Applying Cognitive Principles to Model-Finding Output: The Positive Value of Negative Information

TRISTAN DYER, TIM NELSON, KATHI FISLER, and SHRIRAM KRISHNAMURTHI, Brown University, USA

Model-finders, such as SAT/SMT-solvers and Alloy, are used widely both directly and embedded in domain-specific tools. They support both conventional verification and, unlike other verification tools, property-free exploration. To do this effectively, they must produce output that helps users with these tasks. Unfortunately, the output of model-finders has seen relatively little rigorous human-factors study.

Conventionally, these tools tend to show one satisfying instance at a time. Drawing inspiration from the cognitive science literature, we investigate two aspects of model-finder output: how many instances to show at once, and whether all instances must actually satisfy the input constraints. Using both controlled studies and open-ended talk-alouds, we show that there is benefit to showing negative instances in certain settings; the impact of multiple instances is less clear. Our work is a first step in a theoretically grounded approach to understanding how users engage cognitively with model-finder output, and how those tools might better support users in doing so.

CCS Concepts: • **Human-centered computing**; • **Software and its engineering** → **Formal methods**;

Additional Key Words and Phrases: model finding, Alloy, cognitive science, user studies

ACM Reference Format:
Tristan Dyer, Tim Nelson, Kathi Fisler, and Shriram Krishnamurthi. 2022. Applying Cognitive Principles to Model-Finding Output: The Positive Value of Negative Information. *Proc. ACM Program. Lang.* 6, OOPSLA1, Article 79 (April 2022), 29 pages. https://doi.org/10.1145/3527323

1 INTRODUCTION

Model-finding tools are widely used to reason about a variety of domains. The Alloy (Jackson 2012) tool, for instance, has been applied to network configuration (Maldonado-Lopez et al. 2014; Nelson et al. 2010; Ruchansky and Proserpio 2013), security (Akhawe et al. 2010), software development (Maoz et al. 2011; Rosso and Jackson 2013), distributed systems (Zave 2012), and much else. SMT solvers such as Z3 (de Moura and Bjørner 2008) and the Cooperating Validity Checker (Barrett et al. 2011) also provide model-finding functionality, which is now frequently embedded as the core engine of end-user-facing tools, both in research (e.g., Padon et al. (2016); Seidel et al. (2016)) and industry (e.g., Backes et al. (2019, 2018); Fogel et al. (2015)).

A virtue of model-finders is that they support multiple modalities. In verification, they can search for counterexamples without the manual labor or specialized knowledge required by many deductive methods. In addition, unlike traditional verification engines, model-finders have another use: system *exploration*, without having to write any properties. A model-finder can be asked

Authors' address: Tristan Dyer, atristandyer@gmail.com; Tim Nelson, tim_nelson@brown.edu; Kathi Fisler, kfisler@brown.edu; Shriram Krishnamurthi, shriram@brown.edu, Brown University, Providence, RI, USA.

2475-1421/2022/4-ART79
https://doi.org/10.1145/3527323

to simply produce instances of the model[1], which is presumably helpful to users as they try to comprehend the system. As the Alloy book (Jackson 2012, page 142) states, this sort of analysis:

> ...brings software abstractions to life in three ways. First, it encourages you as you explore, by giving you concrete examples that reinforce intuition and suggest new scenarios. Second, it keeps you honest, by helping you to check as you go along that what you write down means what you think it means. And third, it can reveal subtle flaws that you might not have discovered until much later (or not at all).

Because model-exploration is property-free, it does not yield immediately-actionable concrete counter-examples. Therefore, model exploration requires particularly careful attention to output instances. The intent of showing these instances is that users will gain an understanding of a system or phenomenon, ideally as soundly and completely as possible. Therefore, we should make sure that the instances we present actually accomplish this. Indeed, we believe the same argument also applies to the case of presenting counter-examples to verification. However, this paper focuses on what we believe is the harder problem—because it is more open-ended—of model *exploration*, in the absence of verification goals.

Unfortunately, as we discuss in section 11, very little attention appears to have been paid to the design of model-finder output. The typical model-finder generates a stream of instances, which are shown on screen one at a time, with the user given the choice of looking at the next instance (not even necessarily returning to the previous one). This output might be embellished in some ways: e.g., in domain-specific settings, tools sometimes use visual metaphors common to that domain. Other work (e.g., Clarisó and Cabot (2020); Macedo et al. (2015); Nelson et al. (2013); Saghafi et al. (2015)) argues for applying various *mathematical* principles to the output. But there is little to no evaluation of these methods, and what evaluation there is (e.g., Danas et al. (2017)) is mixed and even negative.

The problem being addressed by model exploration is not, however, new. In its essence, the problem is one of examining concrete instances of an abstract object and arriving at a correct and thorough understanding of that object. Viewed that way, a long line of work in perceptual psychology, broadening into cognitive science and educational psychology, has studied similar questions: there are many domains where people have to generalize from concrete instances, especially visual ones. We discuss this in section 2.

Concretely, those results suggest two specific strategies that can be useful. We translate these into research questions that this paper takes the first steps toward answering:

RQ 1. *What is the value of showing* multiple *instances at a time, rather than just one?*

RQ 2. *What is the value of showing* negative *instances (i.e., non-instances of the model)?*

RQ 3. *What is the value of showing both* multiple and negative *instances at a time?*

In this paper, we use both controlled studies and free-form talk-alouds to examine these questions. We explore several variants of these features, and in general find positive support especially for showing negative instances. We hasten to note that our findings are preliminary, and should be thought of as *formative*. As we discuss in sections 4 and 10, the design space of models is large, and laden with confounding factors; our results apply only to a limited (but carefully chosen) setting. Nevertheless, these results lead to further questions and prompt us to believe that these and other interface ideas deserve much more empirical exploration than they have so far received.

[1]Sadly, the terminology in this space is confusing due to a clash between logic and software engineering. In this paper, we follow Alloy in referring to the input constraint-set—think "specification"—as a *model* (as in software engineering) and a satisfying output valuation as an *instance*. We nevertheless follow logic in using *model finder* for the tool that produces instances—the same sense as used in "model checker" by, e.g., Clarke et al. (2009).

We have implemented interfaces for our studies atop the open-source Sterling (Dyer and Baugh 2021) instance visualizer. We pick Sterling for several reasons:

- Sterling is designed to be used as a plug-in replacement for the Alloy visualizer. Thus Alloy users can already benefit from these ideas.
- Sterling is designed for embedding, rather than being tied to any particular tool (like Alloy). Indeed, other prototype model-finders also use Sterling. Thus, building atop Sterling eases future work with model-finders beyond Alloy.
- Sterling is scriptable, enabling features like custom theming, which we use in our experiments for consistency and clarity. We do not use additional features of Sterling, such as domain-specific visualizations, which may also be beneficial but are outside the scope of this work.
- Sterling eases visual comparison across instances by maintaining the position of atoms, which we believe benefits our work.

We therefore believe our use of Sterling (rather than, say, the Alloy Analyzer itself) will ease the repetition and expansion of our experiments in other settings. However, the results also stand (largely) independent of Sterling and do not require its adoption. Likewise, although we used the Alloy language to build models for our studies, this paper is relevant to visualizing output from any model-finding tool, such as an SMT solver.

2 COGNITIVE SCIENCE BACKGROUND

A model-finder essentially asks the user to construct a high-level understanding from a collection of concrete visual instances. This general problem, removed from the logic context, has been studied for a long time in perceptual psychology and other fields. The seminal work of Gibson and Gibson (1955) on *contrasting cases* demonstrated that learning happens when noticing structural differences across examples that vary in a systematic way. These ideas were translated into a variety of other fields, ranging from identifying the sex of young chickens to distinguishing NATO tanks from Warsaw Pact ones (Biederman and Shiffrar 1987). The book by Gibson (1969) is a guide to much of the early work in the area.

Another important line of work is on learning from examples, with that large body of work summarized by Atkinson et al. (2000). A significant amount of contemporary work combines these threads, showing students collections of concrete examples to help them understand a more abstract concept (such as algebraic manipulation rules or the notion of density). Rittle-Johnson and Star (2007, 2009), and a line of work culminating in Schwartz et al. (2011), shows that students learn better if they see alternatives side-by-side rather than sequentially. Note that these are but a representative handful of a whole body of literature. A useful meta-analysis of the related area of learning from case comparisons is given by Alfieri et al. (2013).) This inspired our original emphasis on showing multiple instances at a time (RQ 1).

The examples used in the aforementioned lines of work, however, are *positive* examples: that is, they are actual examples of the phenomena under discussion. In some cases, this is an explicit part of the theory. For instance, Brown and Hanlon (1970), studying language acquisition in children, claimed that children received little direct correction (a form of negative feedback). Because this leaves open the question of how people then eliminate grammatical errors, this has led to a long line of work on the exact nature of feedback. Marcus (1993), for instance, summarizes some of the ensuing debate and argues for why forms of negative feedback are unlikely to be the cause of unlearning grammatical mistakes.

However, much of this discussion has taken place in the context of the social world, where various norms govern behavior (e.g., correcting people has a social cost; people rarely make ungrammatical sentences on purpose; etc.). In a model-finding setting, we have a virtual world that is not governed by the same constraints. In this setting, reading the literature on variation theory (Marton 2014), a phenomenological approach to concept formation, led us to also consider the possible use of *negative* examples: that is, instances that are not actually members of the phenomenon (for instance, see Marton and Pang (2013) and Kullberg et al. (2017)). We also found supporting evidence for negative instances in Winston (1970) work on machine recognition; and, if not for negative instances per se, then for contrasting with unlike categories (Hammer et al. 2009). This inspired our research question about negative instances (RQ 2), and naturally, therefore, its combination with multiple instances (RQ 3).

We also used the research to drive how we generated instances. For instance, in the **SpaceTravel** model (section 4), four negative instances were chosen explicitly to draw attention to a single constraint violation. This was inspired by ideas from Gick and Paterson (1992) and Hammer et al. (2009) and many others who have reused or replicated these ideas, such as Schwartz et al. (2011). We believe that much more can be learned from the research in this area to carefully target the nature and order of instance generation, controlled by injecting suitable logical propositions into the solver instances for generating subsequent instances.

3 ROADMAP OF THE PAPER

This paper involves many different experimental conditions, each tested in the context of multiple models and over both quantitative and qualitative studies, and across several stages over well over a year. These are all somewhat mutually dependent, but in a paper must necessarily be presented sequentially. Therefore, we provide a roadmap here for how the subsequent sections are laid out.

In all the studies, subjects explored instances of a model in an attempt to understand it. They were not shown the model (i.e., the Alloy specification; we discuss this further in section 10), only visual output. We tested participants' understanding of the constraints by having them classify false and true instances and properties.

- *What models did we use in our studies?* We used a set of carefully-designed models, which we describe in section 4.
- *What were the experimental conditions?* We used a total of seven different conditions, one a control and six new ones based on the literature discussed in section 2. We describe these in section 5.
- *Which instances of the models did we use?* This is also described in section 5. The answer depends slightly on the specific condition.
- *Who participated in our study?* We used crowdsourcing site Prolific for our quantitative study, and qualified university students for the qualitative one. We describe the subjects in section 6.
- *What were the study protocols and how did we evaluate performance?* Section 6 discusses both these issues.
- *What are our quantitative findings?* Section 7 describes what we found.
- *What are our qualitative findings?* Section 8 describes what we found.

Preview of Results. Across all three small models used in our quantitative studies, we find that displaying multiple positive instances simultaneously does not appear to have any positive or negative effect on performance. The introduction of negative instances, however, does appear to provide some benefit. Although conditions perform differently across different models, each model saw a noticeable benefit from providing negative instances alone or mixed with positive instances.

