-hoc categories, such as "things to take out of the house in case of fire", abstract schemas and metaphors. These category types are less dependent on surface similarities and they often

combine things into a common category that may appear very different from each other outside of a specific context. Consider, for example, "things one can stand on" or "positive feedback systems".

The categories in physical Bongard problems are instances of these latter category types. PBP categories are constructed around rules and often combine scenes into the same category that differ in many other aspects, making the scenes appear quite dissimilar on the surface. PBPs are well-defined in that a simple rule that correctly categorizes all scenes always exists, and the task of a solver is to find and vocalize such a rule.

Despite the success of similarity-based models in explaining human behavior data in many categorization tasks, the rule-based approach clearly captured existing aspects of human concept learning, too. In the 90s, the interest in rule-based concept representations was rekindled and there is now a broad acceptance that the two approaches are by no means mutually exclusive. In cognitive neuroscience, there is convincing evidence that humans have multiple category learning systems which function in rule-like and similarity-like manners (Ashby and Maddox [2011]). These systems might be constantly working in parallel and competing with each other as in the COVIS model (Ashby et al. [1998]). Which system is dominating in a particular task will depend on the structure of the to-be-learned categories, and even for a single category, subjects might follow a rule learning approach at first and then gradually move towards an instance and similarity-based approach for identifying new instances – as their growing repertoire of perceived examples permits (Logan [1988], Ashby and Maddox [2005]). PBPs were designed so that they directly require the construction and vocalization of categorization rules, which is the aspect my work focuses on.

I will next describe three influential cognitive models of rule-based category learning in some more detail.

Models of Rule-based Category Learning

INDUCE An early and influential rule-based learning system from the field of AI is the rule-induction algorithm INDUCE by Michalski (Dietterich and Michalski [1981]). The INDUCE algorithm belongs to the family of separate-and-conquer algorithms: It starts with a positive example of the category and constructs a so-called star-rule – a conjunction of feature descriptors that includes the positive, but none of the negative examples. It then removes all positive examples covered by that rule and repeats the process with the remaining positive examples.

Medin, Wattenmaker, and Michalski [1987] explored INDUCE as a process-level cognitive model for human category learning. In their paper, they compared the learning results of the INDUCE algorithm with the performance of human learners using pictures of trains with varying numbers of cars, types of loads and colorings, as well as driving directions as categorization task.

The authors found that INDUCE does a reasonable job as a process model although it lacks the ability to combine an overgeneralizing rule with descriptions of counter-examples. In terms of human strategies, they found that people did not merely prefer the simplest description but had a strong bias towards concept validity over cue validity. That is, they preferred features that are consistent within the category to features that are discriminating best between categories, even in the context of a discrimination task. Furthermore, people preferred conjunctive rules to disjunctive ones.

While my work shares the general approach of using a hypothesis-testing algorithm as a process model for human rule-based category learning, there are at least two important differences. First, the PBPs scenes contain internal relationships, while such relationships were not part of the problem domain and were not considered by the model. Second, I will focus on iterative perception and its interconnection with rule generation. To allow this focus within the scope of my work, each PBP was designed around a single conjunctive rule. According to the findings of the INDUCE experiment, conjunctive rules that cover a concept are a natural starting point for humans in inductive categorization tasks.

RULEX The RULEX model from Nosofski and Palmeri (Nosofsky et al. [1994]) is an influential model of rule-based category learning that is based on the assumption that simple rule-based categories are represented using a logical rule and a set of exceptions from that rule. The process-level version of RULEX is restricted to binary features and assumes a small number of predefined feature dimensions. The training instances are presented one by one to the model in combination with their category labels. The model learns in two stages. In the first stage, it identifies a simple rule based either on a single feature dimension or, if no sufficiently accurate single feature rule can be found, a conjunctive rule involving two features. To identify a rule, the model stochastically chooses one or two feature dimensions based on their predefined saliencies. Then, a rule that reflects the values of the selected feature dimensions in the current training instance is constructed and checked against subsequent training instances. The rule is eventually accepted or rejected based on the ratio of compatible instances, which is compared against a set of thresholds that are parameters of the model. When a rule is accepted, the model proceeds to the second stage, in which it stores all instances that the rule categorizes incorrectly.

In the context of PBPs we are interested in the first stage of this learning process since PBPs are based on well-defined concepts that don't require exceptions. One aspect of the RULEX model that seems well suited for a concept-learning model for PBPs is the use of predefined saliencies to influence the feature dimensions that the model attends to. The aspect of not storing all seen instances in memory and re-analyzing when a new candidate rule is considered is another psychologically appropriate choice, although it is likely the case that at least some information from those in-

stances is retained by a learner so it can influence the choice for the next rule candidate.

Several aspects of the RULEX model are inappropriate for the structural learning situation of PBPs, though. First, the number of possible rules in the experiments RULEX was applied to is small, while for PBP it is typically very large. Second, the number of potentially relevant feature dimensions is small for RULEX, while it is again quite large for PBPs. These two differences allow the RULEX model to work despite a very simple rule candidate selection process that does not consider information gained in previous perceptions. A model that handles PBPs will need to intelligently constrain and prioritize the rule search space in order to find the correct solution among vastly more candidates in a reasonable time.

COVIS Ashby et al. [1998] developed a neuro-cognitive model of concept learning, which contains two separate systems that compete with each other during learning and predicting of category membership. One system, the information-integration system, works in a similarity-based fashion while the other system represents categories through simple rules that can be easily verbalized. The rule-based system initially selects a rule from a small set of predefined rules as the active rule. Each rule has an associated saliency that is increased for the active rule when a consistent example, and decreased when an inconsistent example is encountered. Whenever an example inconsistent with the active rule is presented to COVIS, the rule-based system will select the rule with the highest saliency – which might be either the same or a different rule. The selection process has an added stochastic element favoring rule-switching and an adjustable static bias towards keeping the current rule.

The main contribution of the COVIS model is the integration of two distinct concept learning systems into a single architecture which is closely aligned with human brain structures. This allows modeling race-effects between these two systems, exploration of the situations in which each of the systems excels and modeling of the effects brain anomalies have on concept learning. Yet, similarly to the RULEX model, COVIS assumes extremely simple rule structures in its treatment of rule-based categories. The number of possible rules for describing a PBP scene is larger than the number of rules COVIS works with by several degrees of magnitude. Solving PBPs will require more powerful techniques for focusing on promising subsets of the rule space than the update of a single rule saliency per example that is used in COVIS.

Relational Categories

The rekindling of interest in rule-based learning and the development of models like the ones discussed above seems to capture existing aspects of human concept learning and establishes continuity with the early theorizing on hypothesis testing. Yet, the more recent models of rule-based learn-

ing continue to focus on very simple rule-based categories, whose instances can be represented as short and predefined attribute-value lists. Many real world categories, however, define membership through structure within or between instances. Gentner and Kurtz [2005] characterize such categories, like "bridge", "enemy" or "barrier" as "relational categories" and argue that despite often being overlooked in the categorization research, they are both frequently used and important in cognition. The instances of the categorization task of PBPs, the physical scenes, have a rich inner structure and in many of the underlying categories capture a structural aspect of the scenes, such as "a circle left of a square".

Rational Models

The models of categorization that we have discussed so far are mechanistic models that try to model processes in the human mind. An alternative, complementary class of models is rational models. Instead of trying to predict categorization behavior based on the structure of the mind, rational models predict categorization behavior based on the structure of the environment, assuming that the behavior is perfectly adapted to it. In the rational approach, one first determines what information is available to an individual in a world with uncertainty and what the individual's goals are. The next step is to mathematically derive or estimate the optimal decision an individual should make in given circumstances. Both a good fit or systematic derivations of the observed behavior from the rationally optimal behavior can provide insights into what information is actually taken into account by the individual and how powerful the actual cognitive mechanisms are.

The rational approach can be applied to analyze the learning of concepts based on feature similarity and Anderson [1991] gives a rational analysis of learning attribute-value represented categories. Many categories are structured or relational, though, and the rational analysis has been extended to these, too. One line of research uses a rational, Bayesian model to demonstrate and analyze how structured background information is incorporated into concept learning tasks (Kemp and Tenenbaum [2009], Griffiths and Tenenbaum [2009]). Another line of research applies Bayesian analysis to learning of structured, rule-based concepts by using a generative grammar to construct all possible rules – a language of thought. The priors are defined over the parts of the grammar and posterior probabilities can be computed for each generated rule. A powerful aspect of these grammars is that they are easy to extend to cover relations to other objects and naturally allow for compositionality of concepts (Goodman et al. [2008a,b]).

Rational models can provide important insights into how it is theoretically possible that people learn new concepts from only a few examples and can predict optimal categorization behavior. Unfortunately, they provide little information on processes that allow the mind to achieve the behavior. The design of the PATHS model borrows techniques from rational analysis but integrates them into a process-level model of concept learning. This allows

the model, at least in principle, to account for order effects and to take known cognitive limitations into account.

Analogy-Making

With the exception of the rational models discussed above, concept learning researchers have almost exclusively focused on unstructured attribute-value represented concepts. An active research area that puts the focus on structured knowledge and how it is retrieved aligned and reasoned upon is analogy-making research. In the past decades, there has been considerable success in modeling these processes; Gentner and Forbus [2011] provide a review of the computational models in the field. The process of making an analogy can be decomposed into several subprocesses. During *retrieval*, potential analogs to a target situation are retrieved from memory, during *mapping*, a structural alignment between a source and a target situation is constructed and could be used to *infer* additional information about the target based on what is known about the source. An *abstraction* might be constructed from the results of a comparison and reasoners might *rerepresent* the analogs after a partial match to improve the match. Additionally, the *encoding* of the two analogs plays an important role in the process.

A very influential approach in the analogy-making field is structure mapping theory, SMT, and its implementation in the structure mapping engine, SME (Gentner [1983], Falkenhainer et al. [1989], Forbus et al. [1994]). Both focus on the subprocess of mapping the elements of a source representation onto the elements of a target representation. SMT is domain-general and based on three constraints. First, *structural consistency* of the mapping, specifically 1:1 mappings between elements and parallel connectivity of child elements. Second, *systematicity* meaning that mapping of higher-order relations is preferred to the mapping of lower-order relations or attributes. Third, *tiered identicality* meaning that mapped elements have to be identical or taxonomically close to get mapped unless the mapping is supported by a larger mapping structure. SMT has been used in various other models that extend the focus from the mapping subprocess to, among others, retrieval (MAC/-FAC, Forbus et al. [1995]) and abstraction (SEQL, Kuehne et al. [2000]). The aspect of encoding the analogs that are to be structurally aligned is typically not a part of the SME-based models and they start from an existing symbolic representation instead.

Not all models of analogy-making take a purely symbolic approach like SME and there are connectionist models like LISA (Hummel and Holyoak [1997]), as well as hybrid models like AMBR and DUAL (Kokinov and Petrov [2001]).

In analogy-making models, typically little attention is paid to the perception and encoding of the instances that are to be mapped. One important exception are the systems based on the *fluid analogies (FA)* framework of Hofstadter [1996]. The FA framework describes analogy-making in terms of a number of dynamically interacting components. The first component is a

central blackboard (the workspace), where representations of the instances are created, modified and destroyed by small actions (codelets). The second component is the coderack, which holds the set of actions that can be selected for execution. Actions are stochastically selected for execution based on fixed utilities associated with each action. The third component described in the FA framework is the slipnet, a graph of features, concepts, and their interconnections. The slipnet regulates feature activations, which influence which actions are added to the coderack. Additionally, the dynamic distances of nodes in the slipnet are used to decide whether to allow mapping of non-identical concepts onto each other.

The FA framework is algorithmically instantiated in several domain-specific systems that build analogies on letter strings (CopyCat, Mitchell [1993]), arrangements on dinner tables (TableTop, French and Hofstadter [1992]), musical tunes (Musicat, Nichols [2012]), and others. These specific computational models generally take some liberty in how to implement the FA approach and use, for example, several workspaces or do not use a slipnet at all. The implementation of the FA approach that is closest related to our work and did, in fact, inspire the topic of this thesis, is Harry Foundalis' Phaeaco (Foundalis [2006]). Phaeaco is an FA system that searches for solutions to classical Bongard problems. In contrast to existing systems that work on similar problem domains, Phaeaco does not require an a priori symbolic description of the scenes and instead uses computer vision algorithms to perceive scene images. It resembles a vertical slice through a cognitive system, from low-level perception up to creating symbolic analogies.

The FA framework in general and Phaeaco specifically are closely related both to the PBP problem domain and to my goal of modeling PBP solving with a focus on the interactions of perception and rule-construction. While this goal is framed in terms of concept learning, the question arises whether Phaeaco, or an adjusted version of it, can be used to reach it. An alternative approach would be to design and implement a new model, based on the FA framework. Unfortunately, neither of these options is viable.

Phaeaco itself has, despite its impressive ability solve BPs starting from a pixel-level representation, a number of significant limitations. For example, it cannot ignore parts of a scene as irrelevant, it is not able to interpret relational situations like an object being left of another object and it cannot apply an interpretation that it discovered for one scene to another scene directly. These are key capabilities that I want to capture in a model of structured concept learning. Modifying Phaeaco to add these capabilities is impractical both because the required changes are significant and because I could not obtain the source code.

The alternative of constructing a new model based on the FA framework would be a surprisingly complex undertaking for a number of reasons. One reason is that the FA framework is mostly a philosophical or conceptual framework and is quite unspecific about how to practically implement it. Additionally, it advocates domain specific implementations that require a

nearly complete reimplementation for each problem domain. Finally, there are no guidelines for making the numerous structure and parameter choices required in the implementation of the codelets and the slipnet. These choices are both important and hard to get right, since they influence the resulting recurrent dynamics in the FA system in complicated ways.

While these points illustrate that the implementation of an FA system are more complex and less constrained than one might wish, the more important theoretical question concerns the relative merits of casting PBPs as an analogy-making or a concept learning situation. The common challenge is that a meaningful comparison or abstraction of structured examples like PBP scenes requires an alignment of the elements in the examples (Gentner and Markman [1994]). The difference between the two perspectives is that in analogy-making, the search for a consistent structural alignment typically precedes an optional construction of an abstraction while in rule-based concept learning, the structural alignments are typically implicit results of the constructed abstractions. If a learner extracted the rule "a square supports another objects" from looking at a PBP, the progress of checking the rule on two scenes implicitly aligns the elements in them that are relevant in the abstraction.

I decided on the concept learning perspective in this thesis. The main reason was that a meaningful one-to-one mapping of between the elements of different scenes only exists for very few of the PBPs. Mappings rather exist on the level of a successful abstraction, or gist, of these situations (Hofstadter [1995]). For PBPs, which additionally do not have a deep hierarchical structure of relationships that has to be mapped, a direct approach to constructing abstractions seemed more fruitful.

Summary

A central challenge in modeling rule-based concept learning is to capture the unfolding of low-level perceptual processes and of high-level symbolic rule-construction, as well as their interactions. While the models and algorithms in the discussed literature provide partial answers to this challenge, none of them sufficiently captures the integration of, and interaction between these processes, which is essential for effective concept learning from structured examples in a cognitively plausible way. In my modeling approach, I follow a hypothesis generation and testing account of learning, as it provides – when properly combined with perceptual capabilities – a good basis for integration of rule-construction and perceptual choices, and can be constructed in a way that is compatible with human memory and processing limitations.

2

PERCEIVING PHYSICAL SCENES

To extract relevant information from the static images of physical scenes that make up a PBP, the perception component of the PATHS model has to solve a number of problems. First, while the depicted scenes are static, they describe dynamic, physical setups and the PATHS model has to be able to interpret them as such. It has to be able to predict both the unfolding of events in a scene and the results of imagined interactions with a scene. This need for predictive perception is shared by embodied agents, natural and artificial, that interact with a dynamic world. For such agents, a central goal of perception is to provide relevant information for acting in the world. Since performing an action takes time, it is often useful to see the world as it will be, instead of as it is now – a thought captured in the conceptual momentum research (Hubbard [2005]). Another important benefit of the ability to mentally explore a potential future is that it is a much safer way to figure out questions like "will this branch support my weight?" or "could this rock topple over and crush my foot?". Beyond the importance of predictions for perceiving and acting successfully in physical situations, it can provide a powerful learning signal in general, through the comparison of predictions and perceptions (O'Reilly et al. [2014]). In fact, the case has been made that the entire human mind can be interpreted as a "hierarchical prediction machine" (Clark [2013]).

A second challenge the PATHS model has to solve is to bridge the gap between "raw" perceptual input and the symbolic representations of the concept. In the case of PBP scenes, many features are metric in nature such as distances between objects, while in the formulation of a contrastive categorization rule a binary membership to a feature concept, such as whether or not two objects are close to each other, is preferable. The PATHS model solves this by using fuzzy feature concepts as intermediate representation. The perception of a feature is modeled as a mapping of actual measures in a scene (e.g., 0.36 units distance between two objects) to a degree of membership to a feature concept (e.g., 60% close). During rule-construction, a second mapping transforms the membership degrees to binary memberships (e.g., close). The thresholds of the second mapping are adjustable and can be adapted to the particular context set through previous perceptions of other scenes. While "60% close" might be mapped to "close" in the context of one PBP, it might be mapped to "not close" in the context of another where it helps distinguishing scenes on the left from scenes on the right.

A third challenge results from the fact that PBPs are structured and dynamic and, therefore, have a large, open space of features that can be constructed to describe a scene. Possible features include relationships between

arbitrary pairs of objects, attributes of arbitrary groups of objects, as well as events and attributes of objects that are constructed through imagining the unfolding of a scene over time. The challenge is that the perception of features takes cognitive effort and the number of possible features is large and increases dramatically with the number of objects in a scene. It is not an efficient strategy to perceive everything that could be perceived, so a choice about what to perceive and when must be made. This fundamental necessity of selective perception is central for human cognition and is evident in powerful selective attention mechanisms. In the PATHS model, feature saliencies provide a bottom-up way to guide the perception process, while insights from the rule-construction process can influence what is perceived next in a top-down manner.

I will address the first two points in this chapter and come back to the last point in the next chapter where I discuss the rule-construction process and its interaction with the perception process.

2.1 FEATURE SPACE

To a skilled observer, each scene of a PBP tells a small physical story, or – more accurately – it tells one of many possible stories depending on how it is interpreted. One necessary prerequisite for understanding these physical stories is conceptual knowledge about the aspects that make up the scene. This includes knowledge about rigid objects, their physical behavior, their features, as well as relationships between objects. Just like a human observer, a computational model needs to work with a set of basic a priori known concepts. While perceiving the scenes and searching for a fitting interpretation, more complex features and concepts can be constructed based on this basic set of concepts. When the PATHS model works on a PBP, it starts off without any knowledge about the scenes or the objects in them. Only by actively selecting a feature and a target and perceiving the feature on the target does the model build an internal representations of the scenes. The basic features that are available to the model are listed below. Each of them is associated with a procedure to perceive it using the object outlines that are visible in the PBP scene pictures.

GEOMETRIC PROPERTIES Understanding a scene involves identifying the objects in it, and often their sizes and their shapes. Objects might be grouped together based on shared properties. PATHS can perceive the features *small*, *big*, *triangle*, *square*, *circle*, *rectangle* and the *count* of objects in a group.

SPATIAL PROPERTIES The position of an object in a scene and especially spatial relationships between objects are important to make sense of physical scenes and are used in PBPs. Different from the geometric properties, the spatial properties can change over time in a scene as the physics unfold. PATHS can perceive the object attributes *left*, *right*,

bottom, *top*, *on-ground* and *single* (not close to any other object), as well as the object relationships *left-of*, *right-of*, *beside*, *above*, *below*, *on-top-of*, *close*, *far*, and *touches*. It can also perceive whether the objects in a group are all *close* or *far* to each other or are all *touching* each other.

TEMPORAL PROPERTIES Temporal aspects of a scene include the state of objects at different times during the imagined unfolding of physics, as well as changes over time. Some events like collisions happen at a particular point in time, other events, like a ball rolling down a ramp, happen over a time interval. The PATHS model looks at the scene at two particular times. Time t_{start} is the initial situation depicted in the scenes of a PBP and t_{end} is the time at which, after the unfolding of physics, all objects in a scene come to a halt. All features can be perceived by the model at those two times, and most of the features can have different values at them.

DYNAMIC OR PHYSICAL PROPERTIES The combination of spatial and temporal dynamics gives rise to many physical properties, such as object movements and collisions, the stability of an object under disturbances, the support relationship between objects or how the movements of a particular object are restrained. The PATHS model can perceive the object features *moves*, *stable*, *unstable*, and *moveable-up*, as well as the relationships *supports*, *hits*, *gets-hit* and *collide*.

These concepts cover a large portion of the frequently used vocabulary with which humans described PBP scenes in their solution attempts. There are additional concepts used by humans that are not part of our computational model, though. These include metaphors like "the object is trapped / hidden / protected" to describe an object that cannot be picked up because another object is blocking the way or descriptions like "the remnants of an ancient culture" that a participant used to distinguish an horizontally arrangement of objects from a vertical one. Another type of scene interpretation that is beyond the capabilities of the PATHS model are episodic descriptions of the fate of an "actor"throughout time. Such accounts are especially important when entities with agency are involved, which is not the case in the PBP domain.

Practical knowledge about the features listed above involves the ability to perceive or predict them in a given situation. Additionally, many of the features can be satisfied to different degrees and how binary labels are attached to them can depend on the context of the PBP. For example, a stack of blocks that is interpreted as stable in one PBP, might have to be interpreted as unstable in the context of another PBP in order to find a solution. PBP 8 and 30 are such a set of problems.

2.2 PHYSICAL FEATURES

A particularly powerful way of predicting the unfolding events and object affordances in a scene is to mentally perform physical simulations. While a major approach to model thinking about physical situations has been to represent them symbolically and use logical inferences to reason about them (Forbus [1984]), other researchers have explored the idea of physical reasoning as running mental simulations. The mental simulations do not require abstract knowledge of physical laws and instead rely on a practical understanding of how one situation turns into the next. They are cognitively plausible (Barsalou [1999]) and fit human data better than several rule-based models, for example when making predictions about the stability and fall patterns of Jenga towers (Battaglia et al. [2013]).

In the PATHS model, I use a 2D physics engine for performing mental physics simulations. A physics engine is a computer program that numerically simulates physical scenes and can be used to predict a scene's short-term future and to simulate interactions with the scene. I regard the physics engine as a black box that endows the model with the capability of imagining the unfolding of physical events. While this approach has some cognitive plausibility at an abstract level, it should not be misinterpreted as an attempt to provide a process-level account of mental physics simulation. In fact, the mechanisms within typical physics engines are largely incompatible with what might happen in a person's mind. For example, classical physics engines are carefully handcrafted based on an implementation of Newton's laws and don't afford being learned in an iterative process. Assuming that the mechanisms to perform physical simulations are already in place, there are still a number of aspects of human behavior that are not well captured by them. First, there are no direct means to handle incomplete or inaccurate input data, something that humans have to deal with all the time (although see Battaglia et al. [2013] for a technical way to deal with uncertainty). Second, PE's don't allow predicting qualitative results without simulating all detailed quantitative data. For example, when dumping a sack full of pebbles onto the ground, we should be able to estimate the rough shape of the pile after the pebbles come to a rest, without undertaking the hopeless endeavor of predicting each single pebbles' position in the pile. It might well be that in the human cognition several systems are working together to allow predictions at different levels of detail in physical scenes. In the context of PBPs, the limitations of existing physics engines play only a minor role, so they are well suited to provide the PATHS model with physics simulation capabilities.

Output of a Physics Engine

When a typical physics engine is provided with the positions and outlines of all objects in a scene, it can simulate the unfolding of events to provide

the following object properties for each object at a given time in the future: position and velocity, distance to other objects, and collisions with other objects that have occurred. These properties are used in the implementation of the model's perceptual routines.

In some cases, this requires significant post-processing of the physics output. Taking collisions as an example, there are much more collisions in a physics simulation than one would expect. An object that is resting on the ground will, in each time step, sink slightly into the ground due to gravity, only to collide with the ground and be pushed up again, resulting in a large number of "micro-collisions". In cases where two objects move towards each other and then collide, it might take the physics engine more than a single time step to separate them, resulting in a series of collisions. Both effects are due to the internals of the physics engine and are not part of the concept of a collision that a scene interpreter should have. To filter out all uninteresting collision events, the model first calculates the velocity with which the objects collide. It then only retains collisions that have a velocity above a threshold, which could be thought of as the collisions that would make a perceivable sound in the real world. The model also merges collisions that are temporally close, so that a maximum of four collisions events can be discriminated per second for any given object pair.

All features that are derived from the simulation data are represented as percepts with a degree of membership, or satisfaction, to a concept. The range of the degree is from zero to one, where zero means no membership and one means full membership. This is done to model the gradual nature of most concepts. The distance between objects is captured by the concepts "touch", "close" and "far". The remaining physical concepts, "moves", "stability", "supports" and "movability" require simulation beyond the calculation of an object's state at a particular time.

To perceive the "moves" attribute, the model captures whether an object moves or is about to move at a particular point in time. In order to find out whether an object is about to move, the model triggers a short simulation of the situation and checks whether the object in question moves 0.2 seconds in the future. This identifies objects that hang in mid-air in a PBP scene as moving objects, although they are, according to the rules of PBPS, not yet moving in the moment that is depicted.

Perceiving the other three physical concepts is based on counter-factual reasoning or "Probehandeln" and is described next.

Stability

People have a quick sense of how stably a few objects are configured, just by looking at them. This allows us to foresee potentially dangerous situations such as spilling coffee by placing a mug too close to the edge of a table, or falling down from a badly positioned ladder.

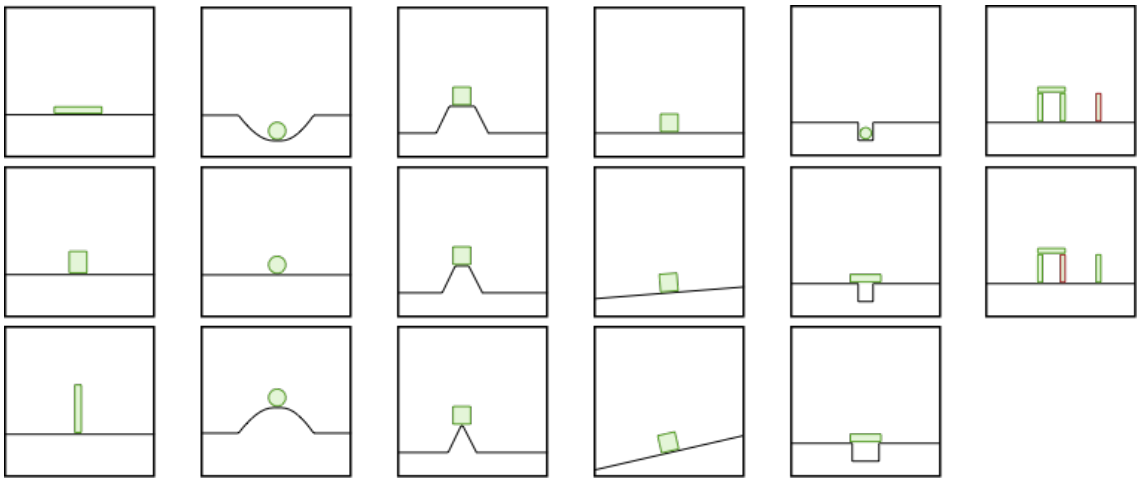


Figure 6: Different stability situations. Each column above has several scenes with objects that vary in stability. The most stable situations are at the top. Different columns show different reasons for stability variations.

In the context of dynamic rigid objects, a natural definition of object stability is an object's ability to withstand external forces without moving. If an object can tolerate stronger perturbations relative to its mass without toppling over or rolling away, we will consider it more stable than an object that can tolerate weaker perturbations before moving.

For a single object resting on the ground, its stability is mostly influenced by the amount of surface area in contact with the ground and the position of its center of mass. When several objects are involved, an object might be stable due to other objects pinning it down. Similarly, a curved or angled ground will affect object stability. Figure 6 shows different types of stability situations, each with several instantiations that vary in their degree of stability.

Capturing all these situations with logical rules would be a complex undertaking. However, there is reason to believe that in order to make judgments about the dynamics and outcomes of physical situations people instead perform mental simulations. Battaglia et al. [2013] compared how well different models of physical reasoning matched the predictions of participants towards which direction an unstable Jenga-tower will fall. They found that a series of physics simulations that additionally accounted for a person's uncertainty about the exact position of Jenga blocks in the scene provided the best fit. This is opposed to accounts stating that humans rely on simple measures like height and asymmetry to reason about potential results.

The PATHS model uses a physics engine to perform mental simulations of a physical scene that allow it to predict the future and to engage in counterfactual reasoning. To assess stability, the physics engine is first used to predict the near future of the scene without any external perturbations. If the

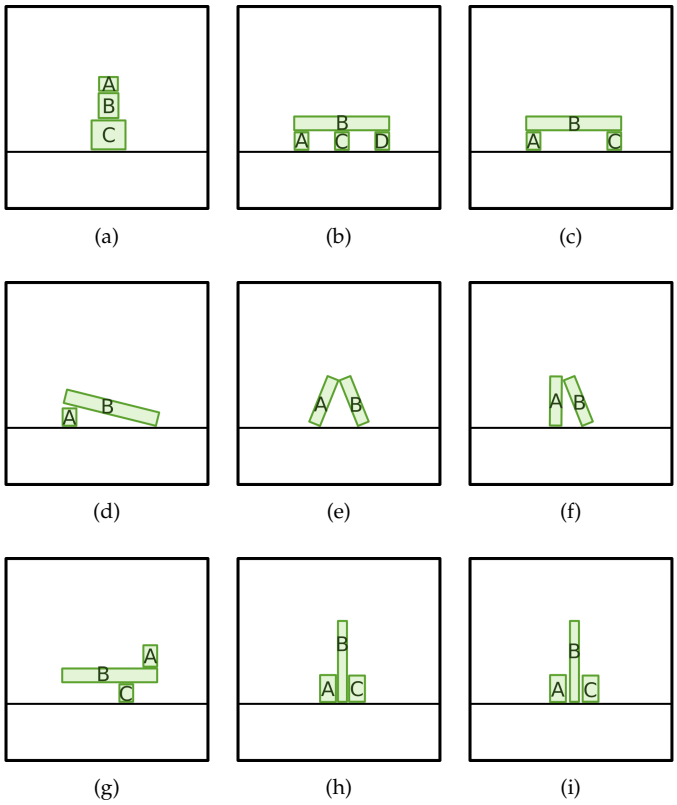


Figure 7: Different types of support situations.

target object significantly moves, it is considered unstable. Otherwise, the algorithm resets the scene and conducts a series of three short simulations, with an increasingly strong horizontal impulse applied to the target object's center of mass at the start of the simulation. This impulse is comparable to hitting a small object with a flick of a finger to observe how it reacts. If the target object topples over, falls down, or rolls away – all significant changes in position and orientation – the object is considered unstable. The perceived stability has a membership value between 0 and 1 that reflects how strong an impulse was needed to push the object out of balance.

This implementation of perceiving stability has the advantage that it both takes the object's shape and the environment into account while being very general, as it covers all situations the physics engine can simulate.

Support

The concept of support is closely related to stability in that it describes whether the presence of an object helps to stabilize another object. Building stable structures that have a proper inner support is obviously an important topic in architecture and structural engineering, both using sophisticated methods for analyzing complex structures. In the context of PBPs, we are interested in a more intuitive understanding of support and generally do

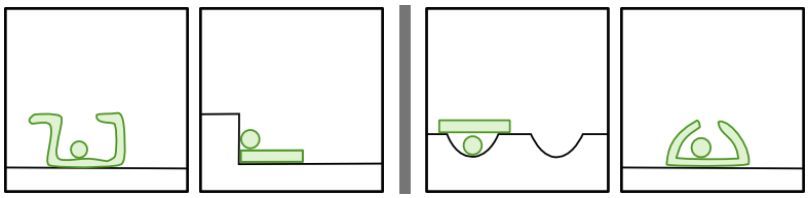


Figure 8: Four scenes taken from PBP 31. They vary in whether the small object can be lifted up or not.

not take into account that an object might break or might be fixed to another object.

Support situations with separate, rigid, non-breakable objects can be organized according to several aspects. The first aspect reflects whether the two objects touch. I will call the support relation *direct* if they do and *indirect* if they do not. In Figure 7a, object C directly supports B and indirectly supports A. The second aspect reflects the redundancy of the support an object provides. An object might be the only supporter of another object, like C is for B in Figure 7c, or it might be part of a group of supporters, like C is for B in Figure 7b. Similarly, a third aspect of support is whether the supported object would actually fall or topple over without the supporter or remain stationary but become less stable (see object A supporting object B in Figures 7g and 7h, respectively). I refer to the latter as a *stabilizing* relationship.

Support relationships can get complicated quickly and I chose to limit the model's perception of support to a relationship between object pairs. In order to find out whether an object supports another object, the model uses the same counter-factual reasoning as it does when perceiving stability. The model "imagines" what would happen when the supporter is removed by running the respective physical simulation. If the potentially supported object starts to move or becomes unstable after the removal of the potential supporter and remains stable if the supporter is not removed, there is a supporting relationship between the objects. The model perceives four different types of support: *direct*, *indirect*, *stabilizing* and *no* support, and currently maps them to gradual memberships to a single "supports" concept.

Movability

In physical scenes, we encounter both objects that we can pick up or move around, and objects that are limited in the way they can be moved. In the context of PBPs, the main reason for potential constraints in how an object can move is that the path of the object might be blocked by other objects or the ground.

The PATHS model uses the notion of an object's *movability* to reflect whether it can be moved by a moderate force. To perceive movability, the model performs a physics simulation in which it continuously pulls on a string that it attaches to the center of the object in question. The change – or

lack of change – in position when pulling with a moderate force is used to judge the movability. Figure 8 shows four scenes that differ in whether the small object can be lifted up.

2.3 SPATIAL RELATIONS

How the objects in a scene are positioned relative to each other captures much about what is going to happen, what might have happened before, and what the objects in the scene might be used for. It is often essential for understanding what a scene is about and a model of physical understanding or a PBP solver will need to perceive and work with spatial information.

Spatial information in a scene includes the object's positions relative to the scene and relative to each other. The relative position of a *target object* in relation to a *reference object* can be described in terms of distance and direction and will depend on the objects' shapes, especially if they are close to each other. All directional spatial relations additionally require a reference frame in order to determine what is *left*, *right*, and so on. In the case of 2D scenes like in the PBPs, the direction of gravity provides us with the y-axis and the first direction of the reference frame. Placing the x-axis perpendicular to it we arrive at the classical Euclidean 2D space.

Related Work

Space and time are fundamental concepts in our lives and various scientific areas have made contributions to the questions of how spatial and temporal information can be perceived, represented, and reasoned upon. There has been active research on representing and reasoning on qualitative spatial and temporal data in the field of A.I. spanning from at least the 80s with the temporal intervals of Allen [1983] up to the present. The focus has been on formulating representations and ontologies that express temporal-spatial knowledge as well as calculi that allow automatic reasoning on a qualitative level. The work has been practically applied to handle queries in geographic information systems. However, building an ontology of space and time that is compact, complete, consistent, and computationally well-behaved turned out to be a very challenging task. Cohn and Hazarika [2001] and Vieu [1997] give a good overview of the field and its development. There are several dimensions along which we organize space and time concepts: topology (space: contains, touches; time: during, ends-with), distance (space: close, far; time: long ago, shortly after), orientation, direction, or order (space: left of, above; time: before, after), position (space: left, right; tomorrow, early), size (space: small, big; time: short, long), as well as the concept of shape (space: round, square; time: –). Due to the richness of spatial concepts, most work in qualitative spatial reasoning focuses on just one of the aspects listed above. Also, each approach has to make the choice

whether the basic elements are treated as points, as by Freksa [1992], or as regions, as in the region connection calculus of Randell et al. [1992]. Many spatial calculi are computationally intractable (NP-hard) in their full form, so an important task has been to identify tractable subsets (Renz and Nebel [2007]). There are approaches that combine spatial and temporal reasoning like the spatial-temporal constraint calculus, which combines Allen's interval calculus with RCC-8 (Gerevini and Nebel [2002]). Other calculi allow reasoning about moving objects (Van de Weghe et al. [2006]).