Our qualitative, talk-aloud study revealed that when viewing negative instances, participants hypothesize about constraints that are being violated and then verify that those constraints are never broken in any positive instances. Further, participant vocalizations on a larger model suggest that: (1) when the number of possible constraints to test is very large, the approach described above does not necessarily lead to any understanding of the model, since participants needed to first pick the right hypothesis to test; and (2) carefully chosen pairs of positive and negative instances, presented in a way to highlight the differences, can assist with hypothesis selection.

4 EXPERIMENT DESIGN: MODELS

It would be impractical to begin with large industrial or academic models, such as Chord (Zave 2012) or web security (Akhawe et al. 2010), because those involve significant domain knowledge, and a layperson (or even a computer scientist without the right training) might find it difficult to reason about the instances alone without any of the other relevant background. Moreover, larger models, which often have larger instances, would add more interface-related variables to the experiment. We opted for small models in this work so that we could obtain preliminary results that could then guide later studies on larger models.

We also decided against using existing small models from Jackson (2012) or other published works, both to avoid similar domain dependencies, and to sidestep familiarity participants may have with these models. Consequently, we constructed our own set of Alloy models for use in these studies. In doing so, we were guided by three broad design goals:

- selecting domains that would give participants enough context to begin making hypotheses, without the possibility that some might be *experts* in that domain;
- selecting constraints that were non-obvious while still being reasonable possibilities within the domain, and which had a variety of different relational shapes; and
- increasing complexity levels between different models, so that we could see how each condition fared as the space of hypotheses increased.

In each model, we embedded four *target constraints* for participants to discover, along with some structural constraints (such as those enforcing the type of a relation) that we did not use for evaluation. We briefly sketch the three models here along with an example instance of each in fig. 1. (Due to space limits, we include the full models along with our artifact.)

For our qualitative study, we introduced a fourth, more complex, model. It describes the layout of a small fictional town and its residents. We present it in section 8.

4.1 FeedingRitual Model

One model crafted a narrative about a fictional species of singing blobs that followed a rigid social ritual when eating. This was directly inspired by the course-requirements model from Montaghami and Rayside (2017), but we wanted to avoid participants' prior knowledge about course requirements from either helping or interfering with their understanding and thus confounding their performance.

The model contained two types: `Blob` and `Food`. The `Food` type was restricted to include exactly one element, which is how a unique constant is usually denoted in Alloy; the nature of the food is irrelevant, only that some blobs are eating. Two additional relations encoded which blobs were eating (`eating`) and which blobs were singing and to whom (`singingTo`). We also wanted to minimize the number of salient features (section 2): if participants were overwhelmed by features, they would presumably yield far noisier data (a hypothesis borne out in section 8).

The model's target constraints were: (1) exactly two blobs must always be eating; (2) a blob can only sing to one other blob at a time; (3) if a blob is singing and eating, the blob it is singing to must also be eating; and (4) blobs cannot sing in a cycle (i.e., the singing relation is acyclic). In the properties test (section 6.2, step 6), we included these plus four invalid rules: (1) a blob can sing to any number of Blobs; (2) a blob must always be singing; (3) a blob can sing to itself; and (4) any number of blobs can be eating.

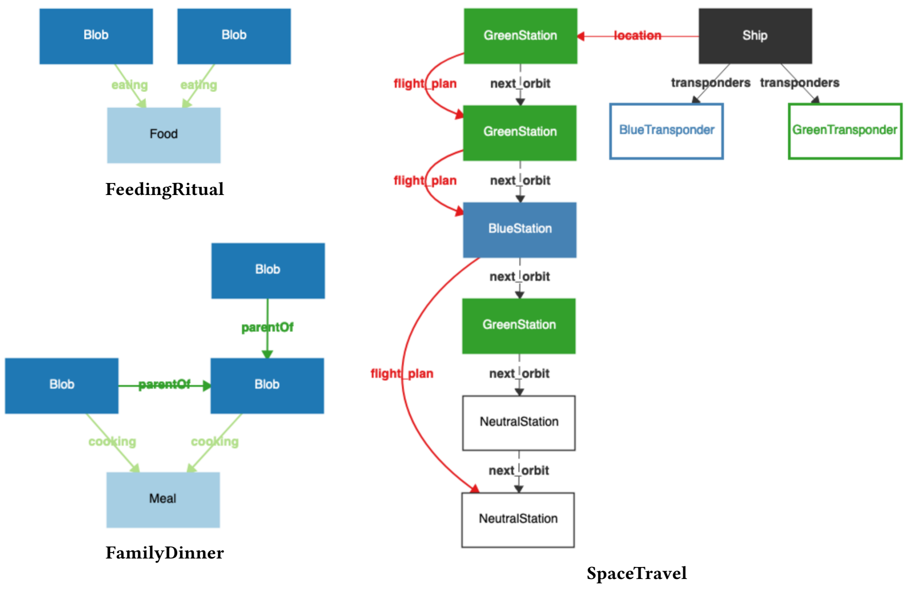


Fig. 1. Valid instances from each of the three models

4.2 FamilyDinner Model

Another model examined social relationships by modeling the rules of a dinner party. To avoid pre-conceptions about *human* relationships, the model was again about blobs. As before, this model contained two types: `Blob` and `Meal`. The `Meal` type was restricted to include exactly one element. The `cooking` relation described which blobs were cooking a meal, and `parentOf` encoded the ancestor relationship between blobs.

The model's target constraints were: (1) exactly two blobs must always be cooking; (2) A blob has at most one child; (3) if a blob has a child and is cooking, the blob's child must also be cooking; and (4) the `parentOf` relation is acyclic. The properties test (section 6.2, step 6), also included four invalid rules: (1) a blob can be a parent of multiple blobs; (2) every blob must cook; (3) it is possible for a blob to be its own parent; and (4) any number of blobs can be cooking.

The similarities between **FeedingRitual** and **FamilyDinner** were deliberate; we discuss this, and the implications thereof, in section 12.

4.3 SpaceTravel Model

The **SpaceTravel** model describes the flight plan of a spaceship traversing colored space stations to move from the outermost orbit around a fictional star to the innermost orbit. We created this model specifically to be larger (in terms of instance size) and more complex (in terms of number of relationships). Furthermore, we changed the idiom from blob social rituals to one that is likely more recognizable: point-to-point travel. By placing the scenario in space, we hoped to avoid any cultural factors that might come with specific forms of travel in recognizable settings, such as driving between cities.

In **SpaceTravel**, stations can be blue, green, or neutral. The spaceship can have blue and/or green transponders. We represented station and transponder color both textually (e.g., `GreenStation` or `BlueTransponder`) and by using actual colors (green and blue) in the visualizer. This reduced the risk of confusion caused by a mismatch between two different perceptual signals: in this case words corresponding to colors and the actual colors used (the Stroop effect (Stroop 1935)).

Instances of the model depict possible flight plans that the spaceship could follow to reach the innermost station from the outermost station; the set of stations is not always the same across instances. The target constraints of the model are: (1) to stop at a blue or green station, the ship must have a transponder corresponding to the color of that station; (2) the ship cannot backtrack by moving to a station further away from its destination; (3) the ship must reach the station at the innermost orbit; and (4) the ship cannot skip more than two stations without docking. The properties test (section 6.2, step 6) for **SpaceTravel** also included four invalid rules: (1) every station must be visited; (2) the ship must have a transponder to dock at a neutral station; (3) the ship cannot visit more than three stations; and (4) the ship may return to a station it has already visited.

5 EXPERIMENTAL CONDITIONS

Our research questions (section 1) seek to learn how two separate variations from the default impact users: showing multiple instances simultaneously (RQ 1) and showing (labeled) negative instances (RQ 2). These questions induce a vast design space of possible studies, especially when the two ideas are combined (RQ 3).

As the goal of this work is formative, we crafted six new experimental interfaces (fig. 2) that capture reasonable points in the design space. We denote the default interface, which reflects the Alloy presentation (one positive instance at a time), as **1+**. The **4+2-** and **6+** conditions are likely not realistic, given that six instances of any appreciable size would be difficult to fit onto a single screen in a meaningful way, but we tested them for completeness on these relatively small models.

These interface designs are intertwined with decisions on *how many* instances to display. Nothing in the literature discussed in section 2 addressed this variable, so we fixed the instance count across all experiments. In our own experience using and teaching Alloy, users exploring instances will view some small number of them before returning to the model itself. To get a good approximation of how many instances are typically viewed, we collected data from students in an introductory formal methods course. Throughout the course, students generated instances from 17 different models, executing around 41,000 run commands, using a version of Sterling modified to collect the number of instances viewed for each run command. Of the times students viewed more than a single instance, we found they rarely viewed more than 12, with an average of 8.33 and median of 5. We therefore displayed 12 (sometimes negative) instances in these studies.

5.1 Obtaining Negative Instances

By definition, a model finder produces instances that satisfy *all* input constraints. Since a negative instance fails to do so, defining and obtaining them requires some precision. The input is a conjunction of constraints $C_1 \wedge \ldots \wedge C_n$ over a given language of relation (or uninterpreted function) symbols. We might, therefore, begin by searching for instances of the negation $\neg(C_1 \wedge \ldots \wedge C_n)$. This is trivial to construct, since models fed to SAT or SMT-solvers are generally phrased as conjunctions of constraints.

Unfortunately, the negate-the-conjunction approach can produce negative instances witnessing the failure of multiple target constraints. Worse, it negates structural constraints (e.g., in **FeedingRitual**, that blobs and food are disjoint types). Both of these flaws risk showing nonsensical instances that distract from the target constraints. Moreover, there is no guarantee that, for every distinct target constraint C_i, a negative instance witnessing the failure of C_i will be produced early enough in the instance stream to be discovered.

Instead, we therefore broke the overall problem into separate sub-problems. For each distinct target constraint C_i, we separately searched for instances that satisfied all constraints except C_i,

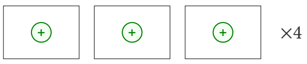
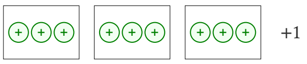
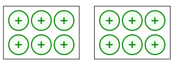
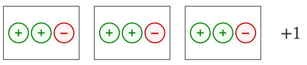
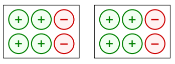
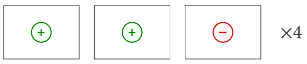
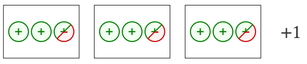
Condition	Description	View Sketch
1+	12 positive instances in total. 1 instance displayed per page. This is the default mode of exploration against which all other conditions are compared.	
3+	12 positive instances in total. 3 instances displayed per page.	
6+	12 positive instances in total. 6 instances displayed per page.	
2+1-	8 positive and 4 negative instances in total. 2 positive and 1 negative instance displayed per page.	
4+2-	8 positive and 4 negative instances in total. 4 positive and 2 negative instances displayed per page.	
++-	8 positive and 4 negative instances in total. 1 instance displayed per page, every third page displaying a negative instance.	
2.5	12 positive instances and 4 negative instances in total. 2 positive and 1 negative instance displayed per page, but the negative instance includes a button that modifies the negative instance to make it a positive one.	

Fig. 2. Summary of Experimental Conditions. The **View Sketch** column presents the sequence of displays a participant in that group would use to try to understand the model. Each rectangle represents a computer screen, and each circle represents an instance. Circles with a + are positive instances and those with a − are negative instances. A ×4 after the sequence indicates that the complete sequence is repeated four times, and a +1 indicates that there is one additional screen of the same type that completes the sequence.

that is, which satisfied $(C_1 \wedge ... \wedge C_{i-1} \wedge \neg C_i \wedge C_{i+1} \wedge ... \wedge C_n)$. Since we had designed every target constraint to be independent of the others, this query was satisfiable in all cases. This let us study negative instances in their best and most consistent light.

5.2 Instance Selection

For each model, we generated separate sets of instances to display during the exploration phase and instance test. All three positive conditions (**1+**, **3+**, and **6+**) used the same instances during the exploration phase, varying only in the mode of display. Similarly, the **++-**, **2+1-**, and **4+2-** used the same instances. The **2.5** condition added one additional positive instance to enable toggling.

For each of these three models, running our study protocol for all seven experimental conditions (fig. 2) requires a total of 17 positive instances and 9 negative instances. Of those, 13 positive and 4 negative instances are used during the exploration phase and the remaining 4 positive and 4 negative are used in the instance test. For the exploration phase, the specific set of instances, along with how those instances were presented, was determined by experimental group. For each model, the same set of four positive and four negative instances was used across all experimental groups during the instance test.

In a model-finding setting, a certain element of "luck" is to be expected in instance generation. The extent to which one instance of billions triggers a productive realization depends on the the user's mental state as much as the tool. Yet, there is value to trying to reduce dependence on luck, as many prior works (Clarisó and Cabot 2020; Cunha et al. 2014; Macedo et al. 2015; Nelson et al.

2013; Saghafi et al. 2015; Sullivan et al. 2017) have argued. We exercised care in selecting the specific instances used, drawing on techniques from preexisting tools where appropriate and falling back on manual selection where needed. It is worth noting that in both cases, the process of selecting these instances can largely be automated.

Blob Models. For both the **FeedingRitual** and **FamilyDinner** models, we took as a starting point the first 4 instances that Alloy itself generated for the models using the SAT4J solver. (We used a scope of up to 6 blobs and 1 meal.) Then, for each of those positive instances, we generated (via the target-oriented model-finding features of Pardinus (Cunha et al. 2014)) one *minimally* distant and one *maximally* distant positive instance. This gave us 12 positive instances in total. For each minimally distant positive instance, we then generated (again using Pardinus and the algorithm in section 5.1) a minimally distant negative instance, yielding 4 negative instances. Finally, for each negative instance, we produced a (distinct) positive instance for use with the **2.5** condition's toggling interface. This positive instance was manually chosen using a method similar to the one used in Bordeaux (section 11) to create a near-miss/near-hit pair that would guide participants towards relevant features.

SpaceTravel *Model.* Because of the increased complexity of **SpaceTravel**, we found that automatically producing maximally-distant instances resulted in quite dense and unruly instances that were difficult to read, and an automatic enumeration of minimally-distant instances was not guaranteed to exercise different dimensions of the instance space. We therefore chose **SpaceTravel** instances manually in a deliberate fashion, which research suggests is a more useful approach than choosing at random (Hammer et al. 2009). All instances were generated using a maximum scope of 6 for all types. For positive instances, we chose instances that were not just minimally distant, to increase the likelihood of feature exploration (rather than encouraging a focus on narrow differences between two nearly identical positive instances). For negative instances, we manually followed the approach in section 5.1, selecting instances that specifically violated exactly one target constraint (of the four that we wanted participants to identify). We manually selected positive instances for the **2.5** togging interface to guide participants towards relevant features, as we did for the blob models.

5.3 Why So Many Experimental Conditions?

This work proceeded in multiple phases. At first, we formulated a small set of quantitative studies, guided by student conversations and other experiences from decades of teaching formal methods. This first round only compared ensembles of instances (**3+**, **6+**, **2+1-**, and **4+2-**) against the default interface (**1+**) and only used the two most basic models (**FeedingRitual** and **FamilyDinner**). The trends that emerged from these early experiments suggested that negative information (or, at least, the contrast between positive and negative) could be useful. Groups that viewed negative instances tended to perform better than those who did not.

However, it was not clear what role the *immediate contrast* between positive and negative instances played. Moreover, participants using negative instances in **FamilyDinner** tended to perform better than those using negative instances in **FeedingRitual**, suggesting that the presence of recognizable relationships impacted the ability to use negative information. Yet, our ability to interpret these results was limited by the simplicity of the two models we had built so far.

We therefore designed the next round of experiments to introduce a more complex model (**SpaceTravel**). We wanted more recognizable idioms than the social rituals of blobs, and hence used point-to-point travel. Still, we did not want excessive interference from cultural travel conventions; hence the location in space. We also added the **++-** condition to test the presence of negative instances without also showing them alongside positive instances. The **++-** group outperformed the default in all three models, suggesting that negative information provides a benefit even without direct juxtaposition with positive instances.

The results from **SpaceTravel** suggested that the value of negative information might be diminished as models become more complex, independent of the recognizability of the domain: the negative-instance conditions performed more closely to the **1+** default than they did in either **FeedingRitual** or **FamilyDinner**.

Nevertheless, it seemed reasonable that the problem might be one of focus: as instances become more complex, finding points of divergence between side-by-side instances can become overwhelming. We opted to add another condition (**2.5**) that would call attention to these differences. **2.5** participants saw a 3-instance ensemble as in **2+1-**, but could toggle between the negative instance and a minimally different positive instance; Sterling allowed us to keep atom position constant for quick visual comparison. We found that participants using this condition did notably *worse* in **FeedingRitual** and **FamilyDinner** but much better in **SpaceTravel**. This led to our final round of studies, where we focused on gathering new qualitative information in the hopes of discovering the cause of this effect.

6 EXPERIMENTAL SUBJECTS, PROTOCOL, AND METRICS

To address the research questions from both quantitative and qualitative perspectives, participants were drawn from two populations. For quantitative analysis, the goal was to recruit enough participants to perform statistical analysis on their performance and develop an understanding of the overall trends when comparing experimental groups. We recruited a total of 848 participants (before the filtering of section 6.3) on the crowdsourcing site Prolific. They were offered USD 9.52/hr, and were required to confirm they were working on a desktop computer and fluent in English. They were given an estimated completion time of 15 minutes and were allowed up to 56 minutes to complete their tasks. They were only allowed to participate in a single study. We expected that most would have no significant experience with formal modeling.

For qualitative analysis, we were able to recruit 7 students at Brown University who had completed a formal methods course the previous semester. The course had used Alloy extensively, so students had experience using Alloy and interpreting Alloy instances. These students were offered a USD 20 Amazon gift card in exchange for approximately an hour of time to complete their tasks. They participated in a talk-aloud study, in which they were asked to vocalize their thought processes while completing the same tasks as the participants from the Prolific population. These results were used to better understand how negative information and multiple instances were used.

A review of the experimental design by our IRB board determined that no IRB application was required for these qualitative studies, as no identifiable private information or identifiable biospecimens were obtained, used, or studied. Nevertheless, we followed standard research protection protocols with all participants.

6.1 Measuring Model Comprehension

We are broadly interested in the role that our experimental conditions play. The most concrete measurement we can apply is to see how well subjects understand the (hidden) model behind the instances shown. To do so, we chose two concrete metrics that measure some aspect of "model comprehension". First, it seems reasonable to believe that a person who has formed a good understanding of the hidden model will be able to more accurately determine if an *arbitrary instance* satisfies the model. Second, better understanding of the model should lead to better judgments about whether the model entails a given *property*. These two activities—instance classification and property classification—form the basis of our empirical evaluation.

6.2 Study Protocol

We tested each condition in fig. 2 using the same experimental protocol. In each study, participants began with an *exploration phase* where they were instructed to explore a set of instances with the intent of determining the constraints of the underlying model. Participants then proceeded to an *instance test* where they were shown different instances and asked to classify each as valid or invalid based on their understanding so far. Participants who did not correctly classify *all* instances were required to repeat the exploration phase (being told that they did not answer enough questions correctly), after which they were required to retake the instance test. Finally, participants were shown a set of possible model *constraints* (in English) and asked to identify which were actually part of the model (the *properties test*).

In more detail, all subjects executed the following steps, varying only in the model used to generate instances and the method of instance exploration:

(1) **Introduction**: Participants read a description of the tasks they will be asked to perform.
(2) **Pre-test**: Participants took a short quiz to ensure that they were able to understand the instances. Quiz questions asked for simple observations such as the number of a specific type of atom or the number of edges with a certain label.
(3) **Explore**: Participants used the visual form of their condition to explore a series of instances, from which they were expected to understand the model.
(4) **Instance Test**: Participants took a quiz in which they had to classify each instance in a set of instances is valid or invalid. For each instance they classified as invalid, they had to give a written description of the rule or rules that the instance breaks. If the participant did not correctly classify all instances in the set, they were returned to the Explore phase for a second (and last) attempt.
(5) **Survey**: Participants took a survey in which they were asked to give a written description of the constraints of the model, what their profession is, and for a description of any written notes they took during the experiment.
(6) **Properties Test**: Participants had to pick out constraints of the model from a list of 8 possible constraints. The list contained 4 actual constraints and 4 non-constraints.

Corresponding to the measurement principle in section 6.1, participants were evaluated on two metrics: instance test score and properties test score. The instance test score is the number of instances (out of 8) that a participant was able to correctly classify as either valid or invalid. Participants who did not correctly classify all 8 instances on their first attempt, and therefore took the instance test twice, have two instance test scores. The properties test score is the number of constraints (also out of 8) that a participant was able to correctly classify as either valid or invalid.

6.3 Filtering Responses

As crowdsourcing sites can sometimes yield poor answers (participants are rushed, uninterested, or bots), we performed some quality control, following Kittur et al. (2008), to remove participants who appeared to not be actively engaged with the study. Concretely, we included two mitigations.

First, the pre-test (step 2) asked simple questions that allowed us to filter out obviously-disengaged participants. For example, given the instances in fig. 1, we asked "which blob is singing?", "how many blobs are cooking?", and "how many flights are in this flight plan?" We expect that bots programmed to select answers at random would (probabilistically) fail this test.

Second, we used the following coding manual to classify participants based on their written answers to the properties test (step 6):

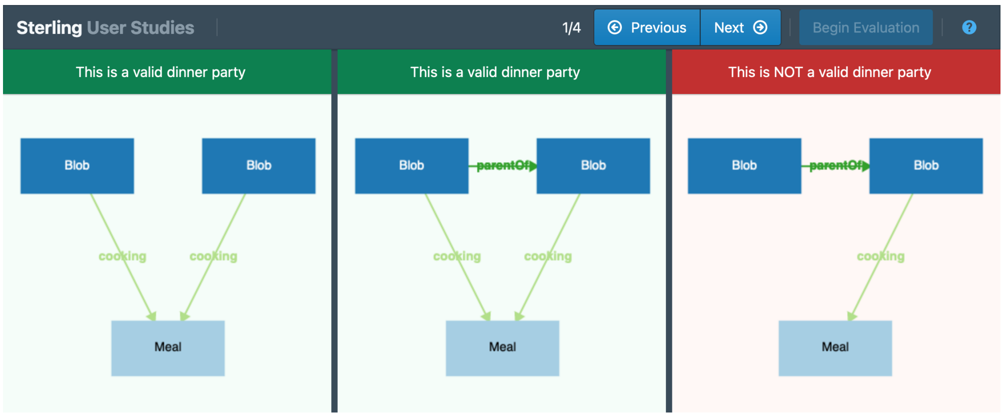

Sterling User Studies
1/4
Previous
Next
Begin Evaluation
This is a valid dinner party
This is a valid dinner party
This is NOT a valid dinner party
Blob
parentOf
cooking
Meal

Fig. 3. The study interface showing two positive and one negative instance of the **FamilyDinner** model.

- **Engaged:** In their written responses, the participant has expressed at least one rule or pattern that is not explicitly given in the instructions or pre-test (such as "blobs are eating" or "blobs are singing").
- **Not Engaged:** In their written responses, the participant has ***not*** expressed any rules or patterns, or has only repeated those given in the instructions or pre-test. (We viewed simply repeating the instructions as a failure to comprehend them, or possibly evidence of an automated response.)