For our purpose of modeling solving PBPs, we are interested in qualitative spatial representations, not the automatic logical inference-making on the representations. Additionally, we need spatial representations that address all of the dimensions of spatial concepts mentioned above and a representation that can capture the vagueness that is inherent in many spatial concepts. For example, the PATHS model should be able to perceive differences in the degree of closeness of two objects, like very close and close. This makes it possible to either use the degree of closeness in a categorization rule directly or to adjust a threshold value used to make a binary decision about closeness depending on the context that is set by the PBP scenes.

Of the spatial representation research that I reviewed, Isabelle Bloch's work on fuzzy spatial relations fits these requirements best (Bloch [1999], Hudelot et al. [2008]). Bloch and colleagues developed a powerful framework that allows to model the inherent vagueness of spatial relations like "to the left of X" and the duality of spatial relations like "left" and "right". It has a robust way of combining spatial concepts such as "close above", and modifying them, such as in "very far". Concepts don't have to be exhaustive or mutually exclusive (the notions of above and left clearly overlap for some objects). It has only a few parameters that must be tuned. Finally, the approach takes the shapes of both reference and target objects into account.

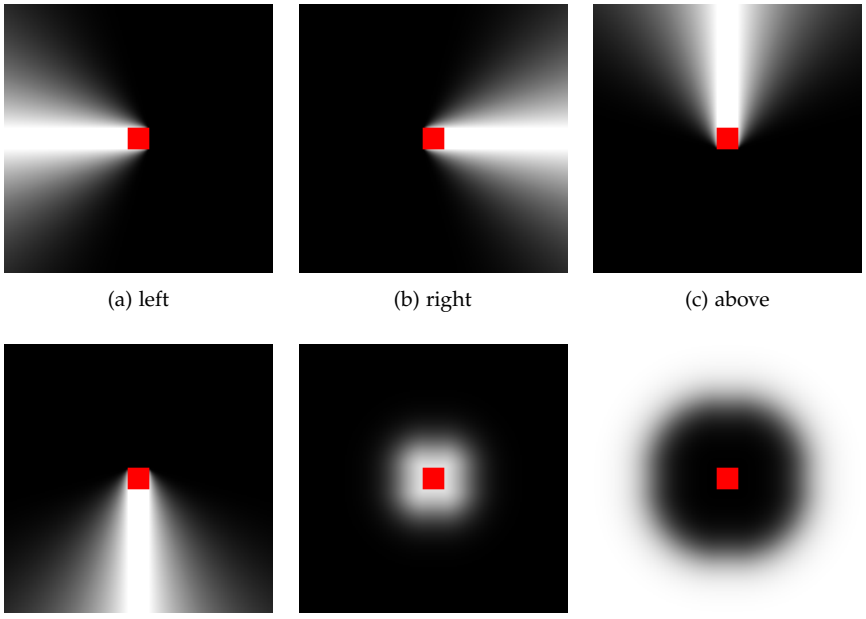
Fuzzy Spatial Relations

Isabelle Bloch's model of spatial relations is based on fuzzy set theory and has strong ties to morphological operations. Hudelot et al. [2008] introduce an ontology of spatial relations covering several aspects including distance, orientation and topology. The ontology and image processing algorithms were developed within a computer vision context and applied to medical images in which a priori qualitative spatial knowledge from an expert system had to be incorporated for the segmentation and interpretation of brain scans. All fuzzy operations work directly on the image space, therefore all objects and relative positions are sampled at a specific resolution. The computed spatial concepts can be used to answer two different questions.

1. To what degree does a given spatial relation hold for two specific related objects, for example, is A left of R?

2. At which locations is a given spatial relation fulfilled for a specific reference object, for example, which places in space are left of R?

The answer to the second question is provided by calculating a fuzzy set, referred to as fuzzy landscape, around the reference object in the same image space that the object is in. The fuzzy set is a function $\mu : \mathcal{S} \to [0, 1]$, that maps each point in the image plane $\mathcal{S}$ onto a membership value between 0 and 1. This membership value corresponds to the satisfaction of the spatial relation in question. Figure 9 shows the fuzzy landscapes of the six basic spatial relations using a square as the reference object.



(a) left (b) right (c) above

(d) below (e) close (f) far

Figure 9: The fuzzy landscapes of six basic spatial relations for a square. The brightness reflects the degree to which a position in the image space satisfies the respective relationship to the square.

To answer the first question, the fuzzy landscape around R is compared to the object A. The degree of relationship membership between A and R is measured by the relationship satisfaction values at the positions in the landscape that are covered by A. Isabelle Bloch uses three values to represent the fuzzy relationship between A and R: the minimum satisfaction value $\prod$, the mean satisfaction value M and the maximum satisfaction value N over all points of A in R for the relative position along the direction α.

$$\prod_{\alpha}^{R} = \min_{x \in A} \mu_{R,\alpha}(x)$$

$$M_{\alpha}^{R} = \frac{1}{|A|} \sum_{x \in A} \mu_{R,\alpha}(x)$$

$$N_{\alpha}^{R} = \max_{x \in A} \mu_{R,\alpha}(x)$$

In the fuzzy set framework, the minimum and maximum satisfaction measure can be interpreted as the necessity and possibility of A and R being in relative direction α, respectively. The definitions shown here are for the case of crisp objects, where each point in the image plane does or does not belongs to an object. This is sufficient for the PATHS model. The notions above generalize naturally to fuzzy objects, which is how Isabelle Bloch introduces them.

Two advantages of this approach are its solid mathematical grounding in fuzzy sets theory and fuzzy morphological operators, and its flexibility. The relative position of two objects with regard to a chosen angle can be calculated quite efficiently in 2D and 3D to measure concepts such as "left" and "above", while morphological structuring elements can be designed for other relations such as the distances "close" and "far". Finally, by using the fuzzy t-norm and t-conorm, conjunctions and disjunctions of spatial maps can be easily calculated in order to construct combinations of spatial concepts such as "far above" and "beside".

I will first describe the algorithm used to calculate the fuzzy landscapes as it was proposed by Bloch [1999] and then describe my improvements to that algorithm. The algorithm is applied to three pairs of relative positions "left of, right of", "above, below", and the distance relations "close, far". These are the basic relations the PATHS model uses for perceiving spatial information. Additional spatial relations such as "beside" or "on-top-of" are modeled as combinations of these basic relations. I follow the bipolar fuzzy sets approach of Isabelle Bloch, in which the fuzzy landscape of each basic spatial relation is complemented by the landscape of its opposing relation. Two concepts are combined by taking the fuzzy union of the positive information and the fuzzy intersection of the negative information. The intersection of two concepts is computed by intersecting the positive information and taking the union of the negative information.

Fuzzy Landscape Algorithm

Isabelle Bloch presents a slow exact and a fast approximate algorithm for calculating fuzzy landscapes for relative positions. The fast algorithm iterates over each pixel of the image space with the reference object at its center twice, first in a forward direction (left to right, top to bottom), then in a backward direction (right to left, bottom to top). The algorithm works in two stages. The goal of the first stage is to find the point in the reference object that is best aligned with the target direction seen from each point in the image space. This is accomplished during the two iterations over all pixels by setting each pixel to the best reference point of its 8 neighboring pixels (or its own reference point, in case it is already the best). Points for which no reference point is set yet are ignored when choosing the best point. Figure 10 shows the best matching reference point for a given point in the image space and the spatial relation "right-to".

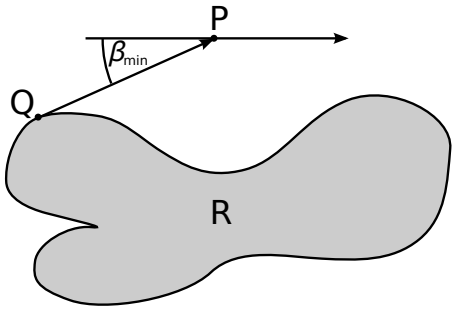


Figure 10: Q is the best reference point for the image point P, since it is the point in the reference object R that is closest to the spatial direction "right".

The different spatial relations are realized by using different functions $\beta : \mathbb{R}^2 \to \mathbb{R}$ that maps the relative position of a reference point $Q \in R$ and a point in S onto a real value expressing how far the relative position of both points matches the spatial relation tested for. I use the following function for the relations "left of", "right of", "above" and "below".

$$\beta_\alpha(P, Q) = \arccos \frac{\overrightarrow{QP} \cdot \vec{u}_\alpha}{\left\|\overrightarrow{QP}\right\|},$$

where $\vec{u_\alpha}$ is the unit vector in the respective direction. For the relations "close" and "far", I use the Euclidean distance between the points

$$\beta_{dist}(P, Q) = \left\|\overrightarrow{QP}\right\|.$$

After iterating over the image two times, each point in S has an (approximately) optimal reference point $Q \in R$ attached and the respective value of β. In the second stage of the algorithm, this value is mapped onto an acceptability value between 0 and 1 by using the following mapping for the relative directions and distances, respectively.

$$f_\alpha(x) = \max\left(0, \left(1 - \frac{2\|x\|}{\pi}\right)^3\right) \tag{1}$$

$$f_{close}(x) = 1 - \frac{1}{1 + e^{a*(b-x)}} \tag{2}$$

$$f_{far}(x) = \frac{1}{1 + e^{a*(c-x)}} \tag{3}$$

During the interpretation of a PBP, an algorithm has to look at up to 16 scenes, each of which might contain several objects with potentially important spatial relations between any object pair. An important property of the algorithm is that the most promising features are looked at first and an exhaustive perception of all possible relationships between objects is avoided. Still, the number of promising object pairs can easily be on the order of 100, so the calculation of the six basic spatial relations requires the PATHS model to calculate up to several hundred fuzzy spatial relations including the fuzzy landscapes they are based on. The performance bottleneck in calculating a fuzzy spatial relationship between object A and reference object R

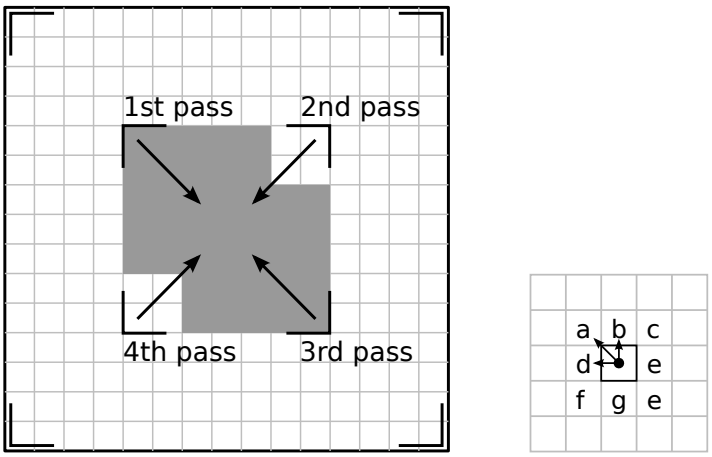
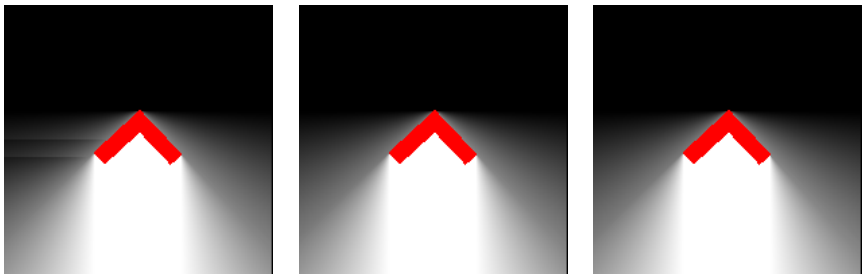


Figure 11: Illustrates the modified algorithm for fast computation of the spatial landscapes. Each of the four passes starts at one corner of the reference object's bounding box and proceeds to the opposite corner of the image space. The right figure shows which neighbor pixels are taken into account in the first pass.

is the calculation of the fuzzy landscape. Its time complexity is $O(\|\mathcal{S}\|)$, the number of pixels in the image space. Choosing the resolution is a trade-off between accuracy and speed of computation. The image space must be large enough so that for a reference object R the object A is still contained inside it, no matter where A was placed in the scene. Therefore, I scale the objects by such a factor that the width and height of the actual scene fit twice inside the image space and place the reference object at its center. I decided to use a resolution of 100 x 100 points.

In the original version of the fuzzy landscape algorithm, it iterates twice over the whole image space and takes into account the complete 8-neighborhood for each pixel. The first pass iterates from the upper left corner to the lower right; the second pass iterates in reverse. I modified the original algorithm based on the observation of how the information "flows" through the image during the iterations. Instead of two complete passes, I use four partial passes: one from the upper left to the lower right, one from the upper right to the lower left, one from the lower right to the upper left, and the last one from the lower right to the upper left. Instead of starting in the corners of the image space, the passes start at the respective corners of the reference object's bounding box. This simple modification reduces the number of pixels that have to be looked at by a factor of up to 2, depending on the size of the reference object. Additionally, in each pass only those three neighboring pixels in the 8-neighborhood of the current pixel are used, which are in the direction opposite to the direction of the current pass, reflecting the direction of information flow. This reduces the number of pixel checks by another factor of about 3. See Figure 11 for an illustration of the algorithm.



(a) original method (b) accurate method (c) improved method

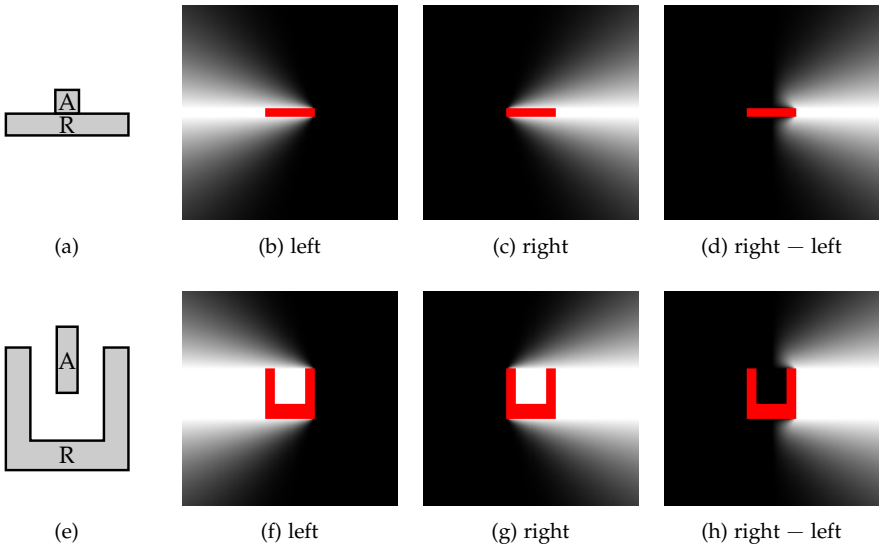
Figure 12: Side by side comparison of the original, the improved algorithm, and the accurate algorithm. The improved version is about five times faster and shows fewer artifacts than the original version.

The advantages of my adaptation of Isabelle Bloch's algorithms are twofold. First, we get a performance gain close to a factor of 6 when the reference object is small in relation to the total image space, which is typically the case. The second advantage is an increased accuracy of the fuzzy landscape, resulting from using four instead of two passes. Figure 12 demonstrates reduced artifacts for an example situation. While I made no attempt to prove that the modified algorithm is always at least as accurate as Isabelle Bloch's original method, across all test cases I looked at the results of the modified algorithm were equally close or closer to the results of the accurate method.

The fuzzy landscape for a specific relation has to be calculated only once for each reference object, as long as its rotational orientation in the scenes remains constant. After calculating the fuzzy landscapes for all basic relations, additional concepts can be defined by combining them.

Bipolar Fuzzy Landscapes

Bloch [2010] describes how to extend the fuzzy spatial framework to include bipolar information. The notion of having a fuzzy set $\mu : \mathcal{S} \to [0, 1]$ is extended to a bipolar fuzzy set, which is a pair of membership functions (μ, ν) where μ captures the positive information of where a spatial relation is known to be satisfied and ν captures the negative information of where a spatial relation known to be unsatisfied. There are two common applications for doing so. First, in many spatial search problems, constraints on possible positions are known and these can be captured by the second part of the bipolar fuzzy set. Second, and more relevant in the context of PBPs, is the case of opposite relations like left and right, above and below, as well as near and far. The opposition in these pairs does imply a form of symmetry, but the parts are neither mutually exclusive nor jointly exhaustive. There are, for example, image regions that are neither left nor right of a reference object while other regions might be both left and right of different parts of a concave object.



(a) (b) left (c) right (d) right − left

(e) (f) left (g) right (h) right − left

Figure 13: Two example situations of an object A placed relative to an object R. If I only considered the fuzzy landscape of the relation *right of*, object A would be described as *right of* R in both cases, which is counter-intuitive since it is too permissive. In a bipolar interpretation of the relation *right of*, I also take the opposing relation *left of* into account as negative information. This yields a more intuitive representation.

The conceptual part of the cognitive model will not work directly on the fuzzy landscapes, but on the qualitative spatial relationships between objects, represented as a single membership value between 0 and 1. To come up with the acceptability or satisfaction value for a concrete relation between two objects, the negative and positive information of the respective bipolar fuzzy landscape have to be combined. This is done by subtracting the negative landscape from the positive landscape at each corresponding point in the fuzzy set and clipping all values below zero. The related object is then placed in this combined landscape at the same relative position and rotation as in the real scene and its mean membership is used as acceptability. See Figure 13 for two examples. Notice that without taking the negative information into account, both examples would lead to a high acceptability of both "A left of B" and "A right of B", which would be counter-intuitive.

This bipolar nature of many spatial relations is captured well by the bipolar fuzzy set theory. All fuzzy and morphological operations generalize nicely to the new setting. What is of interest to us is the fusing of two bipolar fuzzy sets. Taking the conjunctive fusion of two bipolar fuzzy sets (μ_1, ν_1) and (μ_2, ν_2) is done by taking the conjunction of the positive parts $\mu_1, m u_2$ and the disjunction of the negative parts ν_1, ν_2 as negative evidence. The disjunctive fusion of two bipolar fuzzy sets is defined as the disjunction of the positive information and the conjunction of the negative information.

$$(\mu_{and}, \nu_{and}) = (\mu_1 \cap \mu_2, \nu_1 \cup \nu_2)$$
$$(\mu_{or}, \nu_{or}) = (\mu_1 \cup \mu_2, \nu_1 \cap \nu_2)$$

The conjunction $\mu_\cap$ and disjunction $\mu_\cup$ of two fuzzy sets μ_1 and μ_2 are defined as

$$\mu_\cap(i,j) = \top(\mu_1(i,j), \mu_2(i,j))$$
$$\mu_\cup(i,j) = \bot(\mu_1(i,j), \mu_2(i,j)),$$

where $\top$ is the fuzzy t-norm and $\bot$ is the fuzzy t-conorm. In the literature on fuzzy mathematics, there are several well-established fuzzy logic systems with their respective t-norms and t-conorms. Three of the most common ones are the Łukasiewicz logic (Łukasiewicz t-norm and bounded sum t-conorm), the Gödel logic (minumum t-norm and maximum t-conorm) and the product logic (product t-norm and probabilistic sum). Following Isabelle Bloch's example, I will use the minimum t-norm and maximum t-conorm in the PATHS model.

$$\top_{min}(a,b) = \min(a,b)$$
$$\bot_{max}(a,b) = \max(a,b)$$

At the time of calculating an object's membership to a spatial relation with a reference object, I combine the respective positive and negative information of the bipolar fuzzy landscape using the function

$$sub(a,b) = \max(0, a-b).$$

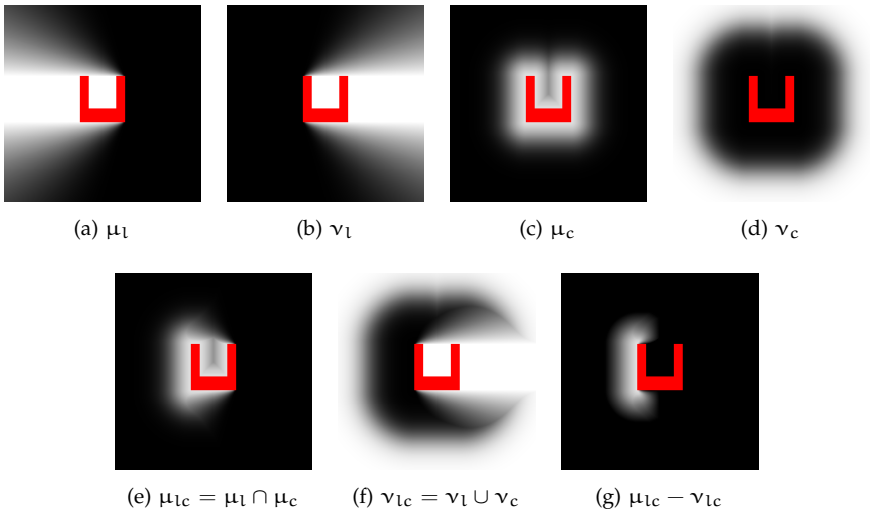
This is the only step that falls slightly outside the mathematical system of fuzzy logics, as "subtraction" in fuzzy terms is normally defined as $a - b := \top(a, \neg b)$. The negation is defined as $\neg a = 1 - a$ for the Łukasiewicz logic and as $\neg a = 1$ if a is 0, otherwise 0. Therefore, the above formula for subtraction is compatible to the Łukasiewicz logic and not for the Gödel logic, where the compatible subtraction is $a - b = a$ if $b = 0$, else 0. In my experience, the above definition works very well and is less drastic than the Gödel subtraction. Using a different subtraction method is unproblematic since the subtraction is only done as a last computation step after all fuzzy operations have been applied.

Combining Spatial Concepts

LEFT AND CLOSE Using the formulas above, I can easily define the spatial concept *left and close* as a bipolar fusion of the concepts *left* and *close*.

$$(\mu_{lc}, \nu_{lc}) = (\mu_l \cap \mu_c, \nu_l \cup \nu_c),$$

were the indices lc, l, and c stand for *left and close*, *left*, and *close*, respectively. The negative information for *left*, ν_l, and for *close*, ν_c, are identical to the



(a) μ_l (b) ν_l (c) μ_c (d) ν_c

(e) $\mu_{lc} = \mu_l \cap \mu_c$ (f) $\nu_{lc} = \nu_l \cup \nu_c$ (g) $\mu_{lc} - \nu_{lc}$

Figure 14: Combination of the spatial relations *left* and *close* for a U-shaped reference object. The first four plots show the positive and negative information of *left* and *close*. The landscapes (e) and (f) are the results of the bipolar conjunctive fusion. The final representation (g) is the difference of both and is used to calculate the degree of membership for a given object to the spatial relation *close and left of the U-shaped object*.

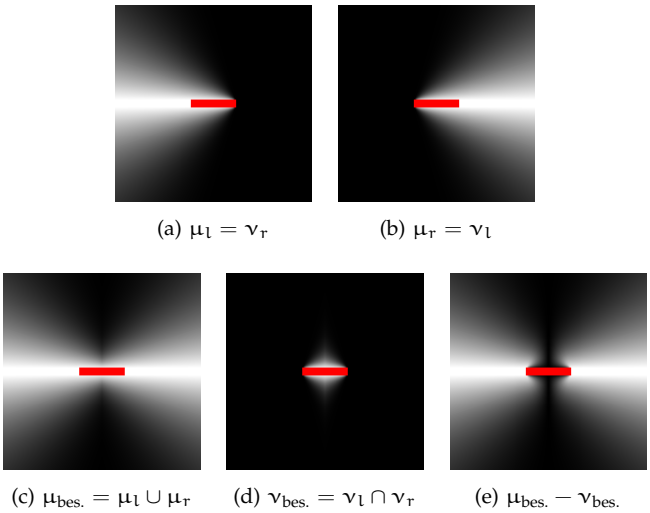
right and *far* relations, respectively. Figure 14 depicts the fuzzy spatial landscapes used in the construction of the relation *close and left*. In the last plot of this figure, the positive and negative information of the bipolar representation are combined in the same way they are combined when calculating the membership of an object to the relation and the given reference object. This representation captures the intuitive meaning of *close and left* much better than just using the positive information.

BESIDE The concept *beside* is introduced as a disjunctive fusion of the *right* and *left* concept.

$$(\mu_{bes.}, \nu_{bes.}) = (\mu_l \cup \mu_r, \nu_l \cap \nu_r)$$

See Figure 15 for the results. Again, incorporating the negative information of both *right* and *left* leads to a more plausible result than just using positive information.

ON-TOP-OF I model the meaning of the concept A *on-top-of* B as A being on top of B and touching B. Since *touches* is not defined as a fuzzy landscape but is fulfilled to a certain degree for two specific objects, *on-top-of* shares that property. I define *on-top-of* as the minimum of the acceptabilities of A *above* B and A *touches* B.



(a) $\mu_l = \nu_r$ (b) $\mu_r = \nu_l$

(c) $\mu_{bes.} = \mu_l \cup \mu_r$ (d) $\nu_{bes.} = \nu_l \cap \nu_r$ (e) $\mu_{bes.} - \nu_{bes.}$

Figure 15: New spatial relation *beside* defined as bipolar disjunctive fusion of *left* and *right*. The first two plots show the positive and negative information of *left* and *right*. The landscapes (c) and (d) are the results of the bipolar disjunctive fusion. The final representation (e) is the difference of both and is used to calculate the degree of membership for a given object to the spatial relation *beside the rectangular object*.

2.4 GROUP ATTRIBUTES

The PATHS model is capable of treating several objects in a scene as a group and can perceive attributes of that group. Objects can be grouped through common attributes, for example "all squares in the scene", through common relationships, for example "all objects left of a circle", or through combinations of attributes and relationships. The attributes *count*, *touching*, *close* and *far* can be perceived on a group.

The model's perceptual routine for determining the number of objects simply returns the correct integer as the attribute value. To decide whether the objects in a group are all close to each other, the model builds the minimal spanning tree (MST) on a fully connected graph that has the objects as nodes and their pair-wise distances as weights of the edges. An MST is defined as a subgraph that is structured as a tree, connects all nodes, and minimizes the sum of edge weights. The degree of membership of a group to the concept *close* is determined by mapping the value of the largest edge in the MST onto the interval $[0, 1]$, using the same mapping function as in the close relationship (see page 35).

The algorithm for perceiving the *touches* group attribute works in the same way with the only difference being that it uses the mapping of the touches relationship to translate the biggest distance in the MST into a membership degree.

Finally, the membership of a group to the concept *far* is computed by using the minimum of all pair-wise object distances in the group and the

mapping of the *far* relationship (see page 35). While the strategy for *close* and *touches* membership calculation is similar to the single-linkage criterion in hierarchical clustering and allows for long chains for objects, the strategy for *far* membership calculation resembles an inverse complete-linkage criterion and requires that there are no two objects that are close to each other.

2.5 CONCLUSION

In this chapter, I described how the PATHS model can perceive geometrical, spatial, dynamic and physical features in a scene. The features include attributes of objects and group of objects, as well as relationships of pairs of objects. The perceptual system of the PATHS model solves a number of challenges inherent in the task of extracting relevant information from PBPs. While its ability to simulate the unfolding of a scene and of imagined actions captures the predictive aspects of perception, its translation of perceptions into fuzzy concept memberships provides an important intermediate step towards building symbolic categorization rules without sacrificing the option of context-dependent reinterpretations. In the next chapter, I describe the rule-construction component of the PATHS model and how it can guide the perception of PBP scenes towards the most relevant features.

3

LEARNING PHYSICAL CONCEPTS

This chapter describes how the perceptual capabilities of the PATHS model laid out in the previous chapter are integrated with a hypothesis generation and testing mechanism. The PATHS model perceives features on objects, object pairs, and groups of objects in PBPs and uses these perceptions to construct structured rule-based representations of the scenes. The perception process and the process of hypothesizing about the correct categorization rule interact with and constrain each other, as the model focuses on the most promising hypotheses. PBP solutions take, as discussed in the first chapter, the form of rule-based concepts that contain all scenes on the left side while not containing any scene on the right side – or vice versa.

In order to focus on the particular aspect of interleaved perception and concept-formation, I make a number of simplifying assumptions in regard to other aspects of the problem domain. The model treats all given training examples as noise-free and deterministic in the sense that the category labels are correct and the positions and shapes of all objects are precisely known. There is no missing data in the common sense of the term, however, practically, the algorithm will have to deal with incomplete and (yet) missing data, as the instances are iteratively perceived. Finally, the PBP domain was constructed using concepts that can be described with a conjunctive rule that covers all positive examples. In this work, I am not interested in rules with complex logical structures and also won't deal with learning exceptions to a logical rule. Therefore, the rule language of the PATHS model wasn't designed to express rules beyond conjunctions of attributes and relationships.

The PATHS model is meant to provide insights at a process-level. It works iteratively on the scenes and is, just like humans, influenced by the order in which the examples are presented. The model is given the outlines and positions of all objects and the ground in the scenes as input; all features that are used in pattern descriptions or rule hypotheses have to be constructed by running perceptual processes on the input data.

3.1 GUIDING PRINCIPLES OF MODEL DESIGN

In the design of the PATHS model, I took inspiration from several models and theories in cognitive science. PATHS is close in spirit to the fluid analogy systems from Douglas Hofstadter's group, including Phaeaco by Foundalis [2006]. It, too, aims to capture the active, iterative nature of perception and its interaction with higher-level cognitive processes. Unlike the focus on structural mapping in the fluid analogy systems, the PATHS model uses a

hypothesis testing approach and treats PBPs as a concept learning problem. Existing rational models of concept learning, such as Goodman et al. [2008a], inspired the use of Bayesian estimation of hypothesis probabilities to drive the decisions made by the model. The following list describes the main principles and ideas that guided the implementation of the PATHS model.

HYPOTHESIS TESTING. The PATHS model learns rules that sort structured instances into categories through an active process of constructing and testing hypotheses. The model's rule space is restricted to conjunctions of object attributes, group attributes and object relationships; the rules can be all-, exists- and unique-quantified. While humans can work with more complicated logical rules, conjunctions are a natural subset and allow us to focus on challenges other than the construction of complicated logical structures.

ITERATIVE AND ACTIVE PERCEPTION. The processes of perceiving PBP scenes and constructing rule-based interpretations of them happen at the same time and influence each other. The model starts working on PBPs without knowledge of any object attributes or spatial relationships. Instead, using the visual outlines and positions of the objects, the model perceives features step by step in order to build scene descriptions and rule hypotheses.

This perception of attributes and relationships, which often involves mental simulations of physics, requires time and effort. Structured scenes with multiple objects contain many potentially features to look at. In order to learn efficiently, PATHS uses feature and object saliencies, as well as information from hypotheses-scene matches in its decisions of what to perceive.

RATIONAL DECISIONS. The PATHS model uses the information from previous hypothesis-scene matches to estimate how probable it is for a hypothesis or for a feature or object in the current scene to be part of a solution. Based on those estimates, it makes a stochastic decision on what to perceive or check next. This is at its core a rational approach – the model makes decisions that are estimated to be optimal in a given situation. Unlike classic rational approaches, the estimation of probabilities is relatively rough and the model takes memory constraints into account by only using the information that was actively perceived so far.

LOCAL ACTIONS. The PATHS model is only allowed to perceive features and check hypotheses on currently visible scenes. This is motivated by human memory constraints and, in particular, the difficulty of reinterpreting an image in memory (Finks et al. [1989]). As a result, the estimation of hypothesis probabilities has to take into account that often different hypotheses have been checked on a different number of scenes. Although the algorithm keeps track of all hypotheses that are created, typically just a few

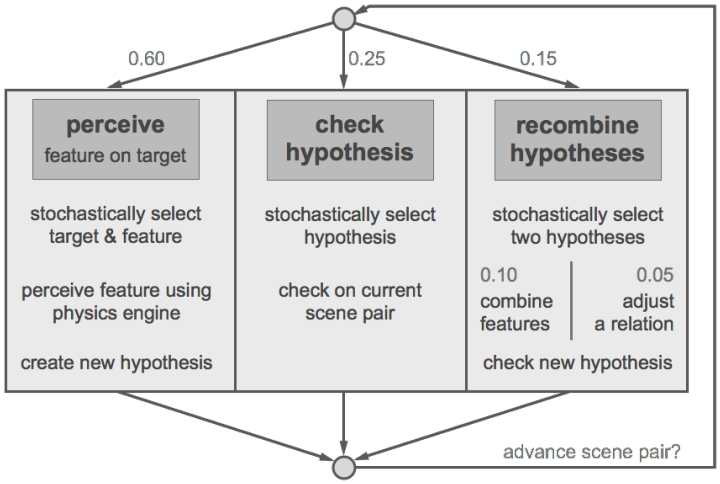


Figure 16: Control flow in the PATHS model. At each step, the model stochastically selects one of four action types with the indicated probabilities. The "recombine hypotheses" block in the diagram contains two similar action types that are selected with the probabilities 0.1 and 0.05. Some actions trigger the creation of a new hypothesis, which in turn triggers a check hypothesis action on the created hypothesis.

hypotheses are actively explored. Only when a promising hypothesis turns out to be wrong, attention is shifted to others.

GRADUAL CONCEPTS AND ADAPTIVE BORDERS. The membership of an instance to a concept is often gradual. For example, when perceiving how 'close' two objects are to each other, they can be very close, close or not so close. At the time the model formulates a rule based on feature membership values, these values are discretized into an all or nothing membership (close or not close). However, the value thresholds that distinguish between these membership states can be adjusted depending on the context. The same object might be called "small" in one context and "big" in another.

3.2 IMPLEMENTATION

The basic architecture of the PATHS model resembles a loop that is repeated until a solution is found. In each iteration through that loop, the model performs three steps. First, it stochastically selects the type of action to perform next from a set of four types. Second, it performs the action, which might involve the stochastic selection of action parameters. Third, it stochastically decides whether to shift attention to a new set of scenes. Some actions can trigger a follow-up action when they are run and in these cases, the model performs the follow-up action before continuing to step three. Figure 16 gives an overview of this architecture. The following text describes the various components of the PATHS model, important aspects of their implementation, and how they work together.

Scenes

The scenes hold physical representations of their objects. Initially, the model only knows about the outlines of the objects and the ground in a scene. Over time, more information is collected and stored through perceiving of object features. A physics engine is used to both predict how the scene unfolds over time and to perceive physical features on objects. Stability, for example, is perceived by observing how much an object moves after poking it.

The model also keeps track of the minimum and maximum feature values for perceived features. This information can later be used to adjust the thresholds of category membership for features.

Switching Between Active Scenes

One way of comparing the performance of the PBP cognitive model to human performance is to look at the influence of the order in which the scenes are perceived. To that end, I present just two scenes of a PBP at a time. The sequence of scene pairs is fixed during a particular trial while the decision of when to uncover the next scene pair is up to the human or cognitive model.

In the case of the cognitive model, this decision is based on how promising the current hypotheses are. If there is a promising solution candidate that was already checked on the current scenes, the model is more likely to move on to the next scenes earlier so the solution candidate can be further verified. If none of the hypotheses are particularly promising, or if a promising one yet has to be checked on the current scenes, the model is likely to continue looking at the current scenes.

Objects and Groups

Objects keep track of all perceptions that were made on them. Groups are constructed by selectors that select a subset of objects in a scene, such as "square objects" or "any object". Each group of objects can be a target for perceiving group features like "object count" and will keep track of all selectors that are known to select that subset of objects in the group.