Responses were coded with other fields, such as numeric scores, hidden so that participants would be filtered objectively, without regard to their effect on study outcomes. To ensure that there would be alignment (beyond chance) between multiple people using the coding manual, we performed a check for inter-rater reliability using Cohen's κ (Cohen 1960) among 3 coders using results from 43 participants. We obtained a κ score of 0.855, which is a strong agreement, and so all subsequent classification was performed by a single coder.

6.4 Study Interface

We built custom interfaces atop Sterling to administer the experiments. We used Sterling because it largely mirrors Alloy's default directed-graph visualizer, with two important differences: it is web based, which eased administering the study, and it holds atom locations constant as the user moves between instances. This means that users can usefully "toggle" between instances without common elements being arbitrarily rearranged. We believe that this reduces the cognitive load of viewing instances; an empirical evaluation focused specifically on the benefits of Sterling is outside the scope of this paper, but we believe such factors are worth separate examination.

The interface (fig. 3) was developed as a JavaScript React application and was deployed on Heroku. We collected data in a MongoDB database. We did not collect Prolific IDs, to protect against any future de-anonymization of those IDs. A navigation bar at the top of the screen provided buttons to navigate through the pages of the study. Intermediate pages provided textual instructions to the participant before the beginning of each section.

The pre-test questions, instance test, properties test, and survey questions were all identical across conditions, with only the interface itself (in the form of instance presentation layout) varying. The pre-test section was presented in a two-column layout, showing an instance in the right column and a multiple choice question prompt related to that instance in the left column. When the participant

Table 1. ANOVA tables for all three models. For each model, analysis is performed separately on the first instance test (I1), second instance test (I2), and properties test (P). Column labels correspond to the standard ANOVA columns: degrees of freedom (dF), sum of squares (SS), mean square (MS), F-ratio (F), and p-value (p) for each model. In the interest of space, the Group (G) and Residuals (R) row labels have been shortened. Lower p-values indicate stronger statistical evidence for an effect.

		FeedingRitual					**FamilyDinner**					**SpaceTravel**				
		dF	SS	MS	F	**p**	dF	SS	MS	F	**p**	dF	SS	MS	F	**p**
I1	G	6	0.56	0.09	3.20	**0.005**	6	0.97	0.16	6.10	**<0.001**	6	0.18	0.03	0.94	**0.466**
	R	217	6.33	0.03			188	5.00	0.03			217	6.95	0.03		
I2	G	6	0.83	0.14	6.03	**<0.001**	6	0.30	0.05	2.14	**0.051**	6	0.26	0.04	1.13	**0.344**
	R	215	4.94	0.02			187	4.36	0.02			215	8.32	0.04		
P	G	6	0.31	0.05	1.55	**0.163**	6	0.17	0.03	1.07	**0.382**	6	0.07	0.01	0.58	**0.745**
	R	197	6.62	0.03			169	4.50	0.03			197	4.04	0.02		

answers each question they are told whether they answered correctly before moving on to the next question. If they answered incorrectly, they are shown the correct answer. All the survey instruments are available in full in our artifact bundle.

During the exploration phase of the study, instances were displayed to the subject in the arrangement corresponding to their assigned experimental group using the Sterling graph view. For example, the screenshot in fig. 3 shows the second set of instances in the 2+1- condition of the **FamilyDinner** study. The exploration interface allowed participants to move both forward and backward (via the "Next" and "Previous" buttons shown in fig. 3). This differs from the default Alloy interface, which does not permit returning to previous instances. We made backtracking possible since, in the multi-instance conditions, subjects could already see multiple instances on screen at a time, simulating the effect of going back and forth. By preventing subjects from ever going back, we would be testing their recall, which was not our goal. By letting them go back and forth, we better measure the effect of seeing multiple instances together at once.

To help reduce cognitive load in experimental groups that include negative instances, all instances were assigned a label indicating whether they were positive or negative, and instances were given a light colored background (green for positive, red for negative), as the figure shows. The specific shades of red and green were chosen to ensure enough contrast for the colorblind to distinguish. Instances in groups with only positive instances had a white background.

During the instance test, a single instance was displayed in the right column of a two-column layout. The left column showed a prompt asking whether the instance was valid, with "Yes" and "No" buttons. Clicking the "Yes" button advanced the participant to the next instance, and clicking the "No" button revealed a text box along with a prompt asking which rule or rules had been broken. The participant could not proceed to the next instance without giving a written response.

We stress that, while we used Alloy internally to construct study instances, and Sterling's visualization is based on Alloy's default view, subjects did not interact with Alloy in any way. They were (intentionally) never shown a formal modeling language; properties, for instance, were written in English. Though this may reduce some authenticity (as we discuss in section 10), it increases the applicability of this work: e.g., subjects could as well have been seeing the output of a solver embedded in some domain-specific tool that never exposes the logic layer.

7 QUANTITATIVE STUDY RESULTS

All results in this section include only participants that were accepted from each study using the quality control filters outlined in section 6.3. No subjects failed the pre-test questions. The only

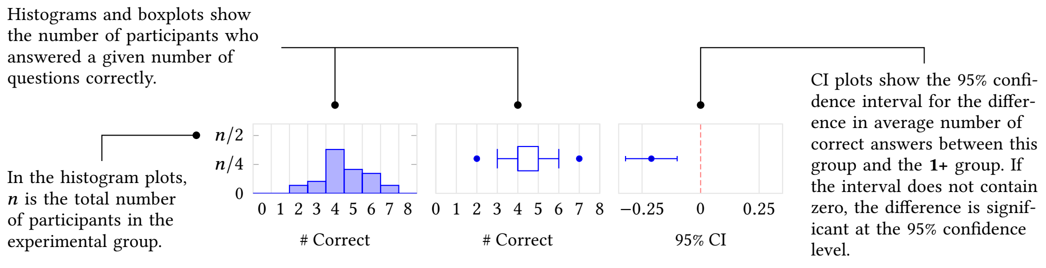


Fig. 4. Second instance test results from the **2.5** group in the **FeedingRitual** study, which are presented here to describe how to read the complete results presented in fig. 5.

participants excluded were therefore those eliminated for failure to provide written responses indicating that they were engaged. Across all quantitative studies, acceptance rates were between 57–92%, with the median and average acceptance rates being 78%.

For each model we collected quantitative performance data from approximately 30 (post-filtered) participants for each group. We analyzed each of the three evaluation metrics using one-way ANOVA (Hays 1994), or analysis of variance, a common statistical method used to determine whether there are significant differences among the means of three or more independent groups. ANOVA results are typically reported in table form, showing the between and within group variability calculated as sums of squares, along with data derived from those values, including the F-ratio and p-value. These results, which we calculated at the 95% significance level, are presented in table 1.

ANOVA results from the **FeedingRitual** study show that there is a significant level of variance for the first ($p = 0.005$) and second ($p < 0.001$) instance tests, but not for the properties test ($p = 0.47$). ANOVA results from the **FamilyDinner** study show that there is a significant level of variance for the first instance test ($p < 0.001$) but not for the second one ($p = 0.051$) or the properties test ($p = 0.34$). Finally, ANOVA results from the **SpaceTravel** study show that there is no significant variance in the first ($p = 0.16$) or second ($p = 0.38$) instance test, or the properties test ($p = 0.75$).

While these results indicate that there are certainly measurable effects present in some cases, we emphasize that these studies are meant to be exploratory. We are less interested in the statistical significance of the results than we are in understanding trends that may indicate further study of an approach is warranted. To better understand how each case performs relative to the default mode of exploration, we perform a post-hoc analysis using Dunnett's method (Dunnett 1955) to individually compare each experimental group to the **1+** group. These results are presented as confidence interval plots in fig. 5, along with histograms and boxplots, for each experimental group.

Crucially, we are *not* performing a multiple-comparisons test that would enable us to contrast the performance of different experimental groups. Dunnett's method only allows us to compare each experimental group individually against the control group, but at a reduced loss of statistical power compared to other methods. Because we are comparing each experimental group *only* to the **1+** group, there is no confidence interval data for the **1+** group. Figure 4 provides a description of how to read these sets of charts.

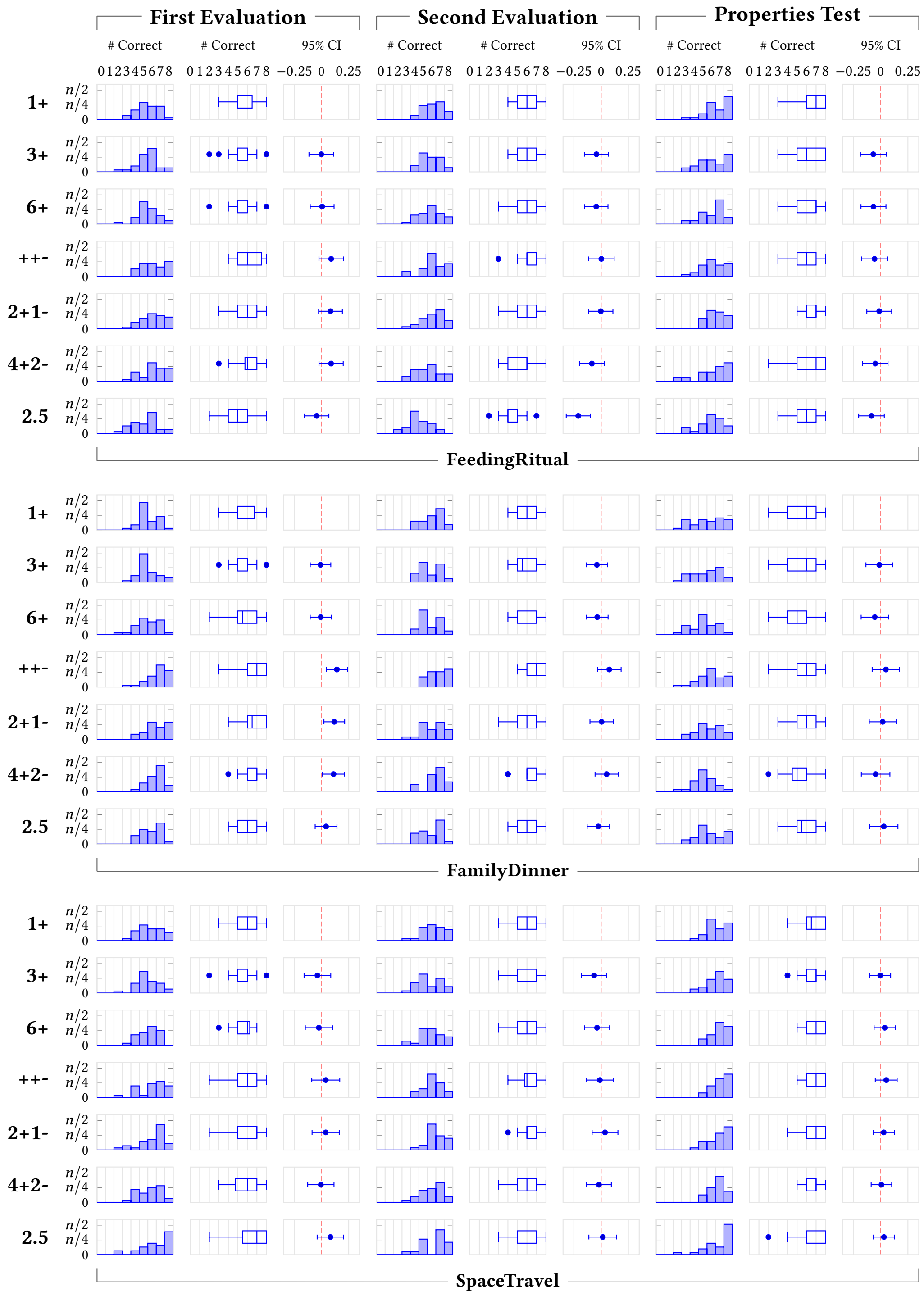


Fig. 5. Quantitative study results. Refer to fig. 4 for a guide on how to read these results.