Features and Percepts

The model currently has 34 built in feature detectors, including detectors for static object properties, such as "size" and "shape", physical properties, such as "stable" and "moves", spatial relationships, such as "left-of" or "close" and group attributes, such as "object count". Each feature detector can perceive its respective feature on any object (or group, whichever is appropriate) and the resulting percept contains the perceived value of the

feature as a membership degree between 0 and 1 (e.g., [A left-of B] = 0.4). The algorithm uses a default threshold of 0.5 to decide whether a feature is considered active or not (e.g, A is "left-of" B or not). This threshold can be adjusted in each selector.

Selectors

Selectors abstract from observations on a specific object or group and represent a structured pattern. They correspond to a possible interpretation of a scene by describing what to look for. The abstraction is both from specific objects to descriptions of those objects' features, and from gradual to discrete feature values. The percept of a feature like "small" is represented with a membership value in $[0, 1]$. When turning that percept into a *feature matcher*, the value is discretized into a binary value "true" or "false", so that the resulting selector can either match objects that are "small" or "not small". The default threshold for the binarization is a membership value of 0.5. This threshold can be adjusted per selector and feature type to account for the observed distributions of feature values in the left and right PBP scenes.

When a selector is applied to a scene, it selects a subset of the scene's objects. Selectors are conjunctions of three kinds of feature matchers: object attribute matchers, object relationship matchers, and group attribute matchers. Each feature matcher maps a source set of objects to a target set of objects that contains all those objects from the source set that have the same feature-value combination as the matcher. For example, a selector that contains a single object attribute matcher "small=true" will select all objects in a scene for which the feature "small" has a membership value above the threshold of 0.5 or an adjusted threshold.[1]

An object relationship matcher contains, in addition to the relationship type and value, a selector for the reference object of the relation. For example, "hits (small)" would select all objects in the scene that hit a small object. Reference selectors are not allowed to contain relationship matchers, which prevents complicated nested structures "a circle that is left to a square that is left to a triangle", but allows for "a circle that is left to a square and is left to a triangle".

When applying a group attribute matcher to a set of objects, it returns the original set if the group of objects in the set has the same feature-value combination as the matcher. Otherwise, it returns an empty set. For example, the matcher "count=3" will return all objects when applied to a scene with three objects or no objects if the scene has a different number of objects in it.

Since all matcher types map object arrays to object arrays, they can be easily combined. For example, a selector consisting of the two matchers

1 I will omit to write "=true" for active features when there is no danger of confusion.

"close-to(square) $\wedge$ count=3" will check whether the number of objects in a scene that are close to a square is three. If so, it selects those three objects, otherwise, it selects none.

If during the application of a selector the values for the selector's features are not yet known for some objects, the model will perceive them first.

Hypotheses

A hypothesis represents a potential solution or partial solution to the current PBP. It consists of a selector, the side of the PBP the selector is describing (left, right or both) and the quantifier that is used with the selector (exists, all or unique). For example, the "small" selector could be applied as "in the left scenes, all objects are small", or as "in the right scenes, there is a small object", or as "in all scenes, there is exactly one small object". Each hypothesis keeps track of whether or not the scenes it has been tested on matched and whether all, some, or exactly one of the objects in each scene matched. This information is used to pick the best fitting side and quantifier for describing the scenes seen so far.

In the model, every hypothesis is associated with exactly one selector, and every selector is associated with exactly one hypothesis. I will use both terms interchangeably when there is no danger of confusion.

Attention Mechanisms

In the PATHS model, the guiding of perception towards regions and aspects of the scenes that are most relevant to the learning process works on several levels. Attention can be shifted to certain objects in a scene, to certain features, or to the aspects necessary for checking a solution hypothesis. During the initial exploration of the PBP scenes, objects differ in their saliency based on their features. Objects that are about to move or are unstable, objects that are spatially separated, or objects that are "oddballs" in another way have a higher probability of being attended to. After perceiving a couple of the scenes, the model will have noticed some reoccurring patterns and will have created some hypotheses. The hypotheses influence the choice of what to perceive next in several ways. First, during the exploration of the scenes, the model is more likely to attend to the objects that play a role in promising solution hypotheses. Second, the model directly checks existing hypotheses on new scenes and during this process only perceives what is necessary to confirm or refute the hypothesis. Third, existing hypotheses can be combined, and the resulting hypotheses influence perception as described in the previous two points.

In summary, what aspects of a PBP scene the model focuses on depends on the a priori salience of features and objects in the scene, as well as information about common patterns gathered from other scenes in the form

of hypotheses. The goal of attending to certain aspects of a scene and ignoring others is to maximize the expected usefulness of new information towards the goal of finding the rule that discriminates between the two categories shown in the PBP. Ideally, the PATHS model will find a solution to a PBP without having to perceive all, or even the majority, of the potentially available features.

Actions

All work done by the model while solving a PBP is organized into small, separate chunks, called actions. A practical way of measuring the model's performance that abstracts from specific computer hardware and implementation details is to count the number of actions that the model performed to solve a PBP. There are four action types that the model can perform, which are: 1) to perceive, 2) to create a hypothesis from a perception, 3) to check a hypothesis against the current scenes, and 4) to combine two existing hypothesis to form a new one. Three of the four action types are triggered top-down by sampling from a fixed multinomial distribution. The perception action is chosen with $p = 0.6$, a hypothesis is checked with $p = 0.25$ and two hypotheses are combined with $p = 0.15$. See Figure 16 for a diagram of the model's control flow. Actions can also be triggered by other actions. A typical example is the perception action triggering a create hypothesis action, which in turn triggers a check hypothesis action to turn a perception into a selector and hypothesis that is then applied to the currently active scenes. Following is a description of each of the action types.

Perception Action

The *perception action* uses one of two strategies with equal probability. Either the action first selects a target from one of the current scenes and then selects which new feature it should perceive on it, or it first selects a feature and then selects a new target for perceiving the feature. The former strategy corresponds to a human looking at an interesting target in a scene and perceiving new properties of it, while the latter corresponds to a human that is looking for targets that have a particularly interesting feature.

In both cases, the target can be an object or a group of objects while the feature can be a group attribute, object attribute, or a relationship between two objects. Both target and feature are stochastically chosen based on their estimated probability of being part of a solution to the PBP. If an object relationship is to be perceived, a reference object for the relationship is chosen in addition to the target object. For features that can change over time, such as an object's position in the scene, the perception action stochastically picks whether the feature is perceived in the initial or final situation, based on a fixed multinomial distribution ($p = .67$ and $p = .33$, respectively).

When perceiving a feature on a target, the model only selects feature-target combinations that were not *actively* noticed before. This biases the model towards the gathering of new information during perception over the revisiting of old information. I consider noticing a feature of a target as *passive* if it happened as part of checking a hypothesis on a scene or while assessing basic feature values in order to perceive a complex feature. For example, checking whether a A is "touching" B in order to find out whether A is "on-top-of" B counts as active noticing of "A on-top-of B", but only as passive noticing of "A touching B". This distinction between active and passive perception reflects whether the focus during checking the feature was on the feature itself or on something else. In other words, it reflects whether someone is implicitly aware or becomes explicitly aware of an aspect in a situation.

If something new is perceived during the perception action, a *hypothesis-creation action* is triggered and turns the percept into a corresponding hypothesis in the next step. See Algorithm 3.1 for a pseudo-code implementation of the perception action.

Algorithm 3.1 Perceive Action

```
procedure RUN_PERCEIVE_ACTION(scene)
    strategy ← pickAtRandom("feature-1st", "target-1st")
    if strategy is "feature-1st" then
        feature ← chooseFeature()
        target ← chooseNewTarget(scene, feature)
    else if strategy is "target-1st" then
        target ← chooseTarget(scene)
        feature ← chooseNewFeature(target)
    end if
    return if (no feature) or (no target)
    time ← chooseTime()
    if feature is a relationship then
        reference_obj ← chooseNewReference(feature, target)
        percept ← feature.perceive(target, reference_obj, time)
    else
        percept ← feature.perceive(target, time)
    end if
    new CreateHypothesisAction(percept)
end procedure
```

The *chooseFeature* and *chooseTarget* methods make a stochastic choice among all features and targets in the current scene, respectively, based on the estimated probability of a feature or target being part of the solution. The *chooseNewTarget*, *chooseNewFeature* and *chooseNewReference* methods make the same kind of stochastic choice, but only among those objects, groups, or features that will lead to new observations. The *chooseTime* method picks "start" with $p = .67$ and "end" with $p = .33$.

Create-Hypothesis Action

The create-hypothesis action turns a percept into a pattern description that can be applied to new scenes. I refer to the pattern descriptions as selectors and each selector is associated with one hypothesis – a potential solution to the PBP that keeps track of all matching results. The create-hypothesis action is triggered through a perception action which passes the new percept of a group's or object's feature to it. It creates a new hypothesis and selector that matches objects with the same feature–value combination as in the percept.

In cases of perceived object attributes, this process is straightforward, while in cases of object relationships and group attributes additional work is necessary. An object relationship is always perceived between two specific objects, so in order to turn a relationship perception into a selector that can be applied to new scenes, the reference object needs to be represented via a selector, itself. Either the "any object" selector or some more specific selector among the existing compatible hypotheses in the workspace might be picked. The reference selector is embedded into the main selector, which describes the relationship feature and value.

If the percept contained a group attribute perceived on a specific group in the scene, a selector that matches this group is picked and combined with the main selector, which describes the perceived group attribute's type and value.

After the new selector and its associated hypothesis are created, they are added to the workspace unless an identical selector already exists. In either case, a check-hypotheses action is triggered for the hypothesis.

Check-Hypothesis Action

The check-hypothesis action applies an existing hypothesis to the scenes in the current scene pair and keeps track of the results. Compatible match results will contribute positively to the estimated potential of the hypothesis while incompatible match results will have the opposite effect. In addition to whether the hypothesis matched the scenes or not, the model keeps track of whether exactly one, or a few, or all of the objects in the scenes match the hypothesis' selector. This information is used to decide on the best logic quantifier ("unique", "exists" or "all") to use in the hypothesis.

Depending on the match results from all scenes a hypothesis was tested on so far, the hypothesis might or might not be a potential solution by itself. Logically, a hypothesis can only be a solution if it so far matched all tested scenes from one side and none of the tested scenes from the other side. If that is not the case, the hypothesis might still be useful since it captures a recurring pattern in the scenes and can contribute to a solution when combined with other hypotheses.

Each time after checking a hypothesis on a new pair of scenes, the action will pick the side and quantifier for the hypothesis that best fit all previous matching results. For example, "in the left scenes, there is a small object"

Algorithm 3.2 Create-Hypothesis Action

```
procedure RUN_CREATE_HYPOTHESIS_ACTION(percept)
    if percept contains an object attribute then
        selector ← createAttrSelector(percept.feature)
    else if percept contains a group attribute then
        group_sel ← chooseSelector(percept.target)
        selector ← createAttrSelector(percept.feature)
        selector ← combine(selector, group_sel)
    else if percept contains an object relationship then
        reference_sel ← chooseSelector(percept.reference)
        selector ← createRelSelector(percept.feature, reference_sel)
    end if
    hypothesis ← createHypothesis(selector)
    addToWorkspaceIfNew(hypothesis)
    new CheckHypothesisAction(hypothesis)
end procedure
```

CreateAttrSelector creates a selector that selects object or groups with the specified attribute, e.g. "is-small" or "has-count=2". *CreateRelSelector* creates a selector that selects objects with the specified relationship relative to any of the objects matched by the specified reference selector. *ChooseSelector* stochastically picks a selector that matches the specified group or object.

could be changed to "in the right scenes, all objects are small". The goal of these adjustments is to find a hypothesis that only matches scenes from one side and is therefore a potential solution. If that is not the case anymore after checking a scene pair, the check-hypothesis action attempts to "repair" the hypothesis by readjusting the concept-membership thresholds of the selector's feature matchers. For example, the selector might be adjusted to accept a larger range of object sizes as being "small". This adjustment of thresholds models the flexibility of a human solver in how he or she applies labels to object properties.

The check-hypothesis action can be triggered through an action that created a new hypothesis which now should be tested on the current scenes. Alternatively, it can be triggered by the model directly without a specific hypothesis to be checked in which case the action makes a stochastic choice among all hypotheses that were not yet checked on the currently active scenes.

Combine-Hypothesis Action

There are two sub-types of actions that both combine existing hypotheses to create new ones. The first one, described here, is the combine-hypothesis action and is two times more likely to be selected than the recombine-hypothesis action described in the next section.

Algorithm 3.3 Check-Hypothesis Action

```
procedure RUN_CHECK_HYPOTHESIS_ACTION(hypothesis, scene_pair)
    if no hypothesis was passed then
        hypothesis ← chooseUncheckedHypothesis(scene_pair)
    end if
    hypothesis.checkScenes(scene_pair)
    hypothesis.adjustQuantifierAndSide()
    if hypothesis matches scenes from both sides then
        hypothesis.tryToAdjustThresholds()
    end if
    if hypothesis is a solution then
        announceSolutionAndStopSearch(hypothesis)
    end if
end procedure
```

ChooseUncheckedHypothesis() stochastically selects an existing hypothesis that was not yet checked on the specified scene pair. Hypotheses with higher estimated utilities are more likely to be selected. *Hypothesis.checkScenes()* applies a hypothesis to the specified scenes and saves the matching results in the hypothesis. *Hypothesis.adjustQuantifierAndSide* selects the main side (left or right) and quantifier (exists, all, unique) that fits the observed matching results best. *Hypothesis.tryToAdjustThresholds()* checks whether modifying the thresholds of the feature matchers used in the hypothesis can make the hypothesis match scenes only from one side of the PBP and, if so, performs the adjustments.

The combine-hypothesis action is triggered directly by the model in a top-down fashion. The action stochastically selects an object from one of the active scenes and picks two hypotheses that match the selected object and were not combined before. Only hypotheses that match scenes from both sides are considered, since combining hypotheses always results in a more specific hypothesis, and hypotheses that have so far matched scenes from only one side are still sufficiently specific. A check-hypothesis action is triggered for the hypothesis created by the conjunction of the two.

The reason to select two hypotheses through a common object they select is to ensure that they aren't incompatible with each other: their conjunction will select at least one object in one of the scenes. This roughly corresponds to a human noticing that the same object is both small and stable, so in addition to the solutions "there is a small object" and "there is a stable object", he or she might now consider the solution "there is a small and stable object".

Recombine-Hypothesis Action

Just like the combine-hypothesis action, the recombine-hypothesis action gets triggered by the model directly and creates new hypotheses by combining parts of existing ones. Instead of combining two complete hypotheses, though, it swaps the reference object selector of a relationship feature in one

Algorithm 3.4 Combine-Hypothesis Action

procedure RUN_COMBINE_HYPOTHESIS_ACTION(scenes)
  object ← chooseObject(scenes)
  hyp_A ← chooseHypothesis(object)
  hyp_B ← chooseOtherHypothesis(object, hyp_A)
  hyp_AB ← combineHypotheses(hyp_A, hyp_B)
  new CheckHypothesisAction(hyp_AB)
end procedure

ChooseObject() stochastically selects an object from the specified scenes. *ChooseHypothesis()* stochastically selects a hypothesis that matches scenes from both sides and whose selector matches the specified object. *ChooseOtherHypothesis()* does the same, but restricts the hypotheses to those that are compatible with the specified one. *CombineHypothes()* returns the conjunction of the two specified hypotheses.

hypothesis with a different selector. For example, based on the hypothesis "there is a square on top of an object", a new hypothesis "there is a square on top of a big object" can be created if there is an existing "big objects" selector.

Specifically, the recombine-hypothesis action first stochastically selects a hypothesis that has a relationship feature and one of the objects in the current scene that matches the relationship's reference object selector. Then, the action searches for a different selector selecting this reference object and, if successful, creates a new hypothesis that is a copy of the original one with the relationship reference selector replaced by the new selector. Finally, a check-hypothesis action is triggered for the newly created hypothesis.

This action can be essential to finding a solution in cases where the model has perceived the correct relationship feature for the solution of the problem but initially picked a reference selector that is incompatible with the solution.

Algorithm 3.5 Recombine-Hypothesis Action

procedure RUN_RECOMBINE_HYPOTHESIS_ACTION(scenes)
  hyp ← chooseHypothesisWithRel()
  rel_idx ← Random.int(hyp.rels.length)
  rel ← hyp.rels[rel_idx]
  object ← chooseObject(scenes, rel.reference_selector)
  sel ← chooseSelector(object)
  hyp_new ← clone(hyp)
  hyp_new.rels[rel_idx] ← sel
  new CheckHypothesisAction(hyp_new)
end procedure

ChooseHypothesisWithRel() stochastically selects a hypothesis that is based on a relationship feature. *ChooseObject()* stochastically selects an object in the given scene that matches the given selector. *ChooseSelector()* stochastically picks a selector among all existing selectors that match the specified object.

3.3 UTILITY ESTIMATION

Whenever the model is selecting a hypothesis to check or combine, or an object and feature to perceive, the choice is made stochastically based on previous hypothesis-scene matches. If, for example, a "circle objects" hypothesis captures that the model has so far only seen circles in the scenes on one side, this gives some credibility to the idea that circles might play a role in a solution to the problem. The PATHS model's choices are based on the estimated probabilities that any of the current features, objects, groups or hypotheses is part of, or represents, a solution to the PBP. I will refer to these probabilities of being useful as *utilities*. Since information about the scenes is gathered iteratively, the estimation of utilities needs to take into account that only partial knowledge is available at any point in time.

In the following, all hypothesis-scene matching results are summarized in a match matrix M. The columns correspond to hypotheses, the rows to the scenes of the PBP and each element $m_{i,j}$ is set to 1 if hypothesis h_j matched scene s_i, to 0 if it didn't match, and is blank if it was not yet tested on the scene. A hypothesis is known to be a solution if it matches all scenes from one side and none from the other, which corresponds to the column vector $(0,\ldots,0,1,\ldots,1)$ or $(1,\ldots,1,0,\ldots,0)$.

Hypotheses

The model estimates the utility of a hypothesis using the following approach. If a hypothesis h_i can be a solution given a particular state of M, which is the case if all tested scenes that matched were from one side and all that didn't match were from the other, the probability $P(h_i|M)$ of the hypothesis being a solution is the joint probability that each of the yet untested scenes is compatible with it. The PATHS model makes the assumption that the probability of a scene matching h_i is 0.5. This estimation is very rough and does not take the complexity of the hypothesis or the number of previous successful matches of that hypothesis into account. Instead, the complexity of the scene is incorporated into the utility estimation as a prior. The resulting formula is

$$P(h_i|M) = 0.5^{blank} P_0(h_i),$$

where $blank$ is the number of scenes that h_i was not yet tested on and $P_0(h_i)$ is set so it declines exponentially with the number of features that are used in h_i. Practically, this ensures both that the more scenes a hypothesis is successfully checked on, the higher its estimated utility gets and that simple hypotheses have a higher chance of being checked and merged.

If a hypothesis mismatches scenes from both sides, it can't be a solution or a part of a solution, since the conjunctive combination with another hy-

potheses always leads to an equally or more specific hypothesis. Therefore, the probability of such a hypothesis is set to 0.

There is a third case, in which h_i is known to match scenes from both sides and matches all tested scenes from at least one of the sides. In this case, h_i cannot be a solution by itself but can contribute to a combined, conjunctive solution. The model estimates its utility as

$$P(h_i|M) = 0.5^{blank+S/2+incomp} P_0(h_i),$$

where $S/2$ is a fixed penalty, set to half the total scene count, and $incomp$ is the number of scene matches that are incompatible with h_i being a solution by itself. In the special case of a hypothesis matching *all* scenes on both sides and containing basis-level features only (shape and size in the case of PBPs), the model sets $incomp = 0$. In practice, this makes hypotheses that are close to being a solution more probable than ones that are farther off, while accounting for the special situation in which the "same" object is showing up in each scene.

Objects and Groups

The probability that any particular object or group plays a role in a solution is estimated based on the sum of the utilities of all current hypotheses that select that object or group:

$$P(o|M) = P_0(o) Z \sum_{h \in H_o} \frac{1}{N_o(h)} P(h|M),$$

where o is an object or group, H_o is the subset of hypotheses that are known to select o, $P(h|M)$ is the estimated utility of hypothesis h, and $N_o(h)$ is the number of targets that h selects in the scene to which o belongs. $P_0(o)$ is the prior probability of the object or group and Z is a normalization factor that ensures the probabilities of all objects or groups in a scene add up to 1. The relative priors for objects and groups depend on the attributes that were perceived on them so far. They give higher probability to objects that are initially moving, unstable, or top-most in a scene and to groups that contain objects that are all close to or touching each other.

The estimation of the probability of an object or group is based on an equal distribution of the probabilities of all hypotheses to the objects or groups they are known to select. The model always considers an "any object" hypothesis, which ensures that each object in the scene is selected by at least one hypothesis. Groups are created as a result of applying a selector, so each group is guaranteed to have at least one associated selector.

Features

In earlier iterations of the model, the choice of a feature for perception was made based on the estimated probability that this feature is used in a so-

lution. The probability was estimated analogously to the object and group formula in the previous paragraph, with a small fixed term being added to the estimations to allow perceiving features that were not present in any of the current hypotheses. The relative priors for features are all set to the same base value, except for the shape and size attributes that are three times as likely to be perceived, and the movement and stability that are two times as likely to be perceived as the other features. This is meant to reflect that humans are more readily perceiving and encoding some features than others, as well as the expectation that PBP solutions will more often depend on these features.

A direct comparison of solution rates and the number of actions towards a solution revealed that simply using the feature priors as utility led to better performance and less variation in the performance. One plausible explanation is that although it is often helpful to focus on the features that led to promising hypotheses, this is already accomplished through the mechanisms of checking promising hypotheses on new scenes and combining them with each other. An additional bias towards perceiving those features seems to push the model too far from an exploration of the whole feature-space towards the exploitation of a potentially wrong initial clue.

The current PATHS model directly uses the feature priors in its stochastic selection of what to perceive next. While this turned out to be the algorithmically better choice for the existing PBPs, it is possible that a version of the model that gets more strongly fixated on current hypotheses would match the human data better.

3.4 A PROBLEM SOLUTION WALKTHROUGH

When the model starts working on a PBP, the first step is to load all the scenes which are provided as SVG images into memory. The objects and the ground in each scene are represented as polygons that describe their outline. This is the sole input that is given. The outlines act as the basis for a physics engine that is used both to simulate how the scenes will unfold and to perceive physical object features like stability.

Throughout the whole solving process, only two of the scenes will be visible at the same time. The algorithm can only work on the currently visible scene pair and proceeds through a predefined sequence of scene pairs. Initially, the model knows nothing about the objects beside their existence, and starts gathering information about the objects in the first visible scene pair. It selects features, such as “large” or “stable”, and objects on which to perceive the features. After a new perception is made, a corresponding selector, such as “large=true”, is created. This selector is then applied to both scenes in the currently visible scene pair, potentially resulting in a number of objects in both scenes that match. The match results and the selector are both captured in a hypothesis, which represents a potential solution or potential part of a solution.

After some perception steps, the model switches to the next scene pair. It can now continue to perceive features on the new objects or check existing hypotheses on the new scenes to gather additional evidence about their likelihood. The third available type of action is to combine existing hypotheses to build more complex ones. For example, "large objects" and "small objects on top of any object" can be combined into "small objects on top of large objects".

The model stops as soon as a hypothesis was checked on all scenes and is determined to be a solution, which means it matches all scenes from one side and none of the scenes from the other side. The search is typically also aborted after a predefined number of actions if no solution was found up to that point.

During a run of the model, it determines the type of the next action by randomly drawing from a fixed multinomial distribution. The elements the chosen action is acting on are determined stochastically based on the information from all hypothesis–scene matches done so far. More promising hypotheses will be checked first; objects and features that play a role in promising hypotheses will be picked with a higher probability for perceiving further features.

I will now describe one particular recorded run of the model solving PBP02. The solution to PBP02 is that the number of objects is one in the scenes on the left and two in the scenes on the right, making it one of the most basic PBPs. Although this particular run is representative of the models usual behavior on PBP02, it is important to understand that any two runs of the model can look quite different to each other due to the stochastic nature of the solution process. The model was configured to to only create selectors for active features, so that it could construct the hypotheses "there is a small object" and "there is a large object" but not the hypothesis "there is an object that is not small".

SCENE PAIR 1 The algorithm started in the situation depicted in Figure 17a, with the first scene pair visible. In the scene pair sequence I chose for this run, the scene pairs always contain scenes from different sides of the PBP. The first few actions the model took were to perceive how "large" the object in scene 1-1 was – it turned out to not be very large at all so no hypothesis was constructed. The model then perceived how "close" the two objects in scene 2-3 were (pretty close) and subsequently created a respective pattern description ("close to any object"). This was turned into a first hypothesis H1 that was tested on both visible scenes and stored as "only in the right scenes does there exist an object that is close to another object". Influenced by the fact that this is a potential solution, the model then switched to the next scene pair.

SCENE PAIR 2 The model then continued to work on the second scene pair as shown in Figure 17b. It decided to check hypothesis H1 on the new

scenes and found that it matched. The model then noticed that Object 2 (the small circle) in scene 2-4 "moves" and created and checked the hypothesis H2 "only in the right scenes does there exist a moving object".

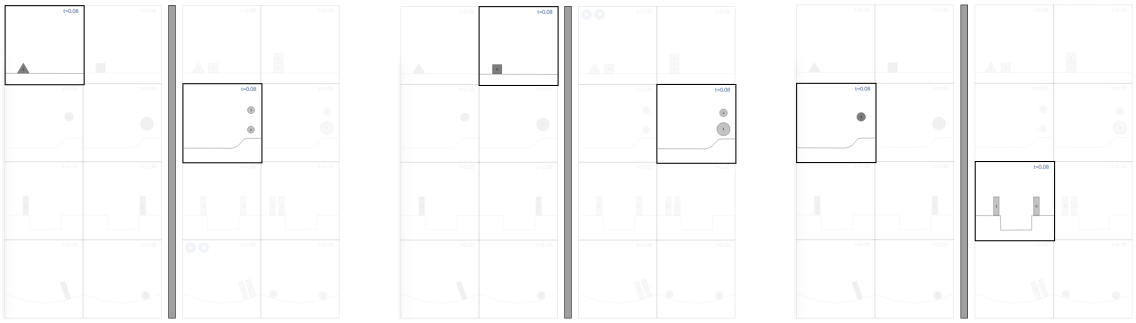
SCENE PAIR 3 After switching to the next scene pair, as shown in Figure 17c, the algorithm checked H1 on the new scenes and found a mismatch. Since H1 did match neither the left nor the right scene, it was excluded from the hypothesis list. Then, the model perceived that Object 0 (the right rectangle) from scene 3-3 is rectangular, which lead to a corresponding hypothesis H3 "only in the right scenes does there exist a rectangular object".

SCENE PAIR 4-8 The algorithm proceeded to look at the scene pairs four to eight, perceiving features, creating and checking hypotheses, and on some occasions unsuccessfully trying to combine existing hypotheses. During this time, it perceived the features "stable", "triangle", "rect", and "unstable" on various objects, leading to a new hypothesis H4 about unstable objects on the right side. Notably, it also perceived the number of objects in a scene in scene pair 7, leading to the hypothesis H5 "only in the right scenes, the number of objects is 2". This is a correct solution, although the algorithm had no way to know this at that time. Hypothesis H3 and H4 were checked on the new scenes and found to match on some, but failed to match on later scene pairs.

SCENE PAIR 1-6 During the last part of the run, the model iterated through scene pairs one to six for a second time. It perceived features like "circle", "triangle", "gets-hit", "count=1", "pos-top", "touching", "collides", "far", and "beside" which led to a number of new hypotheses. An example of a more elaborate one is based on the selector "objects that are beside a circle object at the end". Most importantly though, the algorithm continued to check hypothesis H5, "only in the right scenes, the number of objects is 2", on each of the scene pairs so that after checking it on scene pair six, it had been verified on all the scenes of the PBP. At this point, the algorithm stopped the search and reported the solution. Figure 18 shows all hypotheses that were created during the run.

3.5 CONCLUSION

This chapter described the PATHS model by looking at how it solves actual PBPs, its software architecture and the principles that guided its implementation. Key properties of the model are the tight integration of perception into the concept learning process, which appreciates the active and explorative nature of perception, as well as the combination of rational decisions with process-level modeling. Similar to humans, the model is restricted in its memory in that it can only actively work on currently visible scenes. Finally, the model is able to flexibly change its interpretations of gradual feature val-



(a) 1st scene pair (b) 2nd scene pair (c) 3rd scene pair

Figure 17: The first three scene pairs the PATHS model looked at during a run on PBP02. The model stochastically decides at which times it switches through the fixed sequence of scene pairs.

Hypotheses

R	E	(objects that are 2)	16	1.00
R	E	(circle objects)	4	0.00
R	E	(objects that are far)	2	0.00
L	E	(objects that are 1)	2	0.00
R	E	(objects that are collides-with (rect objects))	2	0.00
R	E	(objects that are beside (circle objects) at the end)	2	0.00
LR	E	(rect objects)	6	0.00
LR	E	(any object)	2	0.00
--	E	(objects that are close (any object))	6	0.00
--	E	(unstable objects)	6	0.00
--	E	(moves objects)	6	0.00

Figure 18: All hypothesis that the PATH model generated during a particular run on PBP02. The left-most column states whether a hypothesis matched scenes on the left, on the right, or on both sides of the PBP. The next column states that all hypotheses were "exists" quantified. The last to columns show the number of scenes each hypothesis watch checked on and estimated utility of each hypothesis. The first hypothesis was checked on all 16 scenes and is known to be the solution, so its utility is 1.

ues to fit the context of a particular PBP. Chapter 5 will compare the model's performance on PBPs to the performance of humans. The studies in which human performance data was collected are described in the next chapter.

4

HUMAN PERFORMANCE ON PBPS

This chapter explores how humans solve PBPs. The common task of all experiments in this chapter is to find and write down a verbal rule that distinguishes between all scenes on the left and all scenes on the right for each presented PBP. What varies between experiments is whether the scenes are presented in a sequence of pairs or all at once, and which kind of scenes are presented close to each other. This influences the type of comparisons that participants make between scenes and has, due to the iterative and active perception and rule-construction processes, a significant influence on their learning performance. The next section discusses the role of comparisons and similarity in concept learning. Section 4.2 describes an eye tracking experiment and Section 4.3 presents a series of four experiments in which the order and arrangement of scenes was manipulated. Besides providing data against which I compare the PATH model's performance, the experiments in this chapter advance our understanding of human concept learning and can inform the design of more efficient presentation schemes for real-world learning material. Physical Bongard problems are an exciting problem domain in this regard, since their open feature-space, dynamic concepts, and inner structure make them comparable in complexity to many real-world concepts that students have to learn.

4.1 THE ROLE OF SIMILARITY IN CONCEPT LEARNING

In inductive learning, one abstracts from trained examples to derive a more general characterization that can lead to both seeing the familiar in new and the new in familiar situations. One particularly powerful technique for inductive learning of difficult, relational concepts is the comparison of multiple cases (Kurtz et al. [2013], Loewenstein and Gentner [2005], Gick and Holyoak [1983]). The benefit of comparison goes beyond establishing similarities between the inputs, it frequently promotes noticing the commonalities and differences between the compared instances, which helps to construct useful generalizations and can change one's representation and understanding of what is compared (Hofstadter [1996], Medin et al. [1993], Mitchell [1993]). In the following, I look into how the type and order of comparisons influence category learning.

In a category learning setting in which category labels are provided, two important types of comparisons are possible: comparisons between instances from the same concept and comparisons of instances from different concepts. The existing research literature does not only suggest different roles

for those two types of comparisons but also makes different predictions as to the factors that influence their effectiveness.

Since comparing instances of the same concept can serve to highlight commonalities between them, it may be beneficial to compare instances that share as few features that are irrelevant for the characterization of the concept as possible. A closely connected notion called "conservative generalization" by Medin and Ross [1989], is that people will generalize as minimally as possible, preserving shared details unless there is a compelling reason to discard them. As within-category objects become more similar, their superficial similarities might be mistaken as defining ones and might lead to a category representation that is too narrow, for example when learning to discriminate pairs of similar-sounding words (Rost and McMurray [2009]), or when learning about which methods to use in exploratory data analysis (Chang et al. [2003]). By varying the irrelevant features possessed by examples with a single category, the relatively stable, deep commonalities stand out and can make hard learning tasks feasible, such as learning relational syntax rules from examples (Gómez [2002]). Another example of the benefit of low within-category similarity when learning from examples are the results of Halpern et al. [1990], who asked students to read scientific passages that included either "near" (superficially similar) or "far" (superficially dissimilar) analogies. The passages that included far analogies led to superior retention, inference, and transfer compared to those featuring superficially similar comparisons, which showed no benefit at all.

All of the studies mentioned above find advantages of low similarity for learning a concept using within-category comparisons. Kotovsky and Gentner [1996] add a potential constraint to this. They argue that a meaningful comparison of structured instances requires first successfully aligning them and this can be too difficult a task for instances that are very different on the surface. Using the notion of "progressive alignment," they demonstrate that especially at the beginning of a learning process, comparing high-similarity instances of the same category can be essential (Gentner [2010]).

For the case of comparing instances between categories, the predictions of the influence of similarity on the learning progress are more univocal. In order to learn how to tell two categories apart, it is best to compare the most similar instances of the two categories with each other, or, more precisely, the instances that have the smallest number of non-discriminative differences. This has the advantage of decreasing the likelihood of spurious differences being chosen as the basis for discriminating the categories, as Winston [1970] described with the notion of "near misses." There is an additional advantage of high similarity for between-category comparisons. When learning to distinguish between several similar concepts, one major difficulty lies in identifying the subtle differences between them and interleaving similar categories can result in increased between-category contrast and discriminability, which in turn enhances learning (Carvalho and Gold-

stone [2013], Birnbaum et al. [2012], Kornell and Bjork [2008], Kang and Pashler [2012]).

In summary, the two lines of arguments described above predict that between-category comparisons should best be made using similar instances, while within-category comparisons should be made using dissimilar instances, as long as the instances are still similar enough to allow for meaningful alignment. Both types of comparisons are potentially important for learning concepts and the best weighting between them will be different across learning situations, depending on the specific task, context, experience of the learner, and structure of concepts (Goldstone [1996]).

4.2 EYE TRACKING STUDY

One powerful way to gain insights into people's cognitive strategies is to record where they are looking at. The trajectory of participants' fixations points during their work on a PBP can tell us, for example, in which order they gather information and whether they spend more time looking for similarities within one side of a PBP or for differences between the sides. To answer questions like these, I conducted an explorative eye tracking study with ten participants solving PBPs. They were shown 33 PBPs on a computer screen and had the task of solving as many of them as possible in about 45 minutes. I used the early version of PBPs that had four scenes per side. The participants were not allowed to return to a problem that they did not solve. Before switching to the next problem, they received feedback on their solution and were told the official solution to the problem they just completed. This was done so all participants would understand the solution concepts of previous problems on arrival at a specific, new problem.

I recorded gaze positions, the time it took participants to solve each problem, and whether they found a correct solution or not. Each participant completed a one-page questionnaire after the experiment containing two questions: 1) How did you solve the problems, what was your strategy? 2) What was difficult and what was easy for you?

A head-mounted eye tracking system was used to record the gaze positions with a temporal resolution of 2 ms. Before recording the eye tracking data, a calibration of the system was done. Right before each individual problem, a short recalibration with a single fixation point at the center of the screen was performed. While attaching the eye tracker cameras to the participant's head has the advantage of potentially very accurate measurements, the downside is that any small slippage of the helmet relative to the participant's head introduces a large systematic error in the estimated gaze positions on the screen. I found that in the one or two minutes the subject works at a single PBP, that is, between two recalibrations, often a substantial slippage occurs. A way to avoid this is to completely restrain head movements and strap the helmet very tightly onto the participant's head. Since both of these are quite uncomfortable for participants, I decided

on the alternative of post-processing the gaze data to remove systematic slippage-induced errors. In cooperation with Samuel John, I developed and applied a post-recording data correction algorithm that strongly increased the gaze data quality in a task-agnostic way by maximizing the entropy of the 2D distribution of the gaze points (John et al. [2012]).