7.1 RQ 1: Simultaneous Positive Instances

To understand the value of showing multiple instances at a time, rather than just one, we compare the **3+** and **6+** results to the **1+** results for each of the three models.

For all three models, the average first instance test scores all fell into the 68–72% range, or between 5 and 6 of the 8 total instances correctly classified as valid or invalid.

Across all three models, the **3+** and **6+** evaluation metrics bear striking similarity to those of the **1+** group. These results suggest that while presenting simultaneous positive instances does not appear to provide any additional value over presenting individual instances, no harm seems to have been done either.

7.2 RQ 2: Labeled Negative Instances

To understand the value of showing negative instances, we compare results from the ++- group to results from the **1+** group for each of the three models. The ++- group uses the same presentation mode as the **1+** group, displaying only a single instance at a time, but every third positive instance has been replaced by a negative one. By comparing the two groups, we can understand how the presence of negative information affects one's ability to determine the constraints of the model. Note that the ++- group is the only one that includes negative instances while still using the default presentation mode of a single instance at a time.

In the **FeedingRitual** study, the average first instance test score was 69% for the **1+** group and 77% for the ++- group. This means that on average, participants in the **1+** group correctly classified 5.5 out of 8 instances while those in the ++- group correctly classified just over 6 of 8 instances. On the second instance test, participants in the **1+** group showed an improvement over their first test scores, averaging 78% correct, while those in the ++- group showed almost no improvement, also averaging 78% correct.

The **FamilyDinner** and **SpaceTravel** studies present strikingly similar trends when considering first and second instance test scores. In the **FamilyDinner** study, the **1+** group correctly classified on average 69% of instances for the first instance test and improved to 76% for the second, while the ++- group correctly classified 84% of instances for both the first and second instance tests. In the **SpaceTravel** study, the **1+** group correctly classified on average 72% of instances for the first instance test and improved to 77% for the second, while the ++- group correctly classified 76% of instances for both the first and second instance tests.

Among these results, only the ++- group in the **FamilyDinner** study showed a significant improvement in the first instance test when compared to the **1+** group ($p = 0.001$), correctly classifying 84%, or 6.7 out of 8 instances. However, the ++- groups from the other studies do appear to follow a similar trend, generally performing better than the **1+** group in all cases.

It appears that participants were able to extract some useful information from the negative instances, as the results suggest that those in the ++- group were able to correctly classify on average one more instance (out of 8) than those in the **1+** group. What's not clear, though, is *how* the negative information is used (a question we explore in section 8), and whether the removal of four positive instances to make room for the negative ones had any negative effect on performance. It is certainly plausible that any negative effect caused by the removal of the positive instances was offset by a comparatively larger positive effect provided by the negative instances. A controlled experiment to test for this effect is one we leave for future study. However, given that there was a measurable positive effect seen in these results, this tradeoff—should it exist—is one that seems reasonable to make.

Within each study, a trend appears when comparing the first and second instance test scores. In all three studies, the **1+** group performed better on average on the second instance test than the first. This suggests that during their second exploration of the instances, participants were able to either discover a constraint they had missed the first time or correct one that they had previously thought to be true. The ++- groups, however, show almost no improvement when comparing their second instance test scores to the first. One might take this to mean that participants in the ++- learned as much as they could during the first exploration phase.

However, if we consider the number of participants who got a *perfect* score on the first instance test, and therefore did not take the second, a different perspective emerges. In each of the **FeedingRitual** and **FamilyDinner** studies only a single participant in the **1+** group got a perfect score on the first instance test and therefore did not take the second one. From the ++- group, however, 8 participants from **FeedingRitual** and 9 from **FamilyDinner** got perfect scores on the first instance test. Considering that the first instance test averages would be considerably lower in the absence of these perfect scores, it appears that participants from the ++- group who took both instance tests did indeed continue to discover constraints during the second exploration, just as the **1+** group did. This effect is less pronounced in **SpaceTravel**, where 4 of the 30 participants in the **1+** group got perfect scores on the first instance test compared to 5 of 25 in the ++- group. Accordingly, the difference in first instance test scores between the two groups is less pronounced, as is the level of improvement exhibited by the **1+** group from first to second instance test.

7.3 RQ 3: Simultaneous Positive and Negative

To understand the value of showing both negative and multiple instances, we compare results from each of the **2+1-**, **4+2-**, and the **2.5** groups to the **1+** group for each of the three models.

In the **FeedingRitual** study, the **2+1-** and **4+2-** groups performed at about the same level when compared to the **1+** group. Each tended to perform slightly better in the first instance test and either the same or worse during the second instance test and properties tests. Interestingly, the **2.5** group performed worse than the **1+** group on all three evaluation metrics, significantly so on the second instance test. Recall that the **2.5** mode of presentation is identical to that of the **2+1-**, with the addition of a toggling button that allows the participant to toggle the negative instance to a positive one. In this case, it appears that this toggling ability was detrimental to their performance.

In the **FamilyDinner** study, the **2+1-** and **4+2-** groups showed trends similar to those same groups in the **FeedingRitual** study, though the performance on the second instance test tended to trend towards higher scores. The **2.5** group in this case tended to perform about as well as the **1+** group in all three evaluation metrics.

In the **SpaceTravel** study, the **2+1-** and **4+2-** groups performed at about the same level as the **1+** group in all three evaluation metrics. The **2.5** group, however, showed some improvement over the **1+** group, particularly in the first instance test. In fact, the **2.5** group showed the best performance out of all groups in this study.

There are two interesting observations to note from these results. First, when comparing the non-toggling ensemble groups (**2+1-** and **4+2-**) to the ++- group, the results show very similar trends. This appears to indicate that it is the presence of negative information that provides some benefit, not the presentation of ensembles. Second, the **2.5** interface, which was designed to draw attention to specific features of the negative instances, appears to be detrimental to performance on the simple blob models while providing some benefit on the more complex **SpaceTravel** model. This trend in particular is one that we set out to explore in our qualitative studies.

7.4 Summary

On instance classification, subjects who saw multiple positive instances at once performed remarkably close to those who were given the default display. In contrast, different models favored different negative-instance conditions on instance classification. Both standalone negative instances and mixed positive-negative ensembles pulled far ahead of the baseline on the blob models, but showed no noticeable improvement on **SpaceTravel**. Ensembles that allowed active toggling between positive and negative appeared to *hinder* subjects on **FeedingRitual**, but (unlike the passive conditions) showed a strong trend of improvement on **SpaceTravel**. Subjects showed little variation on the properties test regardless of condition; we discuss this further in section 12.

8 QUALITATIVE OBSERVATIONAL STUDY

While the results of the quantitative studies suggest that negative instances are valuable, they do not help us understand *how people actually use them.* Understanding usage has implications for which negative instances to show and when, especially in juxtaposition with positive instances. We therefore conducted a talk-aloud study with a small number of participants who had substantial experience with Alloy. To match the quantitative studies, these participants were given the same models, (subset of) interfaces, and instance test questions; they did *not* use Alloy or see the models. They were, however, asked to verbalize what they were noticing and thinking about as they looked at or interacted with the interfaces.

8.1 Study Logistics

Participants and Conditions. We have described the demographics in section 6. Of the seven, one participant could not maintain verbalization sufficiently to provide useful data, so we excluded them. Each of the remaining six worked in one of three presentation modes (**++-**, **2+1-**, or **2.5**), with two participants per mode. We included **++-** to study a stream of instances, **2+1-** as the most promising configuration from the quantitative studies, and **2.5** to explore the impact of toggling.

Protocol. The study sessions were conducted over Zoom. Participants shared their screens while working and we recorded the sessions for later analysis. Participants had agreed to these recordings in advance through a conventional consent form. The study interface was the same one used in the quantitative studies, with the exception of the initial instructions page. In order to explore how participant experience varied with model complexity, each talk-aloud participant worked with three different models, rather than just one as in the individual Prolific studies. We asked participants to verbalize their observations and thoughts while working. The researcher administering the experiments did not interrupt the participants except to encourage them to continue talking aloud.

8.2 Models

The qualitative study used the **FamilyDinner** and **SpaceTravel** models from the Prolific studies. We used **FamilyDinner** instead of **FeedingRitual** since participants had performed better with the former in the quantitative studies. We chose **SpaceTravel** as a more complicated model than either **FamilyDinner** or **FeedingRitual**.

Because we had a subject pool with baseline Alloy experience, we wanted to include an even more complicated model than **SpaceTravel** (with participants working with the models in order of increasing complexity). We therefore introduced a new model, **CityPlanning**, an instance of which appears in fig. 6. It describes a fictional city composed of homes, different kinds of workplaces, and different kinds of people who live and work there. Buildings are connected by roads and bike paths. The model uses a scope of 6 for homes, workplaces, and residents; the size of each profession (banker, teacher, etc.) is further limited to 2 residents.

While the basic constraints might have been guessable (such as the city must be connected by roads and bike paths and everyone must live somewhere), the target constraints were intentionally subtle. Specifically, our evaluations targeted four constraints: (1) banks and houses must be separated

Table 2. Performance on instance tests and properties test for each of the 7 talk-aloud study subjects. Each row presents the results for a single participant. The columns show the number correct out of 8 for each of the first instance test (I1), second instance test (I2) and properties test(P). (Participant 6 was excluded due to not maintaining verbalization.)

		FamilyDinner			**SpaceTravel**			**CityPlanning**		
Participant	**Group**	**I1**	**I2**	**P**	**I1**	**I2**	**P**	**I1**	**I2**	**P**
1	**2.5**	7	5	8	7	7	7	7	5	6
4	**2.5**	8		7	7	7	7	7	4	6
2	**2+1-**	8		6	7	8	6	4	4	4
5	**2+1-**	7	7	6	8		7	4	4	4
3	**++-**	6	7	6	8		8	5	6	4
7	**++-**	8		8	7	8	8	5	5	4

by a bike path; (2) teachers must live next to the school where they work; (3) bankers and teachers cannot live together; and (4) all houses must be reachable by road from a fire station.

CityPlanning yields instances that are structurally more complex than the previous models. With three professions and four types of building, it has roughly the same number of types as **SpaceTravel**. Yet, the relations among those domains are both more numerous (four instead of three in **SpaceTravel** and two in **FamilyDinner**) and less limited (roads and bike paths can form cycles, and houses and workplaces might be shared).
This creates a much larger potential space of constraints for participants to observe. As the research base on negative information (section 2) emphasizes its value for calling attention to specific features, we wanted to explore whether that value exists even as models grow more complex. Moreover, in spite of the apparently intuitive domain, the constraints were, as our qualitative results will show (section 8.3, "Strategies for Hypothesis Formation") quite subtle. The fact that teachers must live close to their schools may initially seem obvious, but no such constraint exists in real life, and a similar constraint on firefighters (which is *not* present in the model) might be even more intuitive.

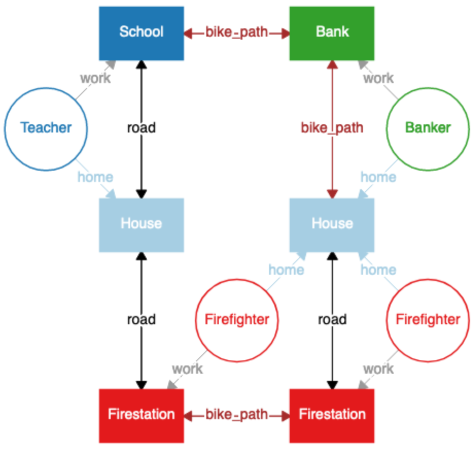


Fig. 6. A valid instance of the **CityPlanning** model.

To confirm that this model is indeed more challenging, we ran an additional Prolific study with it using the **2+1-** configuration. Out of 74 Prolific subjects who started the study, only 48 completed it; of those, only 29 were accepted after quality filtering. Average scores for the first instance test, second instance test, and properties test were 0.45, 0.55, and 0.44 respectively, indicating that participants performed no better than chance. Further, the average time to complete the exploration and instance test sections was around 21 minutes, compared to around 8 minutes for other studies. These results confirm the increased difficulty of the model for Prolific participants.

8.3 Results and Analysis

Table 2 summarizes the participants' numeric results on the instance and properties tests for all three models used in the talk-aloud studies. Although quantitative comparisons are not meaningful here given the small population size, there is useful qualitative information. The drop in scores for two conditions on the **CityPlanning** model suggests that its complexity was useful for studying how negative instances influence hypothesis formation. Moreover, the scores also suggest that the participants had understood the first two models fairly well, and had stayed on task during the sessions. It is interesting that two participants (1 and 4) did notably *worse* on their second instance tests than their first. Unfortunately the data do not provide enough detail to explain; we leave a deeper exploration to a future qualitative study.

We now relate qualitative results. Since we had only two participants for each condition, we did not apply any coding methods to our data. Instead, we focus here on observations that might themselves prompt more detailed future exploration.

Relationships within Groups of Instances. We sought to understand the impact of multiple instances on how participants attempted to form hypotheses. Participants who saw multiple instances together assumed that each group was intended to highlight a specific constraint. For example, a participant in the **2+1-** configuration said:

> So in the two valid ones...so the first one has three flights and this one has two, and both seem to be okay. But this one only has one flight so maybe there's some constraint on like the number of flights, or, like here it seems to be flying from the start to the end directly so there might also be a constraint on that as well, like you can't fly directly to the end.

As described in section 5.2, the positive and negative instances within the same group within **2+1-** did not necessarily highlight the same underlying model feature (unlike in **2.5**, where each negative instance was closely related to the positive instance shown on toggling.) The talk-alouds revealed that participants expected such a relationship within **2+1-** as well, resulting in the lesson that any use of multiple instances may demand careful instance-selection decisions.

Instance Selection and Toggling. In **2.5**, where the negative instances were paired with a closely-related positive instance, participants tended to identify valid constraints more quickly than participants in the other two conditions. While working on **SpaceTravel**, one **2.5** participant said

> Okay so what's wrong with this one? [toggle] Oh, so the transponders [toggle] matters [toggle] because [toggle] you have to [toggle] ... is it because [toggle] if you wanna stop in the middle it has to be the color of your transponders? [Checks transponders on previous pages] Okay so I'm gonna guess this is trying to tell me you have to stop at your transponder's station.

One of the participants in the **CityPlanning** condition used a combination of toggling and revisiting prior instances to narrow down from an observed feature to a specific constraint. In this case, the participant was looking at an instance that was invalid because a teacher's home was not next to their workplace, but the toggled positive instance put the teacher's home next to the school:

> Is this similar [toggle] to the one before? Teacher and banker are living together and that's not allowed, and also … but this one is also like three people living together and I, and you can't have three people living together maybe? [toggle, toggle, observes three living together in left instance] Okay so it's fine to have three people living together, but it's just not the teacher and banker… but is there anything else about this? [toggle] So when it's here the teacher has to make a stop to go to school [toggle] and then

> when it's here the teacher can directly go to school [middle] when it's here teacher goes directly to school [left] teacher goes directly to school… directly to school. So maybe the teacher [toggle, toggle] has to be able to go directly to school [then checks all instances on all pages and validates].

Strategies for Hypothesis Formation. The previously-quoted participant used a negative instance to identify a potential constraint, then confirmed it against the positive instances. We also saw participants identify potential constraints from positive instances, which they then checked against the negative instances (recall that all talk-aloud participants had access to negative instances).

Due to its complexity, the **CityPlanning** model was particularly revealing in how participants used negative instances. Across all conditions, participants consistently started from the positive instances, making hypotheses about basic structural constraints (such as "two bankers can work at the same bank"). Participant 3 stated this strategy explicitly: "my goal right now is to overconstrain so that I catch as much as I can with every instance". Once participants confirmed these basic observations using the negative instances, they worked further with the negative instances to identify constraints on the relations across domains. However, the point at which participants started proposing constraints and their frustrations with doing so differed across the conditions.

The **2.5** participants most quickly began using negative instances to generate hypotheses (though note that ++- participants only received a negative instance on the third screen). They also had relatively more success with generating and confirming hypotheses, and experienced little frustration in doing so. In contrast, both participants in the **2+1-** condition expressed frustration with the process on the **CityPlanning** model. One spent close to an hour on it without seeming to get very far. The other noted that the differences between positive and negative instances were hard to discern and that all instances seemed plausible. The two participants in the ++- condition developed basic hypotheses on the first two models, then began generating ideas for constraints upon seeing the first negative instance. Both went back to the positive instances to check those ideas. Neither participant was able to validate their hypotheses with confidence, with one saying "what could be wrong here? I'm at a loss." One participant also remarked that "I personally probably would have find [sic] more helpful if there were more invalid instances". This speaks to a perceived value of negative instances, though our observations suggest that how those negative instances are juxtaposed with positive ones could have significant impact on their practical utility.

8.4 Summary

The talk-alouds confirmed that participants find negative instances valuable for identifying and validating potential hypotheses. In practice, the way in which negative instances are juxtaposed with positive ones appears critical: participants in the **2.5** condition, who received closely-paired negative and positive instances, experienced more success and less frustration than the other two conditions. Participants in the **2+1-** condition articulated an expectation that negative and positive instances would be related, which strengthens an argument for carefully juxtaposing negative and positive instances regardless of presentation mode.

9 ARTIFACT BUNDLE

This paper's artifact bundle (Dyer et al. 2022) includes:

- the four Alloy models used to generate instances;
- all the study interfaces in all the conditions, to enable reuse and reproduction;
- documentation of the correct answers to all pre-tests, instance tests, and properties tests;
- an enlarged version of fig. 5 for improved legibility; and

- the raw, anonymized, data for each individual in the quantitative studies, including instance and properties test answers and all written responses to instance test and survey questions.

The artifact bundle also includes an experimental model-finder that provides the **2+1-** interface from this study. It accepts Alloy syntax as its source, and uses Sterling as its visualizer. For Alloy `run` commands, the tool generates instances for display in the new interface, while counterexamples from `check` commands are displayed as normal. As it is meant only for demonstration, the model-finder has certain limitations, which we describe in detail in the artifact README file. Future versions of this work are expected to be integrated into the Forge model-finder ((Siegel et al. 2021), available at www.forge-fm.org) for use in an educational setting.

10 THREATS TO VALIDITY

This work naturally has many threats to validity. We discuss each one below.

Internal Validity. We identify two main threats to internal validity. First, any use of paid experimental subjects raises questions of motivation and interest (e.g., our payment may not have sufficed to generate high motivation). To mitigate this, we erected safeguards for both the quantitative and qualitative studies, eliminating participants who were clearly not suitable, but some such participants may have gone undetected. Second, the very small number of qualitative analysis subjects (partially impacted by COVID-19 and student fatigue with online work) forced us to limit the set of conditions used in the qualitative studies, and makes it hard to arrive at general results. In mitigation, we treat that work as especially formative.

External Validity. There are multiple threats to external validity. The largest, perhaps, is that we worked with only a limited set of models and they have a particular flavor. The paper explains in detail our reasons for doing so, but a much larger set of models would be needed to arrive at general conclusions. In particular, we intentionally excluded very large models (so that study time would not become a limiting factor) and models that require domain-specific knowledge (to avoid prior knowledge becoming a confound). Settings that lack these exclusions might exhibit different results (from both a quantitative and qualitative perspective). Secondly, we worked with two specific populations (Prolific subjects and Alloy-trained undergraduates). Subjects with different populations may also see very different results. Finally, our particular strategy for generating negative instances (section 5.1) guaranteed that each instance reflected the negation of one specific target constraint. Our goal was to evaluate negative instances broadly, not to present a specific tool or enumeration strategy. Developing a general tool for finding and presenting negative instances would be interesting; we believe that one of the useful questions our *qualitative* study raises is about which negative instances to show, in light of talk-aloud participants' expectations about how positive and negative instances should be grouped (section 8).

Ecological Validity. Our biggest threat to ecological validity is the constrained nature of our study task. A user of, say, Alloy would also have the source model at hand, not only the output instances. We intentionally excluded this setting for multiple reasons.

First, exposing the source model would introduce many more variables. We could actually be measuring users' understanding of the model or of the modeling language itself, which is complicated and is fraught with potential misconceptions. Any language we used would have these issues, which risk clouding both quantitative and qualitative methods. Second, requiring knowledge of the modeling language would likely also make recruiting subjects for quantitative studies much harder (especially on a crowdsourcing platform). Nevertheless, a natural follow-up to our study is to evaluate the effect of these different output modalities on users who are working with all the affordances that specific tools offer.

However, there *are* practical formal-methods settings that are not far removed from our task: logical models underlie interfaces like system configurators (e.g., you want to purchase a laptop, and there are various constraints on what you can mix-and-match) and security-policy analysis tools. In these settings, end users don't see the formal model (though they do see *some* form of input). Instead, they receive contextual results like failures to configure or descriptions of potential security threats. Although our situation does not exactly match all such uses, these show that the question of ecological validity is actually subtle, and that withholding the model does not *necessarily* damage ecological validity.

11 OTHER RELATED WORK

Section 2 discussed prior cognitive-science work extensively. However, to our knowledge, none of these pertain to formal-methods tools, and most occur in a non-computational context. We therefore discuss other related work from the computer-science literature here.

Kulkarni et al. (2014) use crowdsourcing to examine how exposure to examples at different stages in the design process impacts creativity and conformity. Our studies also provide concrete examples to participants, and it is likely that the comprehension metrics we studied would be correlated with conformity. However, we focus on measuring the value of different enumerations that use side-by-side instances (RQ 1) and negative (RQ 2) information. In contrast, Kulkarni, et al. demonstrate that simply seeing pairs of positive examples has impact.

Bordeaux (Montaghami and Rayside 2017) finds pairs of minimally-different positive instances ("near hits") and negative instances ("near misses"). Crucially, these negative instances, like ours, are not the same as counter-examples to verification, since they are well-defined and (as we show) can be useful even in the absence of verification goals. Likewise, CompoSAT (Porncharoenwase et al. 2018) presents ensembles of instances that each exercise different portions of the underlying model. These works are technically quite appealing, and yet have not been evaluated from a user perspective. Our hope is that this paper will lay the groundwork for more rigorous user-studies of all these works, each backed by cognitive theory. Indeed, our results are especially suggestive of potential benefits from approaches like Bordeaux.

Aluminum (Nelson et al. 2013) and Razor (Saghafi et al. 2015) provide streams of *minimal* instances, along with allowing users to "augment" instances with additional facts. Cunha et al. (2014) and Macedo et al. (2015) find instances as close to (or as far from) a user-provided target as possible. Sullivan et al. (2017) define a test-coverage metric in the model-finding setting and automatically generate instances that increase coverage. Clarisó and Cabot (2020) use graph kernels to cluster instances. Our studies are complementary to all of these. To our knowledge, only minimality has been subjected to formal user studies (Danas et al. 2017). The fact that these experiments showed at best lukewarm results only underscores the value of building tools on a foundation of cognitive theory and rigorous user evaluation.

Computer vision is also concerned with how images can aid structural, analogical, and other types of high-level visual reasoning. It would be interesting to investigate how negative and/or contrasting examples might help in such tasks, perhaps using datasets like as the Abstraction and Reasoning Corpus (ARC) (Chollet 2019) and Relational and Analogical Visual REasoNing (RAVEN) (Zhang et al. 2019).