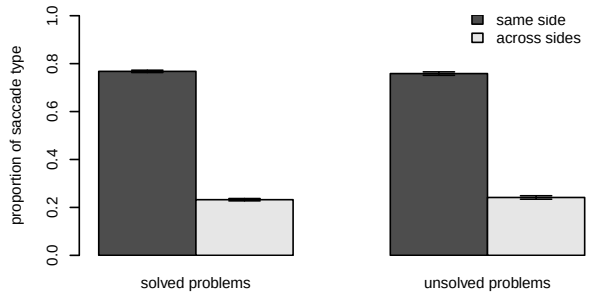


Figure 19: The proportion of same-side and across-side saccades. The data is grouped by whether a problem was correctly solved. The bars represent standard error.

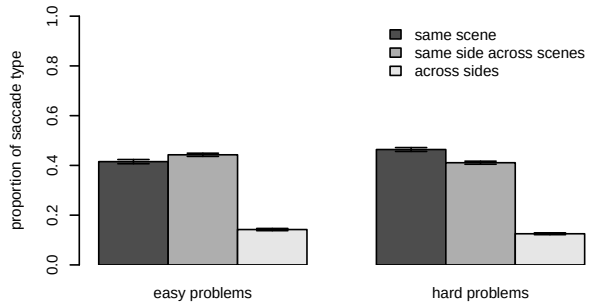


Figure 20: The proportion of within-scene saccades, same-side, and across-side saccades for easy and hard problems. A problem was considered easy if its solution rate was above the median solution rate across problems. The bars represent standard error.

Results

Important aspects of participants' solution strategies are reflected in the order in which they look at the PBP scenes. I analyze this by looking at saccades – a movement of the eye that shifts the gaze from one screen position to another. I grouped saccades into three types: (S1) saccades within one scene, (S2) saccades between scenes from the same side of the PBP and (S3) saccades between scenes from different sides of the PBP.

Figure 19 shows the proportion of same-side saccades (S2) and across-sides saccades (S3) both for solved and unsolved problems. Figure 20 shows the proportions of all three saccade types both for easy and hard problems. A problem was considered easy if its solution rate was above the median

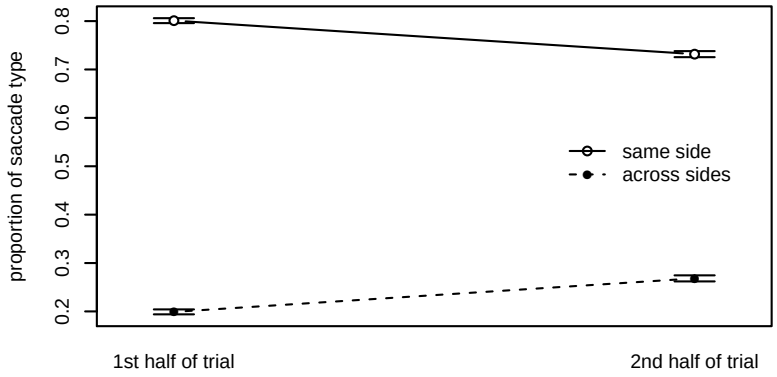


Figure 21: The proportion of same-side and across-side saccades in the first and second half of a trial. In average, participants make significantly more across-side saccades in the second half of a trial. The bars represent standard error.

solution rate across problems. Finally, Figure 21 shows the proportion of same-side saccades (S2) and across-sides saccades (S3) during the first and second half of each trial. I defined the first and second half of a trial based on the total time a participant worked on the PBP in that trial.

I performed a within-subject t-test to test the difference in the proportion of same-side saccades between the first and the second half of a trial. I only analyzed one type of saccade in this test, since the proportions of same-side and across-sides saccades add up to one, so that one is fully determined by the other. The t-test revealed a significant difference in the mean values: $t(753) = -8.6$, $p < 2e-16$. The mean proportion of "across scene" saccades was 0.2 in the first half and 0.27 in the second half. The 95% confidence interval for the difference in the means is $[0.053, 0.085]$. A similar t-tests showed no significant influence of whether a problem was solved on the ratio of saccade types.

Here is a summary of the answers given in the questionnaire. When asked about their strategies, people stated that they *"... first search for similarities (on one side), then compare to the other side,"* as well as *"... imagine the movement."* One person described the following solution strategy: *"After getting used to the problems, look for a list of features (for example, the ones that worked before). If something catches the attention, adapt ordering of features."* Two more responses were: *"first check if the solution is immediately apparent, then check different features, analyze,"* and *"select one or two images and compare with the other side to generate a hypothesis, then check it and if needed repeat."*

When asked about difficulty, participants pointed out that: *"if there is movement that is irrelevant, this makes it harder,"* and *"graduate solutions (a bit vs a lot) were difficult,"* as well as that finding solution that involves *"potential movement or interaction is difficult"*. Another remark was that *"very simple problems can be most difficult because people overthink them."*

Discussion

There were surprisingly little individual differences across subject in their average rates of same-side and across-sides saccades. Additionally, the rates did not significantly depend on the solving success or the difficulty of the problem (Figures 19 and 20), although there was a tendency of participants doing more within-scene saccades for harder problems. A plausible explanation is that harder problems contain more objects per scene, which requires more fixations within a scene during perceiving a scene.

On average, participants moved their eyes to look at a different scene on the same side of a PBP about four times as often as they did to look at a scene on the other side of a PBP. This suggests that participants spent more time on building representations and making comparisons within a category than on contrasting instances from different categories.

The rate of across-sides saccades was significantly higher in the second half of the time spent working on an individual PBP, compared to the first half. A possible explanation is that during the first half, participants first looked at all instances on the left and then at all instances on the right side, whereas in the second half, they were doing more contrastive comparisons between sides to validate solution hypotheses they had come up with. This is consistent with the strategy description quoted earlier (*"first search for similarities (on one side), then compare to the other side,"* and with an inspection of the gaze data from individual trials. In the first couple of seconds, participants typically scan all scenes from one side and then move on to scan all scenes from the other side before they start to show problem- and person-specific gaze patterns.

4.3 SCENE ORDERING EXPERIMENTS

ACKNOWLEDGMENT I designed and carried out the experiments described in this section in collaboration with Paulo Carvalho and Professor Robert Goldstone from the Percepts & Concepts research group at Indiana University in Bloomington, IN, USA. Paulo Carvalho and Professor Goldstone directly contributed to the following aspects of the work presented in this chapter: the compilation of the literature review, the planning of the experimental designs, and the selection of methods for data analysis. I will therefore use the pronoun "we" throughout this chapter. I have described some of the experiments in earlier publications, namely in Weitnauer et al. [2013, 2014, 2015]. I have adjusted, revised and extended the original analyses, descriptions, and plots for this thesis.

The scene ordering experiments deal with the question of how the mode of presentation influences the learning performance. This is a highly relevant question in the context of teaching and learning, for example in schools and universities. To this end, we designed and conducted four successive ex-

	EXP 1	EXP 2	EXP 3	EXP 4
PBP version	1	2	2.1	2.1
training scenes	8	12	16	16
presentation	in pairs	in pairs	in pairs	all at once
test scenes	0	6	4+2	4+2
allow cycling	yes	no	yes	yes
time per pair	4 sec.	manual	manual	manual
factors	schedule	schedule, similarity, side-swap	schedule, within sim., between sim.	schedule, within sim., between sim.
participants	81	38	98	91
main effects	-	similarity	schedule, between sim.	schedule, within sim.

Table 1: Overview of experiments.

periments, where each experiment's design was informed by the results of the previous experiment.

The general idea behind the first three experiments is to present the scenes of a PBP as a succession of scene pairs, instead of showing them all at once. This way the order in which the participants attend to the scenes can be manipulated. In the first experiment, we manipulated the scene ordering along one dimension, the *scheduling* dimension. The scenes inside each pair were either from the same side of the PBP, which we call *blocked condition*, or are from different sides, which we call *interleaved condition*. In the second experiment, we used the redesigned version of PBPs, in which more scenes were added and scene similarity varies in a predefined way, which allowed us to manipulate the *similarity* of temporally close scenes in addition to the *scheduling* condition. In the third experiment, we split the *similarity* condition into two dimensions, the *within-category similarity* and the *between-category similarity*. Finally, in the fourth experiment, we presented all scenes simultaneously and manipulated the similarity structure through varying the spatial instead of the temporal scene arrangement. Table 1 gives an overview of the experiments. We now describe the experimental setups and the results in detail for each experiment.

Amazon Mechanical Turk as Research Platform

Amazon Mechanical Turk (MTurk) is a crowdsourcing Internet marketplace that is operated by Amazon.com. On this marketplace, requesters can post jobs, known as HITs (human intelligence tasks), and humans, known as workers or turkers, can pick among the jobs and complete them for money.

Typical tasks on MTurk include labeling and rating of images or other media. Recently, MTurk has been increasingly used as a platform for conducting psychology studies. If a study only requires a computer and can be designed to run within a browser, MTurk has the advantage of a huge pool of workers that are available 24 hours a day. It is possible to collect data from 100 participants in a few hours instead of a few weeks. Researchers have found that the quality of the data collected on MTurk is comparable to the data from student pools at universities and MTurk is now broadly accepted in the psychology community (Mason and Suri [2012]).

The following experiments were performed on MTurk. Joshua de Loeew provided us with valuable technical support on implementing the experiments using his library jsPsych (de Leeuw [2013]).

1st Experiment

The first experiment was designed to test whether the positive effect of interleaving instances from different categories during learning which was demonstrated by Birnbaum et al. [2012], Kornell and Bjork [2008] and Kang and Pashler [2012] would also apply to category learning in the domain of PBPs. As laid out earlier, the theory behind this effect is that the comparison of scenes from different categories enhances the perception of differences, which is essential for learning, especially in domains with high similarity between the instances. Consequently, our hypothesis was that interleaving the scenes of the PBPs during learning should lead to more correct classifications and category descriptions than blocking the scenes by showing all scenes from one side before showing all scenes from the other side.

After collecting data from a first round of participants, we did not find a significant effect of presentation schedule on the number of correct solutions. The data seemed to support another hypothesis, though, which we decided to test by adding more participants using the same experimental design. It appeared that for hard problems, *interleaving* of the scenes was better, while for easy problems, *blocking* was better. The rationale behind this is that for easy problems the difference of the scenes of one side to the scenes of the other side might catch one's eye at the moment of switching from the one presentation block to the other. Imagine seeing a sequence of photos depicting a variety of red flowers all belonging to the same category. Even if the color is not considered consciously as the defining feature, the blocked presentation of all examples of one category supports the construction of an internal category representation that will most likely include the color feature. If now the first photos of a second category of flowers only contain yellow flowers, this difference will stand out immediately. Interleaved presentation, on the other hand, would be most efficient for difficult problems where small, intricate details in which the two sides differ have to be found.

Participants

We recorded and analyzed the data of 102 participants in total, 62 in a first and another 40 in a second recruiting round. We excluded those participants from the analysis that either didn't finish the experiment or did not get the solution of a single problem right. There remained 81 participants, 49 from the first and 32 from the second round. On average, participants solved 11.6 out of the 23 problems presented.

Material

We selected 23 of the 34 early version PBPs, where each problem has four scenes on the left side and four scenes on the right side. See Figure 22 for an example. The selection was guided by the goal of having a diverse subset of problems that are moderately difficult. Our main measure of how well subjects solved the problems is based on the proportion of subjects that found a correct solution. In order to maximize the information gain, we attempted to adjust the difficulty such that each problem is solved by approximately half of the subjects. We especially wanted to avoid problems that were so easy everybody could solve them or problems that were so hard nobody could solve them, no matter in which condition they were presented.

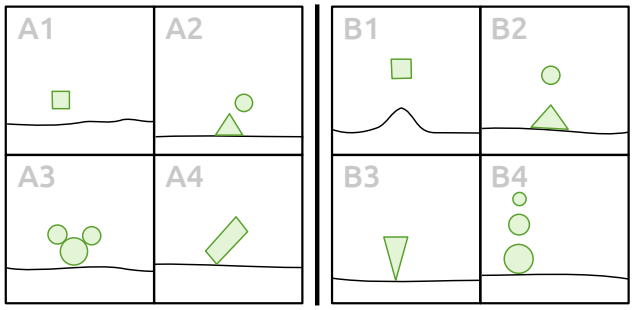


Figure 22: Experiment 1, example PBP. All eight scenes are shown in pairs during training.

We expected strong priming effects depending on what problems a subject saw before working on the current one. We therefore decided to fix the order in which the problems are presented in a way that does not put problems with similar solutions next to each other. Additionally, we decided to show the correct solution after each problem. This way, each subject will have been exposed to the same problems and concepts at any specific point in the experiment. The scene selection and ordering are shown in Table 2.

Design

We manipulated the factor *schedule condition* $\in$ {*blocked*, *interleaved*} in a within-subject manner. Every participant saw half of the problems blocked and half of them interleaved.

PBP	LEFT SIDE	RIGHT SIDE
12	small object falls off	small object stays on top
04	squares	circles
32	objects rotate a lot	obj.s rotate little or not at all
22	objects collide with each other	objects don't collide with each other
08	unstable situation	stable situation
31	circle can be picked up directly	circle can't be picked up directly
27	(potential) chain reaction	no chain reaction
17	objects touch	objects don't touch
23	collision	no collision
26	circle moves right	circle moves left
13	objects form a tower	objects form an arc
30	less stable situation	stable situation
16	the circle is left of the square	the square is left of the circle
24	several possible outcomes	one possible outcome
02	one object	two objects
20	square supports other obj's eventually	square doesn't support other objects
18	object touch eventually	obj.s don't touch eventually
21	strong collision	weak or no collision
09	objects move in opposite directions	objects move in same direction
33	construction gets destroyed	construction stays intact
19	at least one object flies through the air	all object always touch something
05	objects move	objects don't move
28	rolls well	does not roll well

Table 2: The 23 PBPs (version 1) in the order they were shown in the first scene ordering experiment.

Procedure

For each of the 23 problems, the scenes were presented as a sequence of scene pairs so that at any time only two of the eight scenes were visible. The frames of all scenes were always visible, but their content was covered. During the experiment, always two of the scenes which we'll call a scene pair, were uncovered for four seconds and then automatically covered again, uncovering the next scene pair. In order to test our hypothesis that interleaved presentation should lead to better concept learning than blocked presentation we manipulated the *scheduling condition*, which could be *blocked* or *interleaved*. During blocked presentation, the scenes in each pair were taken from the same side and all scenes of one side were shown before the scenes of the other side. The first and second scene pair were composed of scenes from the left side (A-side), and the third and fourth scene pairs are composed of the scenes from the right side (B-side), leading to the pattern ("AA AA BB BB"). During interleaved presentation, scenes from both sides were alternated ("AB AB AB AB"). Figure 23 depicts the positions at which the scenes were shown. For each problem and participant, we shuffled the scenes inside each side of the problem. This way, every participant saw scenes uncovered at the same screen position at the same time, while the scenes were in random order within the sides.

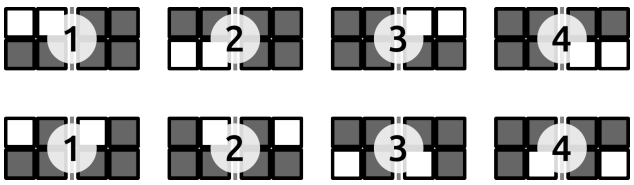


Figure 23: Scheduling for Experiment 1. The positions at which the scenes are shown for blocked (top) and interleaved (bottom) presentation. Each of the four states is shown for four seconds. In each state, two scenes (in white) are shown while the other scenes (in gray) are hidden. Which of the four left and the four right scenes are actually shown at the highlighted positions in the four states is randomized.

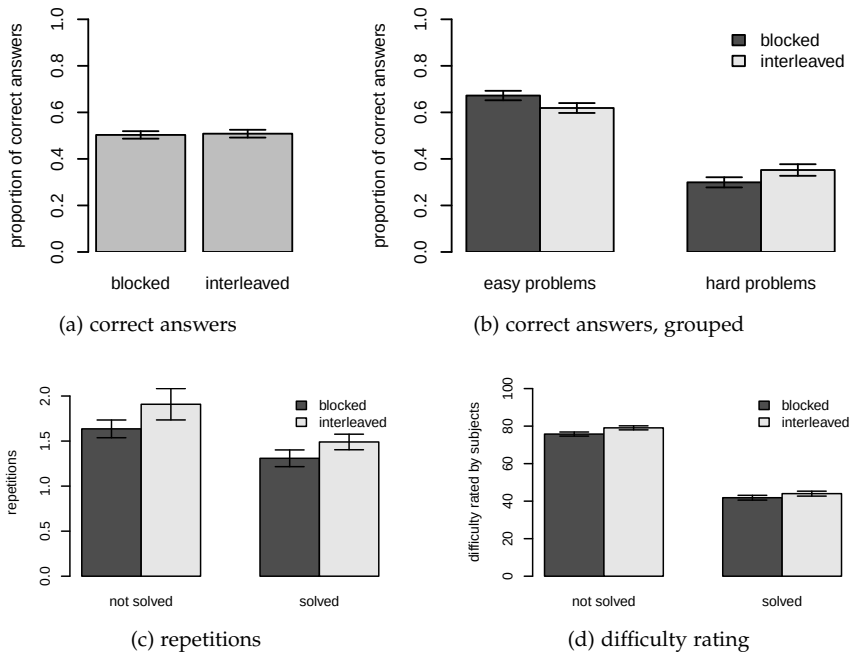
After the participants saw all scenes, they were asked whether they wanted another presentation of all scenes or proceed to enter the solution. After proceeding, they were required to state whether they found a solution, to rate the difficulty of the problem on a scale from 1 to 100 and to type in the solution as free text into two text fields, one for the left side, one for the right side. Finally, they were shown the PBP with all scenes at once together with the official solution.

Results

The written solutions of all participants were manually checked for correctness by trained coders blind to the experimental hypothesis. Solutions which were different from the official solution but valid for the problem

were accepted as correct. Some of the participants had difficulties remembering which concept had been presented on the left and which on the right and provided a correct solution but with sides swapped (e.g., writing "left: all objects are squares" and "right: all the objects are circles" when, in fact, the left-side objects were all circles and the right-side objects were all squares). These cases were counted as correct solutions. We had an agreement of $\alpha > 0.9$ between the coders and had a third trained coder who was blind to the experimental hypothesis resolve each conflicting decision.

We did not find a significant influence of the presentation schedule on the number of correct solutions. The tendency of *interleaving* helping with difficult and *blocking* helping with easy problems did not reach a significant level with the added participants from the second round. The correlation between the subjects' difficulty rating of problems, their success at solving them and the number of repetitions they used was strong in the expected direction. See Figure 24 for the plots.



(a) correct answers

(b) correct answers, grouped

(c) repetitions

(d) difficulty rating

Figure 24: Results of Experiment 1. Contrary to our expectations, there is no significant influence of scheduling condition on the proportion of correct answers (a). There is a tendency of blocking helping easy and interleaving helping difficult problems, but the effects are not significant (b). The number of repetitions (c) and the difficulty rating by the subjects (d) is higher for problems they did not solve compared to problems they solved.

Discussion

Contrary to our expectations based on the reviewed literature, no general advantage of interleaved presentation was present. Both Goldstone [1996] and Carvalho and Goldstone [2012] reason that the effectiveness of within and

between category comparisons for concept learning depends on the structure of the to-be-learned categories, especially on how similar the instances of one category are to other instances of the same category and to instances of other categories. To shed further light onto the influence of similarity on solving PBPs, we decided to control the similarity of the PBP scenes.

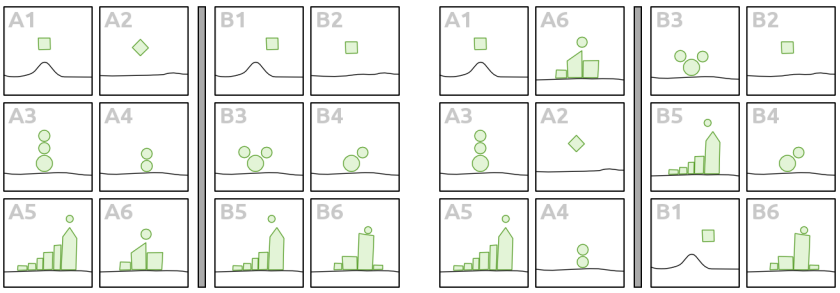
2nd Experiment

In this experiment we compare the effect similarity has on learning performance in blocked and interleaved presentation schedules. Carvalho and Goldstone [2012] recently conducted an experiment with a similar purpose. They manipulated the category structures in a perceptual categorization task towards more or less similarity, both within and between categories, and found this modulates the advantage of blocking and interleaving. The blocked presentation of instances of one category led to better learning performance for the concepts with low similarity while the interleaved presentation was better for very similar concepts.

Our approach is closely related but differs in three aspects. First, we manipulate similarity by grouping concept instances into either similar or dissimilar comparisons, instead of switching between separate sets of categories. Second, we designed the blocked and interleaved schedules in a way that enhances within- and between-category comparison, respectively, while still allowing for both types of comparisons. In this situation, the reviewed research literature makes opposite predictions on whether high similarity of instances shown closely together should help or hurt the induction, so we can directly compare the effect strengths. Third, we use an inductive learning task, physical Bongard problems, with a much larger feature-space than the problem domain used by Carvalho and Goldstone [2012].

We expected to find that grouping by similarity should improve learning performance for the interleaved condition and grouping by dissimilarity should improve performance for the blocked condition. A third hypothesis we wanted to test was whether the difficulty of a problem changes when swapping the left and the right side. This might be the case since subjects always see the scenes of the left side first and some solutions are easier formulated based on one specific side. For example, solutions of the type "left: X; right: $\neg X$" as in "left: the object falls down; right: the object does not fall down" are easier to formulate based on the left side[1]. For solutions of the type "left: X vs. right Y" as in "left: circles; right: squares", we don't expect a difference in solving performance for the side-swapped version.

1 One could argue that a solution could as well be formulated the other way around as "left: the object does not remain on top; right: the object remains on top". We found, however, that the great majority of answers provided by subjects was based on movement and action as in "falls" opposed to the description "remains". In deciding which side is easier to describe, we therefore use an action-centered formulation of the solution.



(a) PBP 24 similar (b) PBP 24 dissimilar

Figure 25: Experiment 2. The two plots show PBP 24 in its similar version where similar scenes are inside the same row (left) and in its dissimilar version where similar scenes are spread across the rows (right). Only the arrangement of the twelve scenes in the upper three rows of each problem is affected, as they are used as training scenes. Six of the eight scenes in the lower two rows, which are not shown here, are randomly selected and used as test scenes.

Material

In order to control for the similarity of scenes, we redesigned the whole set of initial PBPs (version 1). The new problems (version 2) have the same solutions but consist of 20 instead of 8 scenes each, which are aligned in a grid with five rows and four columns (see Figure 26). The scenes are organized in five similarity groups, each with two scenes from the left and two scenes from the right. The scenes inside each group are very similar to each other and only differ in few features. The scenes between two groups are less similar to each other and differ in more features. We also swapped the left and the right side of several problems so that for each problem it was easier or equally difficult to formulate the solution based on the left side compared to formulating it based on the right side. For example, all problems with a solution of the type "X; ¬X" have the "¬X" side on the right side. The selection and ordering of the problems were slightly changed in regard to the previous experiment. PBP 17 was replaced by PBP 11b to introduce the concept of *distance* instead of having two problems about *touching*. PBP 05 was removed because it was too easy. Additionally, the problems order was slightly rearranged in order to positions problems with similar solution themes further from each other to minimize context effects between consecutive problems. See Table 3.

For each of the problems, twelve scenes were used as training scenes. We prepared two versions of each problem by placing the scenes at different positions in the upper three rows. In the *grouped by similarity* version, the scenes were arranged in such a way that the scenes inside each row are similar to each other. In the *grouped by dissimilarity* version, similar scenes were distributed over all three rows. Figure 25 shows an example of the dissimilar and similar version of PBP 24, respectively. We additionally controlled

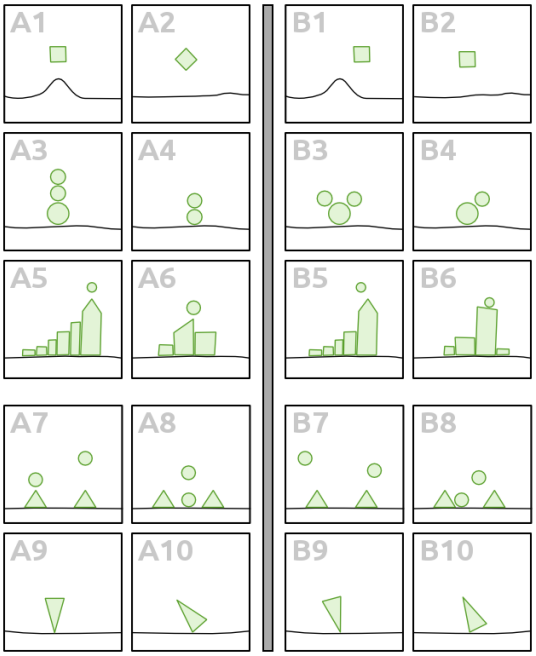


Figure 26: The new version of PBP 24. All PBPs in the new version have 20 scenes that are grouped into five similarity groups. In the arrangement above, scenes from within a row are more similar to each other than scenes across rows.

whether the left scenes would be shown on the left side and the right scenes on the right (*normal condition*) or whether the sides would be swapped (*side swapped condition*).

Procedure

Per problem, twelve of the scenes were used as training scenes and eight as test scenes. During the presentation, two scenes are always displayed simultaneously so that for each problem a sequence of six training scene pairs is shown to the participant. We vary the presentation order of scenes along three independent dimensions with two values each, resulting in eight conditions. The first dimension, *similarity grouping*, controls whether similar scenes are shown temporally close to each other or temporally far from each other. We will refer to the former as the *similar* condition and to the latter as *dissimilar* condition.

The second dimension, *presentation schedule*, controls whether the scenes that are shown simultaneously are from the same or from different categories ("AA BB AA BB AA BB" and "AB AB AB AB AB AB", see Figure 27). We refer to the former as *blocked* and to the latter as *interleaved* condition. In the blocked condition while within-category comparisons are facilitated by presenting scenes from the same category simultaneously, between-category comparisons can still be made between successive scene pairs, although they involve higher memory demands. Analogously, the interleaved condition

PBP	LEFT SIDE	RIGHT SIDE
02	one object	two objects
12	small object falls off	small object stays on top
04	squares	circles
32	objects rotate a lot	obj.s rotate little or not at all
22	objects collide with each other	objects don't collide with each other
08	unstable situation	stable situation
31	circle can be picked up directly	circle can't be picked up directly
27	(potential) chain reaction	no chain reaction
18	object touch eventually	obj.s don't touch eventually
23	collision	no collision
26	circle moves right	circle moves left
13	objects form a tower	objects form an arc
30	less stable situation	stable situation
16	the circle is left of the square	the square is left of the circle
24	several possible outcomes	one possible outcome
20	square supports other obj's eventually	square doesn't support other objects
21	strong collision	weak or no collision
09	objects move in opposite directions	objects move in same direction
33	construction gets destroyed	construction stays intact
19	at least one object flies through the air	all object always touch something
28	rolls well	does not roll well
11b	objects close to each other	objects far from each other

Table 3: The 22 PBPs (version 2) in the order they were shown in the second and third scene ordering experiment.

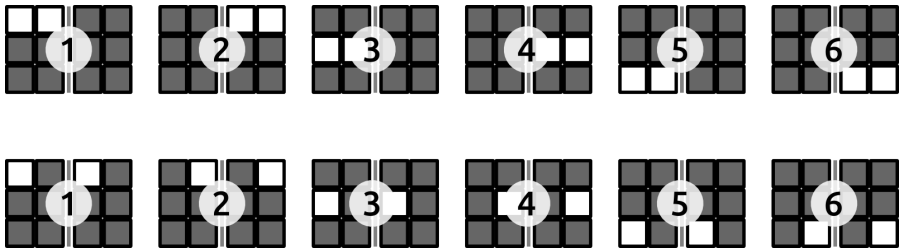


Figure 27: The scene presentation schedule for blocked (top) and interleaved (bottom) presentation. The participant manually proceeds through the six states. In each state, two scenes (in white) are shown while the other scenes (in gray) are hidden.

enhances between-category comparisons but still allows for within-category comparison across successive scene pairs. We are here using the term *blocked* to refer to a slightly different presentation schedule than we did in the previous experiment. Instead of showing all instances of one category before switching to the next, we only block two instances of one category inside the scene pairs and interleave these blocked pairs.

The participants were first given a brief introduction to PBPs including an example problem with a solution. During the experiment, they could proceed through the scene pairs of each problem at their own pace by pressing a key. Each scene was shown exactly once and afterward the subjects were asked whether they thought they had found a solution. Then the subjects needed to classify six test scenes drawn randomly from the eight available test scenes as either belonging to the left side or the right side of the PBP. The test scenes were shown one by one. Finally, they had to type in a description of their solution or their best guess. Before moving on to the next problem, they were shown the problem with all training scenes at once together with the official solution. There was no time limit to the task. At the end of the experiment, participants were debriefed on the study objectives and variables. The original experiment is available online at `http://goo.gl/ndv64h`.

Subjects

We conducted the experiment on Amazon Mechanical Turk. Sixty-seven participants, all US-citizens, took part in the experiment in return for monetary compensation. Of these, we excluded 27 who did not finish all problems (most of them dropped out after seeing only a few) and another two that did not get at least one solution correct across the entire task. There was no need to use catch trials because the subjects were required to write down the solutions as free text. Any cheating or automated answers would have become immediately apparent during our hand-coding of the solutions. The data from the remaining 38 participants was used in the following analysis. On average, participants solved 8.6 out of the 22 problems presented.

Design

We used a 2 × 2 × 2 factorial design. The study condition *presentation schedule* ∈ {*blocked*, *interleaved*} × *similarity grouping* ∈ {*similar*, *dissimilar*} × *side swapping* ∈ {*normal*, *swapped*} was randomly chosen for each problem in a within-subject manner and balanced for each subject.

Results

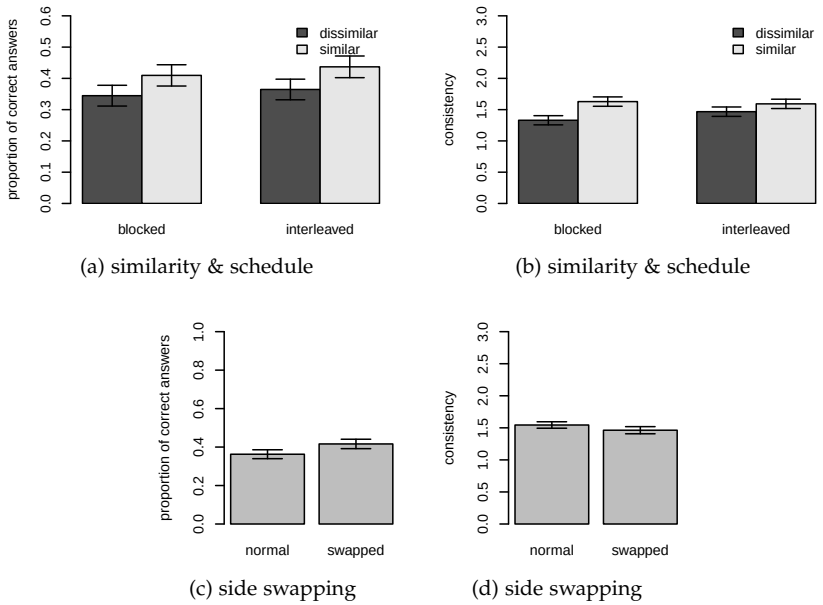
We used two separate measures to evaluate learning success. First, we hand-coded the accuracy of each textual solution given by the participants using the same procedure as in the previous experiment. Like before, alternative solutions and solutions in which the subjects accidentally swapped the description of the left and the right side were accepted as correct. The second measure is based on the proportion of test scenes that were classified correctly. Using this directly would be misleading for cases in which participants mixed up the sides. We therefore developed a consistency measure instead. This consistency measure is defined as $\max(c, 6-c) - 3$, where c is the number of correctly classified scenes being minimally zero and maximally six. The consistency can take values between zero and three, where the latter corresponds to cases where either all test scenes were classified correctly or were all (consistently) classified incorrectly. Figures 28a and Figure 28b show the average of these two measures for all four conditions.

We applied two separate 2 x 2 repeated measures analyses of variance (ANOVA) with *schedule condition* and *similarity condition* as factors to the proportion of correct responses and the consistency measure. These analyses revealed a significant effect of similarity condition, $F(1, 37) = 5.32$, $p = .03$ for the proportion of correct answers measure and $F(1, 37) = 15.7$, $p = .0003$ for the consistency measure. There was no effect of schedule of presentation, or interaction between the two factors for any of the measures (all $p > .05$). We separately analyzed the influence of *side swapped condition* with another ANOVA using just this one factor and found no significant effect.

Discussion

The data analysis revealed a positive effect of grouping scenes by similarity, independent of whether they were presented in a blocked or an interleaved schedule. We argue that this is explained by a strong positive effect of similarity on interleaving which more than compensates for any possible negative effect that similarity had on blocking.

The advantage of similarity for interleaving is in line with our expectations. Goldstone [1996] and the discriminative contrast hypothesis of Birnbaum et al. [2012] predict that direct comparison of instances from different categories highlights their differences (see also Carvalho and Goldstone [2012]). Identifying differences between highly similar scenes is especially effective, as there are fewer superficial differences to compete with the defining one. This insight is already present in the desirable "near misses" in



(a) similarity & schedule (b) similarity & schedule

(c) side swapping (d) side swapping

Figure 28: Results of Experiment 2. There is a significant effect of similarity condition but no significant effect of schedule condition on the proportion of correct answers (a). There is a highly significant effect of similarity condition and no effect of schedule condition on the consistency, which measures how well the subjects could classify the test scenes (b). Swapping of sides has no significant influence on either proportion of correct answers or consistency (c,d).

Winston [1970] work, where instances from different concepts that differ by just one feature are ideal for his algorithmic learner. Near misses provide clear evidence about what features are critical, concept-defining ones. Another possible contributing effect is that it is easier to structurally align two similar scenes than two very different scenes and this alignment process promotes noticing differences (Markman and Gentner [1993]).

What might seem surprising at first is that similarity also improves learning performance in the blocked condition, given that theories like "conservative generalization" by Medin and Ross [1989] predict that similarity for blocked scenes will lead to many superficial similarities and, therefore, inferior performance compared to dissimilar scenes. However, the results can be explained in a way compatible with these theories. We designed both scheduling conditions in a way that allows for within- *and* between-category comparisons. Given this, negative effects of similarity on the former and positive effect of similarity on the latter will compete with each other. In the blocking condition, within-category comparisons were facilitated by showing scenes of the same category simultaneously, while scenes of different categories had to be compared sequentially.

Still, a strong positive effect of similarity on between-category comparison could mask a small negative effect of similarity on within-category comparison and lead to the overall benefit of similarity that we found. This

means, however, that even during blocked presentation, participants exploit between-pair differences to find the solution despite the fact that exploring within-pair similarities has much lower memory demands.

We believe that one important reason for this might be found in the type of categorization task that was used. Due to its open-ended feature space, participants had to identify or construct relevant feature dimensions as a major part of the challenge. Comparing similar scenes from different concepts provides the additional advantage of highlighting such feature dimensions, an advantage that blocking of dissimilar scenes does not provide.

We found no confirmation for our hypothesis that which of the two categories is shown left and which is shown right has an influence on the solving performance. However, the expected effect would have been strongest for a true blocking condition ("AA AA AA AA BB BB BB BB") opposed to the blocking condition ("AA BB AA BB AA BB AA BB") we used. Additionally, for some problems the difference between the complexity of the description of both concepts is smaller than for others. If there is, in fact, an effect of which concept is shown first and which second, these two factors might have prohibited us from observing it.

3rd Experiment

In Experiment 2, we pitted the predicted influence of similarity in within-category versus between-category comparisons against each other by grouping all instances by similarity or by dissimilarity which influenced both the within- and the between-category similarity at the same time. In the remaining two experiments, we disentangle these two types of similarity and manipulate them independently from each other as two separate factors to allow us to draw stronger conclusions.

Experiment 3 is a replication of Experiment 2 using two instead of one similarity factor, resulting in four instead of two different similarity conditions. We expect that for between-category comparisons, a high similarity of the respective scenes should lead to better solving performance while, for within-category comparisons, the compared scenes should optimally be dissimilar. Our interpretation of the previous experiment included that between-category comparisons are more critical to the learning performance than within-category comparisons. This might lead to interleaving being better than blocking because it enhances the more important between-category comparisons. We would expect that this is mainly the case for high between-category similarity, but not for low between-category similarity, which should show as an interaction between scheduling type and between-category similarity.

Participants

We conducted the experiment on Amazon Mechanical Turk. One hundred and eighty-eight participants, all US-citizens, took part in the experiment in return for monetary compensation. Of these, we excluded 90 who did not finish all problems or did not get at least one solution correct across the entire task. Most of them finished less than 4 problems. The data from the remaining 98 participants was used in the following analysis.

On average, participants solved 8.4 of the 22 problems presented.

Material

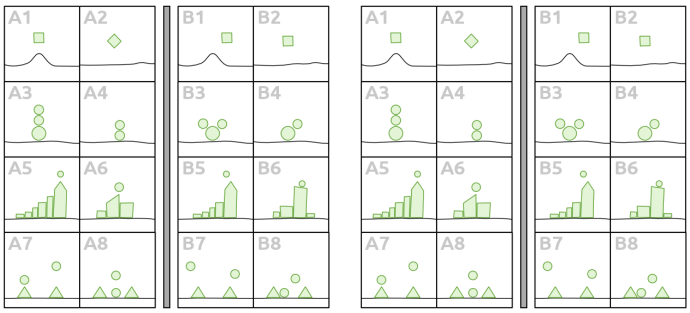
We used the same PBPs as in Experiment 2, but now have four instead of two similarity conditions, each a combination of high or low within-category similarity and high or low between-category similarity. To allow for layouts that make the four similarity conditions clearly distinct from each other, we use 16 scenes from four of the five similarity groups of each PBP during training, which is four scenes more than in Experiment 2.

The scenes are arranged such that in the high within-category similarity condition, scenes of the same category that are presented temporally close to each other are similar to each other. This means during blocked presentation, where within-category comparison is possible within the scene pairs, the scenes inside each pair will be chosen to be similar to each other, while during an interleaved presentation, where within-category comparisons are only possible between successive scene pairs, the scenes will be arranged so that the scenes of successive scene pairs are similar to each other.

Analogously, for the high between-category similarity condition, scenes of different categories that are presented temporally close to each other are chosen to be similar to each other. Since in blocked presentation between-category comparison is only possible between successive scene pairs, scenes will be arranged for high similarity between successive scene pairs. In interleaved presentation, between-category comparisons are possible within the scene pairs, so the scenes in each pair will be chosen to be similar to each other. Figure 29 depicts the arrangement of the scenes for each of the four conditions using PBP 24 as an example. Table 4 lists how many similar and dissimilar comparisons are possible in each condition.

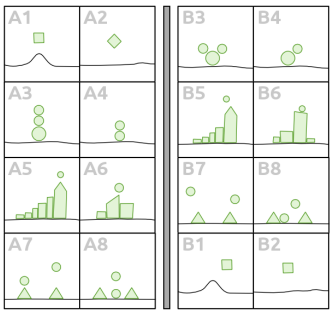
Design

We used a $2 \times 2 \times 2$ factorial design. The study condition *presentation schedule* $\in$ {*blocked*, *interleaved*} $\times$ *within-category similarity* $\in$ {*similar*, *dissimilar*} $\times$ *between-category similarity* $\in$ {*similar*, *dissimilar*} was randomly chosen for each problem in a within-subject manner and balanced for each subject.

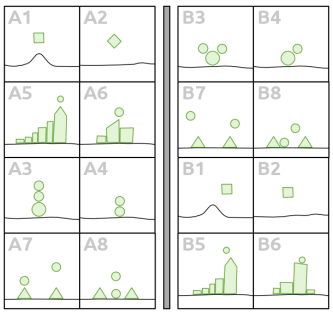


(a) interleaved-sim-sim

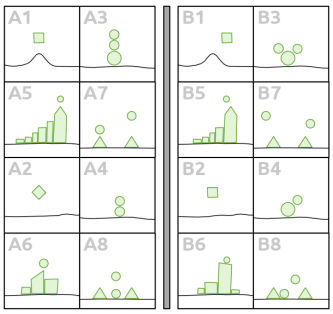
(b) blocked-sim-sim



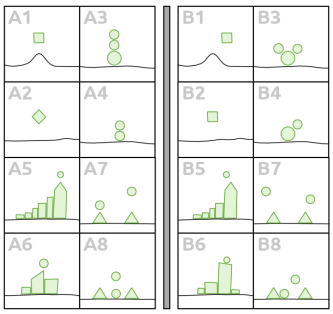
(c) interleaved-sim-dis



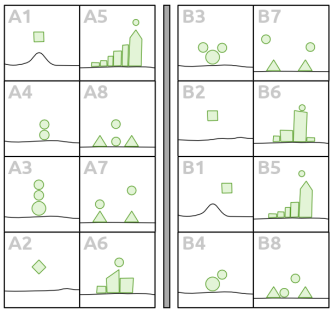
(d) blocked-sim-dis



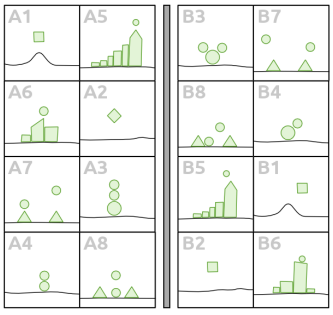
(e) interleaved-dis-sim



(f) blocked-dis-sim



(g) interleaved-dis-dis



(h) blocked-dis-dis

Figure 29: Experiment 3. Arrangement of scenes for the eight conditions interleaved vs. blocked × low vs. high within-concept similarity × low vs. high between-concept similarity. Only the arrangement of the scenes in the upper four rows of each problem is affected, as they are used as training scenes. The scenes in the last row, which are not shown here, are used as test scenes together with two randomly selected training scenes.

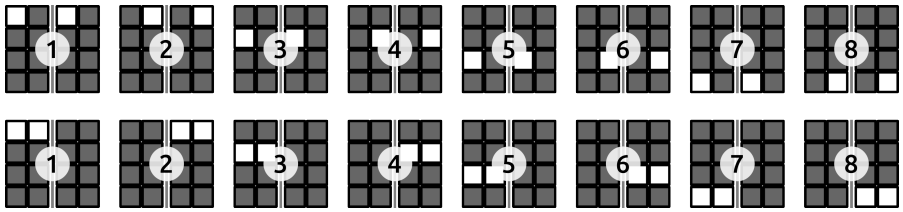


Figure 30: Scheduling for Experiment 3. The positions at which the scenes are shown for blocked (top) and interleaved (bottom) presentation. The subjects proceed manually through the eight states as often as they want.

Procedure

Just as in Experiment 2, the participants were first shown a short introduction to the domain of PBPs, their task and a solved example problem. They then had to solve a series of 22 PBPs presented in an order which was designed to minimize context effects between consecutive problems. For each problem, the participants were presented a sequence of scene pairs through which they could cycle at their own pace. When the participants had seen all scenes at least once, a button appeared on the screen that gave them the option to finish the training for the current problem whenever they liked. The button took them to a page where they had to classify six test scenes, one at a time, as belonging to the left or the right category. The participants were then prompted to type in a free text description of what defined both concepts. After submitting their solution and before continuing with the next problem, they were shown the current problem with all scenes at once and its correct solution.

The material and experimental setup was identical to Experiment 2 with three exceptions. First, the participants were now allowed to cycle through the training scene pairs more than once. This allows participants to revisit scenes and to check new hypotheses on them. After having seen each scene at least once, they could press a button to proceed to the next step whenever they wanted to. Second, the ordering was adjusted to take the more fine-grained similarity conditions into account (Figure 29 and 30). Third, we were showing 16 instead of 12 of the total of 20 scenes per PBP during training. During the classification task, we used the remaining four scenes together with two randomly chosen training scenes as test scenes.

The original experiment is available online at `http://goo.gl/0TrVtB`.

Results

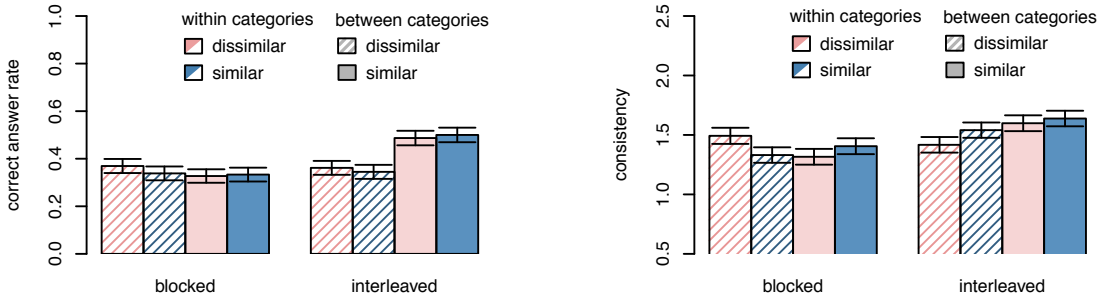
The written solutions of all participants were categorized as correct or incorrect by two trained coders blind to the experimental hypothesis. Cases in which the description of the left and right side was swapped as well as cases with a valid solution different from the official one were counted as correct. The coder agreement was $\alpha = 0.87$. All cases of disagreement were resolved by a third trained coder, also blind to the experimental hypothesis.

CONDITION	PROMINENT COMPARISONS			
	WITHIN		BETWEEN	
	SIM.	DIS.	SIM.	DIS.
int-sim-sim	- (8)	- (8)	8 (8)	- (8)
int-dis-sim	- (-)	- (16)	8 (-)	- (16)
int-sim-dis	- (8)	- (8)	- (4)	8 (12)
int-dis-dis	- (-)	- (16)	- (-)	8 (16)
blo-sim-sim	8 (-)	- (-)	- (16)	- (16)
blo-dis-sim	- (-)	8 (-)	- (12)	- (20)
blo-sim-dis	8 (-)	- (-)	- (-)	- (32)
blo-dis-dis	- (-)	8 (-)	- (4)	- (28)

Table 4: The table depicts how many opportunities each condition provides for making specific types of comparisons side-by-side in a scene pair (left number in each cell) and temporary juxtaposed between two successive scene pairs (right number in brackets). The side-by-side comparison is likely to be much easier to do and therefore to have a bigger influence. The numbers in black are the kinds of comparisons that should be promoted by a condition, while the gray numbers are the kinds of comparisons that should be limited. Compare with the example of an actual scene layout given in Figure 29.

We applied two $2 \times 2 \times 2$ repeated measures ANOVA with *schedule condition*, *within-category similarity condition* and *between-category similarity condition* as factors and a) the accuracy and b) the consistency as defined in Experiment 2 as the dependent variable. For accuracy, this revealed a significant effect of presentation schedule, $F(1,97) = 16.8$, $p < .001$, and of between-category similarity, $F(1,97) = 11.0$, $p = .001$, as well as an interaction between the two factors, $F(1,97) = 19.7$, $p < .001$. For consistency, it revealed a significant effect of presentation schedule, $F(1,97) = 11.1$, $p = 0.001$, as well as a two-way interaction between schedule and between-category similarity, $F(1,97) = 6.6$, $p = 0.01$, and a three-way interaction between all factors, $F(1,97) = 4.7$, $p = 0.03$. There were no other significant effects ($p > .05$). See Figure 31 for a plot of the results.

Additionally to these measures of learning success, we analyzed the time it took participants to solve problems using the measures number of *scene pairs seen* and the logarithm-transformed time. Since the subjects had to look at each of the 16 training scenes at least once, everybody saw at least eight scene pairs per problem while there was no upper bound on how often participants could cycle through the scenes. We used the logarithmic transformation to reduce the influence of cases in which a participant might have taken a short break during solving a problem and to account for the difference of variance between problems of varying difficulty.



(a) accuracy

(b) consistency

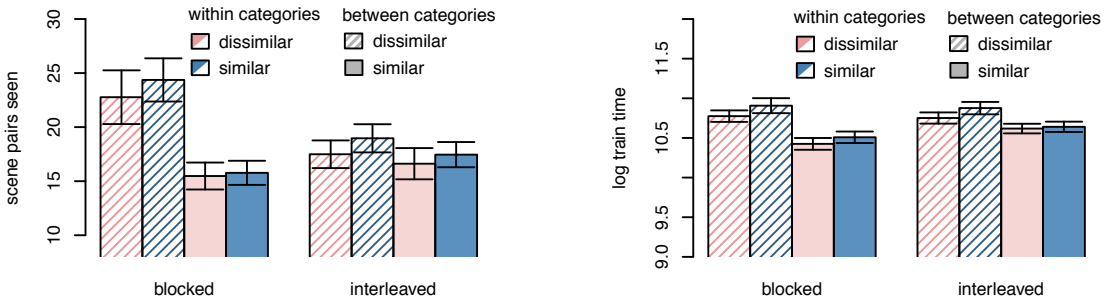
Figure 31: Results for Experiment 3. The left plot shows average accuracy (solution rate) for each of the conditions. The right plot shows average consistency, a measure of how well participants were able to classify test scenes. Both accuracy and consistency were significantly higher in conditions with interleaved presentation and high between-category similarity (scene pairs containing similar scenes from both sides). Error bars represent standard errors.

In this analysis, we only look at the subset of trials that were solved. This results in a number of subjects that do not have at least one problem solved for each of the eight conditions, a situation for which the repeated measures ANOVA is not well-suited. Instead, we apply a generalized mixed effects modeling analysis using the "ezMixed" method of the R package "ez".[2] We applied the mixed effects analysis to the number of scene pairs cycled through and to the logarithmic time to solution, which revealed the significant effects (with evidence of three or more bits) listed in the following table. Figure 32 shows a plot of the results.

DEP. VARIABLE	EVIDENCE
log time	b/w-cat. sim.: 35 bits,
	b/w-cat. sim. – pres. schedule interaction: 3 bits
scene pairs seen	b/w-cat. sim.: 25 bits, pres. schedule: 3 bits,
	b/w-cat. sim. – pres. schedule interaction: 5 bits

The new possibility to view scenes more than once was used 69% of the time. The median number of scene pair views over all problems and participants was 16 which means that typically, each scene was viewed twice.

2 The method computes the likelihood ratio between an unrestricted (with the effect in question) and a restricted model (with the effect in question fixed to zero) and accounts for the additional complexity in the unrestricted model. The likelihood ratio is reported in log-base-2 scale, which means it can be interpreted as "bits of evidence". Three bits of evidence are roughly equivalent to a p-value of 5%. For further details see Lawrence [2013].



(a) scene pairs seen

(b) log train time

Figure 32: Results for Experiment 3. The left plot shows the average number of scene pairs participants chose to look at in each of the conditions. The minimum allowed number was eight. The right plot shows the logarithm-transformed time participants spent on solving. High between-category similarity reduced time and number of scene pairs cycled through. For both plots, only solved trial were taken into account. Error bars represent standard errors.

Discussion

The results above and an inspection of Figure 31 shows that it is the combination of the interleaved schedule and the high between-category similarity that is correlated with a higher rate of correct answers. This is in line with our predictions: The comparison of similar scenes from different categories, which is by far easiest to do during interleaved presentation of high between-category similarity pairs, leads to significantly better learning results.

Despite the expected benefits of low within-category similarity for generalization, we did not find an effect of low within-category similarity, even for the blocked schedule. One possible explanation is that the necessary alignment of the scenes in each pair was too difficult to do for dissimilar scenes. A second possible explanation is that although the blocked presentation introduced a strong bias towards within-category comparisons, the participant might still have tried to build an interrelated characterization of the categories and, therefore, focused on discriminating between the categories by looking for differences between successive scene pairs. The significant benefit of high between-category similarity and the lack of effect of the within-category similarity on the number of scene views for solved problems in the blocked condition supports this hypothesis.

4th Experiment

All experiments so far were done to shed light on the effect of different types of comparisons on category learning by showing a sequence of scene pairs to the learner. In this last experiment, we use a less restricted and more natural way of presentation, in which all scenes of the PBP are shown

simultaneously. The learner is free to choose which scenes he attends to and which scenes he compares to others. We still manipulate the similarity structure of comparisons – in a slightly more indirect way – by arranging the scenes differently across conditions.

In contrast to Experiment 3, we do not specifically promote within- or between-concept comparisons, so the participants are more free to choose the kind of comparison they want to make. For the same reasons as before, we still expect high-between-category and low within-category similarity to positively effect learning performance. To allow a direct comparison of the sequential and simultaneous presentation schedule, we included the best condition from Experiment 3 as an additional condition in this experiment. We had mixed expectations of whether seeing all scenes at once should help or hurt performance. On the one hand, providing all scenes at once allows for more and faster comparisons because the memory constraints don't play as big a role as with the sequential pairs. On the other hand, the simultaneous presentation adds the task of making good decisions on what to look at and what to compare.

Participants

We conducted the experiment on Amazon Mechanical Turk. One hundred forty-three participants, all US-citizens, took part in the experiment in return for monetary compensation. Of these, we excluded 52 who did not finish all problems or did not get at least one solution correct across the entire task. The data from the remaining 91 participants was used in the following analysis. On average, participants solved 11.5 out of the 22 problems presented.

Material

We used the same PBPs as in Experiment 2 and 3 and showed them in the same order (see Table 3). We used 16 training scenes as in Experiment 3 but showed all the scenes simultaneously in four different spatial arrangements. The scenes were aligned so that for the high within-category similarity condition, similar scenes of the same category were placed spatially close to each other while for the low within-category similarity condition, they were placed far from each other. Analogously, adjacent scenes of different categories were similar for the high between-category similarity condition and dissimilar for the low between-category similarity condition. The different spatial alignments are shown exemplarily for PBP 24 in Figure 34. Table 5 lists how many similar and dissimilar comparisons are possible between scenes spatially close to each other in each condition.

The underlying assumption that the different positioning of scenes affects the order in which scenes are most likely attended by the subjects is based on research on visual scan paths. In most situations, people prefer horizontal visual scanning over vertical scanning, where the direction of the

scanning depends on the main reading direction in their cultural context. We can expect western persons to have a strong bias in their visual scanning path towards left-to-right movements and to prefer horizontal over vertical movements (Abed [1991]). Additionally, everything else being equal, it seems likely that a person will prefer comparing two elements close to each other than two element far from each other since it takes less effort in terms of eye movements.

To allow a direct comparison of the sequential presentation schedules of the previous experiments with the simultaneous presentation schedule of the current experiment, we have a fifth condition in Experiment 4 that replicated exactly the best condition of Experiment 3, "interleaved-sim-sim". See Figure 33 for the timing of training scene display in both the simultaneous and the sequential presentation schedule. The test scenes were selected and presented as in Experiment 3.

CONDITION	PROMINENT COMPARISONS			
	WITHIN		BETWEEN	
	SIM.	DIS.	SIM.	DIS.
simult-sim-sim	4 (-)	- (6)	4 (-)	- (-)
simult-dis-sim	- (-)	4 (6)	4 (-)	4 (-)
simult-sim-dis	4 (-)	- (6)	- (-)	4 (-)
simult-dis-dis	- (-)	4 (6)	- (-)	4 (-)

Table 5: The table depicts how many opportunities each condition provides for making specific types of comparisons between scenes that are adjacent horizontally (first number) or vertically (second number in brackets). There is, most likely, a bias to compare scenes that are adjacent, especially horizontally. The numbers in black are the types of comparisons that should be promoted by a condition, while the gray numbers are the kinds of comparisons that should be limited. Compare with the example of an actual scene layout given in Figure 34.

Design

We used a 2×2 factorial design with the factors *within-category similarity* $\in$ {*similar*, *dissimilar*} $\times$ *between-category similarity* $\in$ {*similar*, *dissimilar*}. Additionally to those four conditions in which all scenes were presented simultaneously, there was a fifth sequential condition that replicated the best condition from the previous experiment (interleaved, within-dissimilar, between-similar). One of these five conditions was chosen for each problem presented to a subject in a balanced and within-subject manner.

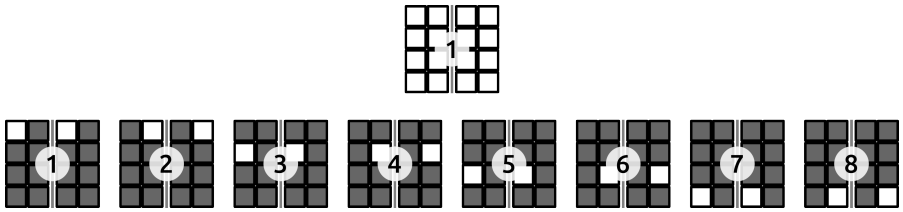


Figure 33: Scheduling for Experiment 4. The figure depicts the timing for training scene presentation in the simultaneous (top) and in the sequential interleaved condition (bottom). The scenes that are shown at a specific stage are marked in white. In the simultaneous condition, all scenes are shown at once. The interleaved condition is identical to the one used in Experiment 3.

Procedure

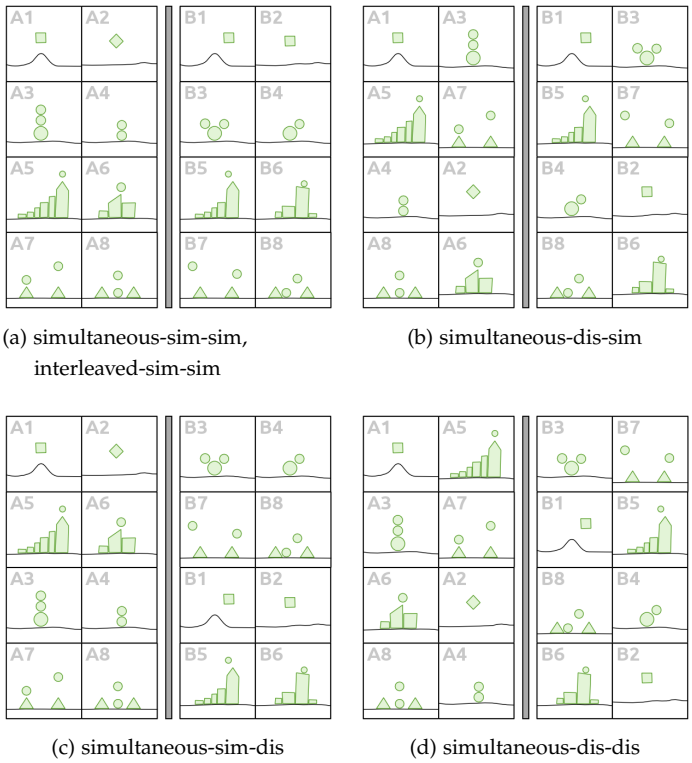
The course of the experiment was identical to that of Experiment 3, except that for the simultaneous presentation schedule, participants could not cycle through scenes pairs but were rather shown all scenes at once and could directly proceed to the next stage. The original experiment is available online at `http://goo.gl/066U59`.

Results

The written solutions of all participants were categorized as correct or incorrect by two trained coders blind to the experimental hypothesis, same as in Experiment 3. Cronbach's α was 0.79. All cases of disagreement were resolved by a third trained coder, also blind to the experimental hypothesis.

We applied two separate 2×2 repeated measures analyses of variance on all problem instances presented with the simultaneous presentation schedule to analyze the effect of the factors *within-category similarity condition* and *between-category similarity condition* with first the proportion of correct answers (accuracy) and second the consistency measure defined in the previous experiment (consistency) as dependent variables. For accuracy, we found a significant effect of *within-category similarity* $F(1, 90) = 8.07$, $p < .01$, while *between-category similarity* had no significant effect and there was no significant interaction. For consistency, we found a marginally significant effect of *within-category similarity* $F(1, 90) = 3.78$, $p = 0.55$ and no effect or interaction with *between-category similarity*. See Figure 35.

To analyze the effort that the solution of a correctly solved problem required, we used the logarithm-transformed time spent on solving as we did in Experiment 3. The number of iterations through a problem's scene pairs can't be used for the simultaneous condition. We used the same type of mixed effects analysis as before and applied it to all solved trials in the simultaneous conditions. It revealed an effect of between-category similarity with 9.6 bits of evidence. All other effects had negative evidence. Figure 36 shows a plot of the data.



(a) simultaneous-sim-sim, interleaved-sim-sim

(b) simultaneous-dis-sim

(c) simultaneous-sim-dis

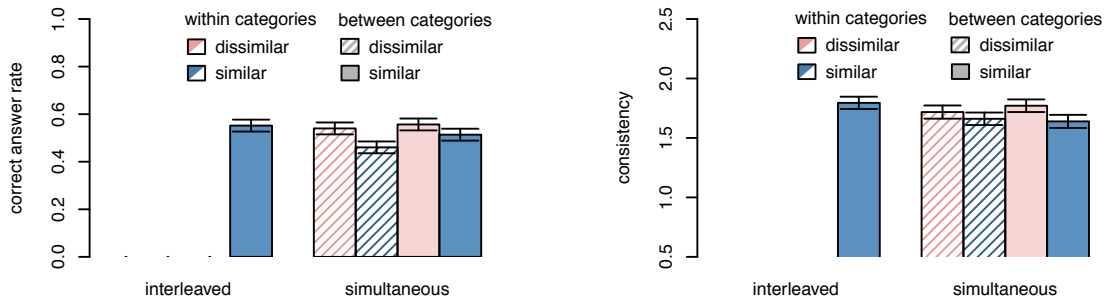
(d) simultaneous-dis-dis

Figure 34: Experiment 4. Arrangement of scenes for the four simultaneous conditions and the one interleaved condition for the example of PBP 24. The interleaved condition has just presented in with high within- and high between-category similarity, while for the simultaneous condition, both similarity types were varied between low and high.

We used four planned t-tests to compare the accuracy in the interleaved condition with the performances in each of the four simultaneous conditions. For the condition "simultaneous-sim-dis", which is the most difficult of the simultaneous conditions, we got a significant difference $t(90) = 3.1$, $p = 0.01$, where p was corrected using the Bonferroni method to account for multiple comparisons. There was no significant difference in reaction times between the interleaved and the simultaneous presentation schedules ($p > .05$).

Discussion

This experiment resulted in two important findings. First, for the simultaneous presentation, low within-category similarity was correlated with better classification performance, which is in line with the body of research that shows low similarity (higher variance) to be beneficial for category learning. We did find the expected positive effect of high between-category similarity on solution time, although not on accuracy. This is different but complementary to the results of Experiment 3 and I will provide an interpretation of the joint results in the general discussion below.



(a) accuracy

(b) consistency

Figure 35: Results for Experiment 4. For the simultaneous condition, the accuracy was significantly higher in conditions with low within-category similarity (similar scenes on each side were placed far from each other). The same effect, though only marginally significant, was found on the consistency of test answers. The interleaved condition replicated the best condition of Experiment 3 and lead to significantly higher accuracy than the worst simultaneous condition, despite the fact that participants only saw two scenes at a time. Error bars represent standard errors.

Second, the interleaved presentation of PBP scenes in the high within-, high between-category similarity condition was significantly better suited for learning than the simultaneous presentation with high within- and low between-category similarity. Although we are comparing the best of the sequential conditions with the worst of the simultaneous conditions, this is still a noteworthy result: If not for generating solution hypotheses, then at least for validating them the simultaneous condition should provide an advantage over the sequential one, in which one never gets to see how all the pieces fit together. Nevertheless, there is a simultaneous condition that is better than a sequential one, which strongly suggests that the process of selecting which scenes should be attended to is a substantial and non-trivial part of the learning task. Presenting just two scenes at a time, which are cho-

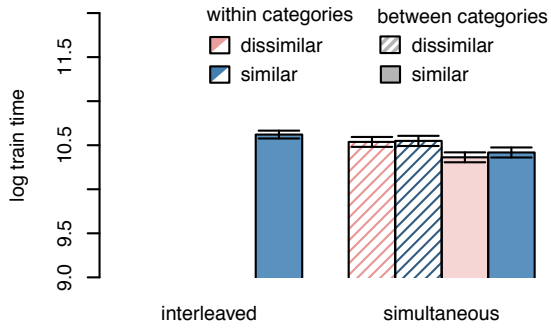


Figure 36: Results for Experiment 4. The plot shows the logarithm-transformed time participants spent on correctly solved trials. Participants were significantly faster in conditions where similar scenes from different categories were placed close to each other. Error bars represent standard errors.

sen to be beneficial to the learning when compared, can promote efficient perception and reasoning strategies which lead to better learning results.

4.4 GENERAL DISCUSSION

One general observation from all four experiments is that solving PBPs is not an easy task for people. On average, each person solved around half of the 22 problems correctly, while typically spending between two and three minutes per problem. This base degree of difficulty is well suited for the experiments, as it avoids ceiling effects.

Over the course of the four experiments, we manipulated whether the scenes in each problem were shown in pairs or all at once, whether similar scenes were presented close or distant from each other and whether within or between category comparisons could be made directly.

There are three major effects we found across the experiments. First, comparing similar cases across the two categories is beneficial to learning (Experiments 2, 3 and 4). Second, comparing dissimilar cases within one category is beneficial to learning (Experiment 4). Third, seeing all category members at once is worse for learning than seeing them in pairs if the paired presentation juxtaposes similar scenes from different sides while the simultaneous presentation does not (Experiment 4).

The first result follows naturally from the prediction based on the notion of discriminative contrast, as discussed in Carvalho and Goldstone [2013] and Kang and Pashler [2012], which states that direct comparison of instances from different categories highlights their differences, together with the insight that comparing similar instances is especially effective since there are fewer superficial differences and the alignment of instances is easier (see Winston [1970] on "near misses" as well as Markman and Gentner [1993]).

The second result is predicted by theories like "conservative generalization" by Medin and Ross [1989] that attribute the advantage of low similarity within-concept comparisons to having less superficial similarities that can be mistaken for the defining similarities.

The question remains of why we did not find both effects in both experiments. My answer is based on the insight that there are different approaches people use when learning concepts. Both Goldstone [1996] and Jones and Ross [2011] argue that learners might either focus primarily on what a category is like (using "inference learning" to build a "positive" or "isolated characterization") or focus on how a category is different from other categories (using "classification learning" to build an "interrelated characterization"). What kind of comparisons are most informative strongly depends on which of these approaches is pursued by the learner. While for building a positive characterization within-category comparisons are of central importance, for building an interrelated characterization the between-category comparisons are more useful.

An interpretation of the results consistent with the different findings across the experiments is that in Experiment 3, participants were focusing on between-category comparisons because they were trying to build an interrelated characterization of the concepts. Naturally, participants would make few within-category comparisons, explaining why the within-similarity condition did not play a significant role. In Experiment 4, subjects might have tried to build a positive characterization instead, and consequently did not pay as much attention to between-category comparisons, explaining the effect of within-category similarity and the lack of an effect of between-category similarity.

There are a couple of reasons why it is plausible to assume that participants chose to look for differences in the sequential presentation and for commonalities in the simultaneous presentation. In the simultaneous presentation, all instances of one category were grouped together on one side of the scene, a layout that allows for quickly scanning all instances to efficiently check for reoccurring patterns and shared features. This assumption is supported by the results of the eye-tracking study discussed earlier. When presented with just two scenes at a time, however, looking for differences might appear as the more efficient strategy: due to the open-ended feature space of the PBP domain, participants had to identify or construct relevant feature dimensions as a major part of the task. Comparing similar scenes from different concepts highlights such feature dimensions, an advantage that comparing dissimilar scenes within one concept does not provide (Carvalho and Goldstone [2013]).

In the experiments described in this chapter, we replicated known effects in human category learning on the new PBP domain. We went beyond existing research with the detailed analysis of the joint effects of within-category similarity and between-category similarity on concept learning performance for blocked, interleaved and simultaneous presentation of examples. In the next chapter, I look into whether these factors effect the PATHS model in the same way as they effect humans.

5

MODEL PERFORMANCE ON PBPS

The PATHS model was designed as a cognitive model and targets a particular slice of human cognition, rule-based concept learning, at a particular level of abstraction, a symbolic hypothesis testing account combined with the perception of scene features. After laying out the architecture of the model in Chapter 3 and collecting human learning data on PBPs in Chapter 4, I now compare the performance of the model to that of humans to see, whether the model successfully captures aspects of human concept learning. This chapter compares human and model performance along several dimensions: the ratio of correct solutions, the time it takes to get to a solution, and the influence of different presentation conditions. The performance data of the model is generated by running it on a replication of the experimental setup of Experiment 3 from the previous chapter. I used Experiment 3, since it implements a powerful manipulation of the kinds of comparisons that are possible, it was the experiment with the largest effect sizes and it presents the scenes in a predefined order, that can be replicated precisely for the model.

This chapter is divided into three sections. In the first section, I describe how the PATHS model was set up to run on the same experimental conditions as participants worked with in Experiment 3. In the second section, I analyze and visualize the data collected from the model runs, and in third section, I interpret and discuss the data.

5.1 EXPERIMENTAL SETUP

In order to compare the performance of the PATH model with that of human participants, I replicated the setup of Experiment 3 for the model. As in Experiment 3, each of the PBP were presented to the model as a sequence of scene pairs. All eight level combinations of the factors "presentation schedule", "within-category similarity" and "between-category similarity" were used as conditions, which determined the order in which scenes were shown. Additionally, the order of the PBP rows was randomized in each trial, which does not influence the similarity structure. Just like the human participants, the model decides when to proceed to the next scene pair and can submit a solution at any time once it has seen all scenes.

There are, however, a number of things that differed from the human experiment. First, the model never submits an incorrect solution, unlike the human participants. Second, the model will give up on a trial if it did not find a solution after a fixed number of actions, which was set to 2500. The participants, on the other hand, were able to give up at any time as long

as they had looked at each scene at least once. Third, the order in which the problems were shown to the model was arbitrary, since the model had no built in mechanism for carrying over activity or experience from one trial to another – it started each trial in the same initial state. Finally, the model was only applied to a subset of the problems that humans worked with. Those problems were the ones that the model was in principle capable of solving, which was problem 2, 4, 8, 11b, 12, 13, 16, 18, 20, 22, 26, 30, and 31. Additionally, the model worked on new problems 35 and 36, which were created to test its ability to work on previously unseen problems. The remaining problems from Experiment 3 and 4 (problems 9, 19, 21, 23, 24, 27, 28, 32 and 33) could not solved by the model because it lacked relevant perceptual capabilites.[1]

Table 6 lists the parameter settings I used for the model. The parameters were not fit to the data, instead they were all set to fixed values before running the experiment. The model could perceive the following features. Object attributes: circle, square, triangle, rectangle, small, large, moves, unstable, stable, single, top-most, on-ground, bottom, top, left, right and movable-up. Group attributes: count, touching, close and far. Object relationships: touch, close, for, on-top, right-of, left-of, above, below, beside, supports, hits, gets-hit and collide. Using these settings and perception capabilities, the model was run 100 times on each problem in each condition, resulting in a total of $100 * 8 * 15 = 12,000$ trials.