12 DISCUSSION

We now discuss a few additional aspects of the research as well as areas for future exploration.

Atom Positions. Sterling uses a graph-layout engine to initially position atoms (blobs, space stations, etc.) on the screen; the relative location of atoms is, by itself, meaningless. As expert users,

Q	1+	3+	6+	++-	2+1-	4+2-	2.5
Q1	21	21	14	10	0	5	−15
Q2	7	1	3	0	0	3	0
Q3	30	29	35	33	5	0	27
Q4	27	21	26	29	20	28	33
Q5	−40	−20	−20	7	14	30	28
Q6	−27	−34	−27	−12	11	−3	20
Q7	27	21	16	10	17	14	7
Q8	−38	−38	−49	−27	−30	−50	−24

Fig. 7. For a given group, the difference in percent of participants from that group in the **FeedingRitual** and **FamilyDinner** studies who correctly answered a given first instance test question (in percentage points). Calculated as ($\%FD_{g,q} - \%FR_{g,q}$) where $\%FD_{g,q}$ is the percent of **FamilyDinner** participants from group g who answered question q correctly, and $\%FD_{g,q}$ is the percent of **FeedingRitual** participants from group g who answered question q correctly.

we had internalized this fact and deliberately created the layouts in all our instances (fig. 1) so that they were as similar as possible. We were surprised to discover that a handful of Prolific participants (30 of 224 for **FeedingRitual**, 14 of 222 for **FamilyDinner**, and 6 of 204 for **SpaceTravel**) had inferred constraints based on this consistent placement. For example, during the **FeedingRitual** study, one participant hypothesized: "The Blobs at the bottom cannot sing to the Blobs at the top." Although this phenomenon was rare, we note it here as another important way that human context can impact the perceived meaning of instances.

Isomorphisms in Mathematics, Not for Humans. As the reader might have guessed, **FamilyDinner** is actually the same model as **FeedingRitual** but with an alpha-renaming. We made this change to assess how robust the results would be across models: it may be reasonable to expect that they should produce the same outcomes. Figure 7 shows a heatmap breaking down the difference in performance per question between groups in the two conditions; though there are many low scores (no difference), there are also some high ones. Furthermore, questions related to specific features are affected more drastically than others (some not at all).

This is, however, not entirely surprising. While structurally identical, the names of the relationships between the blobs and the food have been changed such that blobs now have a `parentOf` relation in place of the `singingTo` relation. In doing so, it seems more likely that participants would have pre-existing ideas about how that relation might be constrained. In particular, they may be more likely to assume that cyclic `parentOf` relationships cannot occur, whereas cyclic singing (i.e., two blobs singing to each other) is entirely plausible. Subjects are also more likely to think about how many children and/or parents a single blob can have, moreso than they might think about how many blobs one can sing to or be sung to by.

Indeed, there are other results in the psychology of reasoning that show that the framing can affect outcomes. For instance, the Wason selection task (Wason 1966) is a well-known example that is considered difficult for people to solve correctly according to the rules of logic. However, later work (Cosmides and Tooby 1992; Stenning and van Lambalgen 2008) has shown that isomorphic problems that change the domain framing can significantly improve performance. Thus, it is not surprising that a mathematically isomorphic model does not yield identical results.

Properties Test Analysis. We have not discussed the properties test results in much detail for any of the studies for two reasons. First, it is the least authentic of the tasks from a perspective of ecological validity. Second, those results showed little variation from the **1+** group across the board.

That said, in the cognitive science literature there is a distinction between “telling apart things in a family” and “telling apart families”, which loosely corresponds to the distinction between identifying instances and identifying properties. However, this is not a connection we have had room to delve into in these studies, and we leave it for future work. Nevertheless, we describe the properties test and report its results since it was part of the study we ran; omitting it would be poor science, especially since it likely added to the cognitive load of participants.

Task Times. In general, participants used different amounts of times under different conditions. We discuss some of these in section 8, but those of course are with tiny sample sizes. Because we cannot know what other tasks Prolific users were doing, we chose to not perform a deep analysis of their timing data. However, the preliminary data we have from them indicates that participants with negative information spent more time exploring instances and required less time to complete instance tests. In the **SpaceTravel** study, for example, compared to the groups with only positive instances, those with negative instances spent on average 25 seconds longer exploring instances before moving on to the first instance test, which they were able to complete 35 seconds faster. This trend is also observed to an extent in the **FeedingRitual** and **FamilyDinner** studies. This suggests that timing, which was not part of our research design, might nevertheless be an important factor—and should be designed and controlled for—in future studies.

Interactivity in Output. The **2.5** results suggest that, at least in some cases, there is real value to making the output of a model-finder *interactive*. Indeed, there are many more opportunities available for interaction. For instance, one can imagine being able to directly manipulate the elements of an instance—e.g., adding or removing nodes or edges—and being informed whether the result is or is not an instance of the model. Another example is that, as instances go by, users can “bookmark” some of them to put in a panel for side-by-side comparison—akin to **6+** or **4+2-**, but chosen by the user. (We have prototyped these interfaces, but not evaluated them.) In addition, there are numerous metaphors in modern user interfaces (e.g., tabbed browsing) that have at least the benefit of familiarity and thus may be worth exploring in this setting. Finally, there are other ideas—such as the minimal exploration method of Aluminum (Nelson et al. 2013)—that can also be put to work.

Suggestion for Tool Developers. Our work suggests that at least negative instances, and perhaps also multiple instances at a time, are worth a much deeper exploration. While we do not claim that any of the interfaces we used in our studies are ideal (or even universally beneficial), we feel it would be beneficial for tools to incorporate forms of negative-instance output as experimental options. By deploying interfaces in real systems and monitoring how users employ them—which will also address questions of ecological validity—we believe a great deal can be learned about the applicability of the principles that drove their design. This will be even more true of more novel forms of interaction in instance output; as Section 8.4 shows, we must take care in how multiple instances are grouped together, as users aim to leverage the constrasts between instances to identify features and constraints of interest.

13 USER STUDIES FOR FORMAL METHODS

Formal methods tools are gaining in importance and popularity. Yet, as our related work (section 11) shows, their human factors aspects have largely been overlooked, perhaps falling between two communities that are currently quite removed. There is growing interest in the usability of proof assistants (e.g., the new HATRA workshop series), which certainly present additional challenges, but even basic questions such as the ones we raise remain largely unexplored.

Some work in this space might take the form of assessing the usability of tools as they are. These can proceed along fairly standard lines familiar to a person versed in human-computer interaction methods. Our work presents a somewhat different roadmap. We instead *ground* our work in theories from outside computer science—specifically, in our case, from cognitive science—to inform how a tool might behave. Combining various results leads us to the numerous conditions (section 5.3) that we explore.

No matter which of these paths one takes, as this paper demonstrates, there are several important cautions to keep in mind. Some of these will be clear to people familiar with human factors and qualitative work; others may be more peculiar to formal methods. The paper is already written to discuss some of these as they arise (as two examples, the design of models in section 4 and the need to filter crowdsourced participants in section 6.3). Here we discuss some higher-level issues:

- Be mindful of confounding factors. Earlier, for instance, we describe the importance of avoiding the Stroop effect (section 4.3). Obviously, there is no comprehensive list of these that one can use as a checklist. Rather, reasoning about them requires approaching experiment design with an "adversarial mindset" (no doubt, we have failed in some ways, too!).
  One particularly important phenomenon to keep in mind is that these tools have numerous "moving parts". It can therefore be very easy to believe one is studying one aspect while inadvertently studying another. For instance, our emphasis was on *output*, but had we included the sources of models, we might have actually been studying Alloy language comprehension (section 10). In return, of course, our study does not allow us to evaluate overall comprehensibility. Big questions like the latter need to be broken down into a series of small studies that carefully control for these phenomena before evaluating them in combination.
- Mathematics $\neq$ life. Mathematical equivalences may not appear that way to humans: for instance, section 12 discusses how humans do not treat isomorphic models identically (or even all that similarly). Therefore, one cannot use mathematics to prune the set of possibilities too much. Indeed, many interesting phenomena are grounded *precisely* in this mismatch between human and mathematical reasoning (as we discuss in the case of the Wason task).
- Acknowledge that not all use cases are the same, even if the underlying formalism is. As we mentioned in section 1, model finders support many different modalities of use: everything from model-checking and verification (where the user may be an expert in both the domain and the modeling language) to automated configurators to security policy checkers. In the latter cases, the user may be, and perhaps *should* be, unaware of the modeling details. The output matters for all users, but their needs and abilities are very different.
- Embrace both qualitative and quantitative methods, as appropriate. Researchers who primarily traffic in technical content are often ignorant or wary of qualitative methods, or dismiss of them as having "small N" and/or not providing p-values. However, qualitative methods are not merely statistically underpowered versions of quantitative studies. Rather, they have their own methods and logic for arriving at useful and valid conclusions. (One example, mentioned in section 6.3, is the use of inter-rater reliability, which helps check that multiple researchers examining the same qualitative data are arriving at similar conclusions.)
- In our experience, qualitative methods provide deep insights that quantitative methods often cannot. For instance, check-boxes are easy to process and quantify, but they often do not capture interesting insights into thought processes that emerge from letting people speak freely (as we see in section 8.3). To really understand *how* formal methods are working for people, qualitative methods are essential.

ACKNOWLEDGMENTS

This work was partially supported by the US National Science Foundation, Defense Advanced Research Projects Agency (DARPA), and Air Force Research Laboratory (AFRL). The views, opinions and/or findings expressed are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. We thank Doug Woos and the Brown PLT reading group for useful exchanges, and especially thank Rob Goldstone for numerous discussions (but stress that any errors are ours alone). We thank John Baugh, who introduced the first author to the others. An anonymous PC member kindly pointed us to the ARC and RAVEN datasets.

REFERENCES

Devdatta Akhawe, Adam Barth, Peifung E. Lam, John Mitchell, and Dawn Song. 2010. Towards a Formal Foundation of Web Security. In *IEEE Computer Security Foundations Symposium*. https://doi.org/10.1109/CSF.2010.27

Louis Alfieri, Timothy J. Nokes-Malach, and Christian D. Schunn. 2013. Learning Through Case Comparisons: A Meta-Analytic Review. *Educational Psychologist* 48, 2 (2013). https://doi.org/10.1080/00461520.2013.775712

R. K. Atkinson, S. J. Derry, A. Renkl, and D. Wortham. 2000. Learning from examples: instructional principles from the worked examples research. *Review of Educational Research* 70, 2 (2000). https://doi.org/10.3102/00346543070002181

John Backes, Sam Bayless, Byron Cook, Catherine Dodge, Andrew Gacek, Alan J. Hu, Temesghen Kahsai, Bill Kocik, Evgenii Kotelnikov, Jure Kukovec, Sean McLaughlin, Jason Reed, Neha Rungta, John Sizemore, Mark A. Stalzer, Preethi Srinivasan, Pavle Subotic, Carsten Varming, and Blake Whaley. 2019. Reachability Analysis for AWS-Based Networks. In *Computer Aided Verification - 31st International Conference, CAV 2019, New York City, NY, USA, July 15-18, 2019, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 11562)*, Isil Dillig and Serdar Tasiran (Eds.). Springer, 231–241. https://doi.org/10.1007/978-3-030-25543-5_14

John Backes, Pauline Bolignano, Byron Cook, Catherine Dodge, Andrew Gacek, Kasper Søe Luckow, Neha Rungta, Oksana Tkachuk, and Carsten Varming. 2018. Semantic-based Automated Reasoning for AWS Access Policies using SMT. In *Formal Methods in Computer-Aided Design*. IEEE, 1–9. https://doi.org/10.23919/FMCAD.2018.8602994

Clark Barrett, Christopher L. Conway, Morgan Deters, Liana Hadarean, Dejan Jovanović, Tim King, Andrew Reynolds, and Cesare Tinelli. 2011. CVC4. In *International Conference on Computer Aided Verification*. Springer Berlin Heidelberg, Berlin, Heidelberg, 171–177. https://doi.org/10.1007/978-3-642-22110-1_14

Irving Biederman and Margaret M. Shiffrar. 1987. Sexing Day-Old Chicks: A Case Study and Expert Systems Analysis of a Difficult Perceptual-Learning Task. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 13, 4 (1987), 640–645. https://doi.org/10.1037/0278-7393.13.4.640

R. Brown and C. Hanlon. 1970. Derivational complexity and order of acquisition on child speech. In *Cognition and the development of language*, J. Hayes (Ed.). Wiley.