PARAMETER	VALUE
action priors	perceive: 0.6, check-hyp: 0.25, combine-hyp: 0.1, recombine-hyp: 0.05
time priors	start: 0.67, end: 0.33
obj. attr. priors	moves: 2, unstable: 1.5, top-most: 1.5, other: 1
group attr. priors	touching: 1.5, close: 1.25, other: 1
feature priors	circle, square, triangle, rectangle, small, large: 3; moves, unstable, stable: 2; other: 1

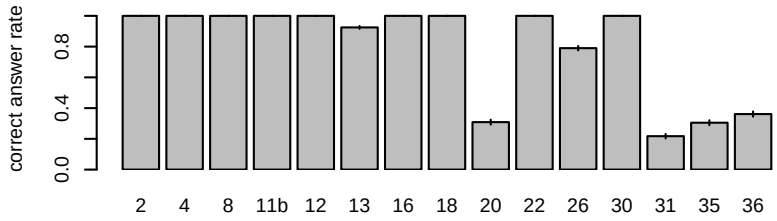
Table 6: Model parameter values used in the experiment.

Additionally, the model was run on another 12,000 trials using a more elaborate way of calculating the probabilities of selecting a specific feature for perception. Instead of just using the prior probability, the prior was updated analogously to the way the probability of selecting a specific object for perception is calculated. This resulted in the model selecting those features that are used in prominent solution hypotheses with a higher probability.

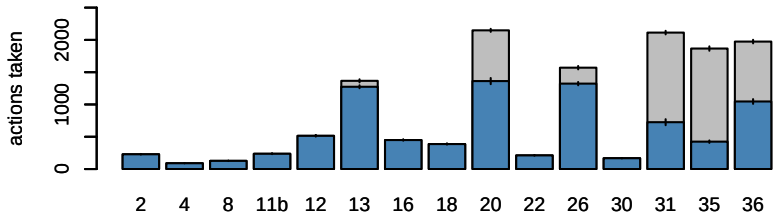
1 The decision of which capabilities I endowed the model with was done before analyzing and comparing the model performance to human performance. I discuss how the model could be extended to solve the remaining problems in Section 5.3.

5.2 RESULTS

Figure 37 shows the model's rate of discovering a solution – its accuracy – in sub-figure (37a) and how many actions it used on average until it found a solution or gave up in sub-figure (37b). I choose a constant action cap at 2500 actions and the model stopped the search if no solution was found after that number of actions. With this action cap, the model reliably found a solution for most of the problems. Because of these ceiling effects, I will not use the model's accuracy directly in subsequent analysis.



(a) rate of solving

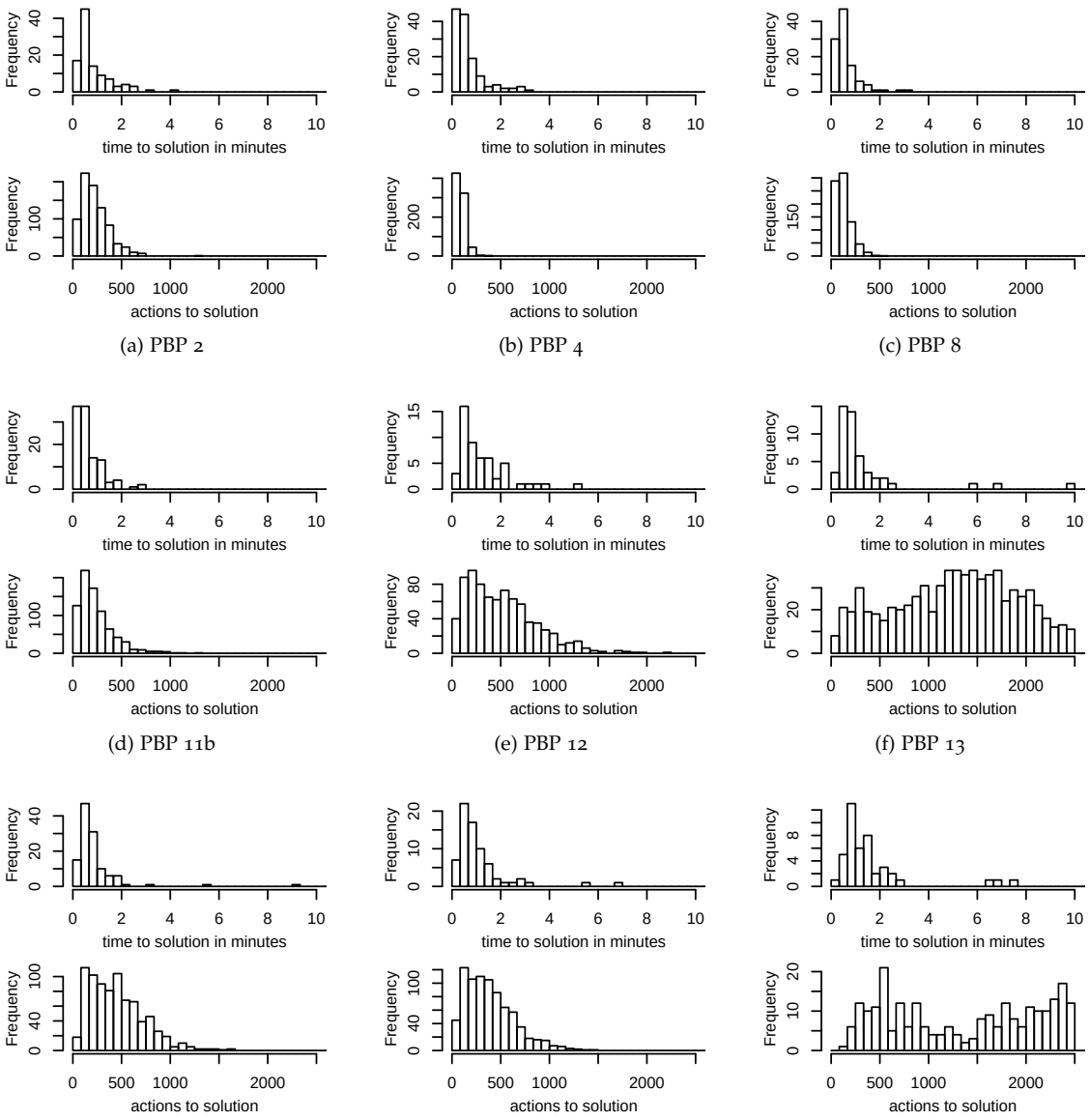


(b) number of actions

Figure 37: (a) The model's rate of finding a solution with a fixed cut-off at 2500 actions. (b) The average number of actions the model took to find a solution (blue) or until it either found a solution or aborted the search at 2500 actions (gray). Error bars represent standard errors. The x-axes show the problem numbers.

Reaction Time Distribution

To gain an overview of the effort it took participants and the model to solve PBPs, I use the time to a solution as the measure of effort for human participants, and the number of actions to a solution for the model. The benefit of the number of actions is that it abstracts from the specific hardware, programming language and differences in the runtime of specific perception actions. I will refer to both the solution time and action count as reaction times or RTs. Figure 38 and 39 show histograms of the RT distributions across participants and model runs, taking only successfully solved trials into account. For the human data, I used the combined data from Experiment 3 and 4. I discuss the data in the next section.

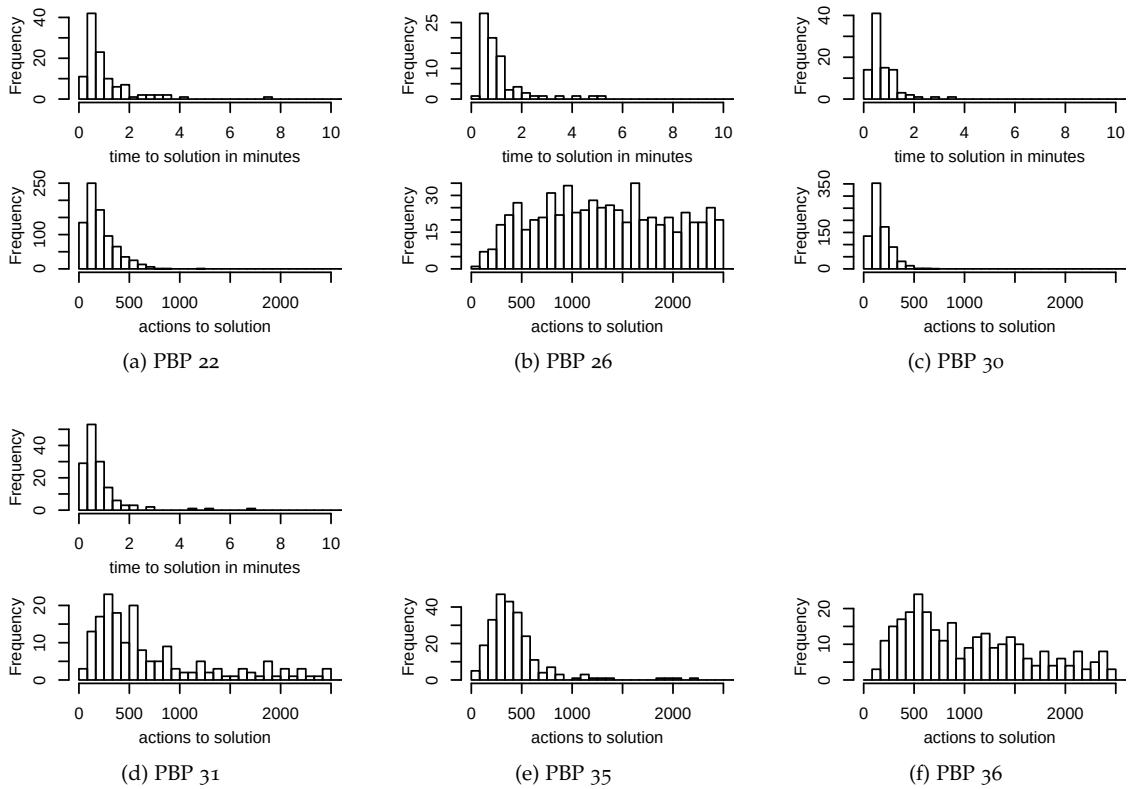


(a) PBP 2 (b) PBP 4 (c) PBP 8

(d) PBP 11b (e) PBP 12 (f) PBP 13

(g) PBP 16 (h) PBP 18 (i) PBP 20

Figure 38: Solution time histograms for PBP 2 to PBP 20. The upper plot in each pair shows the human solution times in seconds, while the lower plot shows the number of actions the model used. Both for humans and the model, only successfully solved trials were considered for the histogram



(a) PBP 22 (b) PBP 26 (c) PBP 30

(d) PBP 31 (e) PBP 35 (f) PBP 36

Figure 39: Solution time histograms for PBP 22 to PBP 36. The upper plot in each pair shows the human solution times in seconds, while the lower plot shows the number of actions the model used. PBP 35 and 36 were not used in the human study and therefore the corresponding plots don't show human solving times. Both for humans and the model, only successfully solved trials were considered for the histogram.

Performance Correlation per Problem

I now compare the average reaction times of human participants and the model on each problem. While further analysis of reaction times is useful for model comparison and for illuminating effects of the presentation condition, the RTs have several problematic aspects. First, the RT data gathered from human participants contains outliers. Second, the RTs in unsuccessful trials are difficult to interpret and result from very different mechanisms in the model and the human participants. The model always searches for a solution until a fixed maximum number of actions, while humans may give up at any time or might submit a wrong answer.

I applied the following techniques to address these problems. First, I excluded all trials in which the reaction time was longer than ten minutes. There are no fast RT outliers since the correct answer could not be guessed.

Second, I took a look at the RTs of successfully solved trials and introduced a difficulty measure that takes both accuracy and reaction time into account.

Figure 40 shows a scatter plot of human versus model reaction times in solved trials. The Pearson product-moment correlation coefficient for the per-problem RTs between humans and the model is 0.77, with $t = 3.96, df = 11, p-value = 0.002$. The 95% confidence interval for a two-sided alternative hypothesis is 0.37 to 0.93.

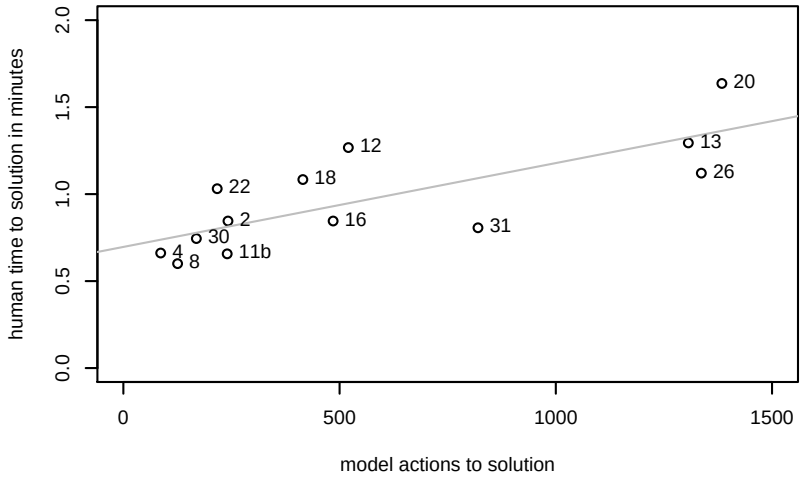


Figure 40: Plot of the average human and model RTs for averaged per problem using only successfully solved trials. The gray line shows a linear fit of the data.

In order to have a measure that takes both the reaction times and the solution rate into account while treating failed trials in the human and model data similarily, I used the difficulty score defined below. The difficulty score combines accuracy and RT information by setting the RT of all unsolved trials to ten minutes for the human data. Any trials that took longer than ten minutes are also set to ten minutes. The model data already has all unsolved trials set to a value of 2500 actions and remains unchanged.

$$\text{difficulty} = \begin{cases} \min(10, \text{RT}), & \text{if solved} \\ 10, & \text{if unsolved} \end{cases}$$

Figure 41 shows a scatter plot of human versus model difficulty scores. The Pearson product-moment correlation coefficient for the per-problem difficulty scores between humans and the model is $r = 0.31$, with $t = 1.0856, df = 11, p-value = 0.3$. There is a single problem, PBP 31, for which the difficulty for humans and for the model was very different. I look into potential reasons for the mismatch on this particular problem at the end of this chapter. If we remove problem 31 from the data, we get a correlation of $r = 0.74$, with $t = 3.4, df = 10, p-value = 0.006$ and a 95% confidence interval for a two-sided alternative hypothesis of 0.28 to 0.92.

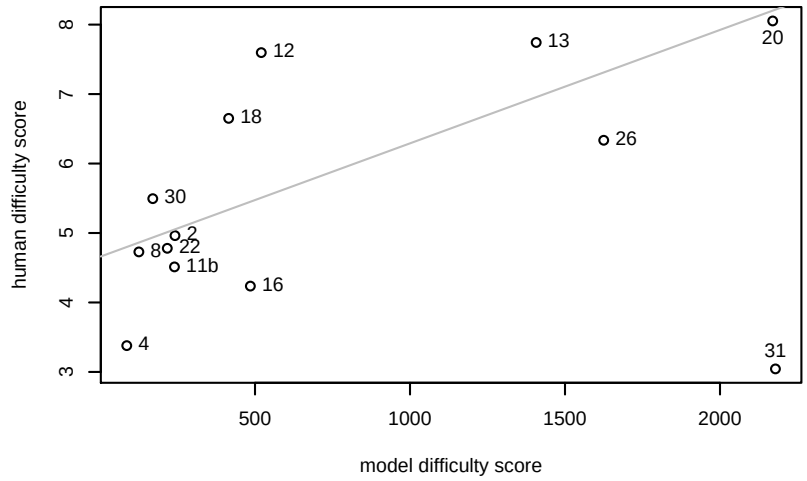


Figure 41: Plot of the difficulty score for humans and the model averaged per problem using all trials. The difficulty score is the average RT with RTs of unsolved trials set to ten minutes or 2500 actions, respectively. The gray line shows a linear fit to the data without problem 31.

Influence of Presentation Condition

In order to analyze the influence of the presentation condition on reaction times, I averaged RTs across problems. It is a well-known effect that with tasks of varying difficulty such as PBPs, the mean and variance between the RT distributions will vary between problems. In order to account for this heterogeneity of variance, I log transform both the RTs of successful trials and the difficulty scores. I use the same conservative removal of outliers from the human data as above, disregarding those trials that took longer than 10 minutes to answer. The log RTs were calculated on the RTs in milliseconds.

I applied a $2 \times 2 \times 2$ repeated measures ANOVA to trials with those problems that could be solved by both human participants and the model; PBP 35 and PBP 36 which were not shown to humans are excluded. The eight presentation conditions are decomposed into the three 2-level factors presentation schedule (blocked vs interleaved), the between-category similarity (similar vs dissimilar), and within-category similarity (similar vs dissimilar). The ANOVA with log RT of the successful trials of the model as dependent variable showed a main effect of presentation schedule, $F(1,99) = 121.4$, $p < 0.001$ and of within-category similarity, $F(1,99) = 14.4$, $p < 0.001$. There was an interaction between presentation schedule and within-category similarity $F(1,99) = 6.7$, $p = 0.01$, as well as between presentation schedule and between-category similarity $F(1,99) = 10.4$, $p = 0.002$. There were no other significant effects ($p > .05$). See Figure 42 for a diagram of the data.

Using the model difficulty score as a dependent variable, the respective ANOVA showed the same significant main effects and interactions as above. The respective values were $F(1,99) = 127.4$, $p < 0.001$ (main: presentation schedule), $F(1,99) = 16.5$, $p < 0.001$ (main: within-category similar-

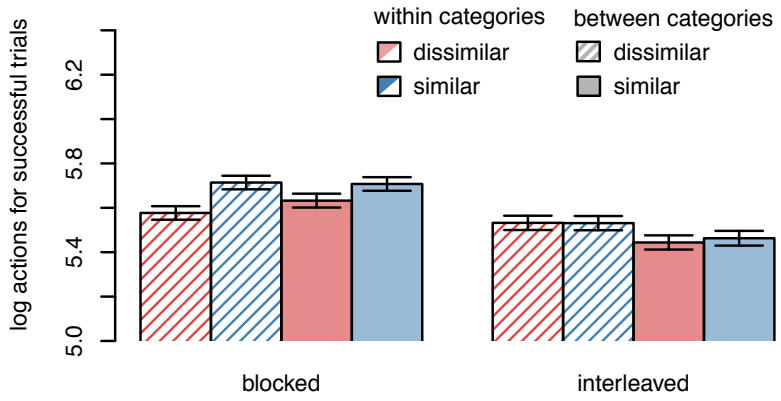


Figure 42: The model's logarithm-transformed number of actions the model took in average on successful trials for the eight conditions. Error bars represent standard errors.

ity), $F(1, 99) = 8.7$, $p = 0.004$ (presentation schedule – within-category similarity interaction), and $F(1, 99) = 21.0$, $p < 0.001$ (presentation schedule – between-category similarity interaction). There were no other significant effects ($p > .05$). See Figure 43 for a diagram of the data.

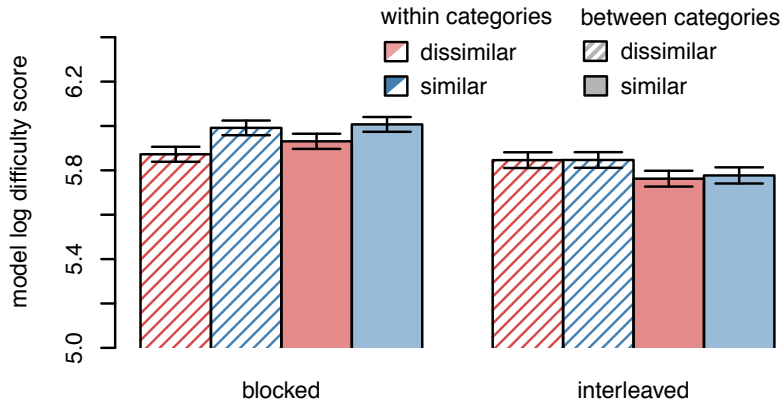


Figure 43: The models logarithm-transformed difficulty score for the model by condition, averaged across all trials. Error bars represent standard errors.

To re-analyze the human data, we cannot use a repeated measures ANOVA, since we are looking at only a subset of the problems the participants worked on. This results in a number of participants that do not have at least one problem solved for each of the eight conditions. I also applied the same generalized mixed effects modeling analysis using the "ezMixed" method of the R package "ez" as in the previous chapter (Lawrence [2013]).

Table 7 lists the bits of evidence found for main effects and interactions with three or more bits of evidence which is roughly equivalent with an p-value of 0.05. This was done for the human data of Experiment 3, which was presented in a sequence of pairs, and for Experiment 4, in which the

scenes were presented simultaneously. Figure 44 and 45 show a bar plot of the data for the logarithmic RT and the logarithmic difficulty, respectively.

EXP	DEP. VARIABLE	EVIDENCE
3	log RT for solved	b/w-cat. sim.: 16 bits
3	log difficulty	b/w-cat. sim.: 25 bits, pres. schedule: 15 bits, interaction of the above: 10 bits
4	log RT for solved	b/w-cat. sim.: 19 bits
4	log difficulty	b/w-cat. sim.: 11 bits, w/i-cat. sim.: 10 bits

Table 7: This table lists the evidence found in a mixed effects modeling analysis on the logarithmic reaction time for solved trials and on the logarithmic difficulty score for all trials. The analysis was done on the human data from Experiments 3 and 4 using the subset of problems the model worked on.

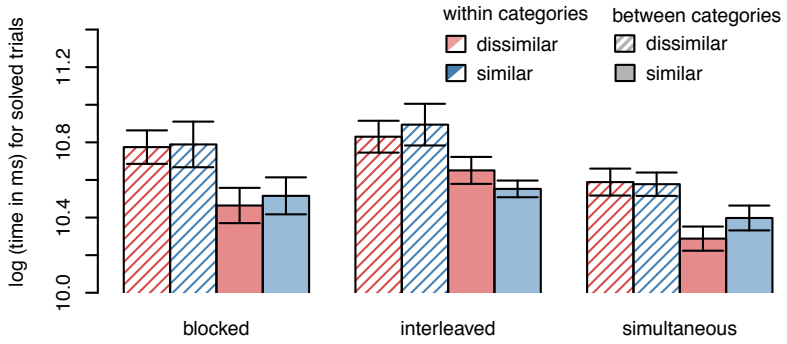


Figure 44: Human participants average logarithm-transformed reaction times on successful trials for the eight conditions. Combined data from Experiment 3 and 4. Error bars represent standard errors.

Efficiency

I have argued that the perception of features in a scene takes cognitive effort and should not be treated as being "free" in models of concept learning. When we instead assume a cognitive cost to extracting feature values, an iterative process of perception that is tightly connected to the rule-construction process can allow a model to learn concepts more efficiently without having to perceive all features in a scene. I analyze this aspect of the PATHS model's performance from two perspective. First, I compare the number of features in a PBP that the model could potentially perceive, given its perceptual capabilities, to the number of features it actually perceives. Second, I analyze how the number of feature *types* that the model is capable of perceiving influences how long it takes the model to solve a PBP.

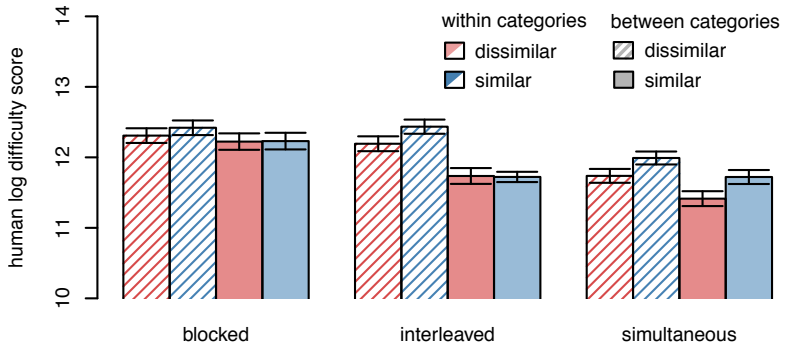


Figure 45: Human participants average logarithm-transformed difficulty scores across all trials for the eight conditions. Combined data from Experiment 3 and 4. Error bars represent standard errors.

In the following analysis of the number of perceivable and perceived features, the number of perceived features reflects the how many feature values the model extracted from a scene. Once a feature is perceived by the model, it's value is cached internally. If the feature value needs to be accessed again, for example when a hypothesis is checked on the scene, the value is retrieved from the cache. This retrieval does not count towards the number of perceived features in the subsequent analysis.

The number of perceivable features in a PBP is the sum of the number of perceivable object attributes, the number of perceivable group attributes and the number of perceivable object–object relationships in all scenes. The model can perceive $F_o = 30$ types of object attributes, $F_g = 7$ types of group attributes and $F_r = 23$ types of relationships. In these figures, features that potentially vary over time, like relative position, were counted twice since the model can perceive them at the start and the end of the simulated unfolding of physical events. With N_o denoting the number of objects in a scene, the number of non-symmetric relationships is $N_r = N_o \cdot (N_o - 1)$, and the number of possible groupings of objects in a scene is $N_g = 2^{N_o} - 1$. We can now compute the number of possible perceptions P for a PBP as

$$P(\text{PBP}) = \sum_{s \in \text{scenes}} \Big(F_o N_o(s) + F_r N_r(s) + F_g N_g(s) \Big) \tag{4}$$

Figure 46 shows the average number of perceived features in relation to the number of perceivable features for each PBP the model solved. Trials in which the search was stopped without success after 2500 actions are included in the plotted averages, too.

A second perspective on the efficiency of the model is to look at how the number of feature *types* that the model is capable of solving influences how fast it solves a problem. For this analysis, I ran different instantiations of the PATHS model on PBP 26. Each instantiation had a different number of feature types that it was capable of perceiving in the scenes. The simplest model instantiation was only able to perceive the three feature types "circle",

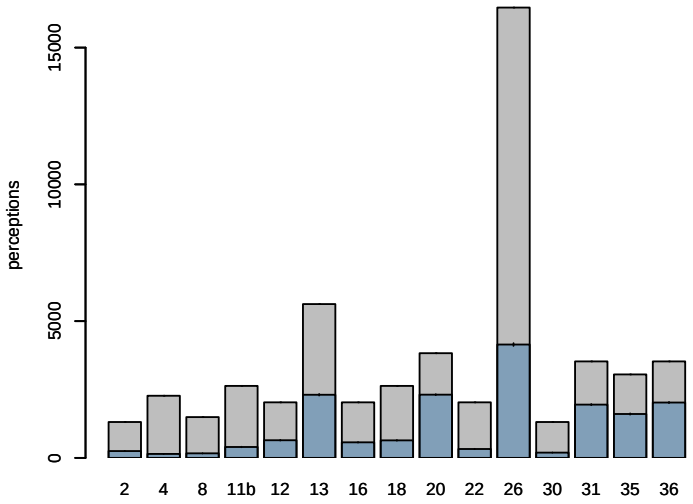


Figure 46: The gray bars represent the number of perceivable features in the PBPs. The blue bars show the average number of features that were perceived by the model across all trials.

"right of" and "left of", which are sufficient to solve PBP 26. Each additional instantiation can perceive one more feature type than the previous one, up until the last instantiation which is identical with the unmodified PATHS model. I chose PBP 26 ("circle moves to the left vs. right") for this analysis for several reasons. First, it is the problem with the most objects and, consequently, a large number of features could be perceived on the scene. Second, the solution of the problem requires the combination of at least two feature types, so that the model had to combine hypotheses to solve the problem. Both aspects lead to a large feature- and rule-space.

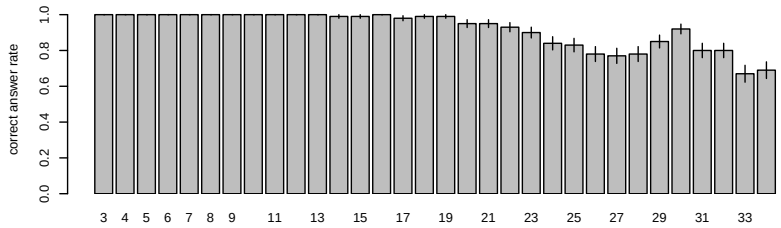
Figure 47 shows the accuracy and the difficulty score for the model working on PBP 26 with different numbers of feature types that it is capable of using. Of the feature types available to the model instantiations, the first three features, as well as feature 29 and 30 ("left" and "right") could be used to construct a solution for PBP 26. Features 13, 21 and 31 ("moves", "top-most" and "unstable") were potentially useful in that they increased the saliency of the circle object when they were perceived on it.

Increasing the number of available feature types does not only increase the number of perceivable features in the PBP scenes, but also the number of hypotheses that can be constructed. The number of hypotheses that the PATHS model can in principle construct is

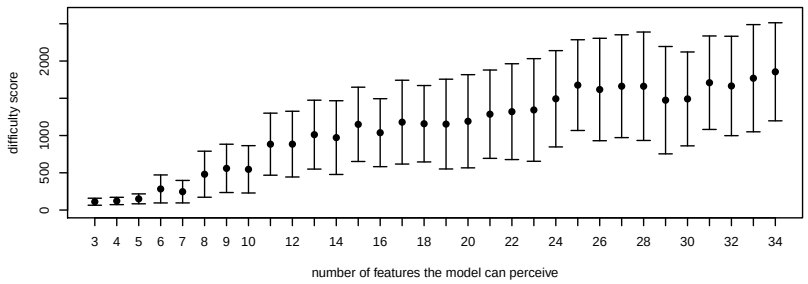
$$H = (F_o + F_r F_o^n + F_g)^m, \tag{5}$$

where F_o, F_r and F_g are the number of object attribute types, object relationship types and group attribute types that are available to the model. The parameter n restricts the number of object attributes that can be used in the reference object selector of a relationship matcher. The parameter m

restricts the number of feature matchers that can be used in a hypothesis. For the original PATH model, the number of constructible hypothesis is $H = 6.4 * 10^7$ for $n = 2, m = 2$ and is $H = 4.1 * 10^{15}$ for $n = 3, m = 3$. The PATHS model does not have a predefined cutoff for the complexity of constructed hypotheses. Instead, more complex hypotheses are less likely to be selected in the model's actions.



(a) rate of solving



(b) difficulty score

Figure 47: The plot shows the performance of different instantiation of the PATHS model working on PBP 26. The x-axis in both plots shows the number of feature types that a particular instantiation was capable of extracting from a scene. In the top plot, error bars represent standard errors while in the bottom plot, they represent standard deviation.

5.3 DISCUSSION

One directly apparent outcome is that the model can solve all of the 13 problems for which I chose to implement the necessary perceptual capabilities. The model was also able to solve two additional problems 35 and 36 that were created after the human studies and model implementation were finished in order to test the model's capability to work on novel problems. Most of the original 13 problems are very reliably solved within 2500 actions; two of them were solved in less than half of the trials (see Figure 37a). In general, the solution rate of the model was much higher than the solution rate of human subjects. The model was not able to solve nine of the 22 problems that were presented to human subjects, because of the limited number of perceptual capabilities I endowed the model with. Adding capabilities to allow the model to solve those remaining problems would be straightforward for some, but would require adjustments to the core system for others.

Table 8 gives an overview of what would have to be added to the system to enable it to solve each of the remaining problems. I expect that adding those capabilities to the model would not have significant effects on the results.

PBP	REQUIRED ADDITIONAL CAPABILITIES
The following would require no modification of the core system.	
21	collisions with ground; measure collision strength
23	collisions with ground
27	stability of a whole group of objects
28	rotational stability
32	object rotation
The following would require minor modifications of the core system.	
9	movement direction; same/different attribute
19	type of motion; event occurrence at arbitrary time
33	spatially defined groups, group integrity over time
The following would require major modifications of the core system.	
19	type of motion and reasoning over event sequences
24	"noisy physics": perform several simulations with slightly different initial conditions and do statistics over the results
27	causality and reasoning over event sequences

Table 8: This is an overview of the capabilities that would have to be added to the model to enable it to solve all 22 problems solved by human participants in Experiments 3 and 4. PBP 19 and 27 appear twice, as differently general solutions would require differently complex adjustments to the model.

In the histograms of per-problem reaction times, the distributions for participants and the model are both skewed to the right (Figure 38 and 39). While for most problems, the distributions of participants and model look qualitatively similar, clear differences are visible for the problems 13, 20 and 26, which all are problems that the algorithm could not reliably solve within 2500 actions. One difference in the underlying process generating the RTs is that while the distribution for the model directly reflects the number of actions required to get to a solution, the time distribution for the human participants is a superposition of two distributions, the time to assumed solution and the time before giving up. While it is difficult to draw any strong conclusion from the histogram data, it seems plausible that the lack of an adaptive stochastic decision on when to give up in the model is at least in part responsible for visible discrepancies.

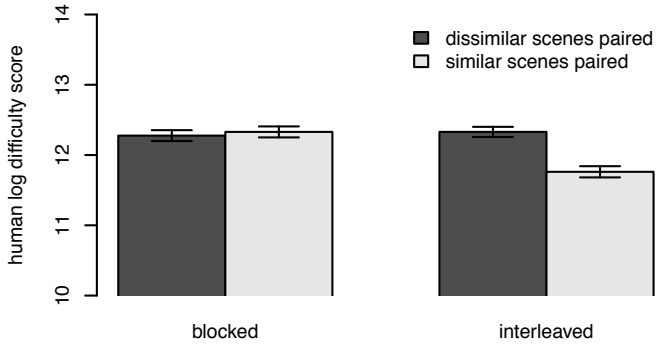
A third observation is that the model is efficient in its solution process in the sense that it perceives only a fraction of the features that are present in a scene (see Figure 46). Additionally, the model scales well in the number

of feature types that it can perceive. Increasing the number of feature types that the model is capable of perceiving increases the number of perceivable features in the scenes linearly for object attributes, quadratically for relationships, and exponentially for group attributes (see Equation 4 on page 104). The number of hypotheses rules that the model can form increases with $O(n^4)$ in the number of object attributes, and quadratically in the number of relationships and group attributes when the hypotheses are restricted to at most two feature matchers per hypothesis and per reference object selector (see Equation 5 on page 105). Despite the quick growth of both the number of perceivable feature and constructible rules with a growing number of available feature types, the number of actions the PATHS algorithm takes to solve PBP 26 increases only linearly with the number of feature types (see Figure 47). This demonstrates the ability of the PATHS model to quickly converge towards the right parts of a large rule and feature search-space.

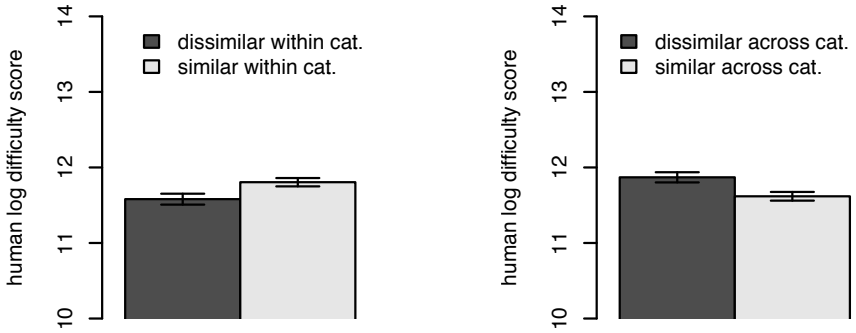
Agreements with Human Results

I compared the difficulty of problems for humans and the model using two measures: the log reaction time (or log action count) on solved trials and a difficulty measure which combines the accuracy and reaction times. In both cases, I found a significant correlation of Pearson $r > 0.7$ between the average human and average model performance. In the analysis of the difficulty measure, I excluded problem 31, which was the only problem for which the difficulty for humans and the model greatly varied. In fact, problem 31 is the easiest of all problems for humans and the hardest of all problems for the model. I will look into the reasons for this peculiar mismatch at the end of this chapter.

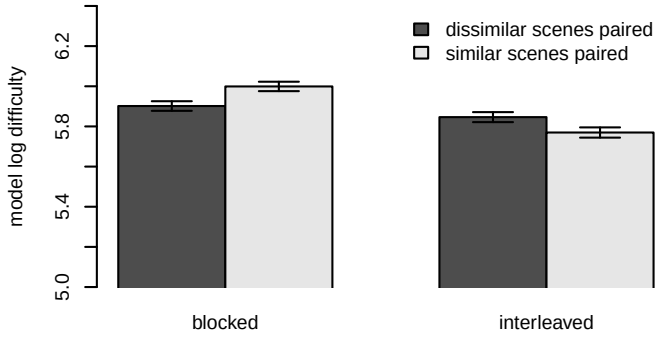
The model provided a good fit to the human data in another way. The effects of manipulating order and type of presented scenes were very similar for both model and human subjects. In Experiment 3 with human subjects, I found a significant advantage of high between-category similarity for conditions in which scenes from both categories were paired (interleaved conditions) in terms of accuracy and log reaction times. In Experiment 4, where all scenes were presented to the participants simultaneously, I found a significant advantage of low within-category similarity. These effects were significant both for the whole set of PBPs shown during the experiments and for the subset that was used with the model. The analysis of the model's performance revealed the same effects: an advantage of high between-category in interleaved conditions and an advantage of within-category similarity in blocked conditions. Figure 48 summarizes these results. For the sequential presentation schedules, it splits by the similarity condition that was promoted by the schedule type (within similarity for blocked and between similarity for interleaved) and combines the similarity condition that was not promoted by the schedule type.



(a) humans on sequential pairs (Experiment 3)



(b) humans on simultaneous scenes (Experiment 4)



(c) PATH model on sequential pairs

Figure 48: Comparison of the effects of presentation schedule and scene similarity on humans and the PATHS model. In Experiment 3 (top plot) which used sequential presentation of scene pairs, pairing similar scenes taken from both categories reduced difficulty. In Experiment 4 (middle plot) which showed all scenes simultaneously, putting dissimilar scenes next to each other within the categories and putting similar scenes next to each other between the categories reduced difficulty. The PATHS model (bottom plot) replicated both effects: pairing of dissimilar scenes within category and similar scenes between categories reduced difficulty. Bars represent standard errors.

Disagreements with Human Results

I conclude this chapter with a discussion of the points were the model does not fit the human results well and the insights we can gain from these.

First, why was there no advantage of low within-category similarity in the blocked condition of Experiment 3? The effect did show in Experiment 4, in which we know from eye tracking data that subjects tend to make more within- than between-category comparisons. One would expect the same to be true in the blocked condition of Experiment 3 since each scene pair consists of two scenes from the same category. One possible explanation is that the pair-wise presentation of the scenes biases people to contrast the two scenes instead of searching for commonalities. The longer reaction times for dissimilar versus similar between-category similarity in the blocked condition are consistent with this interpretation.

Second, although the same effects to similarity conditions were present for the human participants and model, the effect sizes for the model are smaller. The effect size of within-category similarity on the difficulty measure was Cohen's $d = 0.08$ for the model and $d = 0.18$ for the human participants. The effect size of between-category similarity on the difficulty measure was $d = 0.06$ for the model and $d = 0.43$ for the human participants. One potential reason for the difference in the effect sizes is that memory restrictions in human cognition were not sufficiently reproduced in the model. On one extreme, a perfect memory could completely remove any effect of order in learning. With decreasing number of hypotheses and perceptions that can be kept reliably in memory, the differences in performance between the conditions will get bigger.

Third, humans sometimes utilize rich background knowledge when solving the PBPs which can help – or hinder, if it is not aligned with the actual solution of the PBP – focusing on the right objects and features in a scene. This is quite apparent in problem 31 (see Figure 49), which human participants typically solved with ease, but which the model found particularly difficult. All solutions found by the model resolve around the "can-move-up" attribute, with the two most common being "there are can-move-up circles in all left scenes" and "there are small and can-move-up objects in all left scenes". Human participants came up with a wide variety of solutions, using verbs that evoke rich situations. They described the circle as being trapped, enclosed, covered, stuck, protected, imprisoned, contained, boxed in, hidden, surrounded, confined, secured, shielded, having an escape, "can be picked up", "can get out" and as free to move. While there is little to guide the attention of the model towards the right concepts, humans quickly home in on a familiar narrative that is captured in the scenes.

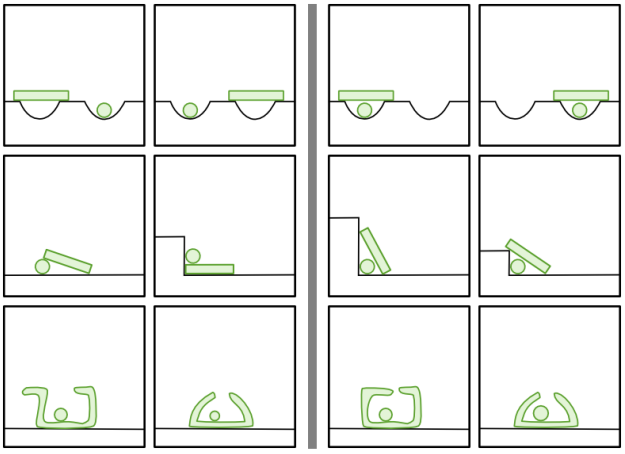


Figure 49: First three rows of PBP 31. All solutions found by the model include the "can-move-up" object attribute.

6

CONCLUSION

In this thesis, I developed a new cognitive model of rule-based concept learning, applied it to a challenging problem domain, and compared its performance to that of humans. The model addresses an important and somewhat neglected topic in the concept learning literature: the role of iterative and active perception and its interaction with rule-construction in human concept learning. Efficient learning from complex examples relies on good choices about the order in which features and rules are explored. Modeling the iterative and concurrent nature of perception and rule-construction allows these processes to mutually guide these choices.

The PATHS model is a process-level cognitive model of human category learning for rule-based categories and can currently solve thirteen of the twenty-two physical Bongard problems that human participants solved. The PBPs were designed as a challenging problem domain with structured, dynamic physical scenes as the instances that are categorized according to logical rules. The model tightly integrates perception with hypothesis generation and testing: perception drives rule formation and hypothesized rules guide further perception. The previous chapter demonstrated the correlation between the model's and human per-problem performance. Furthermore, the model is influenced by similarity and order of presented scene pairs in the same qualitative ways as humans. In this sense, the model achieves both its goal of perceiving and learning structured concepts and of capturing important characteristics of human learning performance.

While the PATHS model was developed from scratch, it combines ideas from several existing approaches. It is essentially a hypothesis generation and testing approach that shares a focus on perception such as described in Douglas Hofstadter's fluid analogy framework. The model picks its actions based on estimations of the probabilities of entertained hypotheses. This combination of process-level modeling with mathematically well-founded Bayesian techniques is a particular strength of the model.

Though the development of the cognitive model was the main goal of this thesis, several related but distinct insights and artifacts were created on the journey. The physical Bongard problems are a novel problem domain that can be used by other researchers. PBPs are attractive due to their open feature-space, their dynamic and structured content and the fact that they allow for easy manipulation of the similarity of scenes presented next to each other. While the studies with human participants were in part done to serve as a comparison for the model, they also directly advanced our understanding of how the mode of presentation and similarity of instances influences human concept learning. Specifically, I demonstrated the benefit of cross-

category comparison of similar versus dissimilar PBP scenes and the benefit of within-category comparison of dissimilar versus similar PBP scenes, mediated by presentation type. Finally, the parts of the PATHS model that perceive dynamic physical attributes of scenes like stability, support, and movability by using a physics engine to perform counterfactual reasoning, can easily be reused in other work.

One possible concern about the PATHS model is that it currently can solve PBPs and nothing else. Even if PBPs are representative for an important class of concept learning situations in people's lives, the question of properties and insights from the PATHS model that can be generalized or "exported" into other domains and contexts remains. I will discuss such properties and insights next, starting with two aspects that deserve more attention in the concept learning field.

The first aspect is integrating perceptual processes tightly with higher-level cognitive processes. Across different fields like active learning, optimal experiment design, analogy-making or memory retrieval this aspect has been integrated into models and algorithms in different flavors, yet in modeling category learning it is still the rule to treat perception as separate and completed before engaging in the "actual concept learning". What I propose and the PATHS model demonstrates is taking a closer look at the dynamics that result from the more realistic approach of tightly interconnected perception and conception.

A second, high-level design choice in the PATHS model that is exportable is the mixture of rational and process-level accounts. This powerful modeling approach that combines the strength of both accounts might benefit other models as well.

Beside these overarching modeling decisions, there are various ways in which the PATHS model as-is can be applied in new contexts to gain insights. I will briefly sketch three such potential future applications.

One straightforward extension would be to implement several variations of the internal probability estimations within PATHS and to test which of the variations fits human data best. Since the probability estimations are central to the model, they influence, among other aspects, the number of hypotheses that the model explores simultaneously, the influence of old versus new information and the balance between explorative and exploitative search behavior.

Another way of applying the PATHS model in a new context would be to model scaffolding or priming effects that arise from letting learners engage with problems in a particular order, for example, in increasing difficulty. In terms of PBPs, an easy problem could scaffold a similar but more difficult problem that follows it. Solving the first problem influences the mental state of the solver in ways that can make solving subsequent similar problems easier and different problems harder. These influences include a raised awareness to the type of solution, the complexity of the solution and the features that played a role in the solution of the first problem. People might

also be more likely to notice structural patterns in the second problem that were present in the first. The PATHS model can account for all of these mentioned effects, yet currently does so between the scenes of a single PBP only. It would be straightforward to extend the model to carry over hypotheses and feature activations from one problem to the next. This would, for example, allow modeling what kind of strategies and biases people carry over between problems.

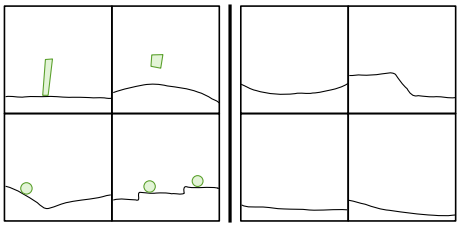
A third opportunity for application is to model the phenomenon of functional fixedness. Functional fixedness describes a cognitive bias that limits a person to use an object in another than its traditional way. In the famous candle problem, Duncker and Lees [1945] gave participants a box of tacks, matches, and a candle and asked them to attach the candle to the wall. Only very few participants came up with the solution to tack the box to the wall and use it as a candle holder. Most participants seemed to be fixated on the box's initial function as a tack container, which prohibited them from reconceptualizing its potential use. Given a situation with an initially empty box, they were much more likely to solve the problem. In terms of PBPs and the PATHS model, functional fixedness could be described as, sometimes overly, committing to a particular interpretation of a situation. There is an obvious trade-off here since a strong functional bias is both a powerful way to reduce the search space in typical problem situations but may also block one from finding a solution in atypical situations. The PATHS model at times shows the same phenomenon of getting stuck in a particular type of interpretation. This happens when a hypothesis looks promising initially and turns out to be wrong later, at which time the positive feedback loop between perception and hypothesizing has led the model into a local minimum in the space of interpretations that is difficult to escape. In such situations, resetting the model and having it start over on the problem – restoring an open mind in a way – can be faster than continuing the search. Although this involves discarding information that the model had already gathered, it can help avoid becoming stuck on a particular "garden path" that can result when unfortunate coincidences are discovered early that only serve to distract from the correct categories. The PATHS model might allow to shine new light on mechanisms of functional fixedness and how they can be limiting or beneficial to problem-solving.

Finally, the PATHS model could contribute towards a central goal of robotics research, the construction of intelligent stystem that humans can naturally interact with. The cognitive ability modeled in PATHS – the construction of verbal concepts from images of physical scenes – is likely one of the prerequisites for future cognitive interaction technology that shares our living spaces, naturally communicates, and fluidly interacts with us.

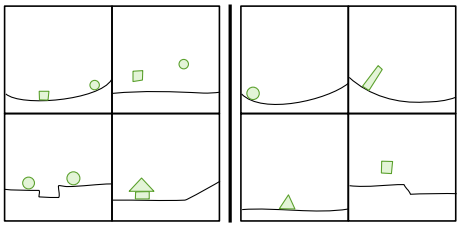
In summary, this work promotes a new focus on interactions between perception and rule-construction in concept learning and provides a new tool, the PATHS model. The new focus brings important questions to our attention, and the tool has already begun to advance answers to them.

A

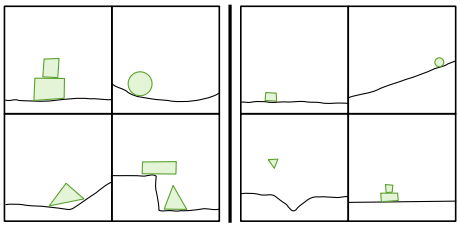
LIST OF PHYSICAL BONGARD PROBLEMS



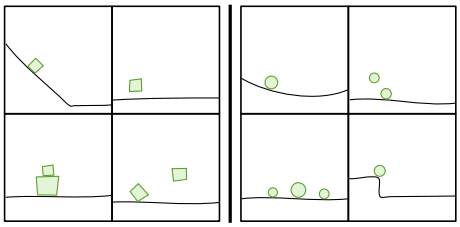
(1) PBP 01



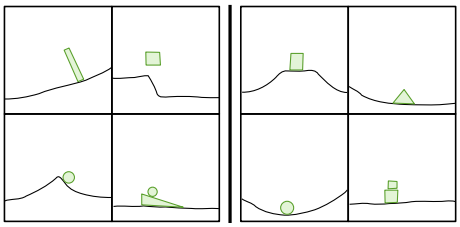
(2) PBP 02



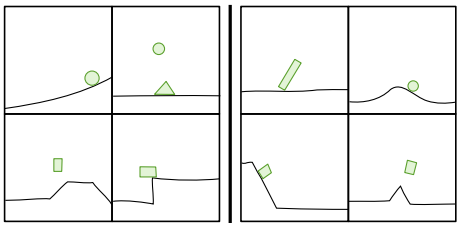
(3) PBP 03



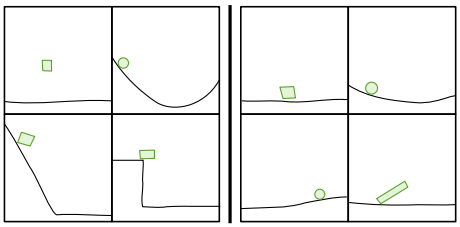
(4) PBP 04



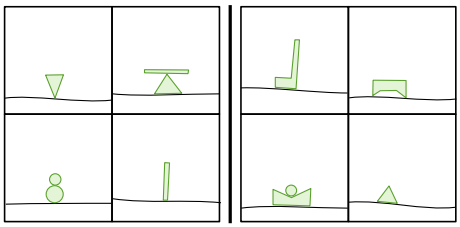
(5) PBP 05



(6) PBP 06

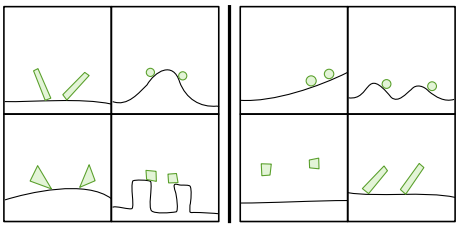


(7) PBP 07

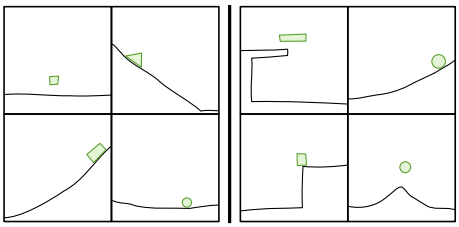


(8) PBP 08

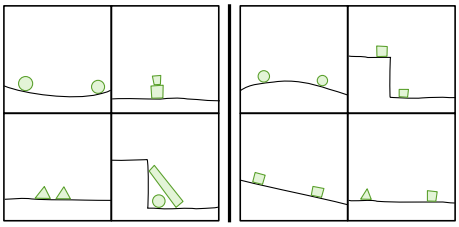
Figure 50: The early / first version of the physical Bongard problems that was used in the eye-tracking study and in Experiment 1. There are 34 problems, each with 8 scenes. Above are the problem numbers one to eight. The solutions to the problems are listed in the next table.



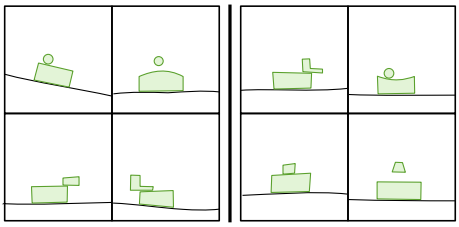
(1) PBP 09



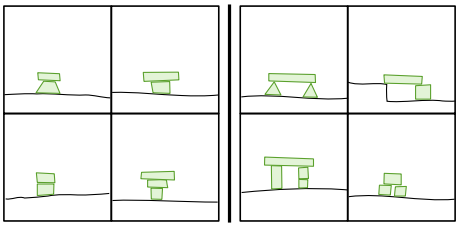
(2) PBP 10



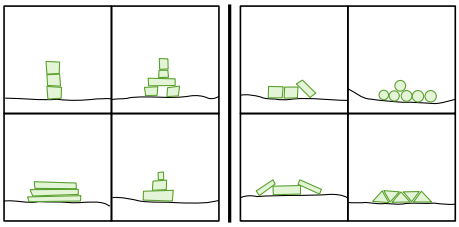
(3) PBP 11



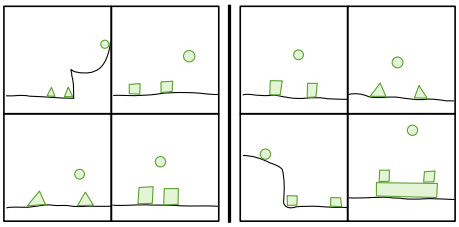
(4) PBP 12



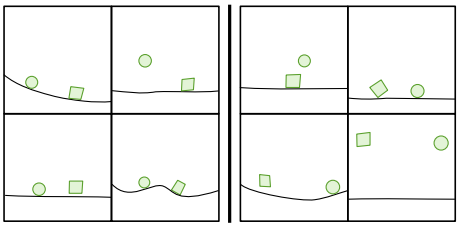
(5) PBP 13



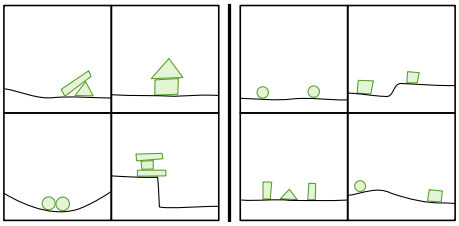
(6) PBP 14



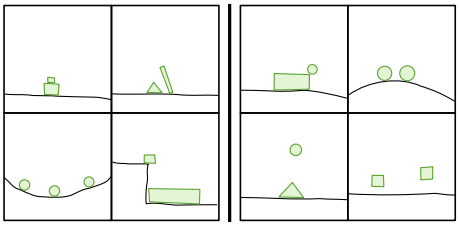
(7) PBP 15



(8) PBP 16

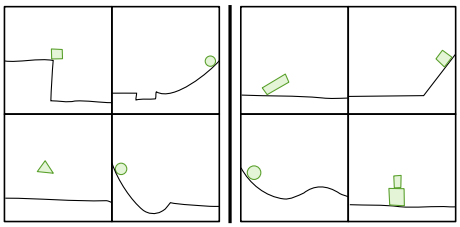


(9) PBP 17

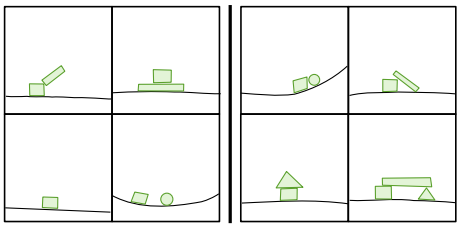


(10) PBP 18

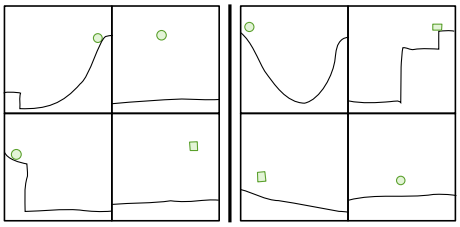
Figure 51: Early / first version of the PBPs, problem 9 to 18. The solutions to the problems are listed in the next table.



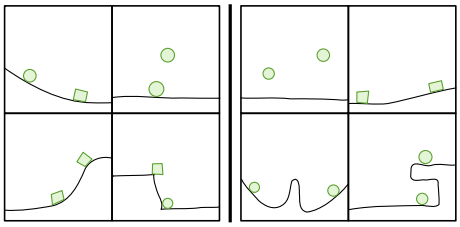
(1) PBP 19



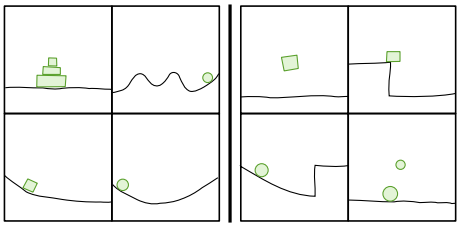
(2) PBP 20



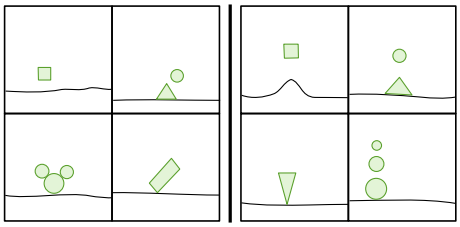
(3) PBP 21



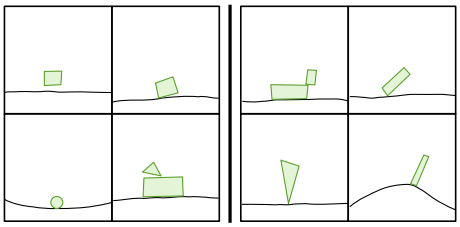
(4) PBP 22



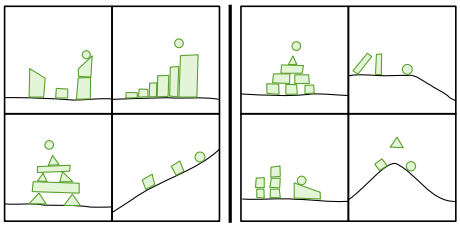
(5) PBP 23



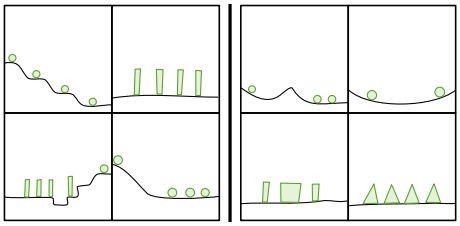
(6) PBP 24



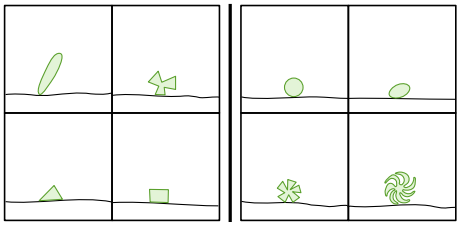
(7) PBP 25



(8) PBP 26

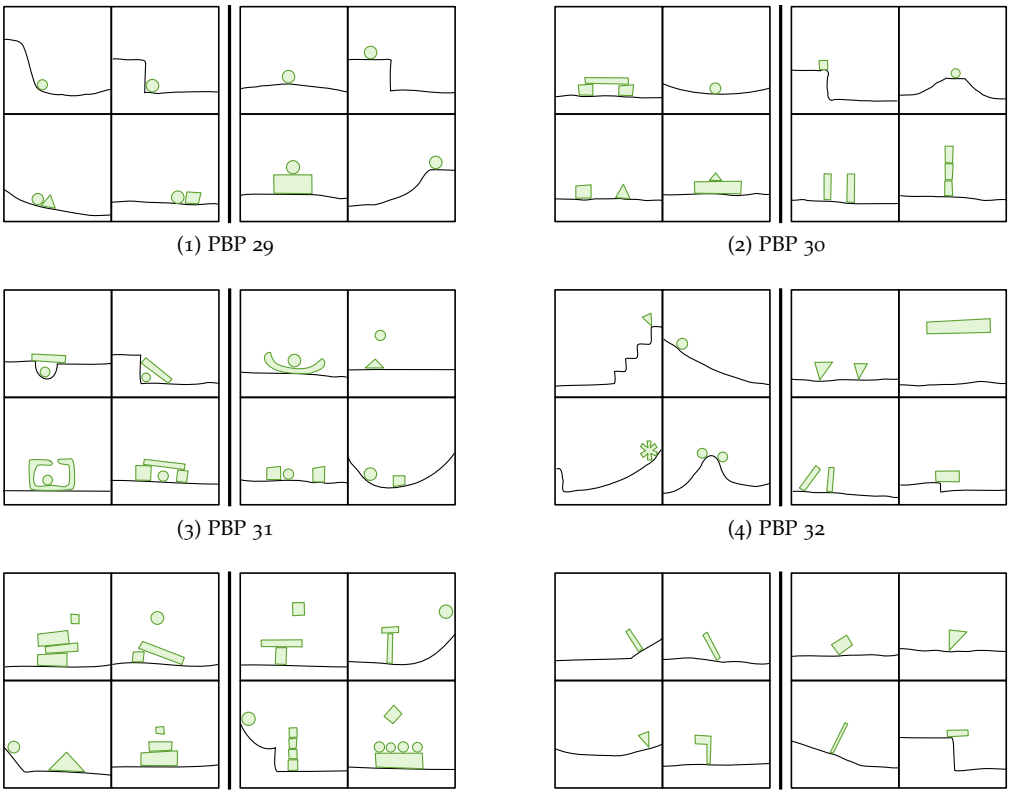


(9) PBP 27



(10) PBP 28

Figure 52: Early / first version of the PBPs, problem 19 to 28. The solutions to the problems are listed in the next table.



(1) PBP 29 (2) PBP 30

(3) PBP 31 (4) PBP 32

(5) PBP 33 (6) PBP 34

Figure 53: Early / first version of the PBPs, problem 29 to 34. The solutions to the problems are listed in the next table.

PBP	LEFT SIDE	RIGHT SIDE
01	objects exist	no objects
02	one object	two objects
03	big objects	small objects
04	squares	circles
05	objects move	objects don't move
06	objects move to the left	objects move to the right
07	fast movement	slow movement
08	unstable situation	stable situation
09	objects move in opposite directions	objects move in same direction
10	rotation	no rotation
11	the objects will eventually be close to each other	the objects are far from each other
12	small object falls off	small object stays on top
13	objects form a tower	objects form an arc
14	vertical construction	horizontal construction
15	circle does not hit right between the other objects	circle hits right between the other objects
16	the circle is left of the square	the square is left of the circle
17	objects touch	objects don't touch
18	object touch eventually	obj.s don't touch eventually
19	at least one object flies through the air	all object always touch something
20	square supports other obj's eventually	square doesn't support other objects
21	strong collision	weak or no collision
22	objects collide with each other	objects don't collide with each other
23	collision	no collision
24	several possible outcomes	one possible outcome
25	objects do not topple over	the object topples over
26	circle moves right	circle moves left
27	(potential) chain reaction	no chain reaction
28	rolls well	does not roll well
29	circle could move to one side	circle could move to both sides
30	less stable situation	stable situation
31	circle can be picked up directly	circle can't be picked up directly
32	objects rotate a lot	obj.s rotate little or not at all
33	construction gets destroyed	construction stays intact
34	object falls to the left	object falls to the right

Table 9: Solutions to all 34 early / first version PBPs.

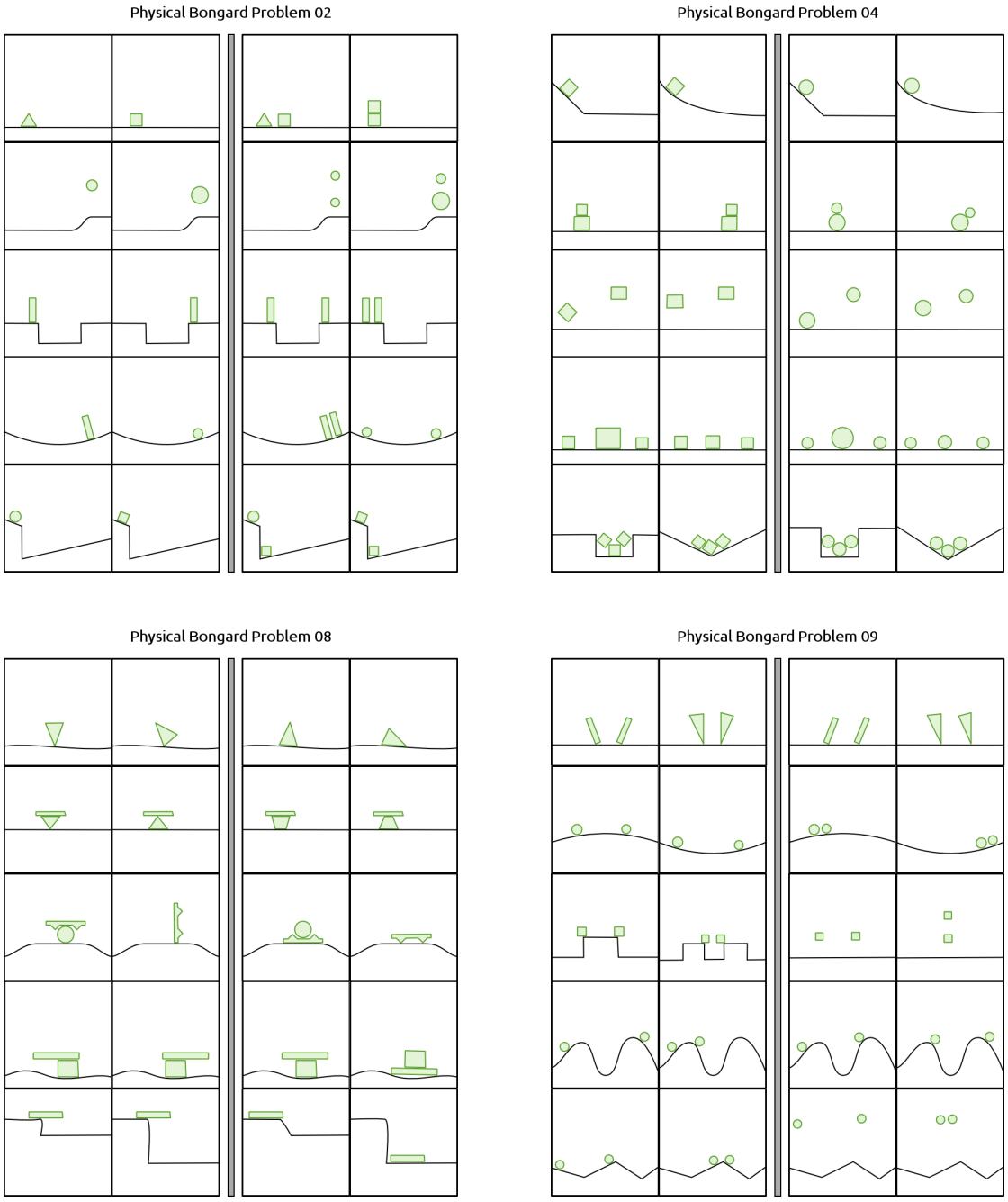


Figure 54: The second version of the physical Bongard problems that was used in Experiments 2 to 4. Twenty-two of the original problems were selected and extended. The version two PBPs contain twenty scenes each and the scenes are grouped into five similarity groups. The similarity groups are arranged by rows above, such scenes within a row are more similar than scenes across rows. The solutions to the problems are listed in the next table.

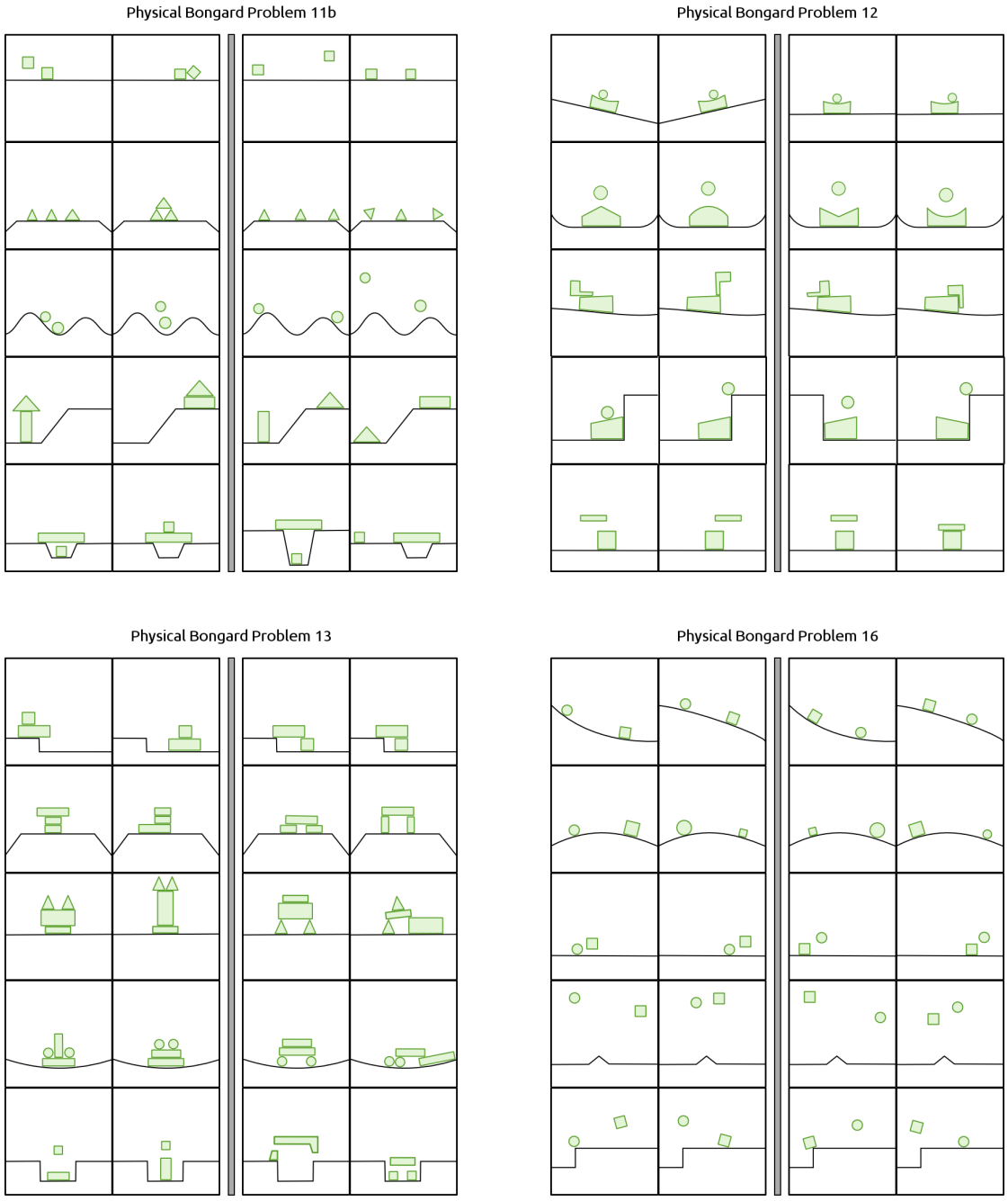


Figure 55: Version two of the PBPs. The solutions to the problems are listed in the next table.