François Chollet. 2019. On the Measure of Intelligence. arXiv:1911.01547 [cs.AI]

Robert Clarisó and Jordi Cabot. 2020. Diverse Scenario Exploration in Model Finders Using Graph Kernels and Clustering. In *Rigorous State Based Methods*. Springer International Publishing, Cham, 27–43. https://doi.org/10.1007/978-3-030-48077-6_3

Edmund M. Clarke, E. Allen Emerson, and Joseph Sifakis. 2009. Model Checking: Algorithmic Verification and Debugging. *Commun. ACM* 52, 11 (Nov. 2009), 74–84. https://doi.org/10.1145/1592761.1592781

Jacob Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement* 20, 1 (1960), 37–46. https://doi.org/10.1177/001316446002000104

Leda Cosmides and John Tooby. 1992. Cognitive Adaptions for Social Exchange. In *The Adapted Mind: Evolutionary Psychology and the Generation of Culture*, Leda Cosmides, John Tooby, and Jerome H. Barkow (Eds.). Oxford University Press.

Alcino Cunha, Nuno Macedo, and Tiago Guimarães. 2014. Target Oriented Relational Model Finding. In *International Conference on Fundamental Approaches to Software Engineering*. Springer, 17–31. https://doi.org/10.1007/978-3-642-54804-8_2

Natasha Danas, Tim Nelson, Lane Harrison, Shriram Krishnamurthi, and Daniel J. Dougherty. 2017. User Studies of Principled Model Finder Output. In *Software Engineering and Formal Methods*. https://doi.org/10.1007/978-3-319-66197-1_11

Leonardo de Moura and N. Bjørner. 2008. Z3: An efficient SMT solver. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems (Lecture Notes in Computer Science, Vol. 4963)*. Springer, 337. https://doi.org/10.1007/978-3-540-78800-3_24

Charles W. Dunnett. 1955. A Multiple Comparison Procedure for Comparing Several Treatments with a Control. *J. Amer. Statist. Assoc.* 50, 272 (1955), 1096–1121. https://doi.org/10.1080/01621459.1955.10501294 arXiv:https://www.tandfonline.com/doi/pdf/10.1080/01621459.1955.10501294

Tristan Dyer and John Baugh. 2021. Sterling: A web-based visualizer for relational modeling languages. In *Rigorous State Based Methods. ABZ 2021*. Springer, Cham, 99–104. https://doi.org/10.1007/978-3-030-77543-8_7 Lecture Notes in Computer Science 12709.

Tristan Dyer, Tim Nelson, Kathi Fisler, and Shriram Krishnamurthi. 2022. *Applying Cognitive Principles to Model-Finding Output: The Positive Value of Negative Information (artifact).* https://doi.org/10.5281/zenodo.6370152

Ari Fogel, Stanley Fung, Luis Pedrosa, Meg Walraed-Sullivan, Ramesh Govindan, Ratul Mahajan, and Todd Millstein. 2015. A General Approach to Network Configuration Analysis. In *Networked Systems Design and Implementation*. 469–483. https://doi.org/10.5555/2789770.2789803

Eleanor J. Gibson. 1969. *Principles of Perceptual Learning and Development.* Appleton-Century-Crofts.

James J. Gibson and Eleanor J. Gibson. 1955. Perceptual Learning: Differentiation or Enrichment? *Psychological Review* 62, 1 (1955), 32–41. https://doi.org/10.1037/h0048826

M. L. Gick and K. Paterson. 1992. Do contrasting examples facilitate schema acquisition and analogical transfer? *Canadian Journal of Psychology* 46, 4 (1992). https://doi.org/10.1037/h0084333

Rubi Hammer, Tomer Hertz, Shaul Hochstein, and Daphna Weinshall. 2009. Category learning from equivalence constraints. *Cognitive Processing* 10, 3 (2009), 211–232. https://doi.org/10.1007/s10339-008-0243-x

William L. Hays. 1994. *Statistics* (5th ed.). Harcourt Brace College Publishers, Fort Worth.

Daniel Jackson. 2012. *Software Abstractions: Logic, Language, and Analysis* (2 ed.). MIT Press. https://doi.org/10.5555/2141100

Aniket Kittur, Ed H. Chi, and Bongwon Suh. 2008. Crowdsourcing User Studies with Mechanical Turk. In *SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 453–456. https://doi.org/10.1145/1357054.1357127

Chinmay Kulkarni, Steven P. Dow, and Scott R. Klemmer. 2014. *Early and Repeated Exposure to Examples Improves Creative Work.* Springer International Publishing, 49–62. https://doi.org/10.1007/978-3-319-01303-9_4

Angelika Kullberg, Ulla Runesson Kempe, and Ference Marton. 2017. What is made possible to learn when using the variation theory of learning in teaching mathematics? *ZDM: The International Journal on Mathematics Education* 49, 3 (2017). https://doi.org/10.1007/s11858-017-0858-4

Nuno Macedo, Alcino Cunha, and Tiago Guimarães. 2015. Exploring Scenario Exploration. In *International Conference on Fundamental Approaches to Software Engineering*. https://doi.org/10.1007/978-3-662-46675-9_20

Ferney A. Maldonado-Lopez, Jaime Chavarriaga, and Yezid Donoso. 2014. Detecting Network Policy Conflicts Using Alloy. In *Conference on Abstract State Machines, Alloy, B, and Z*. https://doi.org/10.1007/978-3-662-43652-3_31

Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. 2011. CD2Alloy: Class Diagrams Analysis Using Alloy Revisited. In *Model Driven Engineering Languages and Systems*. https://doi.org/10.1007/978-3-642-24485-8_44

Gary F. Marcus. 1993. Negative evidence in language acquisition. *Cognition* 46 (1993). https://doi.org/10.1016/0010-0277(93)90022-N

Ference Marton. 2014. *Necessary Conditions of Learning.* Routledge.

Ference Marton and Ming Fai Pang. 2013. Meanings are acquired from experiencing differences against a background of sameness, rather than from experiencing sameness against a background of difference: Putting a conjecture to the test by embedding it in a pedagogical tool. *Frontline Learning Research* 1 (2013). https://doi.org/10.14786/flr.v1i1.16

Vajih Montaghami and Derek Rayside. 2017. Bordeaux: A Tool for Thinking Outside the Box. In *International Conference on Fundamental Approaches to Software Engineering*. 22–39. https://doi.org/10.1007/978-3-662-54494-5_2

Timothy Nelson, Christopher Barratt, Daniel J. Dougherty, Kathi Fisler, and Shriram Krishnamurthi. 2010. The Margrave Tool for Firewall Analysis. In *USENIX Large Installation System Administration Conference.*

Tim Nelson, Salman Saghafi, Daniel J. Dougherty, Kathi Fisler, and Shriram Krishnamurthi. 2013. Aluminum: Principled Scenario Exploration Through Minimality. In *International Conference on Software Engineering*. https://doi.org/10.1109/ICSE.2013.6606569

Oded Padon, Kenneth L. McMillan, Aurojit Panda, Mooly Sagiv, and Sharon Shoham. 2016. Ivy: Safety Verification by Interactive Generalization. In *Programming Language Design and Implementation (PLDI)*. Association for Computing Machinery, New York, NY, USA, 614–630. https://doi.org/10.1145/2908080.2908118

Sorawee Porncharoenwase, Tim Nelson, and Shriram Krishnamurthi. 2018. CompoSAT: Specification-Guided Coverage for Model Finding. In *International Symposium on Formal Methods (FM)*. https://doi.org/10.1007/978-3-319-95582-7_34

B. Rittle-Johnson and J. Star. 2007. Does comparing solution methods facilitate conceptual and procedural knowledge: An experimental study on learning to solve equations. *Journal of Educational Psychology* 99 (2007). https://doi.org/10.1037/0022-0663.99.3.561

B. Rittle-Johnson and J. R. Star. 2009. Compared with what? The effects of different comparisons on conceptual knowledge and procedural flexibility for equation solving. *Journal of Educational Psychology* 101, 3 (2009). https://doi.org/10.1037/a0014224

Santiago Perez De Rosso and Daniel Jackson. 2013. What's Wrong with Git?: A Conceptual Design Analysis. In *SPLASH Onward!* ACM, 37–52. https://doi.org/10.1145/2509578.2509584

Natali Ruchansky and Davide Proserpio. 2013. A (Not) NICE Way to Verify the OpenFlow Switch Specification: Formal Modelling of the OpenFlow Switch Using Alloy. *ACM Computer Communication Review* 43, 4 (Aug. 2013), 527–528. https://doi.org/10.1145/2486001.2491711

Salman Saghafi, Natasha Danas, and Daniel J Dougherty. 2015. Exploring Theories with a Model-Finding Assistant. In *International Conference on Automated Deduction*. Springer, 434–449. https://doi.org/10.1007/978-3-319-21401-6_30

Daniel L. Schwartz, Catherine C. Chase, Marily A. Oppezzo, and Doris B. Chin. 2011. Practicing versus inventing with contrasting cases: The effects of telling first on learning and transfer. *Journal of Educational Psychology* 103, 4 (2011), 759–775. https://doi.org/10.1037/a0025140

Eric L. Seidel, Ranjit Jhala, and Westley Weimer. 2016. Dynamic Witnesses for Static Type Errors (or, Ill-Typed Programs Usually Go Wrong). In *International Conference on Functional Programming (ICFP)*. Association for Computing Machinery, New York, NY, USA, 228–242. https://doi.org/10.1145/2951913.2951915

Abigail Siegel, Mia Santomauro, Tristan Dyer, Tim Nelson, and Shriram Krishnamurthi. 2021. Prototyping Formal Methods Tools: A Protocol Analysis Case Study. In *Protocols, Strands, and Logic*. 394–413. https://doi.org/10.1007/978-3-030-91631-2_22

Keith Stenning and Michiel van Lambalgen. 2008. *Human Reasoning and Cognitive Science*. MIT Press.

J. R. Stroop. 1935. Studies of interference in serial verbal reactions. *Journal of Experimental Psychology* 18 (1935), 643–662. Issue 6. https://doi.org/10.1037/h0054651

Allison Sullivan, Kaiyuan Wang, Razieh Nokhbeh Zaeem, and Sarfraz Khurshid. 2017. Automated Test Generation and Mutation Testing for Alloy. In *Software Testing, Verification and Validation (ICST)*. https://doi.org/10.1109/ICST.2017.31

Peter Cathcart Wason. 1966. Reasoning. In *New Horizons in Psychology I*, B. M. Foss (Ed.). Penguin.

Patrick H. Winston. 1970. *Learning Structural Descriptions from Examples*. AI Technical Report AITR-231. Massachusetts Institute of Technology.

Pamela Zave. 2012. Using Lightweight Modeling to Understand Chord. *ACM Computer Communication Review* 42, 2 (March 2012), 49–57. https://doi.org/10.1145/2185376.2185383

Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. 2019. RAVEN: A Dataset for Relational and Analogical Visual REasoNing. In *Computer Vision and Pattern Recognition (CVPR)*. Computer Vision Foundation / IEEE, 5317–5327. https://doi.org/10.1109/CVPR.2019.00546