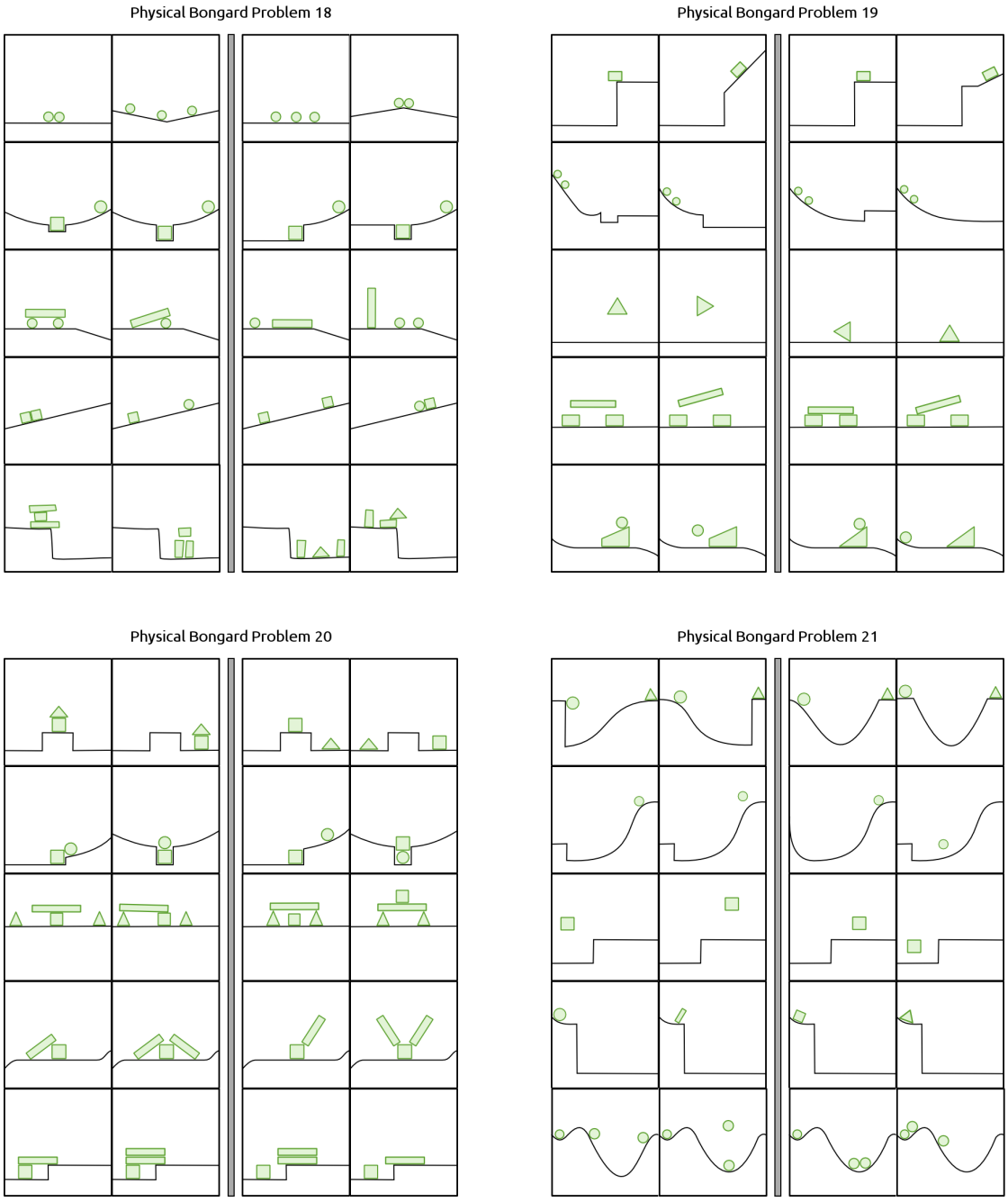


Figure 56: Version two of the PBPs. The solutions to the problems are listed in the next table.

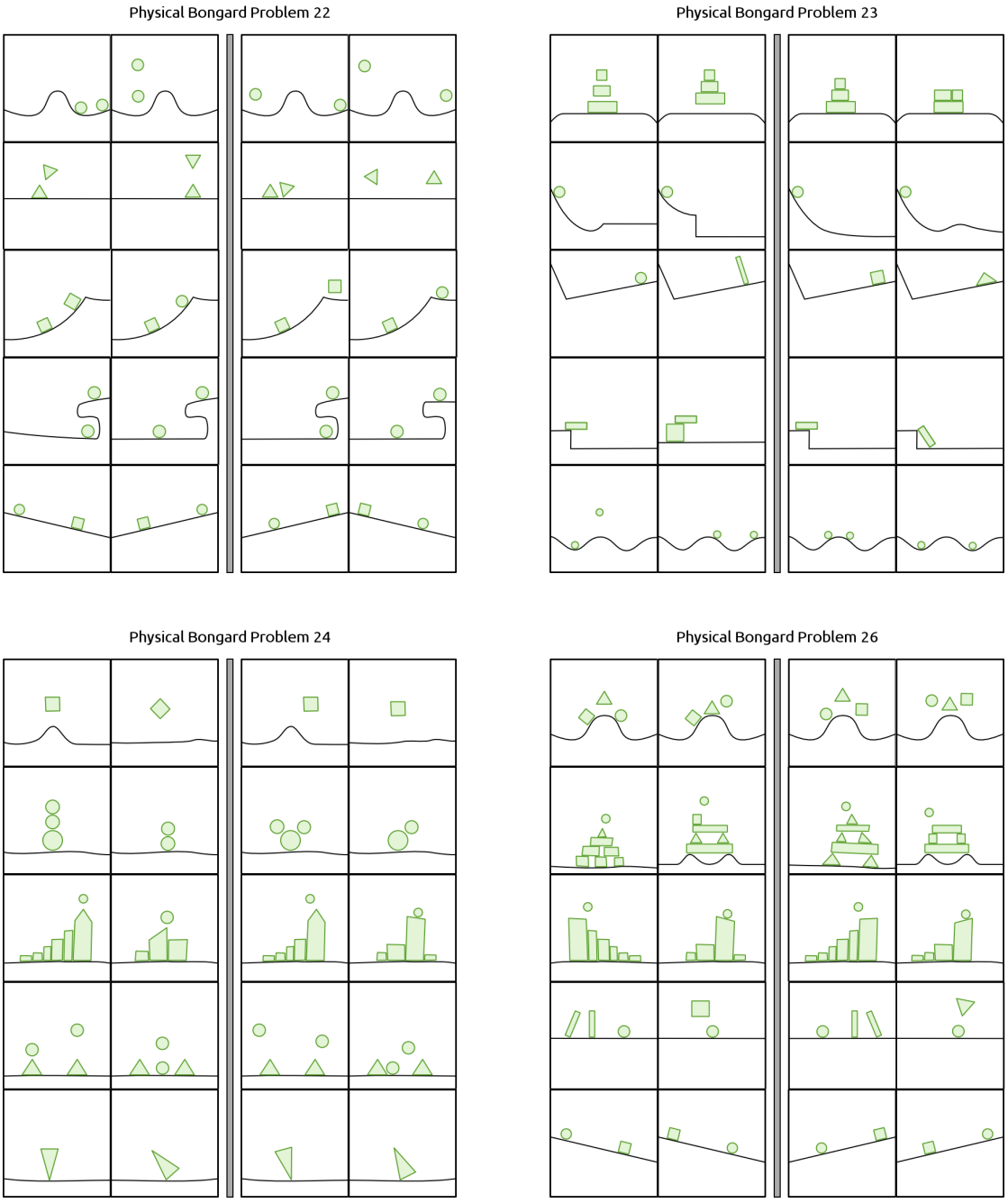


Figure 57: Version two of the PBPs. The solutions to the problems are listed in the next table.

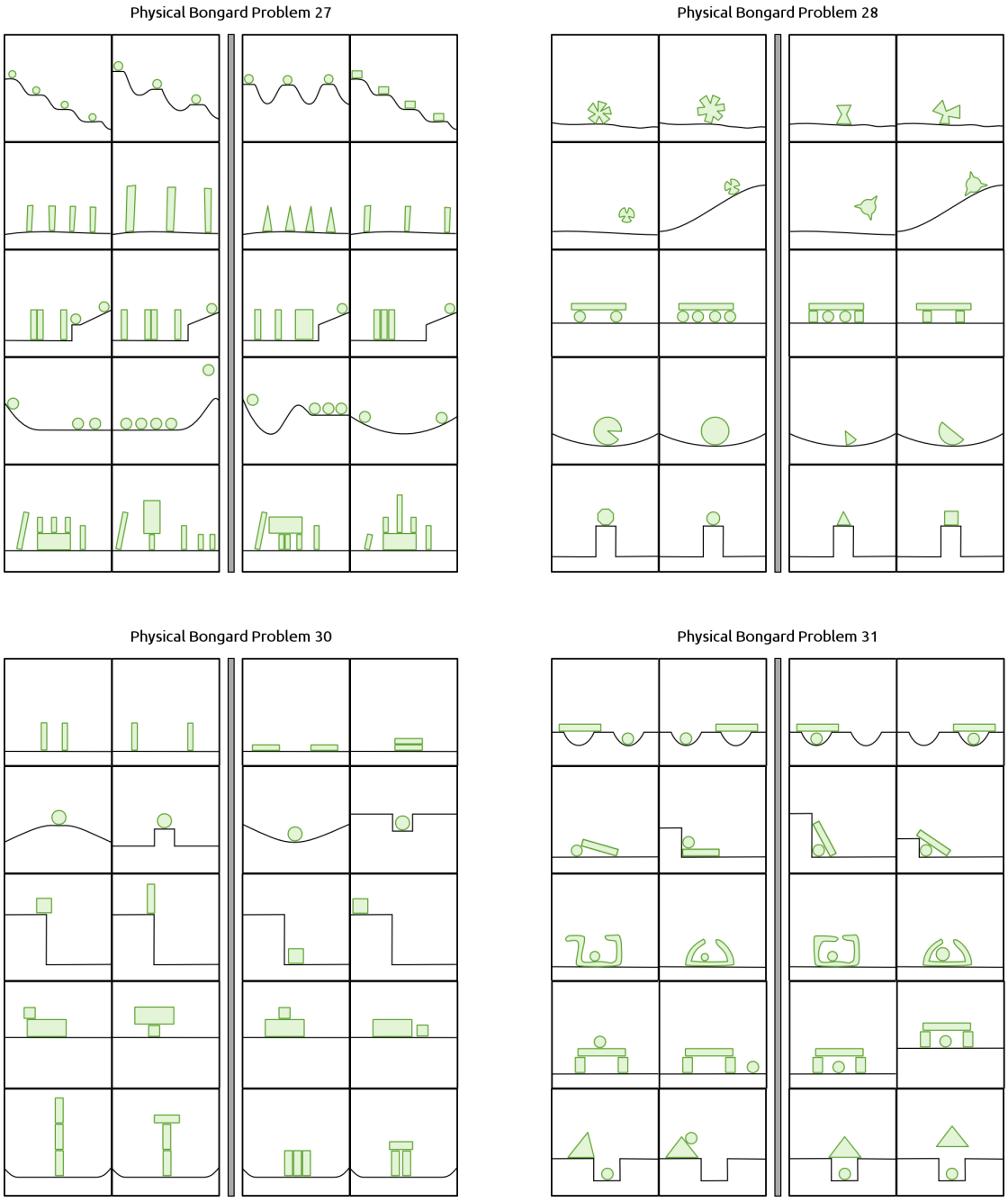


Figure 58: Version two of the PBPs. The solutions to the problems are listed in the next table.

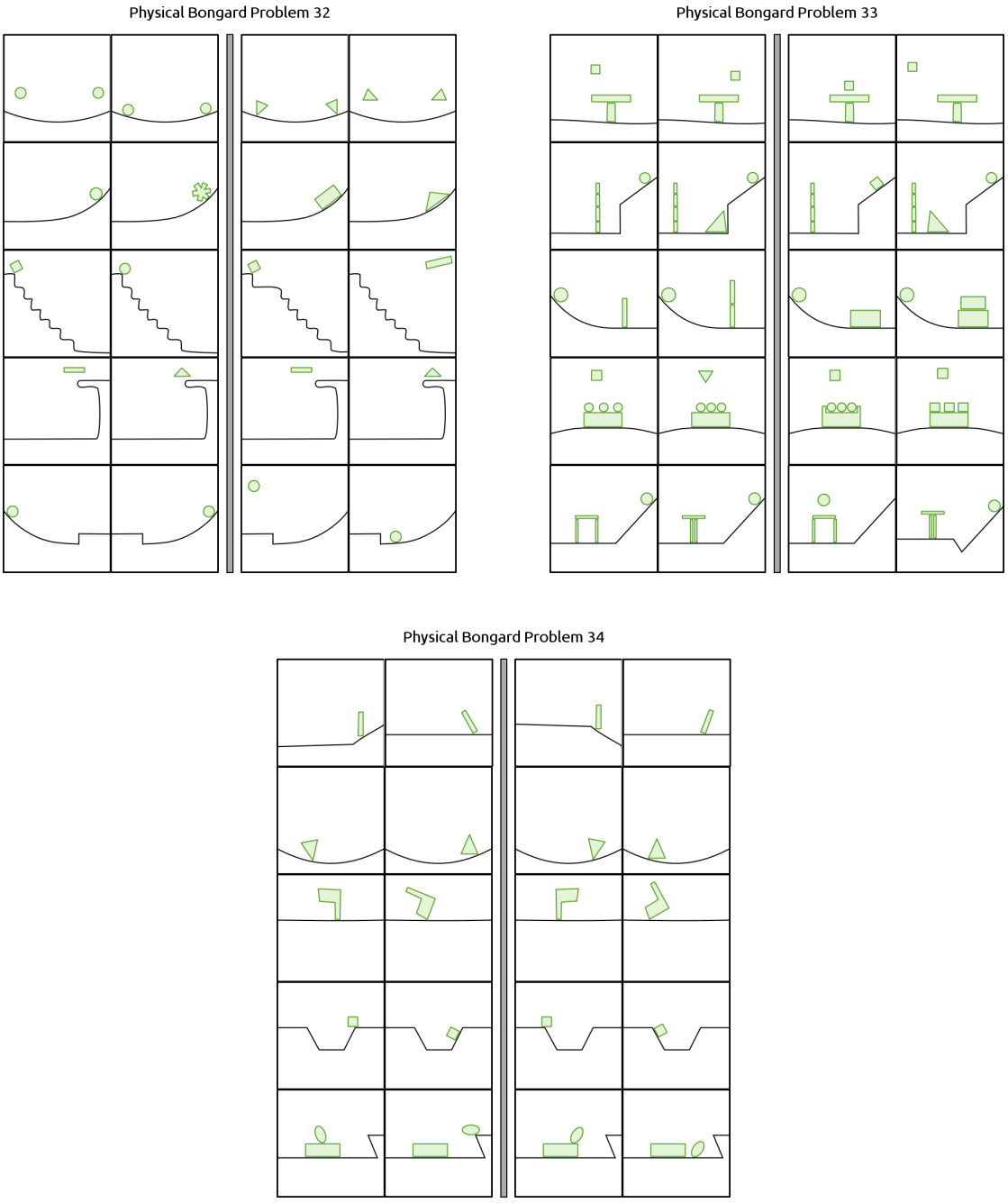


Figure 59: Version two of the PBPs. The solutions to the problems are listed in the next table.

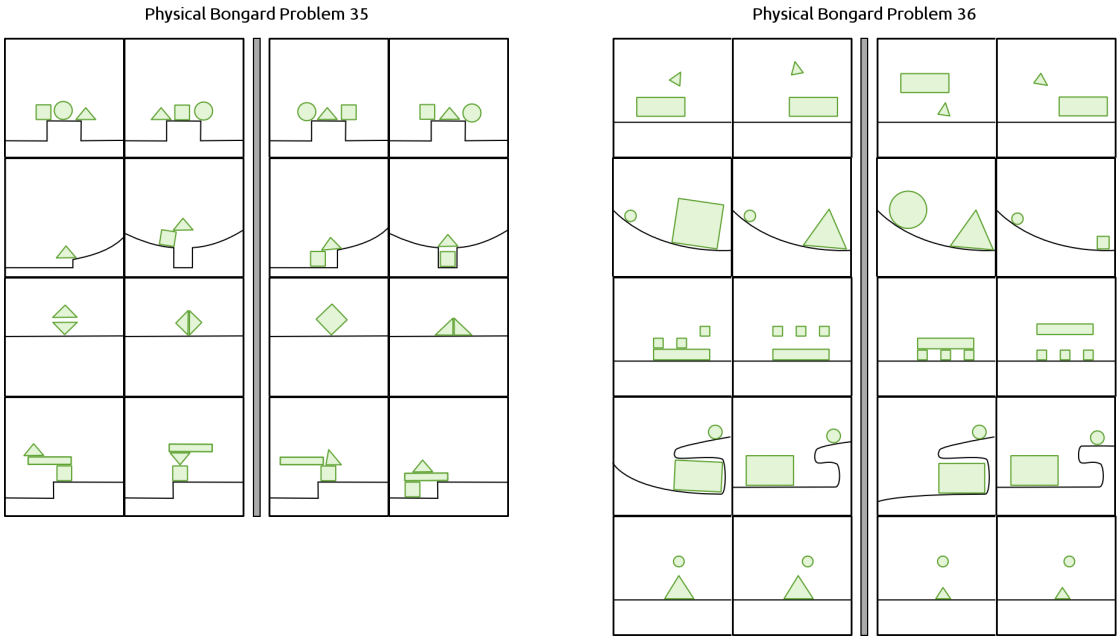


Figure 60: Version two of the PBPs. The solutions to the problems are listed in the next table.

PBP	LEFT SIDE	RIGHT SIDE
02	one object	two objects
04	squares	circles
08	unstable situation	stable situation
09	objects move in opposite directions	objects move in same direction
11b	objects close to each other	objects far from each other
12	small object falls off	small object stays on top
13	objects form a tower	objects form an arc
16	the circle is left of the square	the square is left of the circle
18	object touch eventually	obj.s don't touch eventually
19	at least one object flies through the air	all object always touch something
20	square supports other obj's eventually	square doesn't support other objects
21	strong collision	weak or no collision
22	objects collide with each other	objects don't collide with each other
23	collision	no collision
24	several possible outcomes	one possible outcome
26	circle moves right	circle moves left
27	(potential) chain reaction	no chain reaction
28	rolls well	does not roll well
30	less stable situation	stable situation
31	circle can be picked up directly	circle can't be picked up directly
32	objects rotate a lot	obj.s rotate little or not at all
33	construction gets destroyed	construction stays intact
34	objects fall to the left	objects fall to the right
35	there is a moving triangle	no moving triangle
36	a small objects hits a large object	no small objects hits a large object

Table 10: Solutions to all version 2 PBPs. The first 22 of them were solved by human participants. Problem 34 was used as example problem in the experimental instructions. Problems 35 and 36 were not used with human participants and instead used to test how the PATHS model performs on novel problems.

BIBLIOGRAPHY

Farough Abed. Cultural influences on visual scanning patterns. *Journal of Cross-Cultural Psychology*, 22(4):525–534, 1991.

James F Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

John R Anderson. The adaptive nature of human categorization. *Psychological Review*, 98(3):409, 1991.

F Gregory Ashby and W Todd Maddox. Human category learning. *Annu. Rev. Psychol.*, 56:149–178, 2005.

F Gregory Ashby and W Todd Maddox. Human category learning 2.0. *Annals of the New York Academy of Sciences*, 1224(1):147–161, 2011.

F Gregory Ashby, Leola A Alfonso-Reese, et al. A neuropsychological theory of multiple systems in category learning. *Psychological review*, 105(3):442, 1998.

Lawrence W Barsalou. Perceptual symbol systems. *Behavioral and Brain Sciences*, 22:577–660, 1999.

Peter W Battaglia, Jessica B Hamrick, and Joshua B Tenenbaum. Simulation as an engine of physical scene understanding. *PNAS*, 110(45):18327–18332, 2013.

M.S. Birnbaum, N. Kornell, E.L. Bjork, and R.A. Bjork. Why interleaving enhances inductive learning: The roles of discrimination and retrieval. *Memory & Cognition*, pages 1–11, 2012.

Christopher M Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

Isabelle Bloch. Fuzzy relative position between objects in image processing: a morphological approach. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(7):657–664, 1999.

Isabelle Bloch. Bipolar fuzzy spatial information: geometry, morphology, spatial reasoning. In *Methods for Handling Imperfect Spatial Information*, pages 75–102. Springer, 2010.

Lyle E Bourne. Knowing and using concepts. *Psychological Review*, 77(6):546, 1970.

Gordon H Bower and Thomas R Trabasso. Concept identification. *Studies in mathematical psychology*, pages 32–94, 1964.

Jerome S Bruner, Jacqueline J Goodnow, and A George. Austin. 1956. a study of thinking. *New York, JohnWiley & Sons. BrunerA study of thinking1956*, 1956.

Bruce G Buchanan and Edward A Feigenbaum. Dendral and meta-dendral: Their applications dimension. *Artificial intelligence*, 11(1):5–24, 1978.

Paulo F Carvalho and Robert L Goldstone. Putting category learning in order: Category structure and temporal arrangement affect the benefit of interleaved over blocked study. *Memory & cognition*, 2013.

P.F. Carvalho and R.L. Goldstone. Category structure modulates interleaving and blocking advantage in inductive category acquisition. In N. Miyake, D. Peebles, and R. P. Cooper, editors, *Proceedings of the Thirty-Fourth Annual Conference of the Cognitive Science Society*, pages 186–191, Austin, TX: Cognitive Science Society, 2012.

NM Chang, Kenneth R Koedinger, and Marsha C Lovett. Learning spurious correlations instead of deeper relations. *Proceedings of the 25th Cognitive Science Society. Boston, MA: Cognitive Science Society*, pages 228–233, 2003.

Andy Clark. Whatever next? predictive brains, situated agents, and the future of cognitive science. *Behavioral and Brain Sciences*, 36(03):181–204, 2013.

Anthony G. Cohn and Shyamanta M. Hazarika. Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1): 1–29, 2001.

J. de Leeuw. A javascript library for running behavioral experiments on the web, 2013. URL `https://github.com/jodeleeuw/jsPsych`.

Thomas G Dietterich and Ryszard S Michalski. Inductive learning of structural descriptions: Evaluation criteria and comparative review of selected methods. *Artificial intelligence*, 16(3):257–294, 1981.

Thomas G Dietterich and Ryszard S Michalski. Discovering patterns in sequences of events. *Artificial Intelligence*, 25(2):187–232, 1985.

Thomas G Dietterich, Pedro Domingos, Lise Getoor, Stephen Muggleton, and Prasad Tadepalli. Structured machine learning: the next ten years. *Machine Learning*, 73(1):3–23, 2008.

Karl Duncker and Lynne S Lees. On problem-solving. *Psychological monographs*, 58(5):i, 1945.

Thomas G Evans. A heuristic program to solve geometric-analogy problems. In *Proceedings of the April 21-23, 1964, spring joint computer conference*, pages 327–338. ACM, 1964.

Brian Falkenhainer, Kenneth D Forbus, and Dedre Gentner. The structure-mapping engine: Algorithm and examples. *Artificial intelligence*, 41(1):1–63, 1989.

Jacob Feldman. The simplicity principle in human concept learning. *Current Directions in Psychological Science*, 12(6):227–232, 2003.

Richard E Fikes, Peter E Hart, and Nils J Nilsson. Learning and executing generalized robot plans. *Artificial intelligence*, 3:251–288, 1972.

R. A. Finks, S. Pinker, and M. J. Farah. Reinterpreting visual patterns in mental imagery. *Cognitive Science*, 13(1):51–78, 1989.

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

Kenneth D Forbus. Qualitative process theory. *Artificial intelligence*, 24(1): 85–168, 1984.

Kenneth D Forbus, Dedre Gentner, and Keith Law. Mac/fac: A model of similarity-based retrieval. *Cognitive science*, 19(2):141–205, 1995.

H. E Foundalis. *Phaeaco: A cognitive architecture inspired by Bongard's problems*. PhD thesis, Indiana University, 2006.

Christian Freksa. *Using orientation information for qualitative spatial reasoning*. Springer, 1992.

R French and Douglas Hofstadter. Tabletop: An emergent, stochastic model of analogy-making. In *Proceedings of the 13th Annual Conference of the Cognitive Science Society*, pages 175–182, 1992.

Johannes Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13(1):3–54, 1999.

D. Gentner and K. D Forbus. Computational models of analogy. *Wiley Interdisciplinary Reviews: Cognitive Science*, 2(3):266–276, 2011.

Dedre Gentner. Structure-mapping: A theoretical framework for analogy*. *Cognitive science*, 7(2):155–170, 1983.

Dedre Gentner. Bootstrapping the mind: Analogical processes and symbol systems. *Cognitive Science*, 34(5):752–775, 2010.

Dedre Gentner and Kenneth J Kurtz. Relational categories. *Categorization inside and outside the lab*, pages 151–175, 2005.

Dedre Gentner and Arthur B Markman. Structural alignment in comparison: No difference without similarity. *Psychological science*, 5(3):152–158, 1994.

Alfonso Gerevini and Bernhard Nebel. Qualitative spatio-temporal reasoning with rcc-8 and allen's interval calculus: Computational complexity. In *ECAI*, volume 2, pages 312–316, 2002.

Lise Getoor and Ben Taskar. *Introduction to statistical relational learning*. MIT press, 2007.

Mary L Gick and Keith J Holyoak. Schema induction and analogical transfer. *Cognitive psychology*, 15(1):1–38, 1983.

R.L. Goldstone. Isolated and interrelated concepts. *Memory & Cognition*, 24 (5):608–628, 1996.

Robert L Goldstone. Learning to perceive while perceiving to learn. *Perceptual organization in vision: Behavioral and neural perspectives*, pages 233–278, 2003.

Robert L Goldstone, Alexander Gerganov, David Landy, and Michael E Roberts. Learning to see and conceive. *The new cognitive sciences*, pages 163–188, 2009.

Robert L. Goldstone, Alan Kersten, and Paulo F. Carvalho. Concepts and categorization. volume Volume 4: Experimental psychology of *Comprehensive handbook of psychology*, pages 607–630. Wiley, New Jersey, 2012.

Rebecca L Gómez. Variability and detection of invariant structure. *Psychological Science*, 13(5):431–436, 2002.

Noah D Goodman, Joshua B Tenenbaum, Jacob Feldman, and Thomas L Griffiths. A rational analysis of rule-based concept learning. *Cognitive Science*, 32(1):108–154, 2008a.

Noah D Goodman, Joshua B Tenenbaum, Thomas L Griffiths, and Jacob Feldman. Compositionality in rational analysis: Grammar-based induction for concept learning. *The probabilistic mind: Prospects for Bayesian cognitive science*, 2008b.

Thomas L Griffiths and Joshua B Tenenbaum. Theory-based causal induction. *Psychological review*, 116(4):661, 2009.

D.F. Halpern, C. Hansen, and D. Riefer. Analogies as an aid to understanding and memory. *Journal of Educational Psychology*, 82(2):298, 1990.

Douglas Hofstadter. A review of mental leaps: analogy in creative thought. *AI Magazine*, 16(3):75–80, 1995.

D.R. Hofstadter. *Fluid concepts and creative analogies: Computer models of the fundamental mechanisms of thought*. Basic Books, 1996.

Timothy L Hubbard. Representational momentum and related displacements in spatial memory: A review of the findings. *Psychonomic Bulletin & Review*, 12(5):822–851, 2005.

Céline Hudelot, Jamal Atif, and Isabelle Bloch. Fuzzy spatial relation ontology for image interpretation. *Fuzzy Sets and Systems*, 159(15):1929–1951, 2008.

John E Hummel and Keith J Holyoak. Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review*, 104 (3):427, 1997.

Earl B Hunt, Janet Marin, and Philip J Stone. Experiments in induction. 1966.

Samuel John, Erik Weitnauer, and Hendrik Koesling. Entropy-based correction of eye tracking data for static scenes. In *Proceedings of the symposium on eye tracking research and applications*, pages 297–300. ACM, 2012.

Erin L Jones and Brian H Ross. Classification versus inference learning contrasted with real-world categories. *Memory & cognition*, 39(5):764–777, 2011.

S.H.K. Kang and H. Pashler. Learning painting styles: Spacing is advantageous when it promotes discriminative contrast. *Applied Cognitive Psychology*, 26(1):97–103, 2012.

Charles Kemp and Joshua B Tenenbaum. Structured statistical models of inductive reasoning. *Psychological review*, 116(1):20, 2009.

Boicho Kokinov and Alexander Petrov. Integrating memory and reasoning in analogy-making: The ambr model. *The analogical mind: Perspectives from cognitive science*, pages 59–124, 2001.

N. Kornell and R.A. Bjork. Learning concepts and categories is spacing the "enemy of induction"? *Psychological Science*, 19(6):585–592, 2008.

Laura Kotovsky and Dedre Gentner. Comparison and categorization in the development of relational similarity. *Child Development*, 67(6):2797–2822, 1996.

SB Kotsiantis. Supervised machine learning: A review of classification techniques. *Informatica*, 31:249–268, 2007.

John K Kruschke. Alcove: an exemplar-based connectionist model of category learning. *Psychological review*, 99(1):22, 1992.

Sven Kuehne, Kenneth Forbus, Dedre Gentner, and Bryan Quinn. Seql: Category learning as progressive abstraction using structure mapping. In *Proceedings of the 22nd Annual Meeting of the Cognitive Science Society*, pages 770–775, 2000.

Kenneth J Kurtz, Olga Boukrina, and Dedre Gentner. Comparison promotes learning and transfer of relational categories. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 39(4):1303–1310, 2013.

Pat Langley. The changing science of machine learning. *Machine Learning*, 82(3):275–279, 2011.

Michael A. Lawrence. *ez: Easy analysis and visualization of factorial experiments.*, 2013. URL `http://CRAN.R-project.org/package=ez`. R package version 4.2-2.

J. Loewenstein and D. Gentner. Relational language and the development of relational mapping. *Cognitive Psychology*, 50(4):315–353, 2005.

Gordon D Logan. Toward an instance theory of automatization. *Psychological review*, 95(4):492, 1988.

Bradley C Love, Douglas L Medin, and Todd M Gureckis. Sustain: a network model of category learning. *Psychological review*, 111(2):309, 2004.

Marcus A Maloof and Ryszard S Michalski. Incremental learning with partial instance memory. *Artificial intelligence*, 154(1):95–126, 2004.

A. B. Markman and D. Gentner. Structural alignment during similarity comparisons. *Cognitive Psychology*, 25:431–431, 1993.

W. Mason and S. Suri. Conducting behavioral research on amazon's mechanical turk. *Behavior Research Methods*, 44(1):1–23, 2012.

D.L. Medin and B.H. Ross. The specific character of abstract thought: Categorization, problem solving, and induction. 1989.

D.L. Medin, R.L. Goldstone, and D. Gentner. Respects for similarity. *Psychological review*, 100(2):254, 1993.

Douglas L Medin and Marguerite M Schaffer. Context theory of classification learning. *Psychological review*, 85(3):207, 1978.

Douglas L Medin, William D Wattenmaker, and Ryszard S Michalski. Constraints and preferences in inductive learning: An experimental study of human and machine performance. *Cognitive Science*, 11(3):299–339, 1987.

Prem Melville, Maytal Saar-Tsechansky, Foster Provost, and Raymond Mooney. Active feature-value acquisition for classifier induction. In *Data Mining, 2004. ICDM'04. Fourth IEEE International Conference on*, pages 483–486. IEEE, 2004.

M. Mitchell. *Analogy-making as perception: A computer model*. MIT Press, 1993.

Tom M Mitchell. *The need for biases in learning generalizations*. Department of Computer Science, Laboratory for Computer Science Research, Rutgers Univ., 1980.

Tom M Mitchell. Generalization as search. *Artificial intelligence*, 18(2):203–226, 1982.

Tom M Mitchell, Richard M Keller, and Smadar T Kedar-Cabelli. Explanation-based generalization: A unifying view. *Machine learning*, 1 (1):47–80, 1986.

Stephen Muggleton. *Inductive logic programming*, volume 168. Springer, 1992.

Stephen Muggleton and Luc De Raedt. Inductive logic programming: Theory and methods. *The Journal of Logic Programming*, 19:629–679, 1994.

Eric Paul Nichols. *Musicat: A computer model of musical listening and analogy-making*. PhD thesis, faculty of the University Graduate School in partial fulillment of the requirements for the degree Doctor of Philosophy in the Departments of Computer Science and Cognitive Science, Indiana University, 2012.

Robert M Nosofsky. Attention, similarity, and the identification–categorization relationship. *Journal of experimental psychology: General*, 115 (1):39, 1986.

Robert M Nosofsky, Thomas J Palmeri, and Stephen C McKinley. Rule-plus-exception model of classification learning. *Psychological review*, 101(1):53, 1994.

Randall C O'Reilly, Dean Wyatte, and John Rohrlich. Learning through time in the thalamocortical loops. *arXiv preprint arXiv:1407.3432*, 2014.

J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

David A Randell, Zhan Cui, and Anthony G Cohn. A spatial logic based on regions and connection. *KR*, 92:165–176, 1992.

Jochen Renz and Bernhard Nebel. Qualitative spatial reasoning using constraint calculi. In *Handbook of spatial logics*, pages 161–215. Springer, 2007.

Eleanor Rosch. Cognitive representations of semantic categories. *Journal of experimental psychology: General*, 104(3):192, 1975.

Eleanor Rosch and Carolyn B Mervis. Family resemblances: Studies in the internal structure of categories. *Cognitive psychology*, 7(4):573–605, 1975.

Eleanor Rosch, Carolyn B Mervis, Wayne D Gray, David M Johnson, and Penny Boyes-Braem. Basic objects in natural categories. *Cognitive psychology*, 8(3):382–439, 1976.

G. C. Rost and B. McMurray. Speaker variability augments phonological processing in early word learning. *Developmental Science*, 12(2):339–349, 2009.

Maytal Saar-Tsechansky, Prem Melville, and Foster Provost. Active feature-value acquisition. *Management Science*, 55(4):664–684, 2009.

Burr Settles. Active learning literature survey. *University of Wisconsin, Madison*, 52(55-66):11, 2010.

Herbert A Simon and Edward A Feigenbaum. An information-processing theory of some effects of similarity, familiarization, and meaningfulness in verbal learning. *Journal of Verbal Learning and Verbal Behavior*, 3(5):385–396, 1964.

Nico Van de Weghe, Anthony G Cohn, Guy De Tre, and Philippe De Maeyer. A qualitative trajectory calculus as a basis for representing moving objects in geographical information systems. *Control and Cybernetics*, 35(1): 97, 2006.

Wim Van Laer and Luc De Raedt. How to upgrade propositional learners to first order logic: A case study. In *Machine Learning and Its Applications*, pages 102–126. Springer, 2001.

Laure Vieu. Spatial representation and reasoning in artificial intelligence. In *Spatial and temporal reasoning*, pages 5–41. Springer, 1997.

Erik Weitnauer, Paulo F Carvalho, Robert L Goldstone, and Helge Ritter. Grouping by similarity helps concept learning. In M. Knauff, M. Pauen, N. Sebanz, and I. Wachsmuth, editors, *35th Annual Conference of the Cognitive Science Society*, 2013.

Erik Weitnauer, Paulo F Carvalho, Robert L Goldstone, and Helge Ritter. Similarity-based ordering of instances for efficient concept learning. In *36th Annual Conference of the Cognitive Science Society*, pages 1760–1765. Cognitive Science Society, 2014.

Erik Weitnauer, David H Landy, Robert L Goldstone, and Helge Ritter. A computational model for learning structured concepts from physical scenes. In *37th Annual Conference of the Cognitive Science Society*, pages 2631–2636. Cognitive Science Society, 2015.

P.H. Winston. Learning structural descriptions from examples. Technical report, DTIC Document, 1970.

Janusz Wnek and Ryszard S Michalski. Hypothesis-driven constructive induction in aq17-hci: A method and experiments. *Machine Learning*, 14(2): 139–168, 1994.